DENEYAP
Teknoloji Atölyeleri

ROBOTİK VE KODLAMA LİSE

Micropython

Yazarlar

Prof. Dr. İbrahim ÇETİN
Doç. Dr. Memet ÜÇGÜL
Prof. Dr. Ercan TOP
Prof. Dr. Erman YÜKSELTÜRK

Güncelleme

Doç. Dr. Memet ÜÇGÜL
Haydar Emin ERDOĞAN
Melike ÖZDEMİR

İçindekiler

ROBOTİK KODLAMA DERSİ ÖĞRETİM KILAVUZU 1
1. Hafta: Robotlarla Algoritma ve Hareket 6
GÖZLE VE UYGULA 7
Gözle: Robot Kavramı Üzerine Tartışma 7
Uygula: Algoritma Öğreniyorum 8
Uygula: Çarpım Eşleri Algoritması 8
Gözle: Robot Yapımı 8
LEGO Education SPIKE Prime Yazılımı 10
Gözle: Programlama Alanının Kullanılması 13
Gözle: Robotun Hareket Ettirilmesi 15
Gözle: Robotun Temel Hareketleri 16
Gözle ve Uygula: İleri ve Geri Hareket Eden Robot 18
Uygula: Kare Çizen Robot 19
Uygula: A noktasından B noktasına Hareket 20
TASARLA 23
ÜRET 24
DEĞERLENDİR 24
İLAVE ETKİNLİK 25
Sonsuz İşareti Şeklinde Hareket Etme 25
2. Hafta – Işık, Ses ve Mesafe Sensörü 27
GÖZLE VE UYGULA 28
IŞIK 28
Gözle: Hub Ekranı 30
Uygula: Büyüyen Kalp Efekti 30
Gözle: Sürekli Tekrar Eden Büyüyen Kalp Efekti 31
SES 32
Gözle: Ses Blokları 33
Uygula: Kare Çize Robot Aktivitesi 34
MESAFE SENSÖRÜ 35
Gözle: Mesafe Sensörü 36
Gözle: Belirli Bir Mesafeye Kadar İlerleme 38
Uygula: İstenilen Mesafe Kadar Geri Gitme 38
Uygula: Engele Yaklaştıkça Yavaşlayan Robot 39
Gözle: Robota Yaklaşan Nesne Olursa Ses Çıkar 40
Uygula: Park Sensörü 40
TASARLA VE ÜRET 41
DEĞERLENDİR 43

İLAVE ETKİNLİK 44
Görev-Takımlar Yarışıyor 44
3. Hafta - Kuvvet ve Hareket Sensörleri 45
Kuvvet Sensörü 46
GÖZLE VE UYGULA 49
Gözle: Kuvvet Sensörü 49
Uygula: Kuvvet Sensörü ile Motorların Çalıştırılması 50
Gözle: Kuvvet Sensörü ile İlk Program 50
Gözle ve Uygula: Kuvvet Sensörü Oyunu 51
Gözle ve Uygula: Engele Çarpınca Duran Robot 52
Gözle ve Uygula: Çarpışma Testi 54
Hareket Sensörü 55
GÖZLE VE UYGULA 60
Gözle ve Uygula: Robotun 90 Derece Döndürülmesi 60
Uygula: Kare Şeklinde Hareket 63
Gözle ve Uygula: Yunuslama, Yuvarlanma Işıkları 63
TASARLA 65
ÜRET 67
DEĞERLENDİR 68
İLAVE ETKİNLİK 69
Yönünü Kaybetmeyen Robot 69
4. Hafta – Renk Sensörü ve Sensörlerin Birlikte Kullanımı 71
Renk Sensörü ve Sensörlerin Birlikte Kullanımı 72
GÖZLE VE UYGULA 74
Gözle ve Uygula: Lego Bloğu Kırmızı Renk mi? 74
Gözle ve Uygula: Değil (Not) Mantık Operatörü 75
Gözle ve Uygula: Lego Bloğunun Renk Numarasını Hub Ekranında Gösterme 76
Gözle: Yansıyan Işık Şiddeti 77
Uygula: Belirli Bir Alandan Çıkmayan Robot 78
Gözle ve Uygula: Veya (Or) Mantık Operatörü 80
Gözle ve Uygula: Çizgi Takibi 82
Gözle ve Uygula: Ve (And) Mantık Operatörü 83
TASARLA 83
ÜRET 88
DEĞERLENDİR 90
5. Hafta – Python Programlama ile Robotu Hareket Ettirme 91
GÖZLE VE UYGULA 92
Gözle: Sürüş Modelinde Direksiyon Hareketi 93
Uygula: Direksiyon Hareketini Keşfediyorum 94

Gözle: Direksiyon Hareketinin Detaylarını Öğreniyorum 94
Uygula: İlerleyip Aynı Yere Gelen Robot 95
Uygula: Tekerin Yarıçapını Hesaplıyorum 96
Uygula: 360 Derece Dönen Robot 96
Gözle: Palet Hareketi 97
Uygula: Sürüş Modelinde Palet Hareketi 97
Gözle: Robotun Özelliklerini Ayarlıyorum 98
Gözle: Motoru Belirli Bir Açıya Getirme 98
Uygula: İşaretçi Direk Eşleştirmesi 100
Gözle: Motoru Belirlenen Başlangıç Noktasına Göre İstenen Bir Açıya Getirme 100
Uygula: Üç Turda İşaretçi Direk Eşleştirmesi 102
Gözle: Belirli Bir Derece/Tur/Saniye Hareket Etme 102
Uygula: Belirli Bir Derece/Tur/Saniye Hareket Etme 102
TASARLA 103
Tasarla: Cisimleri Kenardan Ortaya Taşıyorum 103
ÜRET 106
DEĞERLENDİR 106
İLAVE ETKİNLİK 106
Sonsuz Üzerinde Hareket 106
6. Hafta – Işık Matrisi, Hoparlör ve Hub Düğmeleri 108
GÖZLE VE UYGULA 109
Gözle: Işık Matrisinde Yazı Yazdırmak 110
Gözle: Değişkenlere Giriş 110
Gözle ve Uygula: 0'dan 5'e kadar sayıyorum 111
Gözle ve Uygula: 0'dan 5'e kadar sayıyorum 112
Uygula: Geri Sayım Yaptırıyorum 113
Gözle: Piksel Kontrolü ile Işık Matrisi 113
Uygula: Geri Sayım Öncesi Yanıp Sönme 114
Gözle ve Uygula: Ekrana Resim Çiziyorum 115
Uygula: Ok Animasyonu 115
Gözle ve Uygula: Sürekli Tekrarlanan Görevler (While Döngüsü) 115
Gözle ve Uygula: Listeler 116
Gözle: Robotum Ses Veriyor 117
Uygula: Kare Şeklinde Hareket Ederken Ses Çıkarıp Yön Gösteren Robot 117
Gözle: Hub'ın Sol ve Sağ Düğmeleri 118
Uygula: Düğmeye Basılma Süresi 119
TASARLA 119
Tasarla: Dans Eden Robot 119
ÜRET 120

DEĞERLENDİR 120
7. Hafta – Kuvvet Sensörü 122
GÖZLE VE UYGULA 123
Gözle: Kuvvet Sensörünü Tanımlayıp Kullanıyorum 124
Gözle: Koşul İfadeleri 124
Uygula: Sürekli Tekrar ve Koşul İfadesinin Birlikte Kullanımı 125
Uygula: Döngü, Koşul ve Değişkenin Bir Arada Kullanımı 126
Gözle: Miktar Belirtmeden Sürüş Modeli ve Motorları Hareket Ettirme 126
Gözle ve Uygula: Engele Çarpınca Geri Giden Robot 128
Uygula: Sürekli olarak Engele Çarptığında Geri Dönen Robot 129
Gözle: Kuvvet Sensörüne Her Basıldığında Bir Kere Bip Sesi Çalan Program 130
Uygula: Kuvvet Sensörüne Her Basılıp Çekildiğinde Bir Kere Bip Sesi Çalan Program 131
Gözle ve Uygula: Uygulanan Kuvveti Ölçüp Işık Matrisinde Yazdırıyorum 131
Uygula: Kuvvet Miktarına Göre Değişen Bip Sesi 132
Gözle: İki Olay Arasında Geçen Süreyi Saniye Cinsinden Hesaplıyorum 132
Uygula: Sürüş Modelinin Maksimum Hızını Buluyorum 133
TASARLA 134
Kuvvet Sensörü Oyunu 134
ÜRET 135
DEĞERLENDİR 135
İLAVE ETKİNLİK 135
8. Hafta – Hareket (Gyro) Sensörü 137
GÖZLE VE UYGULA 138
Gözle: Hub'ın Açısal Hareketlerini Öğreniyorum 139
Gözle ve Uygula: Sürüş Modelini 90 Derece Döndürüyorum 141
Uygula: Kare şeklinde hareket 142
Gözle: Hub Yönünün Algılanması 143
Uygula: Sürekli Yukarıyı Gösteren Ok Resimleri 144
Uygula: Yunuslama Açısına Göre İleri Geri Dönen Tekerlekler 144
Gözle: Hub ile Yön Değişimine Bağlı İşlem Yapma 145
Uygula: Sürekli Yukarıyı Gösteren Ok Resimleri - Yeni Versiyon 146
Gözle: Hub'a Tıklama, Hub'ı Sallama ve Düşürme Hareketleri 146
Uygula: Hub Hareketine Göre Ses Çıkaran Robot 147
Gözle: Tıklama Sayısı 148
Uygula: Tıklanma Sayısı Kadar İleri Giden Robot 148
TASARLA 148
Hareket Sensörü ile Programlanabilen Robot 148
ÜRET 149
DEĞERLENDİR 150

İLAVE ETKİNLİKLER 151
9. Hafta – Mesafe Sensörü 155
GÖZLE VE UYGULA 156
Gözle: Sol Gözünü Kırpan Robot 156
Uygula: Gözlerini Sırayla Kırpan Robot 157
Uygula: Fonksiyon Kullanarak Göz Kırpma 158
Gözle: Belirli Bir Mesafeye Kadar Giden Robot 159
Uygula: Fonksiyon Kullanarak Belirli Bir Mesafeye Kadar Giden Robot 160
Gözle ve Uygula: Öndeki Aracı Takip Eden Robot 161
Uygula: Öndeki Aracı Takip Eden Gelişmiş Robot 162
Gözle: Duvar Takibi Yapan Robot 163
Uygula: Duvar Takibi ve Sağa Dönüş 165
TASARLA 167
Hedef Tespit Robotu 167
ÜRET 171
Hedef Tespit Robotu 171
DEĞERLENDİR 173
İLAVE ETKİNLİKLER 173
Kaç Duvar Olduğunu Sayan Robot 173
LEGO Kutusu Etrafında Dönen Robot Yarışması 174
10. Hafta – Renk Sensörü 176
GÖZLE VE UYGULA 177
Gözle: Cismin Rengini Belirliyorum 177
Gözle: İleri ve Geri Komutunu Renkler ile Veriyorum 178
Uygula: Renk Sensörü ile Programlanabilen Robot 179
Gözle: Siyah Rengi Görene Kadar İlerleyen Robot 180
Uygula: Renk Kodları ile Harekete Başlayan Robot 181
Gözle: Ham Renkleri Öğreniyorum 181
Uygula: Turuncu Rengi Algıladığında Bip Sesi Çalan Robot 182
Gözle ve Uygula: Yansıyan Işık Yoğunluğunu Belirleyen Faktörler 183
Gözle ve Uygula: PID Algoritması ile Çizgi Takip Eden Robot 184
Uygula: Dikdörtgen Şeklinde Çizgiyi Takip Eden Robot 187
TASARLA 188
Lego Parçasıyla Robotu Kodluyorum 188
ÜRET 193
Lego Parçasıyla Robotu Kodluyorum 193
DEĞERLENDİR 197
11. Hafta- Proje Hazırlıyoruz 199
Haftanın Amacı: 199

Haftanın Kazanımları:..... 199
GELİŞTİRME – TEST ETME 200
12. Hafta- Proje Hazırlıyoruz 201
Haftanın Amacı:..... 201
Haftanın Kazanımları:..... 201
SUNUM 202

ROBOTİK KODLAMA DERSİ ÖĞRETİM KILAVUZU

Dünyada 21.yüzyıl ile başlayan yenilikler yaşamın her alanını etkilemektedir. Hayatımızdaki hızlı değişim ve gelişim, her gün beraberinde yeni bilgiler, yeni teknolojiler ve yeni yaklaşım tarzları ortaya çıkarmaktadır. Bu gelişmelere bağlı olarak küresel eğitimde ve işgücü piyasasında ihtiyaç duyulan öğrenen ve çalışan özelliklerinde değişimler gözlenmektedir. Bu nedenlerden, eleştirel düşünme, problem çözme, analiz ve sentez yapma, işbirlikli öğrenme ortamlarında çalışabilme, doğru ve güncel bilgiye kolay bir şekilde ulaşabilme ve yenilikçi olabilme gibi üst düzey becerilerin kazanılması beklenmektedir. Alan yazında birçok çalışma da göstermiştir ki iyi planlanan programlama eğitiminin, 21.yüzyıl becerileri (P21, 2020) olarak da tarif edilen çeşitli üst düzey becerilerin kazanımında olumlu katkılar sağlamaktadır (Gürer, Çetin & Top, 2019; Yükseltürk & Altıok, 2017).

Programlama ile öğrencilerin bilgi okuryazarlığı, algoritmik düşünme ve problem çözme becerilerinin kazanılmasında etkili olsa da bunlarla sınırlı değildir. Programlama eğitimi ile öğrencilerin araştırma yapması sağlanarak özgün bir projenin baştan sona gelişmesine yardımcı olunabilir. Ayrıca, yapılacak grup çalışmaları ile sorumluluk becerilerinin, hazırlanacak projeler ile de işbirlikli çalışma becerilerinin kazanılmasına da katkı sağlanabilir (Gürer, Çetin & Top, 2019; Yükseltürk & Altıok, 2017).

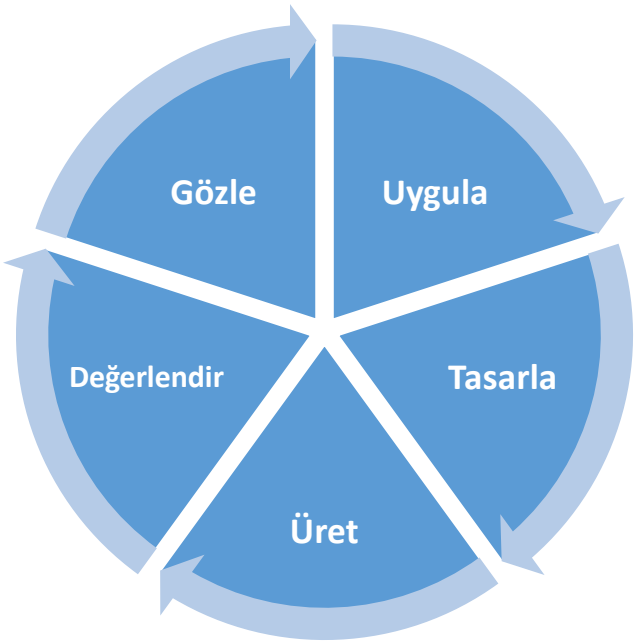
Günümüzde bireylerden bilgi ve iletişim teknolojileri okur-yazarı olmanın yanında hem eğitim hem de iş hayatında farklı dijital araçları kullanarak karşılaştıkları problemlere çözümler üretmesi beklenmektedir. Araştırmacılar hem temel bilgisayar bilimleri kavramlarını hem de programlama becerilerini bilgi işlemsel düşünme ile ilişkilendirmektedir. Bilgi işlemsel düşünme ise temel olarak bir problem çözme sürecidir ve beş bileşeni içerir (Wing, 2008, 2011):

(i) soyutlama, probleme ilişkin toplanan ve çözümlenen bilgiler ve çözümlerin sınıflandırılması yoluyla kuralların kavramsallaştırılması
(ii) algoritma, problemler için yeni çözümlerin inşa edilmesi
(iii) otomasyon yada otomatikleştirme, geliştirilen kavramsallaştırmaların farklı sorunlarda da verimli ve etkili bir biçimde işlemesi
(iv) problemleri ayrıştırma, problemlerin daha küçük anlamlı alt problemlere parçalanması
(v) genelleme, problemlere ilişkin üretilen çözüm desenlerinin benzer problemlerde de uygulanabilir olması

Bu derste, öğrencilerin öncelikle bilgi işlemsel düşünme becerisinin geliştirilmesi için robotik programlama eğitiminin verilmesi planlanmaktadır. Yapılandırmacı (constructivist) eğitim yaklaşımını temel alınarak tasarlanan derste, öğrenenin rolü aktif bir şekilde bilgiyi inşa etmek, eğitmen rolü ise öğrencilerini pedagojik olarak destekleyerek keşfetme veya buluş yapabilme ortamlarını hazırlanmak olarak belirlenmiştir. Ders boyunca hazırlanan etkinliklerde programlanabilir robotik materyaller öğrenenlerin çeşitli işlevlere sahip yaratıcı robot tasarımları sağlanacaktır. Ayrıca, öğrenciler tasarım becerilerini kullanarak inşa ettikleri robotları programlayabilecek, böylelikle programlamanın temellerini de öğrenecekler ve öğrendiklerini somut projelere dönüştürebilecektir. Özetle bu dersin amaçları şu şekilde sıralanabilir:

- Bir problemin çözümü için gerekli olan algoritma ve programlama ile ilgili temel kavramları kullanmak
- Robotların belirli amaçlar için programlamalarını sağlayarak, bilgi işlemsel düşünme çözme becerilerini geliştirmek
- Etkinlikler esnasında gerçek hayattan problemlerin sunulduğu öğrenci merkezli bir öğrenme ortamı oluşturarak, öğrencileri çok yönlü ve disiplinler arası düşünmeye yönlendirmek
- Öğrencileri grup çalışmalarına yönlendirerek, onların sosyal becerilerini, iletişim ve iş birliği becerilerini ve liderlik ve sorumluluk alabilme becerilerini geliştirmek

Bu temel amaçlar doğrultusunda ders verilirken daha önceki Robotik ve Kodlama kitabında detaylarının anlatıldığı GUTÜD (Üçgül, Çetin, Yükseltürk ve Top, 2021) "gözle, uygula, tasarla, üret ve değerlendir" öğrenme döngüsü kullanılacaktır. Bu aşamalar özetle şöyle devam edecektir.



Resim 1.1 Öğrenme Döngüsü

Gözle: Öğrenme döngüsü bu bölümle başlar ve öğretmenin daha aktif olduğu bölümdür. Öğretmen öğrencilerin geçmiş bilgilerini aktive eder ve onların dikkat ve motivasyonlarını sağlamaya çalışır. Ayrıca, öğretmen ilgili konuları uygulamalı olarak (göstererek) anlatır. Öğrencilere sorular sorabilir ve öğrencilerin sorularını yanıtlayabilir.

Uygula: Bu bölümde öğretmen öğrenenlerden gözle esnasındaki gösterilen uygulamaların aynısını/ bir benzerini ister veya onlarla yapar. Anlatılan konulara göre gözle ve uygula bölümleri birkaç kez tekrar edilebilir.

Tasarla: Bu bölümde öğrenciler daha aktif rol üstlenir. Öğretmen rehber görevini üstlenir ve öğrencilere takıldıkları noktalarda destek olacaktır. Tasarla aşaması öğretmen tarafından öğrenenlere bir problem verilerek başlar. Öğrenenlerden öncelikle bu problemin çözümünü tasarlamaları istenir. Tasarlama aşamasında, öğrenenler temel itibarıyla bilinenler ile istenenler

arasındaki bağı kurarak bir plan üreteceklerdir. Bu amaçla, öğrenenler bilgi işlemsel düşünme becerisinin temel bileşenlerini kullanması sağlanacaktır.

Üret: Bu bölümde öğrenciler yine aktif rol üstlenmeye devam eder ve öğretmen rehber pozisyonundadır. Üret aşamasında, öğrenenlerden bir önceki adımda tasarladığı planı kullanarak probleme algoritmik bir çözüm üretmesi istenir. Öğrenenler bilgisayar ve robot başında çalışarak gerekli donanımsal ve yazılımsal çözümleri geliştirirler.

Değerlendir: Bu bölümde temel hedef, öğrenenin, öğrenme sürecinde yaşadıkları ve öğrendikleri üzerine düşünmesini sağlamaktır. Böylelikle, öğrenen problem çözme süreci, dersin konusu ve kendisi ile ilgili gözlemler yaparak yeni öğrenmeler, kendini değerlendirme ve planlama açısından fırsatlar elde edecektir.

EŞLİ PROGRAMLAMA ve GRUPLAR ARASI İLETİŞİM

Bu ders sürecinde etkinlikler yapılırken eşli programlama ve gruplar arası iletişime önem verilmektedir. Öğrenenler kendilerine verilen görevlerden uygun olanlarını eşli programlama yaparak tamamlayabilecektir. Eşli programlamada iki öğrenen, bir robot veya bilgisayar karşısında yan yana oturarak tasarım, algoritma, kod yazma veya hata ayıklama için işbirlikli çalışır. Eşli programlama eşli araba yarışlarına benzetilebilir. Eşli araba yarışlarında sürücü arabayı kullanırken kılavuz sürücüye yön tayini konusunda yardımcı olur. Eşli programlamada bilgisayarı veya robotu kullanan kişiye sürücü denir. Sürücünün görevi robotun istenenleri gerçekleştirmesi için tasarımı ve kodlamayı yapmaktır. Sürücünün yanındaki kişiye kılavuz denilir. Kılavuzun görevi çıkan problemler veya ana problem için çözüm üretmek, bu süreçte ortaya çıkan hataları belirlemek ve sürücünün nasıl çalıştığını değerlendirmektir. Eşli programlamada, araba yarışından farklı olarak, sürücü ve kılavuz düzenli olarak yer değiştirir. Öğrenenlerin her iki görevden de öğreneceği şeyler vardır. Bu yüzden rol değiştirme çok önemlidir. Öğretmen öğrenenlerin periyodik olarak görev değiştirmesini sağlamalıdır. Ayrıca, etkinlikler boyunca gruplar arası bilgi alışverişine izin verilmesi önerilmektedir. Gruplar arasında paylaşımcı bir ortam oluşturulmalıdır. Fakat bu paylaşım, komple bir çözümün paylaşımı şeklinde olmamalıdır. Öğrenenler; çözüm yolları, stratejiler ve eksik bilgiler gibi konular için paylaşım yapabilirler. Ayrıca, bu eğitim programı içerisinde zaman zaman gruplar arası yarışlar düzenlenmiştir. Fakat bu yarışlar, sınıfın paylaşımcı ortamını zedeleyecek içerikte uygulanmamalıdır.

ROBOTİK KİTLER VE PROGRAMLAMA ORTAMI

LEGO Education SPIKE Prime Seti

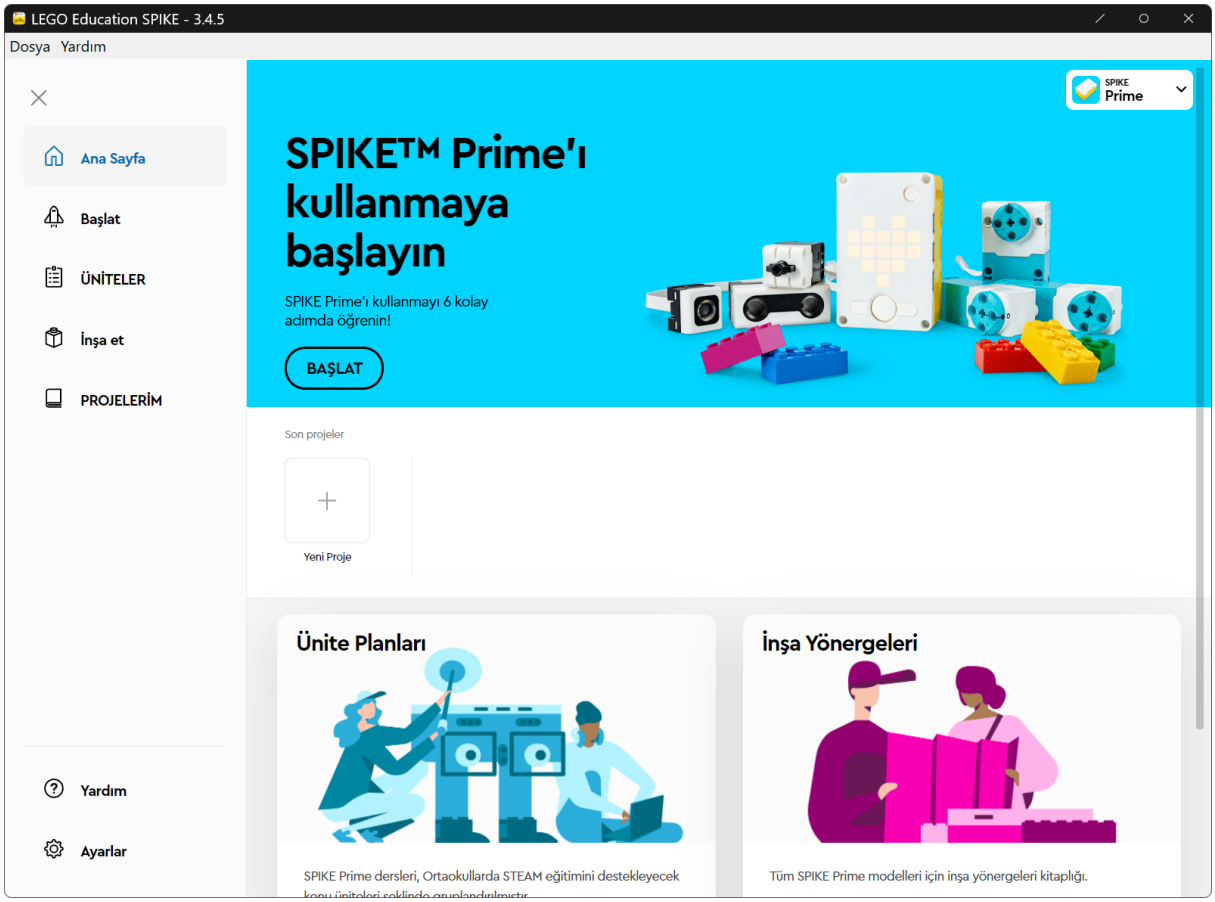
Bu eğitimde LEGO firmasının 2020 yılında piyasaya sürdüğü SPIKE Prime kullanılacaktır. 528 parçadan oluşan bu setin içinde programlanabilir Hub, Mesafe Sensörü, Kuvvet Sensörü, Renk Sensörü, Büyük Motor, iki Orta Motor dâhil birçok teknik parça mevcuttur. SPIKE Prime ve eklenti

seti ile inşa edebilen farklı robot tasarımları mümkündür. Aşağıdaki Resim 1.2'de SPIKE Prime robot seti görülmektedir.

Resim 1.2 LEGO® Education SPIKE™ Prime Seti

SPIKE Prime Yazılımı

SPIKE Prime yazılımı robotları programlamak için kullanılan ücretsiz olarak indirilen ve üç farklı programlama ortamını destekleyen bir araçtır. SPIKE Prime yazılımının programlama arayüzünün giriş bölümü aşağıdaki Resim 1.3'te görüldüğü gibidir. Ünite Planlarıyla farklı konularda (İcat takımı, Bir İş Kurma gibi) ders içerikleri sunulmaktadır. İnşa Yönergeleriyle de farklı robotların tasarımlarına örnekler verilmiştir. "Yeni Proje" düğmesi tıklanarak yeni bir proje oluşturulabilir. Bu aşamada kullanıcılara 3 farklı programlama ortamı seçeneği sunulmaktadır. Simge Blokları ve Sözcük Blokları seçenekleri blok tabanlı programlama ortamı ile desteklenmektedir. Sözcük Blokları ile ise yaygın olarak desteklenen sürükle bırak ortamlarında (Örn. Scracth) programlama yapılabilmektedir. Bu iki ortamda da öğrenciler blok hâlinde var olan veya kendilerinin oluşturdukları komutları sürükle bırak yöntemiyle taşıyarak programları oluştururlar. Komutlar bloklar hâlinde bulunduğundan dolayı, öğrencilerin söz dizimini öğrenmek için fazladan zaman ayırmaları gerekmez ve program yazarken hata yapma ihtimalleri ortadan kaldırılır. Bu sayede öğrenciler zihinsel kaynaklarını problem çözmeye yönlendirebilirler. Bu eğitim boyunca "Sözcük Blokları" programlama ortamı tercih edilecektir. Ayrıca, metin tabanlı programlama yaklaşımlarından Python programlama dili de SPIKE Prime tarafından üçüncü seçenek olarak desteklenmektedir. Metin tabanlı programlama yaklaşımı profesyonel programcılar tarafından yıllardır kullanılmaktadır. Bu yaklaşıma alıştıktan sonra karmaşık / ileri seviye programların oluşturulması daha kolay olabilmektedir. Diğer yandan bu ortamda öğrenciler yazacakları her bir komutu en ince detayına kadar tam olarak bilmek zorundadırlar.



Resim 1.3 SPIKE Prime Yazılımı

Eğitimde Kullanılacak Çalışma Alanları

Öğrenciler ders boyunca bazı etkinliklerde çalışma alanları/kâğıtları kullanacaklardır. Bu çalışma alanlarıyla ilgili bilgiler haftalık içeriklerde detaylı bir şekilde verilmektedir.

Kaynakça

Gürer, M. D., Çetin, I., & Top, E. (2019). Factors affecting students' attitudes toward computer programming. *Informatics in Education, 18*, 281–296.

P21 (2020). Framework for 21st Century Learning - Battelle for Kids, http://www.battelleforkids.org/networks/p21 adresinden 22.05.2022 tarihinde erişim sağlanmıştır.

Üçgül, M., Çetin, İ., Yükseltürk, E. & Top, E. (2021). *Robotik ve Kodlama Ortaokul*. TÜBİTAK Yayınları, Ankara.

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical transactions of the royal society of London A: mathematical, physical and engineering sciences, 366*(1881), 3717-3725.

Wing, J. (2011). Research notebook: Computational thinking—What and why. *The link magazine,* 6, 20-23.

Yükseltürk, E., & Altıok, S. (2017). An investigation of the effects of programming with Scratch on the preservice IT teachers' self-efficacy perceptions and attitudes towards computer programming. *British Journal of Educational Technology, 48*(3), 789-801.

1. Hafta: Robotlarla Algoritma ve Hareket

Haftanın Amacı:

Bu haftanın amacı eğitim süresince kullanılacak robot kavramının tüm öğrenciler tarafından anlaşılmasını sağlamak ve robot seti ile robot setinin programlanacağı SPIKE Prime yazılımının grafiksel arayüzünü tanıtmaktır. Ayrıca, robot seti kullanılarak mekanik robot tasarımları hazırlandıktan sonra programlamaya giriş yapılarak öğrencilerin algoritmanın ne olduğunu tanımlaması ve basit bir algoritma oluşturabilmesi de hedeflenmektedir. Öğrencilere robotlarının belirli bir mesafeyi alabilmesi için gerekli programlama adımlarını öğretmek ve motorların tur, derece ve santimetre cinsinden çalıştırılarak robotun hareket etmesini sağlamaktır.

Haftanın Kazanımları:

- Öğrenciler robot kavramını temel düzeyde açıklar.
- Öğrenciler algoritma kavramını açıklar.
- Öğrenciler basit bir algoritma örneği oluşturur.
- Öğrenciler robot setini tanır.
- Öğrenciler robot setiyle robotik tasarım yapar.
- Öğrenciler robotik setini programlamak için grafik arayüzünü (SPIKE Prime yazılımı) kullanır.
- Öğrenciler robotun hareket etmesi için gerekli programlama adımlarını oluşturur.

Kullanılacak Malzemeler:

Robot seti ve bilgisayar

Ekler:

Konu anlatımında ekran görüntüleri verilen örnek programlar, ayrıca dijital formatta rehber öğretmenlere sunulmuştur. Örnek program dosyalarının numaralandırılmasında, konu anlatımındaki resim numaraları temel alınmıştır. Bu hafta aşağıda sıralanmış programlar ekte rehber öğretmenlere sunulmuştur.

Program_1.5_Temel_Motor_Bloklari.llsp3
Program_1.6_Ileri_ve_Geri_Hareket_Eden_Robot_1.llsp3
Program_1.6_Ileri_ve_Geri_Hareket_Eden_Robot_2.llsp3
Program_1.7_Robotun_Hareketlerine_Ornek.llsp3
Program_1.8_Bir_Motorun_Kullanilmasi.llsp3
Program_1.9_Kare_Seklinde_Hareket_Eden_Robot.llsp3
Program_1.12_Ornek_Program.llsp3
Program_1.14_Lego_Parcasini_Alip_Geri_Getir.llsp3
Program_1.16_Ornek_Program.llsp3

GÖZLE VE UYGULA

Gözle: Robot Kavramı Üzerine Tartışma

Rehber öğretmen, öğrencilerin "robot", “algoritma” ve “programlama” gibi kavramlar hakkındaki önbilgilerini öğrenmek ve bu kavramların bütün öğrenciler tarafından tam olarak anlaşılabilmesi için aşağıdaki sorulardan bazılarını sorarak derse başlayabilir.

- Robot nedir?
- Robotların özellikleri nelerdir? Robotlar nerelerde kullanılır?
- Uzaktan kumandalı bir oyuncak ile robot arasında ne gibi farklılıklar vardır?
- Robotlar hangi parçalardan oluşur?
- Algoritma/ program nedir?

Rehber öğretmen; robot, algoritma ve program kavramları için aşağıdaki tanımları dikkate alarak öğrencileri yönlendirebilir veya gerektiği yerde (öğrenciler kavramı yanlış tanımladıklarında veya doğru olandan çok farklı bir tanım yaptıklarında) bu tanımları doğrudan kendisi yapabilir.

Rehber Öğretmen İçeriği – Robot, Algoritma ve Program

Robot: *Türk Dil Kurumunun hazırladığı Türkçe Sözlük’te “Belirli bir işi yerine getirmek için manyetizma ile kendisine çeşitli işler yaptırılabilen otomatik araç.” şeklinde tanımlanır. Oxford İngilizce Sözlük’te ise “Özellikle bir bilgisayar tarafından programlanan, karmaşık bir dizi işlemi otomatik olarak gerçekleştirebilen makine.” şeklinde tanımlanır. Her iki tanımda da vurgulanan nokta, robotların bir görevi otomatik olarak yerine getirmesidir. Eğer bir makinenin hareketleri bir insan tarafından kontrol ediliyorsa bu makineye robot denilemez. Ama makine çevresini algılayabiliyor ve buna göre hareket ediyorsa, örneğin bir engeli algılayıp yön değiştirebiliyorsa, bu makine robot olarak adlandırılır. Robotlar çevrelerini sensörleri aracılığıyla algılar. Robotlar kullanılan sensöre bağlı olarak; nesnenin uzaklığı, nesnenin rengi, ışık miktarı, ses şiddeti, nem oranı gibi birçok çevresel veriyi algılayıp işlemcileri ile yorumlayabilir ve programları dâhilinde bu verilere tepkide bulunabilirler. Bu durumu göz önüne alındığımızda robotu, “çevresini algılayabilen ve algıladığını yorumlayarak bağımsız tepkiler verebilen makine” şeklinde tanımlayabiliriz.*

Algoritma: *Algoritma, belirli bir problemi çözmek veya belirli bir amaca ulaşmak için çözüm yolunun adımlarının tasarlanmasıdır. Aslında algoritma sadece bilgisayar bilimlerinde değil hayatın birçok alanında kullanılır. Örneğin günlük yaptığınız rutin işlerde, yemek yapımında ya da basit bir matematik işlemi çözümünde kullanılabilir. Örneğin çay demleme sürecini düşünerek yapılan adımları sıralayabiliriz: çaydanlığa suyu koy, ocağı aç, suyun kaynamasını bekle, çayı koy, kaynayan suyu üzerine ekle, 15 dakika bekle, çayı servis et.*

Program: *Temel olarak bir algoritmanın bilgisayar veya robot için uygulanmasıdır. Programlar bilgisayara veya robota yapması gereken şeyleri (algoritmanın adımlarını) bir programlama dili vasıtası ile aktarırlar. Kişi ile bilgisayar arasındaki iletişimi sağlarlar ve geçmişten günümüze farklı programlama dilleri kullanılmaktadır.*

Uygula: Algoritma Öğreniyorum

Rehber öğretmen, sayı tahmini oynayacaklarını söyler ve 1 ile 100 arasında rastgele bir sayı tutar. Öğrencilerden bu sayıyı sırayla tahmin etmesini ister. Öğrenciler sayıdan küçük tahminde bulunursa "Yukarı", büyük tahminde bulunursa "Aşağı" ifadesini kullanır. Bu işlemi doğru tahmin edilinceye kadar devam ettirir. Doğru tahmini yaptığında "Tebrikler" deyip oyunu bitirir. Ayrıca her seferinde sayı isterken kaçıncı deneme olduğunu da tahtaya yazar. Bu oyunu oynanırken hangi algoritmayı takip ettiklerini tahtaya beraberce yazılır.

1. 1 ile 100 arasında rastgele sayı tut,
2. Tahmin et,
3. Eğer tahmin, tutulan sayıdan küçükse "yukarı", büyükse "aşağı" mesajını ver,
4. Eğer tahmin, doğru ise "tebrikler" mesajını ver,
5. Kaçıncı denemede bulunduğunu hesapla.

Başka bir algoritma etkinliğine geçilir. Sınıftaki öğrencilere üzerinde 1 ile 24 arasındaki sayıların yazılı olduğu kartlar rastgele dağıtılır. Bu sayılar veya kartlar artırılabilir. Daha sonra bir öğrenci seçilir. Sadece bu öğrenci duyacak şekilde aşağıdaki algoritma verilir:

1) Her bir öğrenciye git,
   a) Sayısını sor,
   b) Eğer öğrencinin kartındaki sayı 3'ün katkı ise öğrenciyi gruptan ayır ve tahtanın önüne getir.

Seçilen öğrenci diğer öğrencilere sadece kartlarındaki sayıyı sormalıdır ve ipucu oluşturacak cümleler kurmamalıdır. Bütün öğrencilerin kartlarındaki sayılar sorulmasının ardından rehber öğretmen, diğer öğrencilerden, tahtaya çıkarılan öğrencilerin sayılarına bakarak hangi işlemin yapılmış olabileceğini söylemelerini ister. Öğrenciler yapılan işlemi bildikten sonra onlardan bu işlemin algoritmasını yazmaları istenir.

Uygula: Çarpım Eşleri Algoritması

Öğrencilerden, çarpımları 20 olan çarpım eşlerinin (1-20, 2-10, 4-5) bulunması için bir algoritma yazmaları istenir. Öğrenciler buldukları algoritmaları sınıfça tartışabilirler.

Son olarak algoritmanın bir görevi gerçekleştirmesi için net bir şekilde tanımlanmış ve sıralanmış adımlardan oluşması gerektiği vurgulanır. Eğer görevleri açıkça tanımlanmadıysa veya adımların sıralaması uygun değilse robotun veya bilgisayarın istenilen görevi yerine getiremeyeceği belirtilir.

Gözle: Robot Yapımı

Robot kavramıyla ilgili temel tartışmalar yapılıp algoritma uygulaması tamamlandıktan sonra rehber öğretmen öğrencilere LEGO Education SPIKE Prime setini tanıtır. Robotu yaparken dikkat edilmesi gereken noktaları belirtir. Parçaların sağlam olduğundan fakat düşme, çarpma gibi durumlar sonucu kırılıp yıpranabileceklerinden bahseder. Parçaların kaybolmaması için robot seti ile gelen veya var olan tasnif kutusunun kullanılması önerir.

Rehber Öğretmen İçeriği –Robot Seti İçeriğinin Tanıtılması

Hub: *SPIKE Prime sisteminin beyni, programlanabilir Hub'dır. Bu gelişmiş kullanımı kolay tuğla biçimli aygıtın 6 giriş/çıkış portu, 5x5 ışık matrisi, Bluetooth bağlantısı, hoparlörü, 6 eksenli hareket sensörü ve şarj edilebilir pili bulunur.*

Motorlar: *Robot seti içerisinde hareketi sağlayan bir büyük ve iki orta motor bulunmaktadır. Büyük boy açılı motor, entegre dönme sensörü ve gerçek düz çizgi kontrolü için mutlak hizalama özelliğine sahiptir ve yüksek güçlü, yüksek torklu uygulamalar için idealdir. Orta boy açılı motor ise, düşük profilli tasarım, mutlak hizalama ve 1 derecelik doğruluk özelliğine sahip entegre dönme sensörlüdür.*

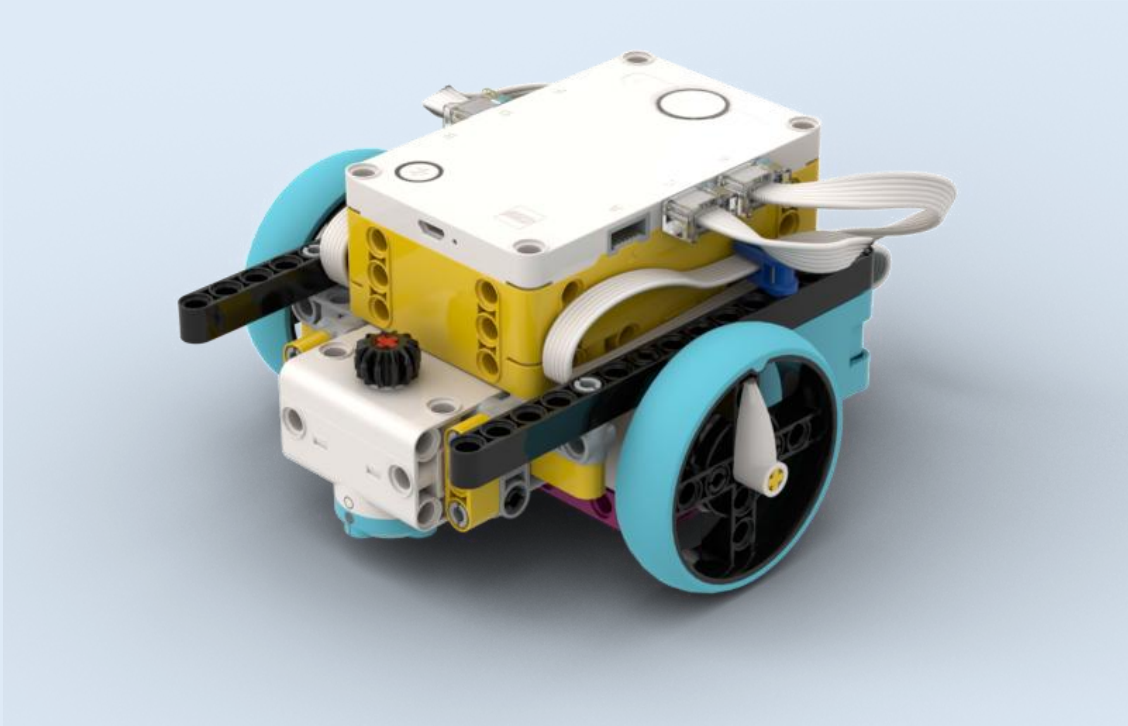
Sensörler: *Etkinliklerde kullanılan robotlarda farklı sensörler bulunmaktadır. 1-200 cm menzilli, +/- 1 cm doğruluk ve programlanabilir LED gözlere sahip* ***mesafe sensörü****, 8 rengi ayırt edebilir ve karanlıktan parlak gün ışığına kadar yansıyan ışığı ve ortam ışığını ölçen* ***renk sensörü*** *ve doğru, tekrarlanabilir sonuçlar için 10 Newton'a (~1 kg) kadar basınçları ölçen ön düğmeye basıldığında, bırakıldığında ya da çarpıldığında kullanılabilen* ***dokunma sensörü*** *bulunmaktadır.*

Lego Teknik Parçaları: *Robotun farklı şekillerde birleştirilebilmesine olanak sağlayan dişliler, tekerlekler, akslar, kablolar ve diğer lego teknik parçalarını içermektedir.*

Öğrencilerden "Sürüş Modeli 1" yönergesine uyarak ilk robotlarını oluşturmaları istenir. Öğrenciler grup çalışması sırasında görev paylaşımı yapmaları konusunda cesaretlendirilir. Daha sonra öğrencilere robot setleriyle temel tasarım robotunu yapmaları için yeteri kadar süre verilir.

Not

Sürüş Modeli 1 yönergesine, SPIKE Prime yazılımındaki 'İnşa et' seçeneğinden erişilebilir. Bu robot modeli toplamda 34 adımda inşa edilebilir.



Rehber öğretmen, robotlar oluşturulduktan sonra, daha önce hazırladığı birkaç programı robot üzerinde çalıştırarak gösterir. Bu aşamada amaç, öğrencilere kod yazmayı öğretmek veya yüklenen kodun robot üzerinde nasıl çalıştırılacağını göstermek değildir. Amaç, robota basit bir görev yaptırarak öğrencilerin robota karşı olan ilgilerini artırmaktır.

LEGO Education SPIKE Prime Yazılımı

Rehber Öğretmen İçeriği – SPIKE Prime Yazılımı

Robot setlerinin programlanması için gerekli yazılımın bilgisayarlarda kurulu olup olmadığı kontrol edilir. Eğer uygulama kurulu değilse LEGO Education SPIKE Prime Yazılımı bilgisayarlara kurulur.

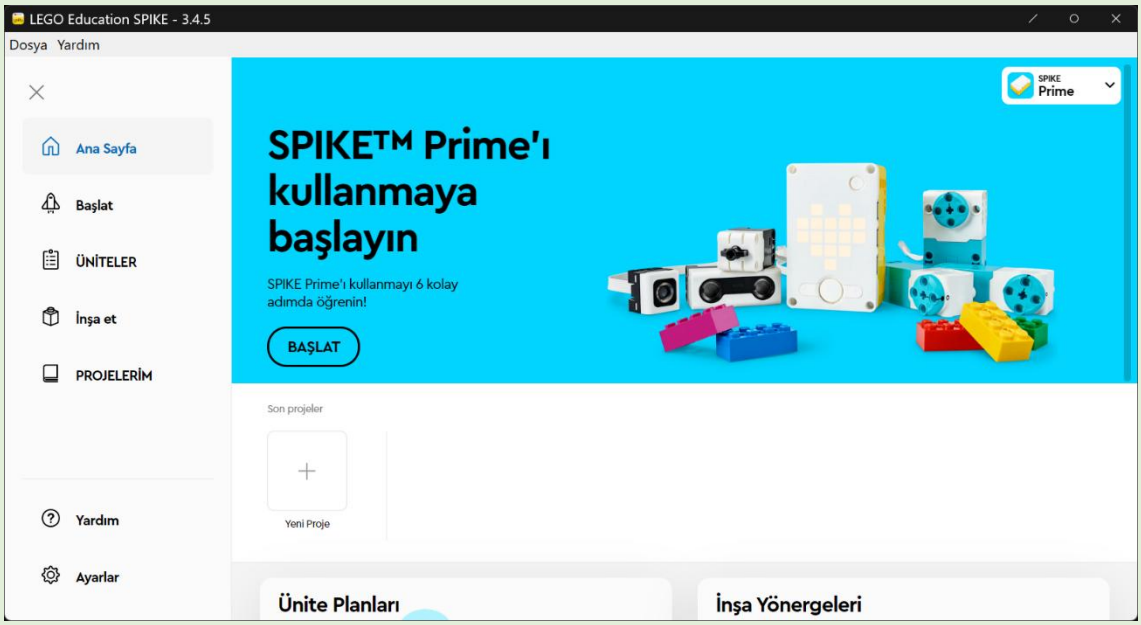
Not

LEGO Education SPIKE Prime Yazılımına
https://education.lego.com/en-us/downloads/spike-app/software
adresinden uygun işletim sistemi seçilerek ulaşılabilir.

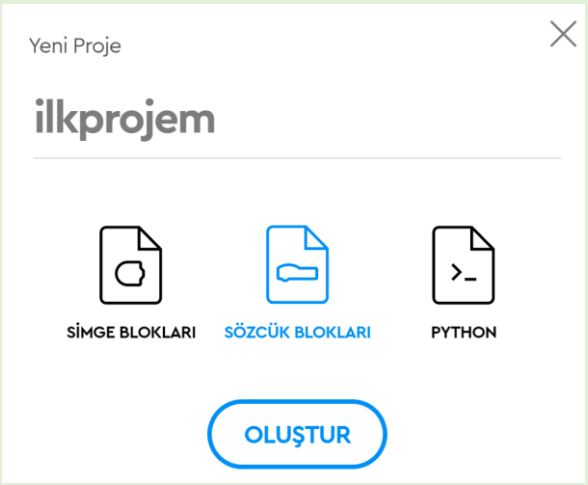
Robot setinin programlanacağı yazılımın kurulumu gerçekleştirildikten sonra indirilen uygulama çalıştırılıp aşağıdaki ekran görüntüsüne ulaşılır. Bu aşamada elimizdeki sete göre seçim yapılır (SPIKE Prime seçilir).

Gelen arayüzde neler olduğu gösterilir. Başlat, Üniteler ve İnşa Et bağlantıları üzerinden Öğretici etkinlikler, Ünite planları ve İnşa yönergeleri hızlıca gösterilir. Ayrıca her bir projemizin saklandığı söylenir ve PROJELERİM bağlantısı ile var olan projelere ulaşılabileceği anlatılır. İlerleyen haftalarda bu başlıklardan faydalanacağı hatırlatılır. Daha sonra yardım menüsünden programlama ekranındaki blokların özelliklerinin anlatıldığı gösterilir. Ayrıca, ayarlar

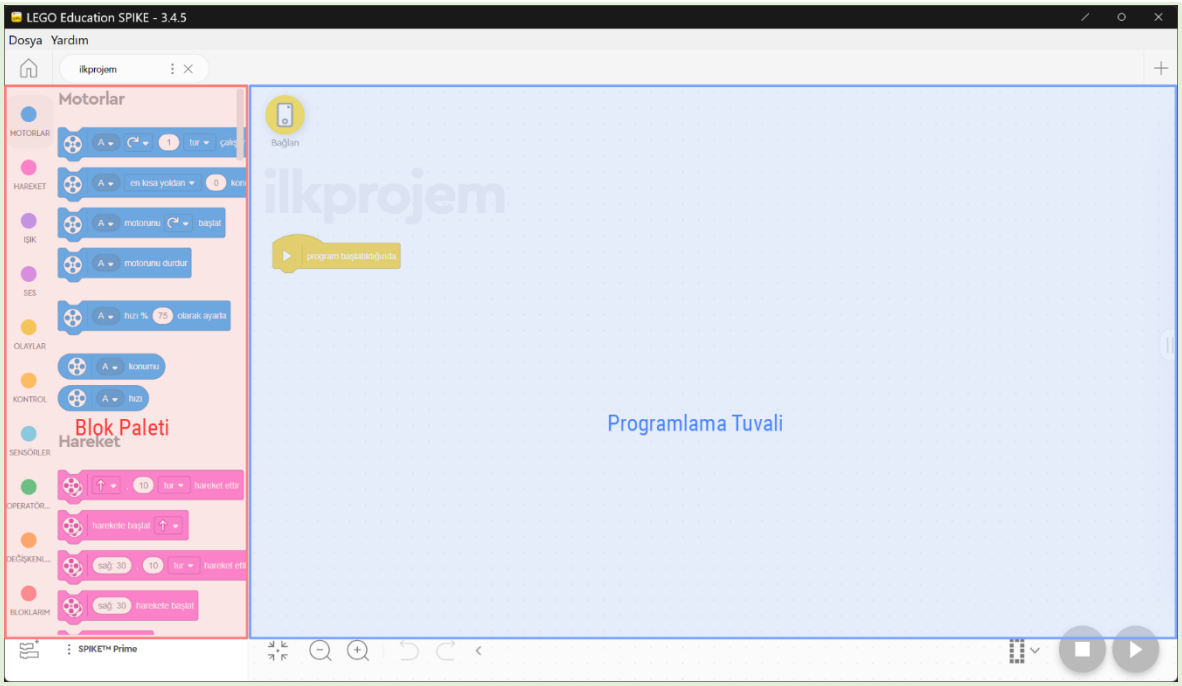
menüsünden de arayüzün özelliklerinin değiştirileceği vurgulanır. Örneğin dil seçeneğiyle Türkçe seçilir.



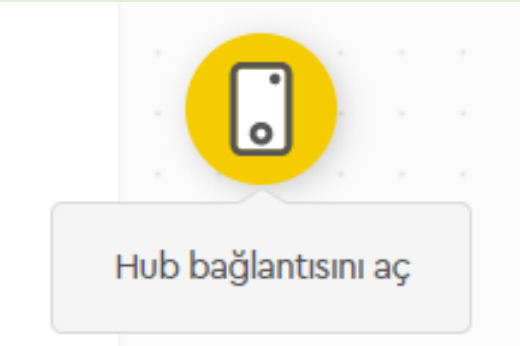
Arayüzden "Yeni Proje" seçilerek ilk uygulamaya başlanır. Gelen ekrandan oluşturulacak projenin adı verilir. Örneğin projenin adı "ilkprojem" olarak giriş yapılır ve Sözcük Blokları seçilerek programlama arayüzüne geçilir.



Yeni proje grafiksel arayüzü (Blok Tabanlı Programlama) açılır ve arayüzde neler olduğu gösterilir. Grafik arayüzü ile Hub arasındaki bağlantının USB veya Bluetooth ile gerçekleştirilebileceği vurgulanır. Önce kablo ile sonra da Bluetooth ile bağlantıların nasıl yapılacağı anlatılır. Daha sonra yazılımının grafiksel arayüzünün bölümleri tanıtılır.



Programlama arayüzü temel olarak iki kısımdan oluşur. Robotun programlanabilmesi için program bloklarının Blok Paletinden Programlama Tuvalinin (Alanının) içerisine sürüklenip bırakılması gerekir. Programlama alanında, varsayılan olarak yukarıdaki resimde görüldüğü gibi "program başlatıldığında" bloğu bulunur. Programlama alanında yaklaştırılan bloklar birbirine yapışır. Böylece sürükle-bırak yaklaşımı ile istenilen programın oluşturulması sağlanır. Programlama blokları, yukarıdaki resimde görüldüğü üzere, Blok Paletinde farklı kategoriler altında gruplanmıştır. Bu kategoriler Motorlar, Hareket, Işık, Ses, Olaylar, Kontrol, Sensörler, Operatörler, Değişkenler ve Bloklarım'dır.



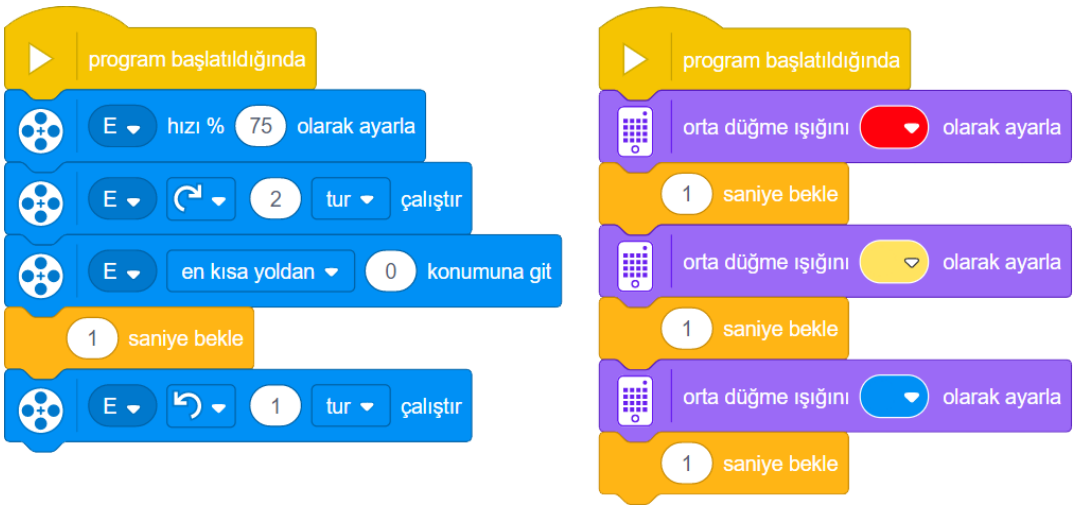
Arayüzdeki sol üst bölümde bulunan "Hub Bağlantısını Aç" özelliği ile Hub'a bağlı olan motorlar ve sensörler görülebilir, ayrıca daha önce yazdığımız programlar yönetilebilir. Programlama tuvalinin sağ altındaki oynat düğmesi (sarı düğme) ile hazırlanan programları Hub'a göndererek çalıştırabilir. "Depolama Konumu Seçin" özelliği ile programların hangi alana yüklenebileceğine karar verilebilir. Hub, sıfırdan başlayarak 19'a kadar toplamda 20 adet programı hafızasında tutabilir.

Gözle: Programlama Alanının Kullanılması

Robot, istenilen program bloğunun programlama alanına sürüklenip bırakılması ve blokların parametrelerinin değiştirilmesi ile programlanır. Program, blokların programlama tuvalinde yer alan "*program başlatıldığında*" bloğunun altına birbiri ardına eklenmesiyle oluşturulur. Program çalıştırıldığında, robot, "*program başlatıldığında*" bloğundan başlayarak program bloklarında belirlenen görevleri sırasıyla gerçekleştirir. Aşağıdaki resimde blokların örnek bir yerleştirilmesi görülmektedir.

Resim 1.1 Blokların Yerleştirilmesi

Robotun birden fazla işlemi aynı anda gerçekleştirmesi isteniyorsa, programlama alanında birden fazla şapka bloğu kullanılarak, iki veya daha fazla programın eş zamanlı çalışması sağlanabilir. Paralel işlemlere bir örnek Resim 2.2'de sunulmuştur. E motoru çalışırken ederken Hub ışığının rengi de değiştirilmektedir. Paralel işlemlerde farklı yığıtlarda, birbiri ile çelişen görevler verilmemelidir.

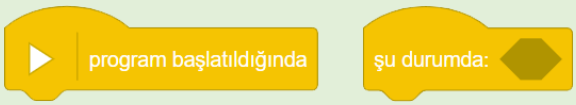


Resim 1.2 Paralel İşlemler

Rehber Öğretmen İçeriği – Blok Türleri

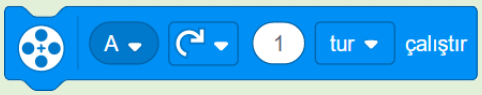
Şapka Blokları

Şapka blokları, her diziyi başlatan bloklardır. Yuvarlak bir tepe ve altında bir çıkıntı şeklindedirler. Bu nedenle sadece altına blok eklenebilir. Örnek şapka blokları:



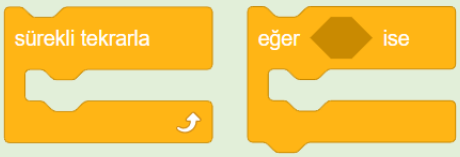
Yığıt Blokları

Yığıt blokları, ana komutları gerçekleştiren bloklardır. Üstlerinde bir çentik ve altlarında bir tümsek vardır. Bunlar motorları hareket ettiren ve ışıkları yakan bloklardır. Örnek bir yığıt bloğu:



C Blokları

Bu bloklar C şekilli bloklardır ve tamamı kontrol kategorisinde bulunur. C'lerin içindeki blokları bir döngüye sokar veya bir koşulun "doğru" olup olmadığını kontrol eder. Örnek C blokları:



Haberci Bloklar

Bu bloklar, bir rakam veya dizi olabilecek değerleri içerir. Bir sensör değerini de içerebilir veya bir değişken değerini de taşıyabilir. Örnek haberci blokları aşağıda verilmiştir.

Bloole Blokları

Bu bloklar koşul belirtirler, ya doğru ya da yanlıştırlar. Altıgen şekle sahiptirler. Örnek bloole blokları aşağıda verilmiştir.

Durdurma Blokları

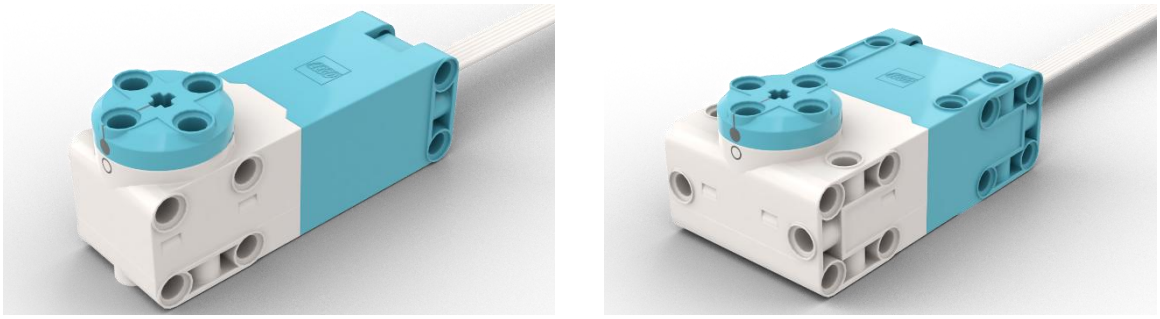
Bu bloklar komut dizilerini sonlandırır. Üstleri girintili ve altları düzdür. Bu sebeple altlarına herhangi bir blok yerleştirilemez. Örnek bir durdurma bloğu:

Programlama Yığıtı

Bir araya getirilmiş bir blok dizisidir.

Gözle: Robotun Hareket Ettirilmesi

SPIKE Prime robot setinde robotu hareket ettiren iki çeşit motor bulunur. Bunlar, Resim 1.3'te görüldüğü gibi orta (medium) ve büyük (large) motor olarak adlandırılır. Set içerisinde bir büyük motor ve iki orta motor mevcuttur. Büyük motor, entegre dönme sensörü ve gerçek düz çizgi kontrolü için mutlak hizalama özelliğine sahiptir; bu nedenle yüksek güçlü ve yüksek torklu uygulamalar için idealdir. Orta motorlar ise hızlı tepki veren robot tasarımlarında tercih edilir. Ayrıca, mutlak hizalama ve 1 derecelik doğruluk özelliğine sahip entegre dönme sensörüne sahiptir.



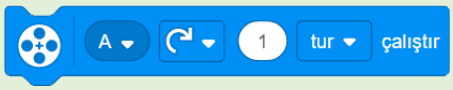
Orta Motor Büyük Motor

Resim 1.3 Büyük ve Orta Motor

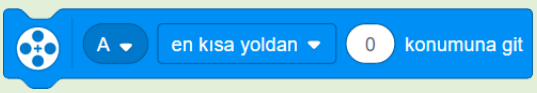
Rehber Öğretmen İçeriği – Motorlar ve Hareket Blokları

Robotu hareket ettirecek program blokları Motorlar ve Hareket blokları sekmesinde yer alır. Motor blokları ya motorların çalışmasını sağlar ya da motorlardan bilgi alır. Hareket blokları ise, iki motoru senkronize çalışmasını sağlar. Bu hafta sık kullanacağımız bazı bloklarla ilgili bilgiler aşağıda verilmiştir, bloklarla ilgili daha fazla bilgiye yardım menüsünden ulaşabilirsiniz.

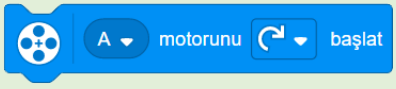
Motoru Belirli Süre Çalıştır *bloğu, seçilen bir veya birden fazla motoru saat yönünde ya da saat yönünün tersine, belirli bir tur, saniye veya derece çalıştırır.*



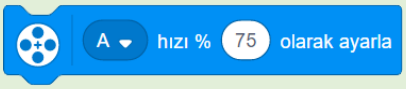
Motor Konumuna Git *bloğu, bir veya daha fazla motoru belirli bir konuma göre ayarlar. Motor, saat yönünde ya da saat yönünün tersine çalışacak şekilde veya belirtilen konuma en kısa yoldan gidecek şekilde ayarlanabilir. Konum aralığı 0-359 derece arasındadır.*



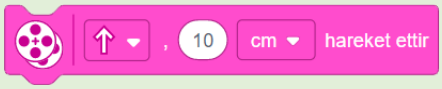
Motoru Başlat *bloğu ile bir veya daha fazla motoru sürekli olarak saat yönünde ya da saat yönünün tersine çalıştırır.*



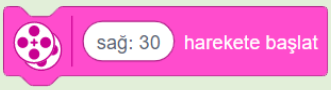
Motor Hızını Ayarla *bloğu ile bir veya daha fazla motorun hızını -100 ile 100 arasında ayarlanabilir. Eksi değerler, motoru ters yönde çalıştırır. Varsayılan hız %75'tir.*



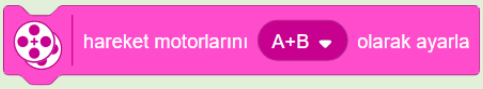
Belirli Süreyle Hareket Ettir *bloğu, sürüş modelini belirli bir santimetre, inç, saniye, derece veya tur cinsinden saat yönünde veya saat yönünün tersine döndürür.*



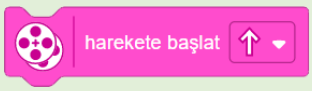
Direksiyonla Harekete Başla *bloğu, sürüş modelini yönlendirme olanağıyla ileri hareket ettirmeye başlar.*



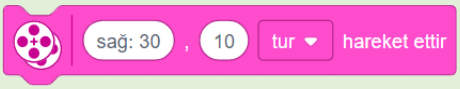
Hareket Motorlarını Ayarla *bloğu ile iki sürüş motorunun bağlanacağı portları tanımlar. Portlar tanımlanmazsa robot hareket etmez.*



Hareket Başla *bloğu, sürüş modelini ileri veya geri hareket ettirmeye başlar.*

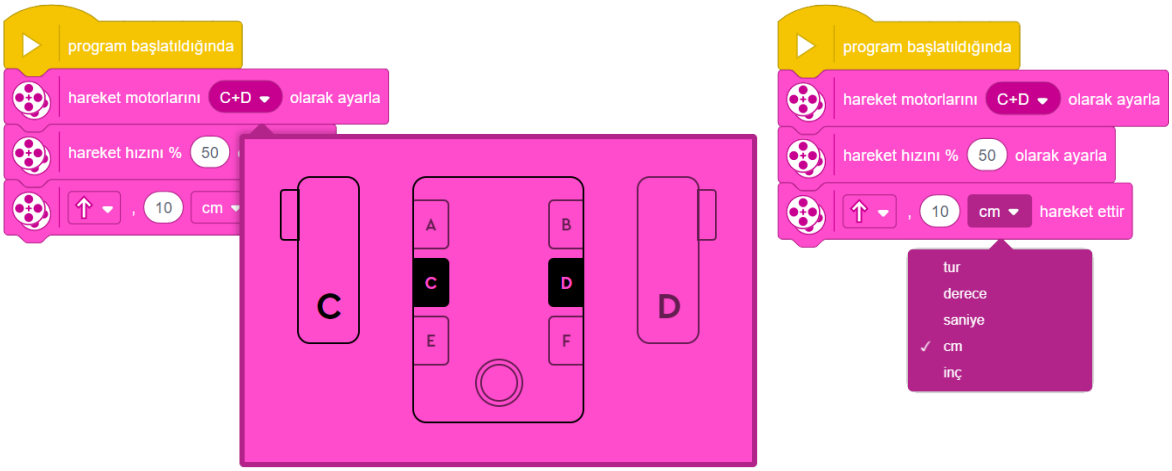


Belirli Bir Süre Direksiyonla Hareket Ettir *bloğu, sürüş modelini belirli bir tur, derece, saniye ileri hareket ettirir. Düz bir çizgide gitmesi için "0", kendi çevresinde dönmesi için "100" ve "-100" arasındaki değerler kullanılır.*



Gözle: Robotun Temel Hareketleri

Robotun iki motorunun aynı anda hareket ettirilmesi ve motorun pozitif veya negatif yönde dönmesini, robotun ileri veya geri gitmesini sağlar. İlgili bloğun özelliğiyle (örneğin Resim 1.4'te sol bölümde C+D tıklandığında) hangi portlara bağlı olan motorların kontrol edileceği görülebilir. Eğer motorlar farklı portlara bağlanmışsa bu kısımdan portlar değiştirilebilir.



Resim 1.4 Temel Hareket Bloğu ve Ayarlanması

Hareket bloklarından motorların seçimi yapıldıktan sonra hızları ayarlanabilir. Resim 1.4'ün sağındaki bölümde olduğu gibi motorlar %50 hızla gidecek şekilde ve ileri hareket etmesi için cm, inç, tur, derece ve saniye gibi özellikleri ayarlanabilir.

Resim 1.5 Temel Motor Blokları ve Ayarlanması

Resim 1.5'teki kod bloğu ile robot 10 cm ileri gider. Ardından, yalnızca C motoru bir tur çalıştığı için D tekerleği merkezinde robot kendi etrafında 180 derece döner. Motor seçimi, yönü ve tur, derece ile saniye gibi özelliklerin değiştirilmesi, robotun dönüş yönü ve miktarını belirler.

Not

Özellikle motorların dönerken seçilen yönleri aynı olsa bile, motorların montajına göre robotun hareketine farklı etkileri olabileceği vurgulanmalıdır. Örneğin, aşağıdaki program çalıştırılarak robotun hareketi gözlemlenmeli ve bu durum öğrencilerle tartışılarak oluşabilecek yanlış algıların önlenmesi sağlanmalıdır.

Gözle ve Uygula: İleri ve Geri Hareket Eden Robot

Direksiyon hareketi bloğu anlatıldıktan sonra robotun bir miktar düz gidip kendi etrafında dönmeden geri geldiği bir program çalıştırılıp gösterilir.

Resim 1.6 İleri ve Geri Hareket Eden Robot (iki alternatif yöntemle)

Öğrencilerden benzer bir uygulamayı bu seferde kendi etrafında dönerek tur, saniye ve derece seçeneklerini kullanarak yapmaları istenir.

Resim 1.7 Robotun Hareketlerine Başka Bir Örnek

Ayrıca öğrencilerden sadece bir motoru kullanarak dönüş yapıp geri gelen programı da yazmaları beklenir.

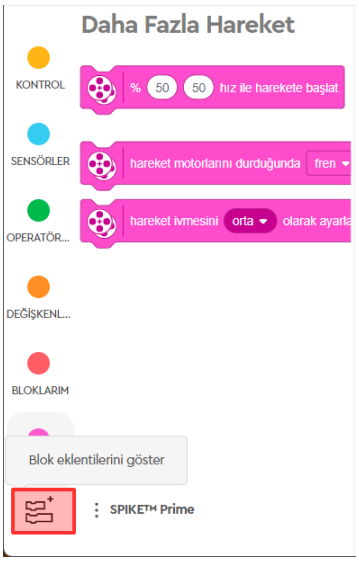
Resim 1.8 Bir Motorun Kullanılması

Uygula: Kare Çizen Robot

Motor bloklarını ve hareket bloklarını kullanarak robotun dönüşünde meydana gelen değişiklikleri gözlemleyen öğrencilerden, kare şeklinde bir rotada hareket eden robotun programını oluşturmaları istenir. Bu etkinlik sırasında, öğrencilerin robotun bulunduğu yerde veya tek tekerlekli hareketle dönebilmesi için gerekli düzenlemeleri keşfetmeleri sağlanır.

Resim 1.9 Kare Şeklinde Rotada Hareket Eden Robot

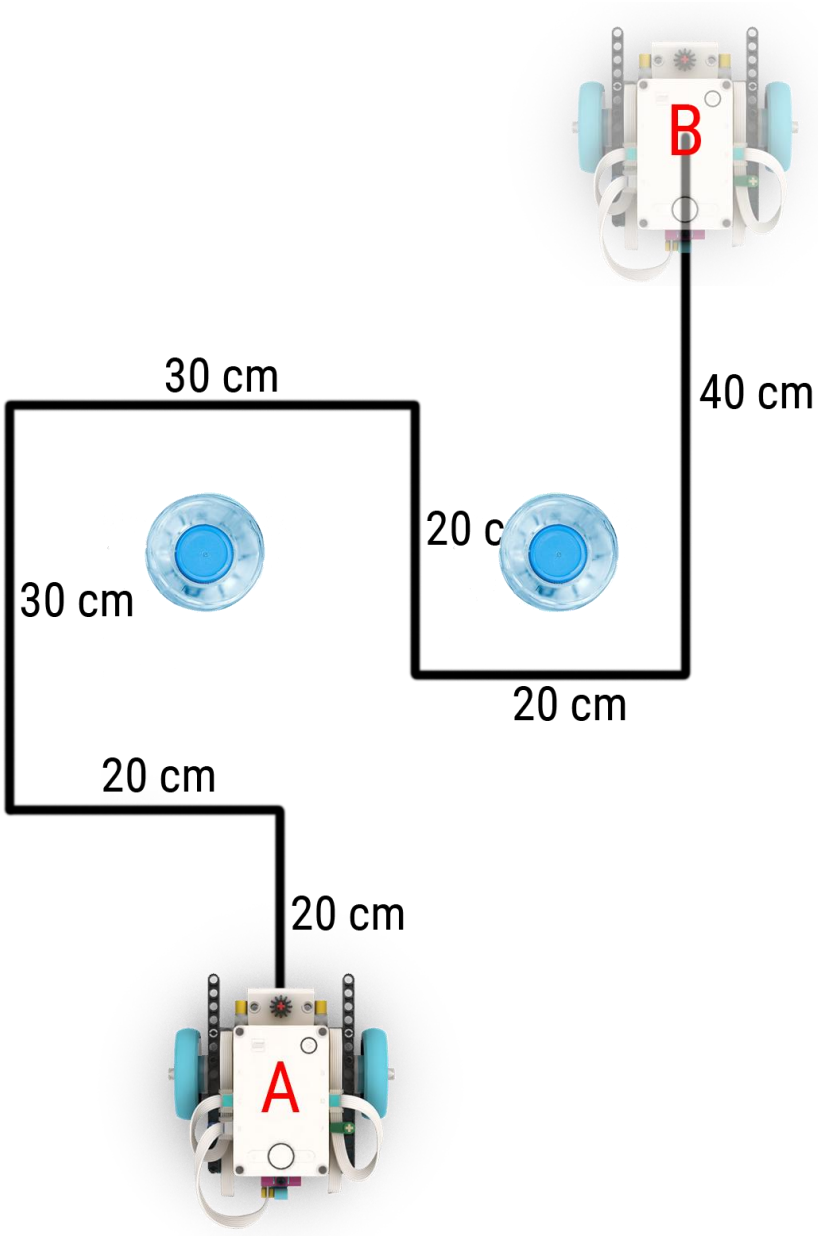
Kare şeklindeki bir rotada hareket eden robotun programlanması, ilerleyen haftalarda döngü konusu anlatıldıktan sonra daha az kod bloğu kullanılarak gerçekleştirilebilecektir. Ayrıca, bu aşamada 'Daha Fazla Motor' ve 'Daha Fazla Hareket' eklentilerinin de kullanılabileceği belirtilmelidir. İlgili eklentileri seçmek için sol alt köşede bulunan 'Blok eklentilerini göster' düğmesine tıklanması gerekmektedir. Bu eklentiler, temel bloklara ek olarak bazı ekstra özellikler sunarak programlama sürecini zenginleştirecektir.



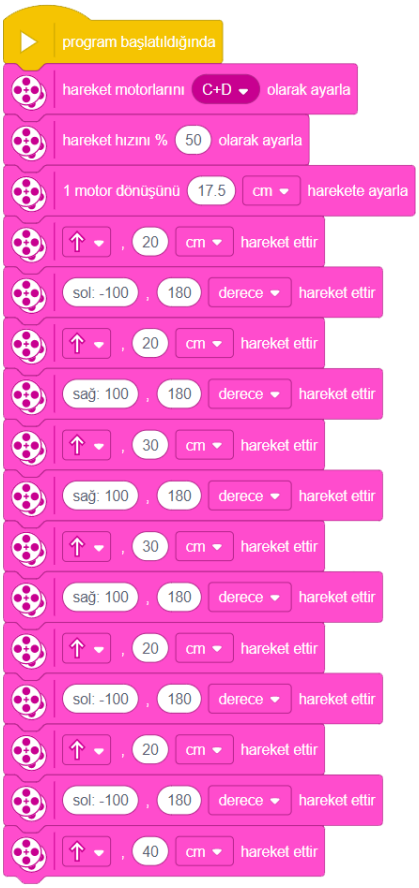
Resim 1.10 Daha Fazla Motor ve Daha Fazla Hareket Eklentileri

Uygula: A noktasından B noktasına Hareket

Bu etkinlikte, robotun Resim .11'de görüldüğü gibi A noktasından B noktasına gitmesi için gerekli programın oluşturulması istenir. Robotun rotası üzerinde boş pet şişeler bulunduğu görülmektedir. Robotun şişelerin çevresinden dolaşılması gerekmektedir. Ayrıca, robotun rotası üzerinde gideceği uzaklıklar belirtilmiştir.



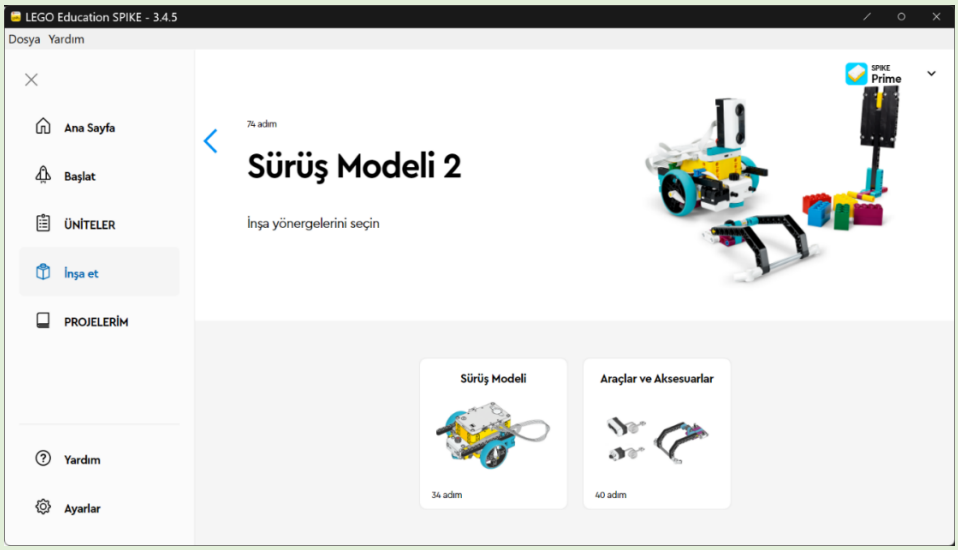
Resim 1.11 Robotun Rotası



Resim 1.12 Örnek Program

Rehber Öğretmen İçeriği – Farklı İnşa Yönergeleri

SPIKE Prime yazılımının 'İnşa Et' veya 'İnşa Yönergeleri' bölümlerinden farklı robot modellerinin nasıl tasarlanacağı incelenebilir. Örneğin, bu aşamada Sürüş Modeli 2 adımlarını takip ederek öğrenciler, istedikleri aksesuarları robotlarına ekleyebilirler. Sensörler ilerleyen haftalarda kullanılacağından, bu hafta yalnızca aksesuar eklemeleri yeterli olacaktır.



Öğrenciler, belirtilen adımları izleyerek büyük motor ile hareket eden kolu monte edebilirler.

TASARLA

Gün içerisinde robotları hareket ettirmeyi öğrenen öğrencilerden, bu bilgileri kullanarak robotlarının yeni bir görevi tamamlamaları istenir. Örneğin, robotlarının belirli bir mesafe düz ilerleyip durmasını, sonra robotlarına takılı olan kolu indirip LEGO parçaları ile yapılmış bir kutuyu (Resim 1.13) tutarak kendi etrafında 180 derece dönmesini ve başlangıç noktasına getirmesini sağlayan bir program tasarlamaları istenir. Tasarlama aşamasında öğrencilerden aşağıda örnek olarak verilen "Tanımlama" ve "Fikir Üretme" süreçlerini sırayla gerçekleştirmeleri sağlanır.

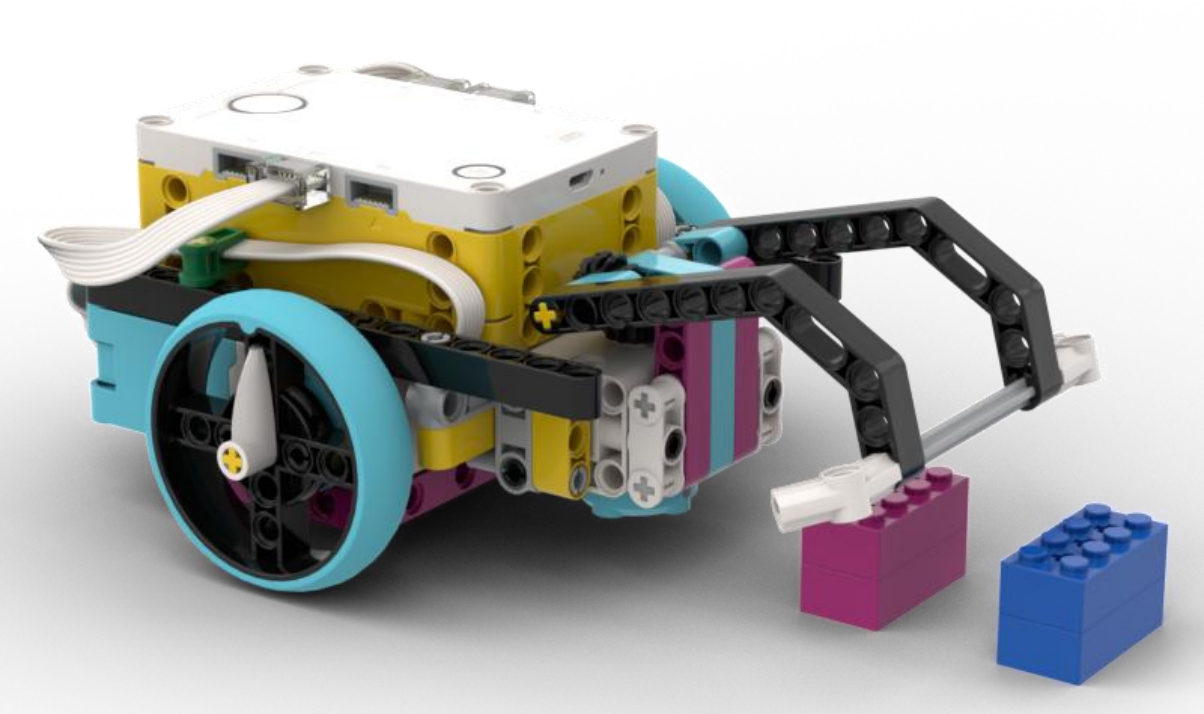
Tanımlama: Bu aşamada öğrencilerden kâğıt ve kalem kullanarak programın algoritmasını veya akış diyagramını hazırlamaları istenir. Öncelikle robotun istenilen hareketleri yapabilmesi için neler gerektiğini belirlemeleri ve maddeler hâlinde yazmaları beklenir. Zorlanan öğrencilere rehber öğretmen aşağıdaki adımları yapabileceğini söyleyerek yardımcı olabilir.

Örneğin robot;

- 20 cm ileri gidecek ve duracak,
- Kolunu kutunun üzerine indirecek,
- Kendi etrafında kutuyla birlikte dönecek,
- Başlangıç noktasına geri dönecek.

Fikir üretme: Bu aşamada öğrencilerin tanımlamada belirlenen işlemlerin nasıl yapılabileceği ile ilgili fikir yürütmesi beklenir. Örnek olarak öğrenciler aşağıdaki maddelere benzer fikirler üretebilir:

- Öncelikle robotun hangi hareketleri yapacağı planlanmalıdır. Planlanan her bir hareketin tanımı yapılmalıdır (Bu aşamada öğrenciler hareketlerin programlanmasını hızlıca deneyebilirler):
    - İlk harekette 20 cm ileri gidip durur
    - Etrafında 180 derece döner.
- Robot kolunun indirilmesi için büyük motor kullanılır.
- Kolun kutuyu kavraması sağlanır.
- Robotun kendi etrafında 180 derece dönmesi sağlanır.
- Robotun başlangıç noktasına geri gelmesi sağlanır.



Resim 1.13 Lego Parçasını Alıp Geri Getiren Robot

ÜRET

Bu bölümde öğrenciler aktif rol üstlenerek verilen problemi çözerek, robotu tasarlayıp programlamayı tamamlarlar. Rehber öğretmen öğrencilere yalnızca zorlandıkları noktalarda destek olur. Öğrenciler bilgisayarda ve robot üzerinde çalışarak gerekli yazılım çözümlerini geliştirirler. Örnek bir çözüm Resim 1.14'te verilmiştir.

Resim 1.14 Lego Parçasını Alıp Geri Getiren Örnek Program

DEĞERLENDİR

Günün sonunda öğrencilerle aşağıdaki sorular üzerinden bir tartışma yürütülür:

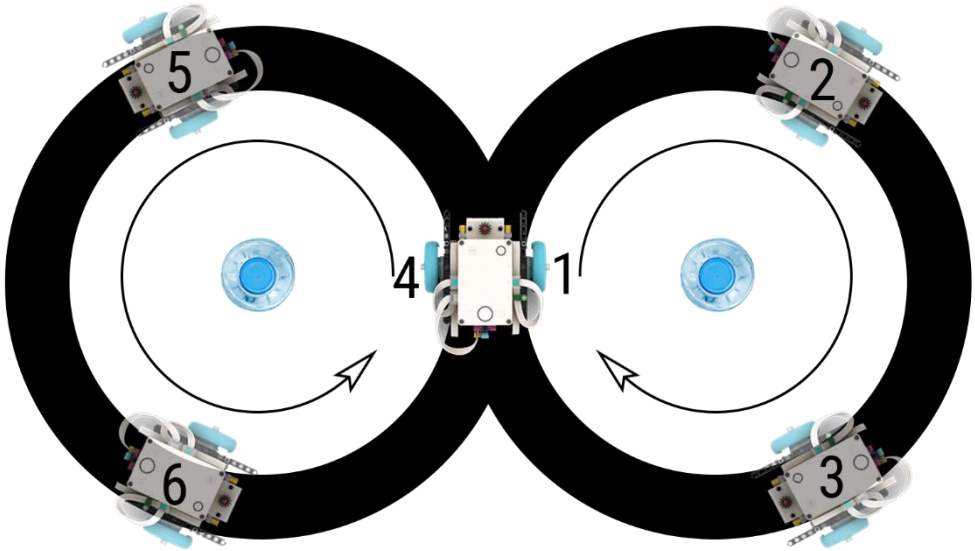
- Robotun setinin içinden ne gibi parçalar çıktı?
- Motor ve hareket bloklarının kullanarak robotların hareket ettirilmesinde ne gibi farklılıklar oluşuyor?
- Daha Fazla Blok Eklentilerini neden kullanma ihtiyacı hissettiniz?
- Sizce robotun hareket ettirilmesi için teker büyüklüğünün bir önemi var mıdır?
- Programınızda tur sayısını değiştirmeden sadece teker büyüklüğünün değiştirilmesi durumunda ne gibi sonuçlara yol açar?

Bu soruların cevaplarına göre farklı robotların da tasarlanabileceğine dair örnekler verilebilir.

İLAVE ETKİNLİK

Sonsuz İşareti Şeklinde Hareket Etme

Bu ilave etkinlik zaman kalması durumunda yapılabilir. Amaç, motor ve hareket bloklarını kullanarak robotların hareket etmesini farklı bir örnek üzerinde yapılmasının sağlanmasıdır. Bu ilave etkinlik için iki nesne (örneğin iki boş su şişesi olabilir) sınıf içinde müsait bir alana, aralarında 40-50 cm olacak şekilde yerleştirilir. Robotun bu nesnelere dokunmadan etraflarında "8" veya "sonsuz sembolü" çizerek hareket etmesi sağlanır. Robotun izleyeceği yol Resim 1.15'deki gibi olabilir.



Resim 1.15 Örnek Rota

Etkinliği zenginleştirmek için iki nesne arasındaki mesafe artırılabilir ya da azaltılabilir. Nesne sayısı da artırılabilir veya farklı nesneler yerleştirilebilir. Ayrıca robotun başlangıç noktasına göre algoritma ve tasarım değişiklik gösterebilir. Programı oluşturmaya başlamadan önce grupların tasarlama adımı için yukarıda bir örneği verilen tanımlama ve fikir üretme süreçlerini gerçekleştirmeleri gerekir (Rehber öğretmen kendisine verilen aşağıdaki örnek çözümü inceleyebilir ama çözümü öğrencilerle paylaşmamalıdır, görevi öğrencilerin kendilerinin yapmasına izin vermelidir).

Resim 1.16 Örnek Program

2. Hafta – Işık, Ses ve Mesafe Sensörü

Haftanın Amacı:

Bu hafta Hub ekranının özelliklerinin tanıtılması, ekranın ışık ve parlaklığının çeşitli desenlerle programlanması, Hub ve kullanılan cihazlar ile çalınacak seslerin programlanması, mesafe sensörünün tanıtılması ve çeşitli problemlerin çözümünde mesafe sensörünün kullanılması için gerekli programların oluşturulması hedeflenmiştir.

Haftanın Kazanımları:

- Hub ekranının özelliklerini ifade edebilir.
- Hub ekranının istenilen şekilde görünmesi için gerekli blokları program akışında kullanabilir.
- İstenilen seslerin oluşturulması için gerekli blokları program akışında kullanabilir.
- Mesafe sensörünün çalışma mantığını ve kullanma yöntemlerini ifade eder.
- Mesafe sensörünü farklı amaçlar için programlama adımlarını oluşturabilir.
- Robotun farklı mesafelerdeki nesnelere göre davranması (hareket etmesi, ses çıkarması, hızını ayarlaması, Hub ekranında farklı simgeler göstermesi) için gerekli programlama adımlarını oluşturabilir.

Kullanılacak Malzemeler:

Robot seti ve bilgisayar.

Ekler:

Bu hafta aşağıda sıralanan programlar ekte rehber öğretmenlere sunulmuştur.

Program_2.3_Sonsuz_Dongu_ve_Donguyu_Tekrarla.llsp3
Program_2.4_Ilerlerken_Ses_Cikaran.llsp3
Program_2.5_Kare_Cizen_Robot.llsp3
Program_2.9_Belirli_Bir_Mesafe_Kadar_Ilerleme.llsp3
Program_2.10_Istenilen_Mesafe_Kadar_Geri_Gitme.llsp3
Program_2.11_Engele_Yaklastikca_Yavaslayan_Robot.llsp3
Program_2.12_Yaklasan_Nesne_Olursa_Ses_Cikar.llsp3
Program_2.13_Park_Sensoru.llsp3
Program_2.14_Ondeki_Araci_Takip_Et.llsp3

GÖZLE VE UYGULA

IŞIK

Rehber Öğretmen İçeriği – Işık Bloğu

Hub ekranı, 5 satır ve 5 sütundan oluşan toplam 25 pikselden meydana gelir. Her bir piksel ayrı ayrı programlanabilir ve piksellerin parlaklığı 10 farklı seviyede ayarlanabilir. Orta düğme ışığı ise 11 farklı renkte yanabilir. Ayrıca, program aracılığıyla girilen metinler, akan metin şeklinde Hub ekranında görüntülenebilir.

Hub'ın ekranını programlamak için blok paletindeki "Işık Blokları" kategorisi kullanılır. "Işık Blokları"nda piksellerin açılması/kapatılması, piksellerin belirlenen sürede açılması, ekrana metin yazılması, piksellerin parlaklığının yüzde olarak ayarlanması, belirli bir pikselin parlaklığının ayarlanması, Hub ekran yönünün değiştirilmesi ve mesafe sensörünün ışıklarının düzenlemesi işlemleri için bloklar bulunur. Bu hafta etkinliklerde kullanılacak yeni bloklar ve her bir bloğun açıklaması aşağıda verilmiştir.

Işık Blok Paleti

Farklı parçaların ışıklarını açma, kapatma ve yoğunluğu belirleme işlemlerine olanak sağlar.

Işık Matrisini Belirli Saniyede Aç

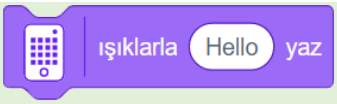
Hub ekranındaki piksellerin her birinin ışık miktarını ve ekranda görüneceği süreyi belirlemek için kullanılır. Belirtilen süre dolduğunda pikselleri kapatır.

Işık Matrisini Aç

Hub ekranındaki piksellerin her birinin ışık miktarını belirlemek için kullanılır. Başka bir işlem yapılmadığı sürece oluşturulan desen ekranda görünür.

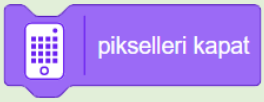
Işıklarla Yaz

İçerisine yazılan metni, ekranda bir seferde yalnızca bir harf akacak şekilde ışık matrisinde görüntülemek için kullanılır.



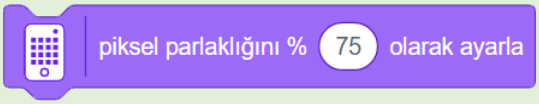
Pikselleri Kapat

Işık matrisindeki tüm ışıkları kapatmak için kullanılır.



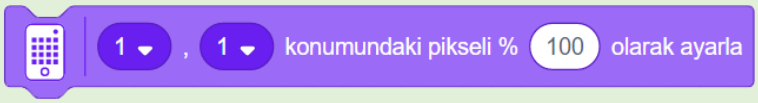
Matris Parlaklığını Ayarla

Işık matrisini kullanacak sonraki bloklar için ışık matrisinin parlaklığını ayarlamak için kullanılır. Parlaklık belirtilmediyse varsayılan değer %100'dür.



Piksel Parlaklığını Ayarla

Belirtilen pikselin parlaklığını ayarlamak için kullanılır, diğer pikselleri etkilemez.

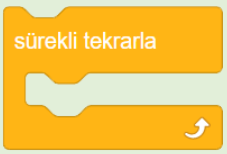


Kontrol Blokları Paleti

Bekle yapıları, döngüler ve koşullar gibi blok yürütmenin doğal akışını değiştirebilecek blokları içerir.

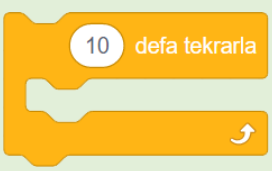
Sonsuz Döngü

İçinde yer alan blokları sırayla sürekli çalıştırmak için kullanılır. Robotun orta düğmesi veya "Tümünü Durdur" bloğu *kullanılarak durdurulur.*



Döngüyü Tekrarla

İçinde yer alan blokları sırayla ve belirlenen sayıda çalıştırmak için kullanılır. Döngüden sonra blok varsa çalıştırmaya devam eder yoksa program durur.

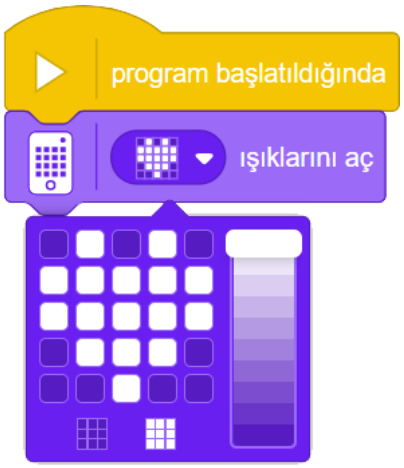


Gözle: Hub Ekranı

Rehber öğretmen, Hub ekranının 5 satır ve 5 sütundan oluşan toplam 25 pikselden meydana geldiğini ve her bir pikselin ayrı ayrı programlanabildiğini ifade eder. Her pikselin yalnızca beyaz renkte yanabildiğini, parlaklığının 10 farklı seviyede ayarlanabildiğini ve orta düğme ışığının 11 farklı renkte yanabileceğini belirtir. Ayrıca, program aracılığıyla girilen metinlerin akan metin şeklinde Hub ekranında görüntülenebildiğini de aktarır.

"*Işık Blok*" paketinde piksellerin açılması/kapatılması, piksellerin belirlenen sürede açılması, Hub ekranına metin yazılması, piksellerin parlaklığının yüzde olarak ayarlanması, belirli bir pikselin parlaklığının ayarlanması, Hub ekran yönünün değiştirilmesi ve mesafe sensörünün ışıklarının düzenlenmesi işlemleri için blokların bulunduğunu anlatır.

Rehber öğretmen, öğrencilere öncelikle Hub ekranında kalp görüntüsü oluşturacaklarını söyler. Bunun için "*Işık Blok*" paletinden "pikselleri aç" bloğunun programlama tuvalindeki "program başlatıldığında" bloğunun altına sürüklenmesi ve piksellerin ışığının tek tek ayarlanması gerektiğini gösterir (Resim 2.1). Açık olan piksellerin en yüksek parlaklığa getirilmesini ister.



Resim 2.1 Işık Matrisini Aç Bloğu

Uygula: Büyüyen Kalp Efekti

Rehber öğretmen öğrencilerden, bütün piksellerin ışığının kapalı olarak başlayıp yavaş yavaş büyüyerek bütün ekranı kaplayan bir kalbe dönüşen program akışını yapmalarını ister. En ortadaki karakterden başlayarak ve ekran tasarımları arasında belirli bir süre ekleyerek büyüyen kalp efekti yapılabilir. Örnek çözüm Resim 2.2'deki gibi olabilir.

Resim 2.2 Büyüyen Kalp Efekti Program Blokları

Gözle: Sürekli Tekrar Eden Büyüyen Kalp Efekti

Rehber öğretmen, büyüyen kalp efekti programındaki kodun bir defa çalıştığında bittiğini söyler. Efekti birden fazla tekrar etmek istediğimizde veya bu efektin sürekli oynatılması gerektiğinde hazırlanan kodların kopyalanıp art arda eklenmesi yerine "*Kontrol Blokları*" paketinden *"sonsuz döngü"* ve "*döngüyü tekrarla"* bloklarının kullanılabileceğini aktarır. "*döngüyü tekrarla"* bloğunda istenilen sayı kadar içerdeki blokların çalıştırılacağı ve varsa "*döngüyü tekrarla"* bloğundan sonraki bloğun çalışacağını, başka blok yoksa bu bölümün biteceği vurgulanır. *"sonsuz döngü"* bloğunun ise içerisindeki blokları program açık olduğu sürece tekrarlayacağı belirtilir. Büyüyen kalp efekti programındaki kodların *"sonsuz döngü"* ve "*döngüyü tekrarla"* blokları ile çalıştırılıp, öğrencilerin de kendi kodlarını bu iki blokla çalıştırmalarını ister.

Resim 2.3 Sonsuz Döngü ve Döngüyü Tekrarla Program Kodları

SES

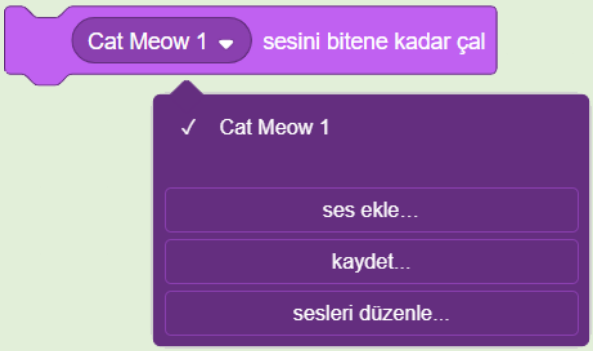
Rehber Öğretmen İçeriği – Ses Bloğu

Ses Blokları Paketi

Kullanılan cihazı ve/veya Hub'ı kullanarak sesler çalmaya olanak sağlar.

Sesi Bitene Kadar Çal

Seçilen sesi çalar ve bitene kadar sonraki bloğa geçmez. Blok ile açılan pencerede 200'den fazla ses arasından seçim yapılabilir, yeni kayıt yapılabilir veya seçilen sesler düzenlenebilir.

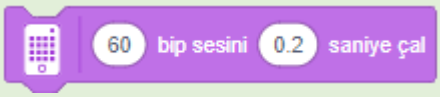


Sesi Başlat

Seçilen sesi çalar ve hemen sonraki bloğa geçer. Blok ile açılan pencerede 200'den fazla ses arasından seçim yapılabilir, yeni kayıt yapılabilir ve seçilen sesler düzenlenebilir.

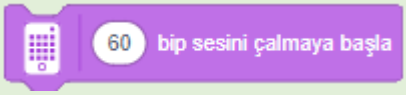
Bip Sesini Belirli bir Saniye Çal

Seçilen bip tonunu belirlenen süre için çalar.



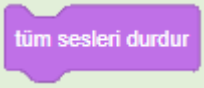
Bip Sesini Çalmaya Başla

Seçilen bip tonunu program bitene veya durdurulana kadar çalar.



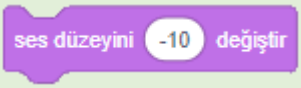
Tüm Sesleri Durdur

Çalınmakta olan (bip ve ses dosyaları gibi) tüm sesleri durdurur.



Ses Seviyesini Değiştir

Çalınmakta olan sesin seviyesini çalındığı ses yüksekliğine göre azaltmak veya artırmak için kullanılır. Varsayılan ses seviyesi %100'dür.



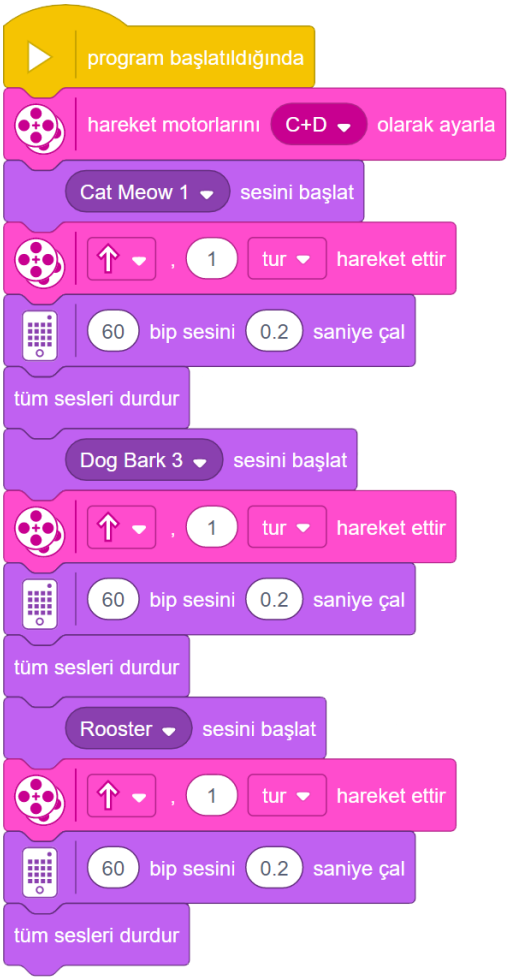
Gözle: Ses Blokları

Rehber öğretmen, 'Ses Blokları'nın öğrencilerin cihazlarından (bilgisayar) ve Hub'larından sesler çalmak için kullanıldığını belirtir. 'Ses Blokları'ndaki 'sesi bitene kadar çal', 'sesi başlat', 'bip sesini çalmaya başla', 'tüm sesleri durdur', 'ses perdesi efektini değiştir', 'ses perdesi efektini ayarla', 'ses efektini temizle', 'ses seviyesini değiştir', 'ses seviyesini ayarla' ve 'ses seviyesi' blokları gösterilir. Projelerine sadece eğlence amacıyla değil, aynı zamanda hata ayıklama amacıyla da ses blokları eklenebileceğinden bahseder. Örneğin, bir kod parçasının ne zaman tamamlandığını belirtmek için kullanılabilirler.

Not

Hub'da yalnızca 'bip' sesi blokları oynatılır; diğer sesler cihazınızda (dizüstü bilgisayar, tablet vb.) oynatılır. Bu nedenle etkinlik esnasında cihazın sesinin açık olduğundan emin olunuz. Aksi halde programda yer alan seslerden yalnızca 'bip' sesleri duyulabilir.

Robotun 3 tur düz ilerlemesi, ilerlerken önce cihazdan kedi, köpek ve horoz sesi çıkarması ve her turdan sonra da Hub'dan bip sesi çıkarması için gerekli kod bloğunun oluşturulması istenir. Resim 2.4'deki gibi bir program akışı oluşturulabilir.



Resim 2.4 İlerlerken Ses Çıkaran Robot Program Kodları

Uygula: Kare Çize Robot Aktivitesi

Öğrencilerden, robotun 20 cm'lik bir kare üzerinde ilerlemesi için bir program oluşturmaları istenir. Programı oluştururken 'döngüyü tekrarla' bloğunu kullanmaları, robot düz ilerlerken Hub'ın ilerleme yönünde ok işareti göstermesi, robot dönerken bip sesi çıkarması ve döndüğü yönde Hub ekranında ok işareti göstermesi sağlanmalıdır. Rehber öğretmen, uygulamanın sonunda başladığı yere en yakın şekilde karesini bitiren robotun oyunu kazanacağı bir aktivite yapacaklarını belirtir.

Bütün öğrenciler programlarını tamamladıktan sonra aktiviteye başlanır. Aktiviteye başlamadan önce robotun yerdeki teker izleri başlangıç çizgisi olarak çizilir. Daha sonra, sırayla bütün grupların robotlarının kareyi tamamlama aktivitesini yapmaları istenir. Yarışmayı kazanan robotun belirlenmesi için her grubun kare işlemini tamamladıktan sonra başlangıç çizgisine olan mesafeleri ölçülür ve not alınır (Örnek çözümde verilen 'belirli süreyle hareket ettir' bloğundaki mesafe değerinin, robotun hareket ettiği zemindeki materyal, pil durumu ve çözümde kullanılan bloklar gibi çeşitli nedenlerden dolayı farklılık gösterebileceği öğrencilere hatırlatılır).

Resim 2.5 Kare Çizen Robot Program Kodları

MESAFE SENSÖRÜ

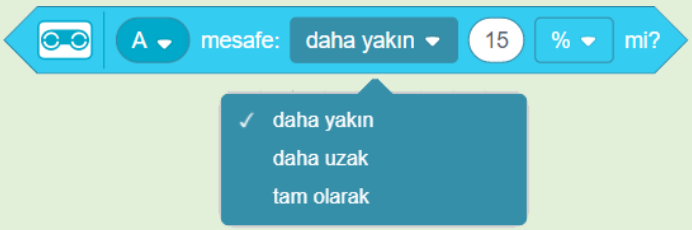
Rehber Öğretmen İçeriği – Mesafe Sensörü

Mesafe sensörü, ileriye doğru yüksek frekanslı ses dalgaları gönderir; bu ses dalgaları insanlar tarafından duyulamaz. Gönderilen ses dalgaları, karşıdaki nesneye çarparak geri yansır ve sensör tarafından algılanır. Ses dalgasının gönderilmesi ile geri dönüşü arasındaki süre, mesafe sensörünün nesneye olan uzaklığını hesaplamak için kullanılır. Algılama aralığı yaklaşık 200 cm'dir; bu mesafeden daha uzaktaki cisimleri algılayamaz. Mesafe sensörü, uzaklık ölçümlerini yüzde, santimetre veya inç cinsinden gerçekleştirebilir. Ayrıca, sensörün üzerinde dört adet programlanabilir ışıklı bölüm bulunmaktadır.

Bu hafta etkinliklerde kullanılacak mesafe sensörü ile ilgili sözcük blokları ve açıklamaları aşağıda verilmiştir.

Sensörler Blok Paleti

Mesafe Şu mu?



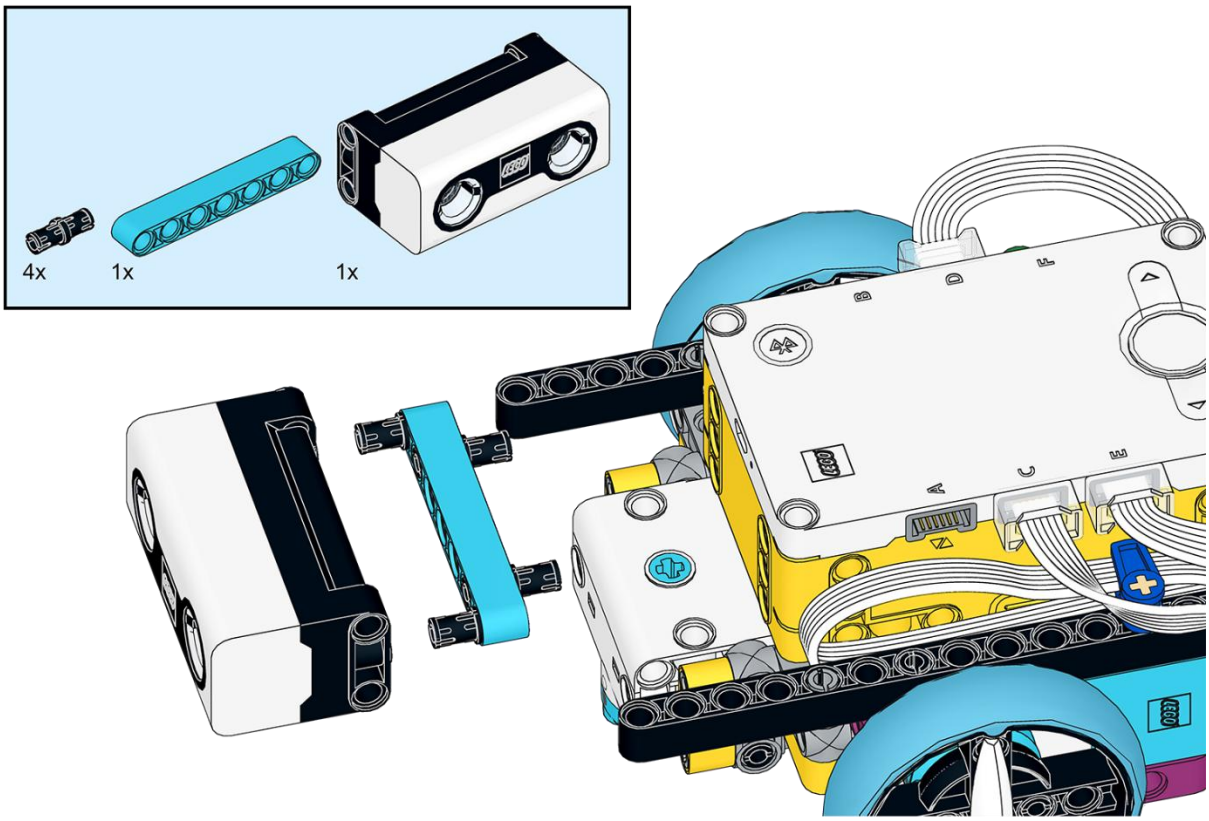
Bu blok, mesafe sensörünün santimetre, inç veya yüzde olarak belirtilen mesafeden yakın (<), uzak (>) veya eşit (=) olması durumunda "doğru (true)" sonucunu verir.

Mesafe

Bu blok, mesafe sensörünün algıladığı mevcut mesafeyi santimetre, inç veya yüzde olarak bildirir. Sensörün aralığı 0-200 cm'dir.

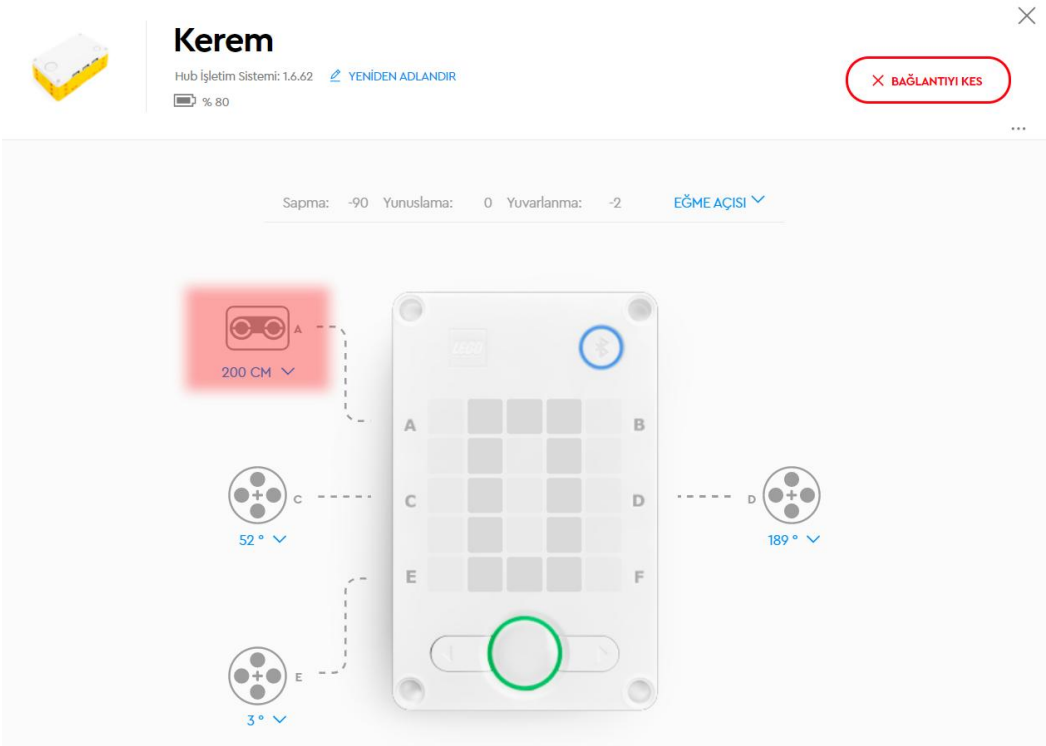
Gözle: Mesafe Sensörü

Rehber öğretmen, mesafe sensörünün özelliklerini detaylı bir şekilde anlatır; bu sensörün nasıl çalıştığını, algılama aralığını ve uygulama alanlarını örneklerle açıklar. Öğrencilere mesafe sensörünün Resim 2.6'daki gibi sürüş modeli robotuna nasıl entegre edileceğini gösterir.



Resim 2.6 Mesafe Sensörü Montajı

Rehber öğretmen, öğrencilere SPIKE yazılımını açmalarını ve yeni bir proje oluşturmalarını ister. Ardından, robotlarıyla Bluetooth veya kablo ile bağlantı kurmalarını ister. Bağlantılar tamamlandıktan sonra, A portuna mesafe sensörünün takıldığını kontrol etmek için aşağıdaki 'Hub Bağlantısı Aç' seçeneğini kullanmalarını söyler.



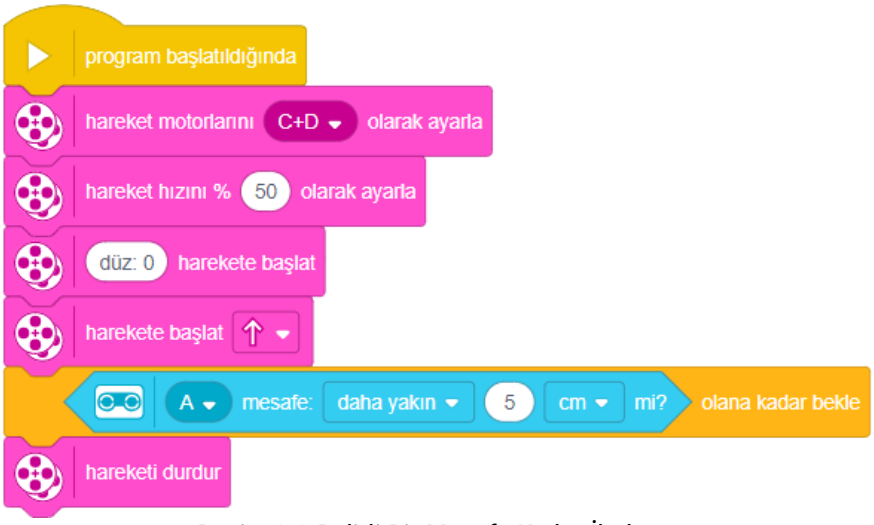
Resim 2.7 A Portunda Mesafe Sensörü

Mesafe sensörü üzerindeki ışıkların yakılması için, Resim 2.8'de gösterilen programın oluşturulması gerektiği açıklanır. Bu süreçte öğrencilere "ışıklarını yak" bloğunun temel işlevleri kısaca anlatılır. Öğrencilerin, öğrendiklerini pekiştirmek amacıyla farklı denemeler yapmaları için cesaretlendirilir.

Resim 2.8 Mesafe Sensörünün Işıklarının Yakılması

Gözle: Belirli Bir Mesafeye Kadar İlerleme

Rehber öğretmen, robotun A numaralı porta bağlı mesafe sensöründen gelen uzaklık değeri 5 cm'den küçük olana kadar ilerlemesini ve bu mesafe 5 cm'den az olduğunda durmasını sağlayan Resim 2.9'daki programı hazırlar.



Resim 2.9 Belirli Bir Mesafe Kadar İlerleme

Bu adımda, programın nasıl yazıldığı ekrana yansıtılarak öğrencilere gösterilir ve çalışma mantığı açıklanır. Rehber öğretmen, öğrencilerden yukarıdaki blokları SPIKE programlama alanına sürükleyip bırakmalarını ve robotu çalıştırmalarını ister. Kod çalıştıktan sonra, nesne ile mesafe sensörü arasındaki mesafe ölçülür ve öğrencilerden buldukları değeri (ilk örnek için 5 cm) bir kâğıda yazmaları istenir. Ardından, aynı işlemler 10 cm, 15 cm ve 20 cm için tekrarlanır. Öğrencilere elde ettikleri verilerin anlamı sorulur ve robotun verilen değerde tam olarak duramayacağı, hata payı bulunduğu açıklanır.

Uygula: İstenilen Mesafe Kadar Geri Gitme

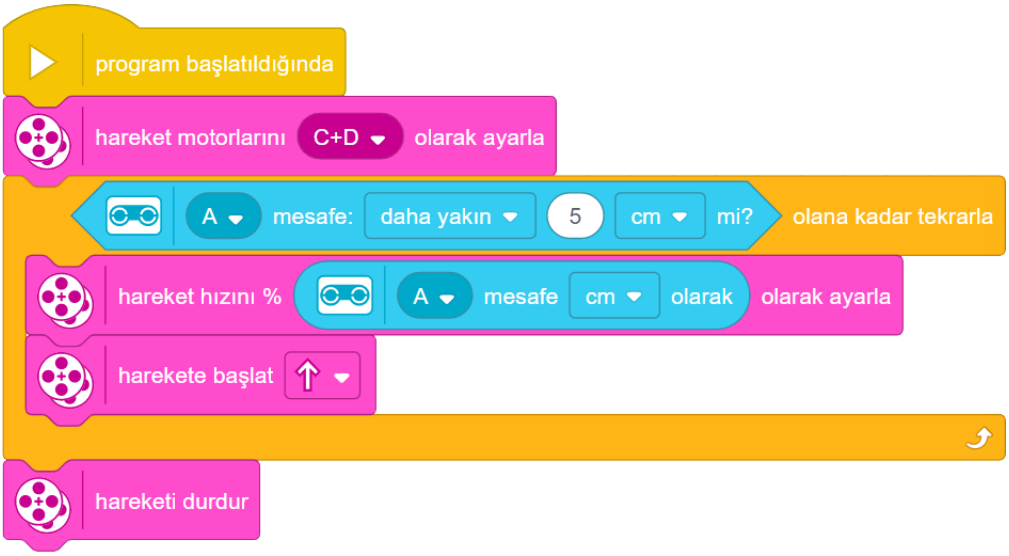
Bu uygulamada, öğrencilerden mesafe sensörünü kullanarak robotlarının, önündeki engelden 20 cm geriye gittikten sonra durmasını sağlayacak bir program oluşturmaları istenir. Öğrenciler,

Resim 2.10'da gösterilen programı oluşturmaları için yönlendirilir. Programlarını hazırladıktan sonra çalıştırmaları ve sonuçları gözlemlemeleri istenir. Robot çalıştırıldıktan sonra, öğrencilerden aldıkları mesafeyi ölçmeleri ve bu mesafeyi 20 cm'ye olabildiğince yaklaştırmak için programda değişiklik yapmaları talep edilebilir.

Resim 2.10 İstenilen Mesafe Kadar Geri Gitme Program Kodları

Uygula: Engele Yaklaştıkça Yavaşlayan Robot

Bu etkinlikte, karşısındaki engele yaklaştıkça yavaşlayan ve belirli bir mesafe kaldığında durabilen bir robot programı oluşturulacaktır. Öğrencilere, mesafe sensörü ile ölçülen uzaklık değerinin robotun hareketini sağlayan motorların güç değeri olarak aktarıldığında, mesafe azaldıkça motorların gücünün de azalacağı ve böylece robotun engele yaklaştıkça yavaşlayacağı detaylı bir şekilde açıklanır. Resim 2.11'de görülen programın, engele yaklaştıkça yavaşlayan ve 5 cm mesafede durabilen bir robot için hazırlanabilmesi amacıyla öğrencilere yeterli süre tanınır. Gerekli görüldüğünde, programın mantığı tekrar gözden geçirilerek öğrencilere daha iyi kavratılması sağlanır.



Resim 2.11 Engele Yaklaştıkça Yavaşlayan Robot Program Kodları

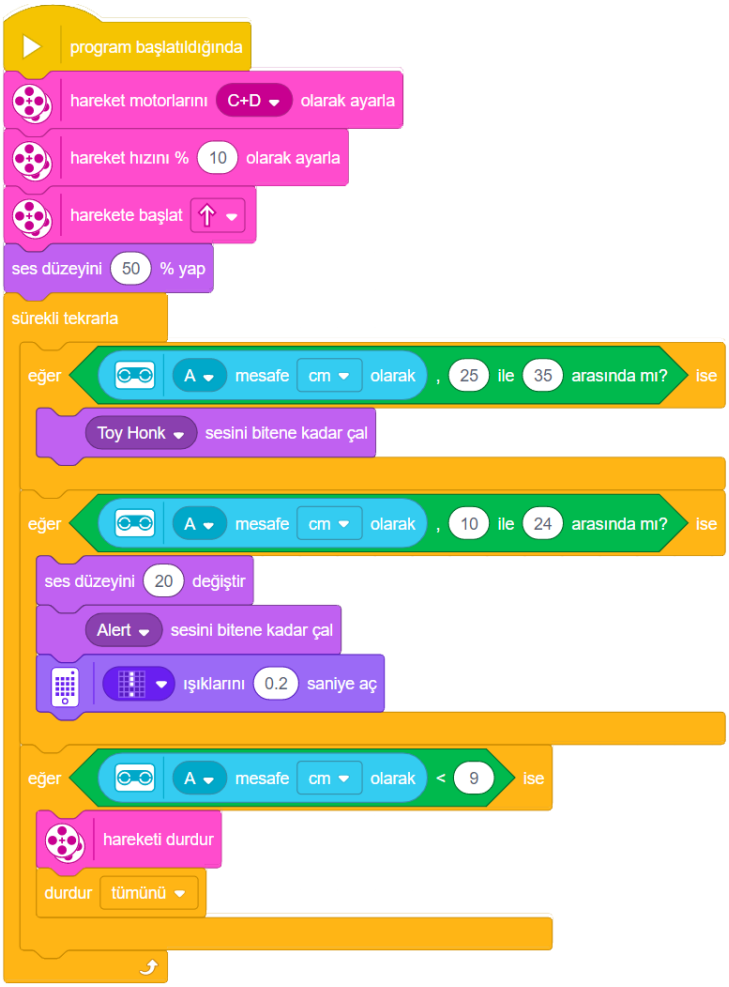
Gözle: Robota Yaklaşan Nesne Olursa Ses Çıkar

Rehber öğretmen, Sürüş Modeli robotuna 5 cm'den daha az bir mesafede bir nesne yaklaştığında ses çalan bir program yazacağını duyurur. Öncelikle, mesafe sensörünün düzgün çalıştığını kontrol eder ve programlama alanına 'sürekli tekrarla' döngüsünü sürükleyip bırakır. Bu döngünün kullanım amacını açıklayarak, programın sürekli olarak mesafe değerini kontrol etmesi gerektiğini vurgular. Ardından, mesafenin 5 cm'den az olup olmadığını kontrol edecek 'eğer ise' bloğunu ekler. Çalmak istediği sesi kütüphaneden seçerek kodunu tamamlar ve koşul doğru olduğu sürece sesin çalındığını öğrencilere gösterir.

Resim 2.12 Robota Yaklaşan Nesne Olursa Ses Çıkaran Program Kodları

Uygula: Park Sensörü

Bu etkinlikte, öğrencilerden robotu taşıtlardaki park sensörlerine benzer bir şekilde programlamaları istenir. Robot, belirli bir mesafeye yaklaştığında uyarı vermeli, mesafe azaldıkça ses seviyesi yükselmeli ve Hub ekranında '!' işareti görüntülenmelidir.



Resim 2.13 Park Sensörü Program Kodları

TASARLA VE ÜRET

Bu etkinlikte robotun önünde bulunan bir arabayı (araba olmak zorunda değil herhangi bir nesneyi) takip etmesi sağlanacaktır.

Öğrencilerden öndeki arabayı ya da nesneyi takip eden, yani öndeki araba ilerledikçe ilerleyen bir robotun program kodunun nasıl yazılacağı üzerinde düşünmeleri istenir. Gruplara çözümü kendi kendilerine üretmesi için zaman verilmelidir. Bu süreçte, öğrenciler grup olarak tartışırlar. Gerektiğinde rehber öğretmen onlara yardımcı olabilir. Ders programı boyunca, tasarlama sürecinde olduğu gibi, öğrencilerin aşağıda örnek olarak da verilen iki adıma benzer bir süreci gerçekleştirmeleri gerekir.

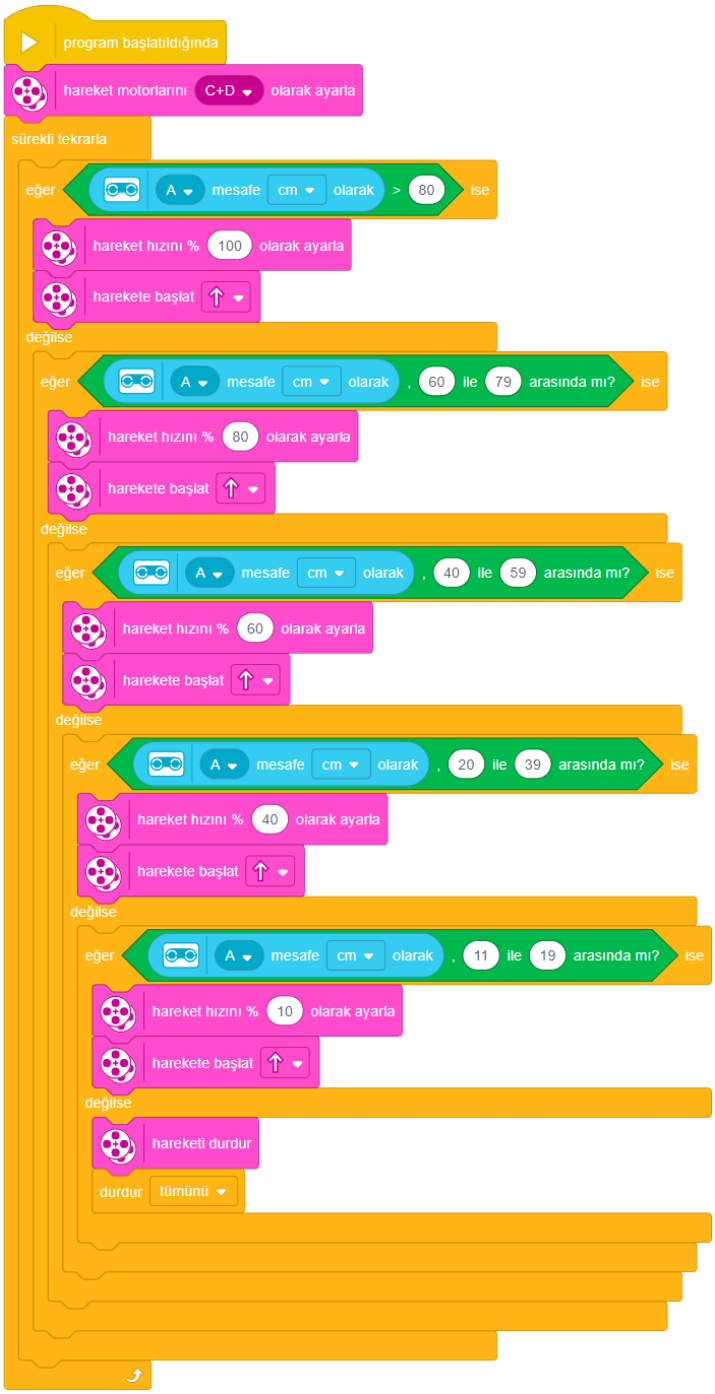
Tanımlama: Öğrencilerin öncelikle istenilen takip işlemi için neler gerektirdiğini belirlemesi ve bunları sıralaması gerekir. Örneğin;

- Robot öndeki araç ile arasındaki mesafeyi sürekli kontrol etmeli.
- Öndeki araç hareket ederse hareket etmeli.
- Öndeki araç hızlı hareket ediyorsa robot hızlanmalı, yavaş hareket ediyorsa yavaşlamalı.
- Öndeki araç durunca durmalı.

Fikir üretme: Bu aşamada öğrencilerin yukarda belirlenen işlemlerin nasıl yapılabileceği ile ilgili fikir yürütmesi gerekir. Örneğin, öğrenciler aşağıdaki maddelere benzer fikirler üretebilir.

- Öndeki araçla arasındaki mesafeyi sürekli kontrol etmek için mesafe sensörü ve döngü kullanılabilir.
- Öndeki araçla arasındaki mesafe 80 cm'den büyükse hareketin hızı %100 olarak ayarlanacak.
- Öndeki araçla arasındaki mesafe 60 cm'den büyük ve 80 cm'den küçükse hareketin hızı %80 olarak ayarlanacak.
- Öndeki araçla arasındaki mesafe 40 cm'den büyük ve 60 cm'den küçükse hareketin hızı %60 olarak ayarlanacak.
- Öndeki araçla arasındaki mesafe 20 cm'den büyük ve 40 cm'den küçükse hareketin hızı %40 olarak ayarlanacak.
- Öndeki araçla arasındaki mesafe 10 cm'den büyük ve 20 cm'den küçükse hareketin hızı %10 olarak ayarlanacak.
- Öndeki araç 10 cm'den yakınsa robot duracak.

Temel olarak bu adımlarla robottan istenilen işlemler yapılabilir. Ama öğrenci isterse robotun öndeki araca daha yakın gitmesini veya öndeki aracın hızına daha fazla ayak uydurmasını da sağlayabilir. Mesafeye bağlı olarak robotun yavaşlaması da sağlanabilir. Her grubun çözümü farklı olabilir, önemli olan bu fikirlerin gruplar tarafından ortaya konulması ve ortaya konulan fikirlerin problemin çözümünü sağlayabilmesidir. Rehber öğretmen gözetiminde öğrenciler çözüm için tasarımlarını yaptıktan sonra bilgisayar ve robot başında çalışarak istenilen görevi yerine getirmelidir.



Resim 2.14 Öndeki Aracı Takip Et Program Kodları

DEĞERLENDİR

Günün sonunda rehber öğretmen öğrencilere aşağıdaki soruları yönelterek öğrenilenlerin birlikte özetlenmesini sağlamalıdır.

- Size bugün Tasarla ve Üret adımında verilen problemi nasıl tanımlarsınız? (Problemi kendi cümleleri ile ifade etme).

- En çok hangi görevde zorlandınız? Bu zorlukların üstesinden nasıl geldiniz? (Problemin çözümü için hangi stratejileri kullandınız ve neden bu stratejileri seçtiniz?) Yeteri kadar tartışma ortamı oluşmazsa rehber öğretmen aşağıdaki soruları kullanarak tartışma ortamı yaratmaya çalışır.
    - Öndeki aracı takip eden programda, robotun hareket etmeye başlamasını nasıl sağladınız?
    - Öndeki aracı takip eden programda, öndeki araç durunca robotun durmasını nasıl sağladınız?
- Kullandığınız yöntemler bu sıkıntıları gidermede başarılı oldu mu?
- Grup arkadaşınızla anlaşmazlığa düştüğünüz durumlar oldu mu ve bunların üstesinden gelmek için neler yaptınız?
- Grup arkadaşınızdan ne/neler öğrendiniz?

Değerlendirme, öğrencileri sıkmadan, her bir soru için verilen cevaplar tatmin edici bir düzeye ulaşıncaya kadar devam ettirilebilir.

İLAVE ETKİNLİK

Görev-Takımlar Yarışıyor

Bu bölümde iki yarışma yapılır. Öğrencilerin yarışmalara takım olarak katılması gerekir. Yarışmalardaki amaç robotun ileride bulunan bir engele kadar ilerlemesi ve geri gelerek başlangıç noktasına vardığında durmasıdır. Başlangıç noktasından 1 metre uzağa engel konulur. Robotların engele 10 cm mesafe kala geri dönmesi ve başladıkları noktaya gelmesi gerekir. Robot engele 8 cm'den fazla yaklaşırsa veya 12 cm'den daha uzak bir mesafedeyken geri gitmeye başlarsa yarışmacılar elenir. Ayrıca robot geri geri giderken ilk başlangıç noktasını 2 cm'den fazla geçer ya da başlangıç noktasına 2 cm'den daha fazla mesafe varken durursa yarışmacılar yine elenir.

Yarışma 1: Bu yarışmada amaç başlangıç çizgisine en yakın durmaktır. Programı yazmaya başlamadan önce grupların tasarlama adımı için yukarda bir örneği verilen tanımlama ve fikir üretme sürecini gerçekleştirmesi gerekir. Bu yarışma için verilen süre rehber öğretmen tarafından belirlenir. Yukarıda belirtilen kurallarla birlikte bu süreyi geçiren yarışmacılar elenmiş sayılır. Bütün grupların robotları işlemlerini bitirdiğinde başlangıç noktasına olan uzaklıkları ve işlemi tamamlama süreleri kayıt altına alınmalıdır. Başlangıç çizgisine en yakın duran robot yarışmayı kazanır. Eşitlik durumunda ise hızlı gelen robot birinci sayılır.

Yarışma 2: Bu yarışmadaki amaç görevi en kısa sürede tamamlamaktır. Programı yazmaya başlamadan önce grupların tasarlama adımı için yine yukarıda bir örneği verilen tanımlama ve fikir üretme sürecini gerçekleştirmesi gerekir. Yarışma kurallarına uyan robotlardan görevi en hızlı tamamlayan birinci olur. Eşitlik durumunda başlangıç noktasına daha yakın olan robot birincili sayılır.

3. Hafta - Kuvvet ve Hareket Sensörleri

Haftanın Amacı:

Kuvvet sensörü öğrencilere tanıtılarak, kuvvet sensörünün; basılma, sert basılma ve bırakılma parametreleri ve ölçülen kuvvet miktarının farklı programlarda kullanılması sağlanacaktır. Ayrıca öğrencilerin hareket (gyro) sensörünü kullanarak, Hub'ın üç eksendeki dönme hareketlerini programlarında kullanabilmelerini sağlayacak temel bilgi ve becerilerin kazandırılması amaçlanmıştır. Öğrencilerin Hub'ın yunuslama, yuvarlanma ve sapma açılarını kullanarak farklı programlar hazırlayabilmeleri hedeflenmiştir.

Haftanın Kazanımları:

- Öğrenciler kuvvet sensörünün basılma, sert basılma ve bırakılma durumlarını, program akışında kullanabilir.
- Öğrenciler kuvvet sensörü kullanarak, kuvvet ölçümü yapabilir.
- Öğrenciler sensörlerden aktarılan verilerin grafiğini çizdirebilir.
- Öğrenciler birden fazla değişkenin grafiklerini yorumlayabilir.
- Öğrenciler hareket sensörü ile farklı eksenlerdeki dönme açısını tespit ederek program içerisinde kullanabilirler.
- Öğrenciler robotun herhangi bir eksende istenilen açı miktarında dönmesini sağlayabilirler.

Kullanılacak Malzemeler:

Robot seti, bilgisayar ve çalışma alanı (mat).

Ekler:

Konu anlatımında yer alan örnek programlar dijital formatta rehber öğretmenlere sunulmuştur. Örnek program dosyalarının numaralandırılmasında, konu anlatımındaki resim numaraları temel alınmıştır. Bu hafta aşağıda sıralanan programlar, çalışma alanı (mat) ve video dosyası ekte rehber öğretmenlere sunulmuştur. Etkinlik öncesinde her grup için bir tane Mat_3_A4.pdf dosyasının çıktısı alınmalıdır.

Program_3.3_Kuvvet_Yuzde_Hareket.llsp3
Program_3.4_Oyun_Newtonx10_cm.llsp3
Program_3.7_Kuvvet_Engele_Carpinca_Duran.llsp3
Program_3.8_Carpisma_Testi.llsp3
Program_3.15_90_Derece_Hareketi.llsp3
Program_3.17_Kare_Hareketi.llsp3
Program_3.19_Su_Terazisi_Isiklari.llsp3
Program_3.22_Tasarla_Uret_Hedefe_Kilitlen.llsp3
Program_3.23_Sallaninca_Rastgele_Sayi.llsp3
Program_3.25_Yonunu_Kaybetmeyen_Robot.llsp3
Video_3.1_Tasarla_Uret_Gorev.mp4
Mat_3_A4.pdf

Kuvvet Sensörü

Rehber Öğretmen İçeriği – Kuvvet Sensörü

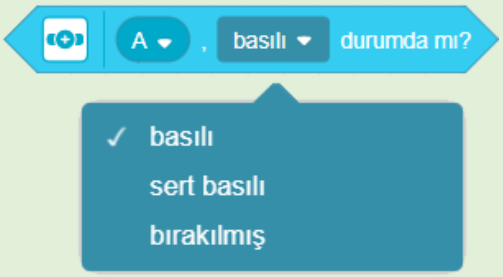
Kuvvet sensörü, EV3 robot setinde bulunan dokunma sensörlerine birçok yönden benzer. Kuvvet sensörü, bir dokunma sensörü gibi basıldığında ve bırakıldığında durumlarını algılayabilir. Ancak kuvvet sensörü, dokunma sensöründen farklı olarak 10 Newton'a kadar kuvveti ölçebilir. Ayrıca sert basıldığında (5N üzeri kuvvetle basıldığında) durumunu da algılayabilir.

Newton, kuvvet birimi olup simgesi N'dir. Terim, fizik bilimine yaptığı katkılar nedeni ile İngiliz bilim adamı Isaac Newton'un adı ile anılır. Kütlesi 1 kg olan bir cismin hızını, 1 saniye içerisinde 1 m/s arttırmak için o cisme uygulanması gereken kuvvet, newton cinsinden 1N = 1kg.m/s^2 olarak tanımlanır.

Bu hafta etkinliklerde kullanılacak yeni bloklar ve her bloğun açıklaması aşağıda verilmiştir.

Sensörler Blok Paleti

Basılı mı?



Bu blok, bir "Boole Bloğu" olduğundan doğru (true) veya yanlış (false) değerlerini alır. Kuvvet sensörü basılmış (0 N'dan daha büyük bir kuvvet uygulanmış) sert basılmış (5 N'dan daha büyük bir kuvvet uygulanmış) veya serbest bırakılmış (hiçbir kuvvet uygulanmamış) olduğunda doğru sonucu verir.

Basınç

Bu blok kuvvet sensörüne uygulanan basıncın değerini Newton veya yüzde olarak alır. Kuvvet sensörünün teknik özellikler kitapçığında 2,5 - 10 Newton'u tespit edebildiği belirtilmiştir. Newton olarak ölçülen basınç değeri maksimum 10N, yüzde olarak 100'e karşılık gelir.

Değişkenler Blok Paleti

Değişkenler, programlarda sayıları, metinleri veya diğer bilgilerin tutulmasına olanak tanır. Değişkenler paletinde, yeni değişkenler oluşturabilir, mevcut değişkenler programlarda

kullanılabilir ve değiştirilebilir. Değişkenler paletinde 'Bir Değişken Oluştur' düğmesi tıklandığında açılan pencereden değişkenin ismi girilerek yeni bir değişken oluşturulur. Oluşturulan yeni değişken ismi ile üç farklı program bloğu listeye eklenir.

Değişken

Bir değişken oluşturulduğunda, değişkenin ismi ile bir blok oluşur. Örneğin 'kutu' isimli bir değişken oluşturulduğunda, yukarıdaki gibi bir blok oluşur ve bu blok değişkenin aldığı değeri yansıtır. Bu blok, diğer blokların içerisine yerleştirilerek kullanılabilir. Örneğin robotun değişken değeri kadar tur, cm, saniye ileri gitmesi sağlanabilir.

Değişkeni Şuna Ayarla

Bu blok, seçilen değişkene istenilen değerin atanmasını sağlar. Örneğin sensörden ölçülen değerin saklanması için kullanılabilir.

Olaylar Blok Paleti

Basınç Şu Olduğunda

Bu "Şapka Bloğu" kendisine bağlı olan tüm blokların kuvvet sensörüne basıldığında, sert basıldığında, sensör bırakıldığında veya uygulanan basınçta herhangi bir değişiklik algılandığında çalışmasını sağlar.

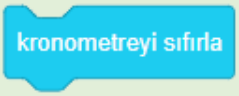
Bu blok sadece belirtilen olay durumunda tetiklenir. Kuvvet sensöründe basıncın değişmemesi halinde blok yeniden tetiklenmez.

Sensörler Blok Paleti

Kronometre

Bu blok program çalışmaya başladığı andan itibaren geçen sürenin saniye cinsinden değerini alır. Program her başlatıldığında kronometre değeri sıfırlanır.

Kronometreyi Sıfırla

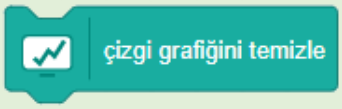


Bu blok, sayesinde programın akışı içeresinde istenilen durumda kronometre değeri sıfırlanır.

Blok Eklentileri – Çizgi Grafiği

Çizgi grafiği blokları SPIKE Prime Uygulaması arayüzünde varsayılan olarak açık değildir. Çizgi grafiği bloklarına ulaşabilmek için, blok paletinde sol alt köşede yer alan "blok eklentilerini göster" düğmesi tıklanarak, çizgi grafiği kutucuğunun seçilmesi gerekmektedir.

Çizgi Grafiğini Temizle

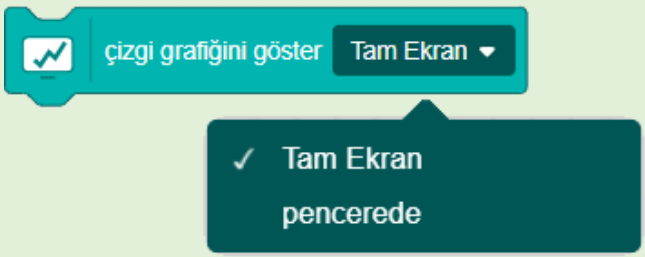


Bu blok, çizgi grafiği ekranını temizler. Önceki ölçümlere ait tüm verileri siler.

Değeri Çizgiye İşaretle

Bu blok, girilen değere veya değişkene (sensör veya kronometre verisi gibi zamanla değişen değerlere) ait verilerin zamana bağlı grafiğini istenilen renkte çizgi grafiği olarak çizilmesini sağlar.

Çizgi Grafiğini Tam Ekran Göster



Bu blok, çizgi grafiğinin tam ekran ya da yüzen bir pencerede açılmasını sağlar.

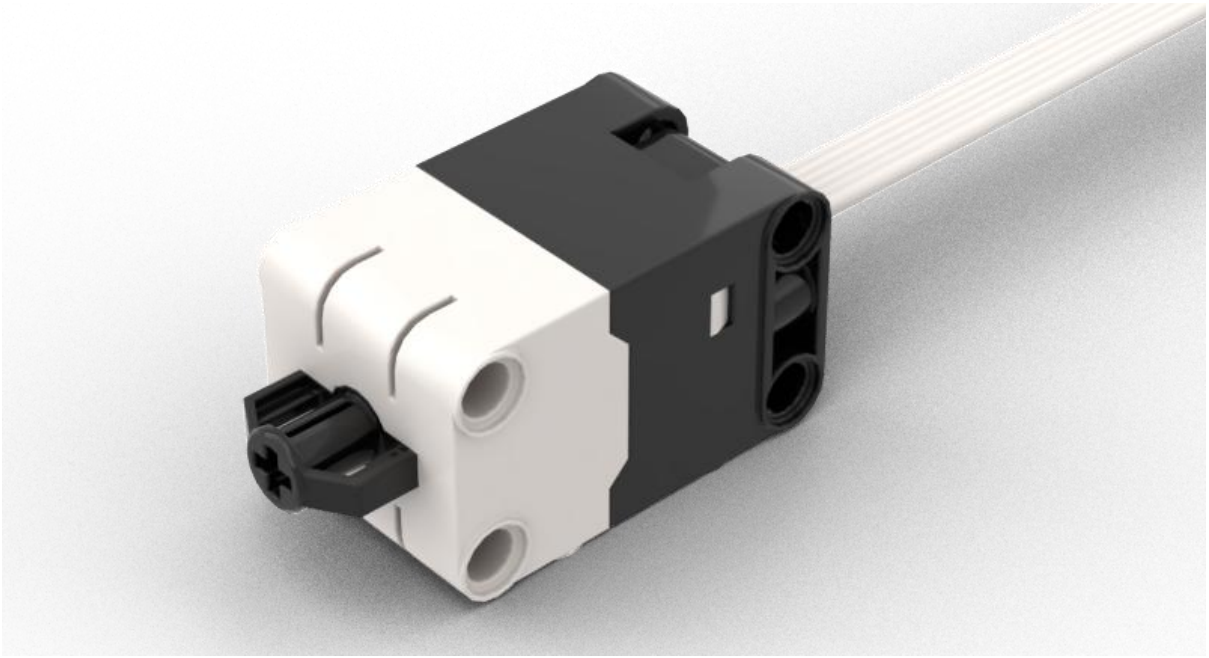
Not

Etkinliklerde hazırlanması planlanan programların ekran görüntüleri, ilgili konu içerisinde sunulmuştur. Ayrıca program dosyaları da rehber öğretmenlerle paylaşılmıştır. Bu paylaşımlar rehber öğretmenlerin etkinliklere hazırlanmasında kolaylık sağlamak içindir. Gözle etkinliklerinde rehber öğretmen ilgili programları öğrencilere açıklayarak adım adım hazırlamalıdır. Tasarla ve üret etkinliklerinde ise rehber öğretmen programları ve ekran görüntülerini öğrencilerle kesinlikle ***paylaşmamalıdır.***

GÖZLE VE UYGULA

Gözle: Kuvvet Sensörü

Rehber öğretmen, geçen hafta öğrendikleri mesafe sensörünü öğrencilere hatırlattıktan sonra, bu hafta kuvvet sensörü ile devam edeceklerini ifade ederek, robot setindeki kuvvet sensörünü göstererek özelliklerini anlatır.



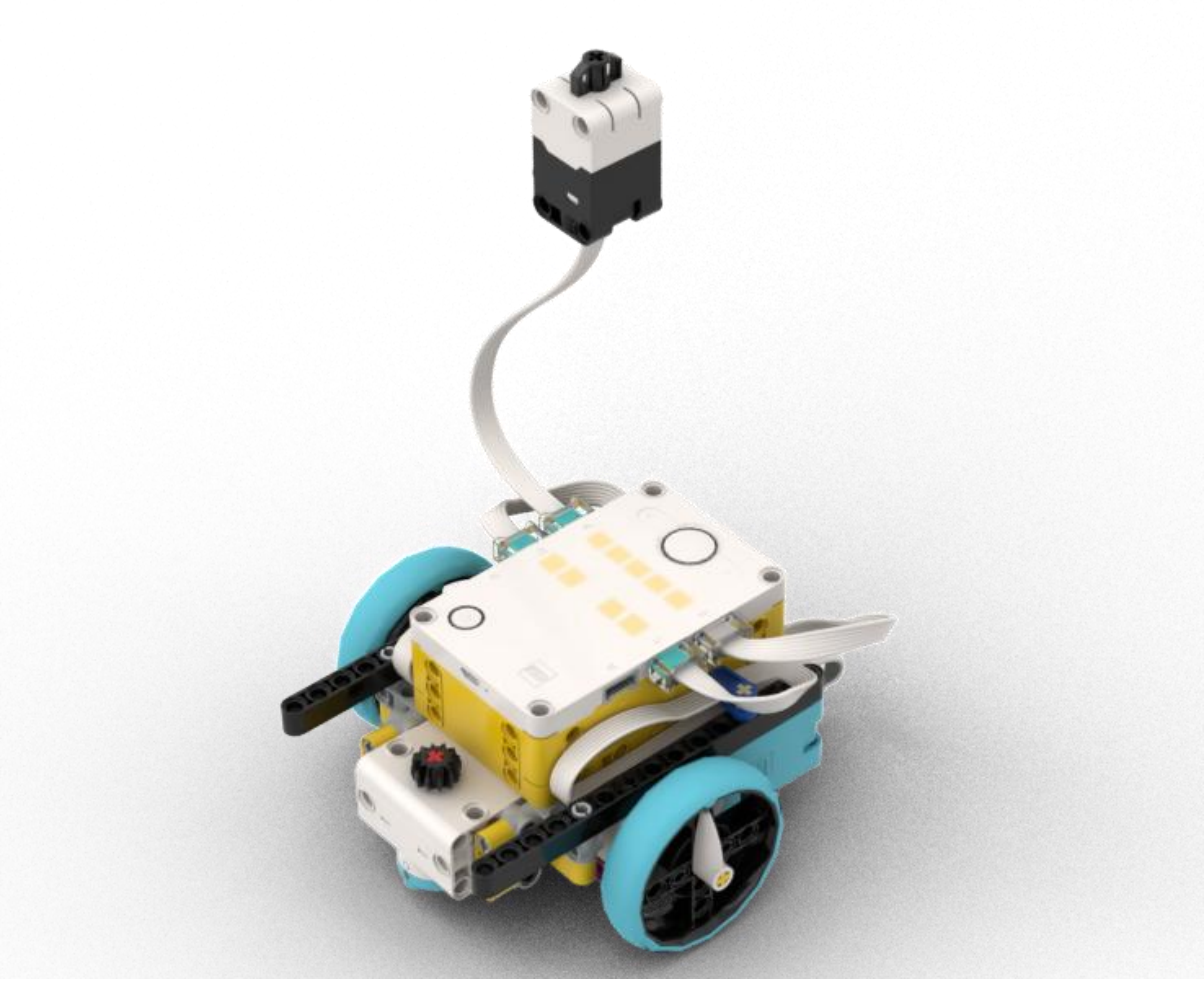
Resim 3.1 Kuvvet Sensörü

Rehber öğretmen, öğrencilerden kuvvet sensörünü Resim 3.2'de görüldüğü gibi Sürüş Modeli tasarımında Hub'ın F portuna bağlantısını yapmalarını ister. E portunda ise büyük motorun bağlı olduğundan emin olmalarını ister.

Robot açık ve ekranda kalp sembolü görünüyor iken, bir kez daha açma/kapama düğmesine kısa süreli basıldığında kuvvet sensörünün bağlı olduğu port (F) ile büyük motorun bağlı olduğu port (E) arasında ışıklar ile bir bağlantı oluştuğu gözlemlenecektir. Bu durumda, kuvvet sensörüne dokunulduğunda büyük motor dönmeye başlayacaktır. Kuvvet sensörüne ne kadar büyük bir kuvvet uygulanırsa, büyük motorun da o kadar hızlı döndüğünü gözlemlenecektir.

Uygula: Kuvvet Sensörü ile Motorların Çalıştırılması

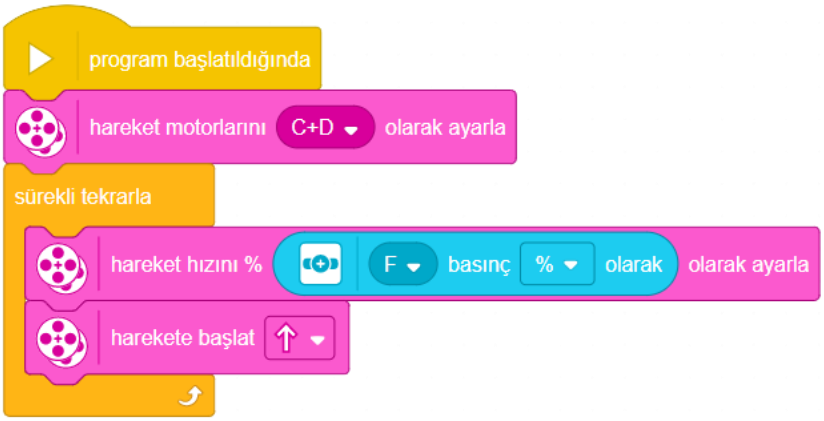
Rehber öğretmen öğrencilerden anlatıları uygulayarak büyük motoru kuvvet sensörü ile çalıştırmalarını ister. Benzer şekilde diğer motorların bağlı olduğu portun karşısındaki porta herhangi bir sensör takıldığında, sensörden okunan veri motorun dönme hızını ayarlayacaktır. Diğer motorlar için de kuvvet sensörü ile denemeler yapıldıktan sonra, öğrenciler bir kez daha açma/kapama düğmesine kısa süreli basarak tekrar program moduna (kalp simgesini olduğu moda) geçerler.



Resim 3.2 Kuvvet Sensörü ile Motor Hareketi

Gözle: Kuvvet Sensörü ile İlk Program

Rehber öğretmen öğrencilere bir önceki uygulamada yalnızca bir motorun çalıştırılabildiğini belirtir. İki motorun eş zamanlı çalışmasını sağlayacak, robotun kuvvet sensörüne uygulanan basınç miktarına göre ileriye doğru hareket hızının ayarlanabildiği programı öğrencilere anlatır. Rehber öğretmen programlarında basınç değerini yüzde ve Newton olarak kullandıkları farklı iki program hazırlamalarını ve programların sonuçlarını değerlendirmelerini ister. Örnek program Resim 3.3'te sunulmuştur.



Resim 3.3 Kuvvet Sensörü Yüzde Hareket Örnek Program

Rehber Öğretmen İçeriği – Kuvvet Sensörü

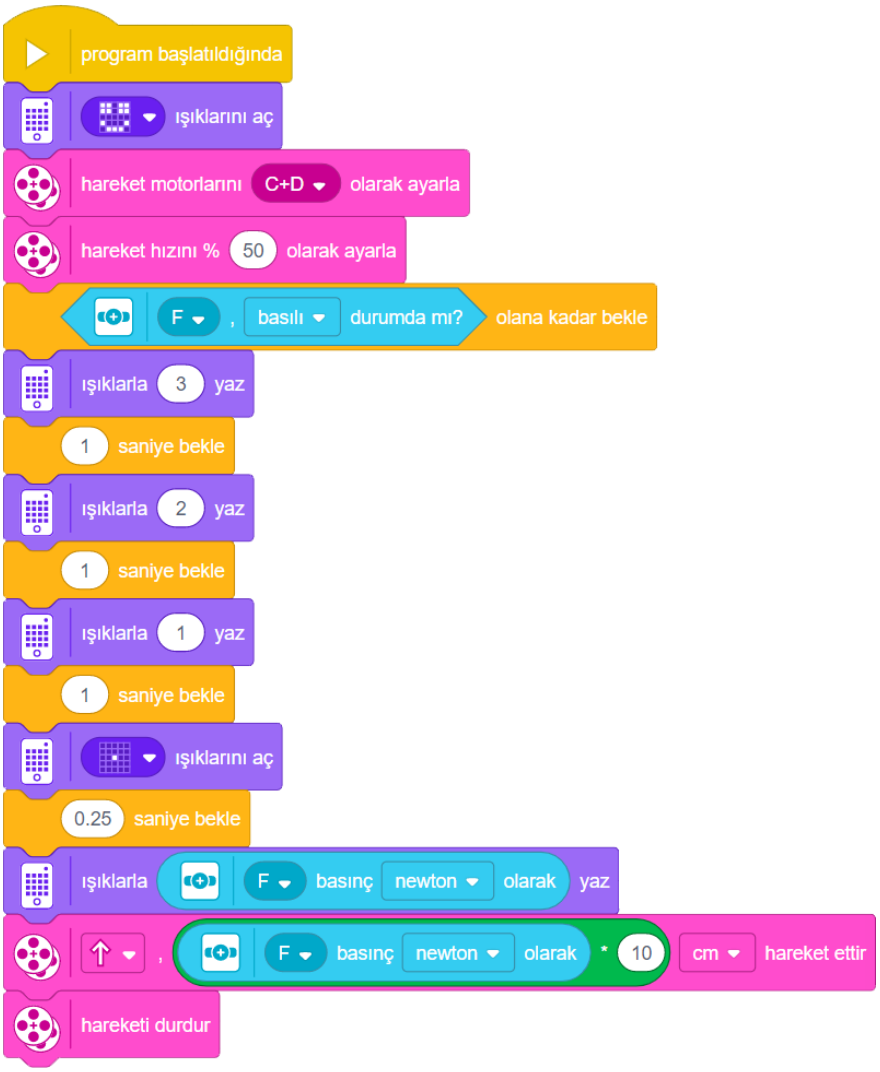
Basınç Newton olarak ölçüldüğünde, kuvvet sensörü ile maksimum 10 Newtonluk bir kuvvet ölçülebildiğinden, robotun hareket hızı maksimum %10 olabilir.

Basınç yüzde olarak ölçüldüğünde, 10 Newtonluk basınç %100 olarak ölçülecek ve robotun hareket hızı maksimum %100 olabilecektir.

Gözle ve Uygula: Kuvvet Sensörü Oyunu

Rehber öğretmen Resim 3.4'te gösterilen programı öğrenciler ile paylaşarak aynı programı hazırlamalarını ister. Sorasında rehber öğretmen programın nasıl çalıştığını öğrencilere anlatır. Program çalıştırıldığında robotun ekranında gülen yüz şeklinde ışıklar yanacaktır. Öğrenciler robota takılı olan kuvvet sensörüne kuvvet uyguladıkları anda, robot ekranında 3'ten geriye doğru sayılmaya başlayacak ve sıfır anında ekranda bir piksel yanacaktır (Program slotlarını gösteren sıfır rakamı ile karıştığından sadece bir pikselin yanması ile sıfır durumu gösterilmiştir). Robot, Hub ekranında tek pikselin yandığı anda kuvvet sensöründen ölçülen basınç miktarının on katı santimetre ileri gidecektir. Hareket esnasında ölçülen kuvvetin Newton cinsinden değeri ekranda görülecektir.

Öğrenciler bir hedef uzaklık belirleyip örneğin 70 cm (belirlenecek hedef uzaklık 10 cm'nin katları olmalıdır) belirlenen uzaklığa bir lego parçası yerleştireceklerdir. Programı çalıştırıp kuvvet sensörüne uygulayacakları basınç ile robotun istenilen miktarda giderek hedefe ulaşmasını sağlayacaklardır. Grup üyeleri sırasıyla deneme yaparak robotu hedefe ulaştırmalıdır. Sonrasında farklı uzaklıklar ile aynı uygulama gerçekleştirilebilir.



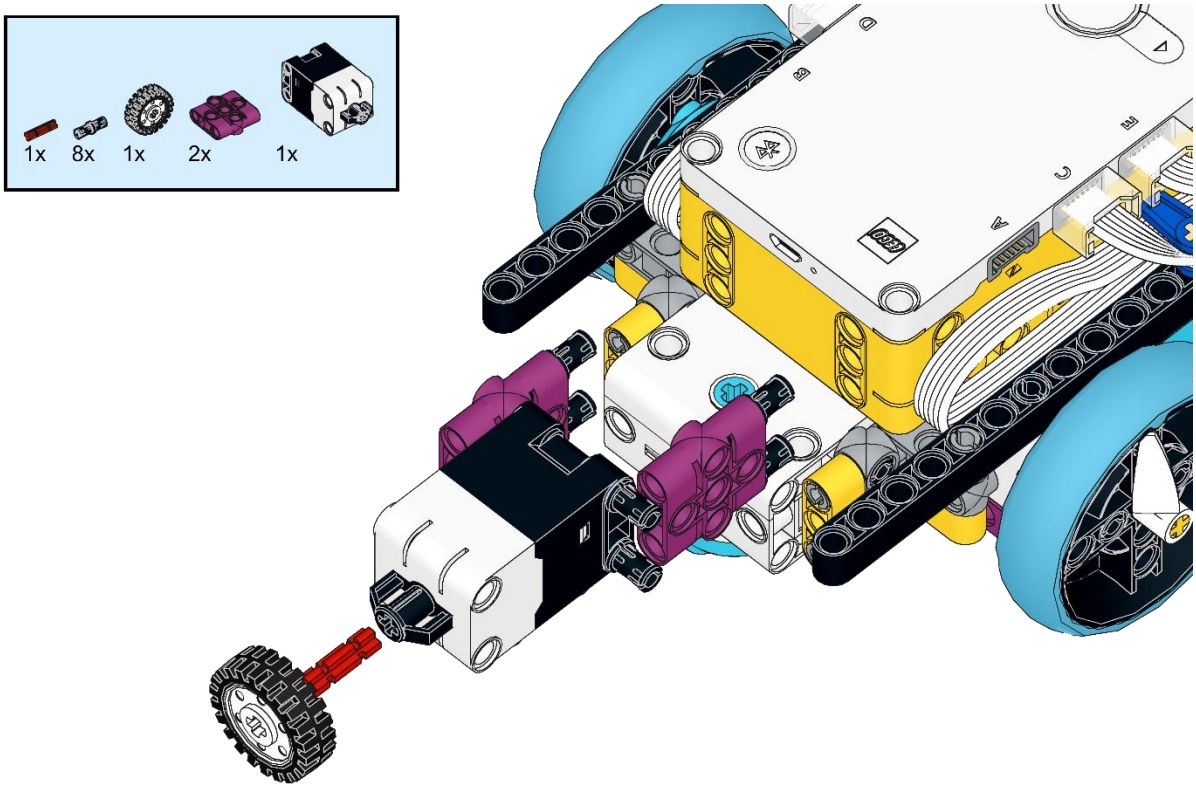
Resim 3.4 Kuvvet Sensörü Oyunu Programı

Not

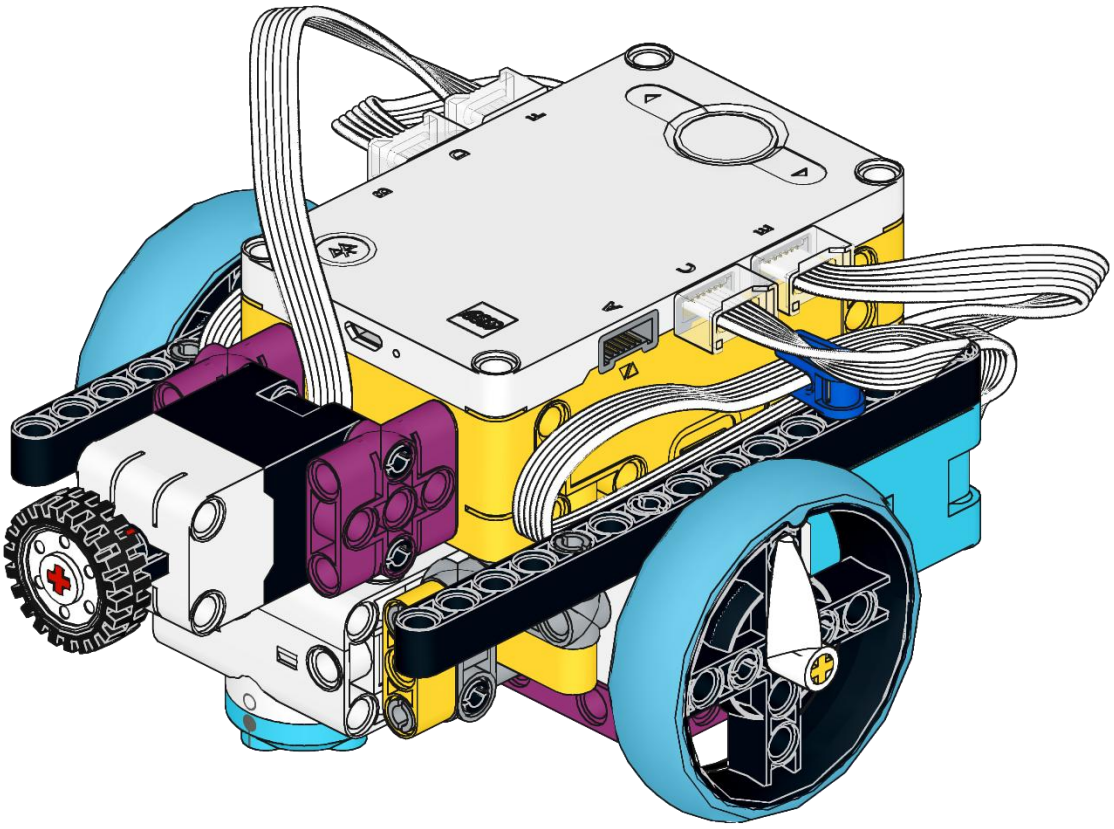
Rehber öğretmen, öğrencilerden uygulana kuvvet miktarını yorumlamalarını isteyebilir. Günlük yaşantımızda uzunluk, ağırlık gibi kavramlarla ilgili bir öngörümüz var iken kuvvet konusunda öngörümüz pek yoktur. Örneğin oyunda uyguladıkları 7N'luk kuvveti baz alarak, bir pet şişeyi sıkıştırmak için en az kaç Newtonluk bir kuvvet gerekeceğini tartışabilirler.

Gözle ve Uygula: Engele Çarpınca Duran Robot

Rehber öğretmen öğrencilere Resim 3.5 ve Resim 3.6'da görüldüğü gibi dokunma sensörünü sürüş modeline birleştirmelerini ve robotlarını ileri doğru giderken bir engele çarpması durumunda duracak şekilde programlamalarını ister. Örnek program Resim 3.7'de verilmiştir.



Resim 3.5 Kuvvet Sensörü Montajı Birinci Adım



Resim 3.6 Kuvvet Sensörü Montajı İkinci Adım

Resim 3.7 Engele Çarpınca Duran Robot Örnek Programı

Gözle ve Uygula: Çarpışma Testi

Rehber öğretmen öğrencilere arabalarla yapılan çarpışma testleri hakkında bilgileri olup olmadığını, öncesinde herhangi bir çarpışma testi videosu izleyip izlemediklerini sorar. Video paylaşım sitelerinden otomobillerin çarpışma testi videolarını öğrencilere izletir.

Not

youtube.com sitesinden Euro NCAP sözcükleri aratılarak uygun birkaç video izletilebilir. Örnek olarak aşağıda birkaç video bağlantısı paylaşılmıştır.

https://www.youtube.com/watch?v=2indFTVpQ_Y

https://www.youtube.com/watch?v=vgfKw6ypvDg

Rehber öğretmen öğrenciler ile çarpışma testlerinin neden yapıldığını, otomobillerin hasar almasına neyin sebep olduğunu, otomobillerin hızının çarpışmanın şiddetine olan etkisini öğrencilerin tartışmalarını sağlar.

Resim 3.8'de verilen programı öğrenciler ile paylaşır ve programı açıklar. Programdaki hareket hızını %20, %40, %60, %80 ve %100 olarak değiştireceklerini ve her bir hız değeri için grafikte okunan maksimum kuvvet miktarını kaydetmelerini ister. Program bir engele (duvara) doğru düz bir şekilde ilerleyen robotun çarpışma başladığı anda kuvvet sensörü verilerinin grafiğinin çizilmesini sağlayacaktır. Ayrıca program robot çarpışma sonrasında engelden uzaklaştığında (öğrenciler robotu aldıklarında) motorları durduracak ve kuvvet sensörü grafiğini tam ekran olarak açılacaktır.

Rehber öğretmen öğrenciler ile buldukları sonucu tartışır. Robotun hızının çarpışma anında oluşan kuvvete etkisi sorgulanır.

Resim 3.8 Çarpışma Testi Programı

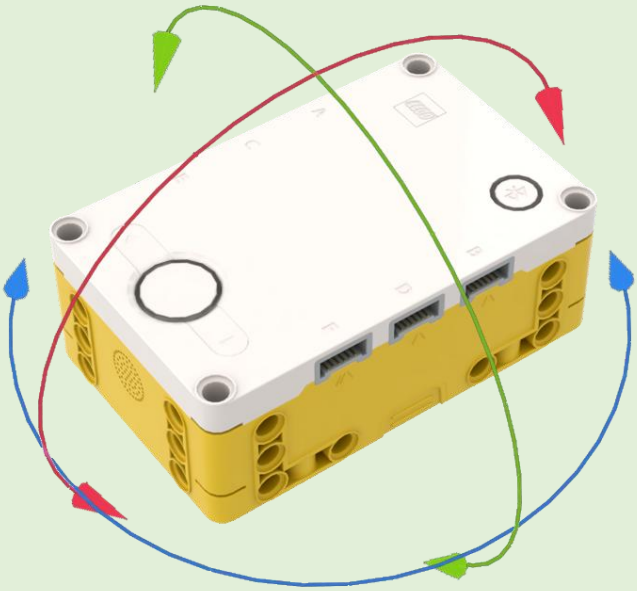
Hareket Sensörü

Rehber Öğretmen İçeriği – Hareket Sensörü

Gyro sensörü olarak da isimlendirilen hareket sensörü açısal dengenin korunması ilkesi ile çalışan ve yön ölçüm işlemlerinde kullanılan sensörlerdir. Günümüzde akıllı telefonlar, navigasyon sistemleri, oyun kumandaları ve robotlar gibi birçok teknolojik üründe kullanılmaktadırlar.

Spike Prime robot setinde hareket sensörü Hub ile bütünleşiktir. EV3 robot setinde olduğu gibi ayrı bir sensör olarak robot setinde yer almamaktadır. Ayrıca EV3 robot setinde yaşanan kayma problemi (robotun dönme hareketi durduğu halde, sensörün açı değerinin artmaya devam etmesi) Spike Prime setinde giderilmiştir.

Ayrıca EV3 setinde yalnızca bir eksende açı ölçülebiliyor iken, Spike Prime setinde altı eksenli hareket sensörü bulunmaktadır. Spike Prime robot setindeki hareket sensörü, üç eksenli ivmeölçer ve üç eksenli jiroskop sensörlerini içermektedir. Böylece bütünleşik hareket sensörü ile üç eksendeki dönme hareketinin açısal değeri ve ivmesi ölçülebilmektedir. Ayrıca sensör Hub'ın sallandığını, dokunulduğunu ve düştüğünü algılayabilmektedir. Resim 3.9'da üç eksendeki dönme hareketi gösterilmiştir.



Resim 3.9 Hub'ın Üç Eksendeki Dönme Hareketi

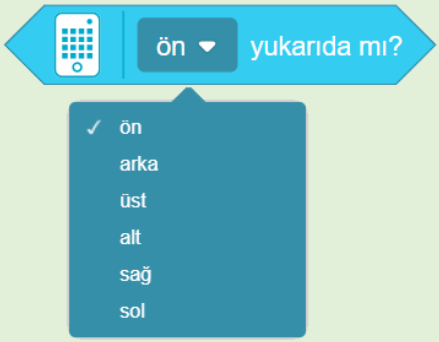
Bu hafta etkinliklerde kullanılacak hareket sensörü ile ilgili kod blokları ve açıklamaları aşağıda verilmiştir.

Sensörler Blok Paleti

Eğildi mi?

Bu blok bir "Boole Bloğu" olduğundan, Hub'ın seçilen yönlerde (dört yön oklarından herhangi biri), herhangi bir yönde (sol üst köşedeki oklar simgesi) veya düz durması (sağ üst köşedeki Hub simgesi) durumlarında doğru (true) sonucunu verir.

Hub yönü şu mu?



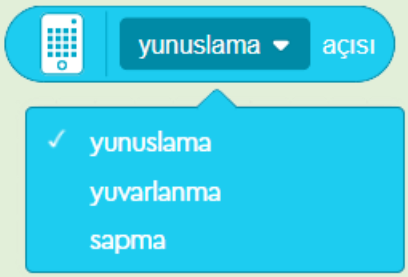
*Hub seçilen yönü yukarı geldiğinde doğru (true) sonucunu verir. Hub'ın ekranın bulunduğu yönü **üst**, pilin takıldığı yüzeyi **alt**, şarj girişinin olduğu yönü **arka**, hoparlörünün olduğu yönü **ön**, B,D,F girişlerinin olduğu yön **sağ** ve A,C,E girişlerinin olduğu yön ise **sol**dur. Resim 3.10'da Hub'ın yönleri gösterilmiştir.*

Resim 3.10 Hub'ın Yönleri

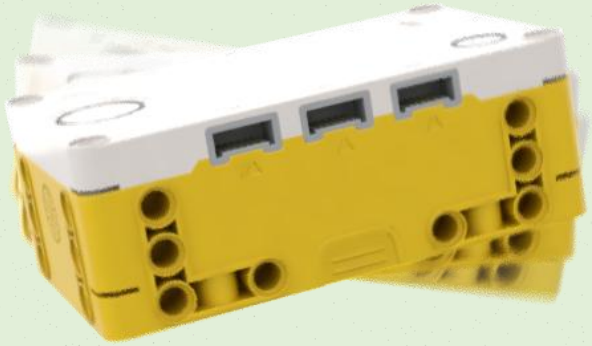
Sallanıyor mu?

Bu blok Hub'ın belirlenen sallanma, tıklatılma ve düşme hareketlerini gerçekleştirdiğinde doğru (true) sonucunu verir.

Hub Yunuslama Yuvarlanma Sapma Açısı



*Bu blok, Hub'ın **yunuslama**, **yuvarlanma** ve **sapma** açı değerlerini verir. Hub'ın üç eksendeki dönmesini ifade eden yunuslama (Resim 3.11), yuvarlanma (Resim 3.12) ve sapma (Resim 3.13) hareketleri aşağıda sunulmuştur.*

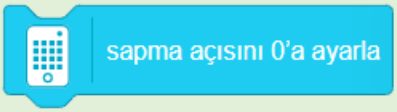


Resim 3.11 Yunuslama

Resim 3.12 Yuvarlanma

Resim 3.13 Sapma

Hub Sapma Açısını 0 Olarak Ayarla



Bu blok Hub'ın o anki sapma açısını sıfır (0) olarak ayarlar. Bu blok çalıştığı anda Hub'ın ön-arka yönünün doğrultusu sıfır derece olarak kabul edilecektir.

Olaylar Blok Paleti

Eğildiğinde

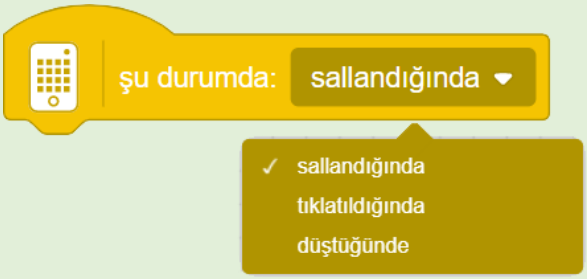
Bir "Şapka Bloğu" olan "eğildiğinde" bloğu, Hub'ın düz bir konumda iken, belirlenen yönlerde bir hareketin gerçekleşmesi durumunda kendisine bağlı olan tüm blokları çalıştırır.

Bu blok sadece belirtilen olay durumunda tetiklenir. Hub yeni bir yönde eğilmediği sürece yeniden tetiklenmez.

Hub Yönü Yukarı Baktığında

Bu blok seçilen yönün yukarı olması durumunda, bağlı olan tüm blokları çalıştırır. Bu blok sadece belirtilen olay durumunda tetiklenir. Tekrar tetiklenmesi için Hub'ın yönünün değişmesi gereklidir.

Hub Sallandığında



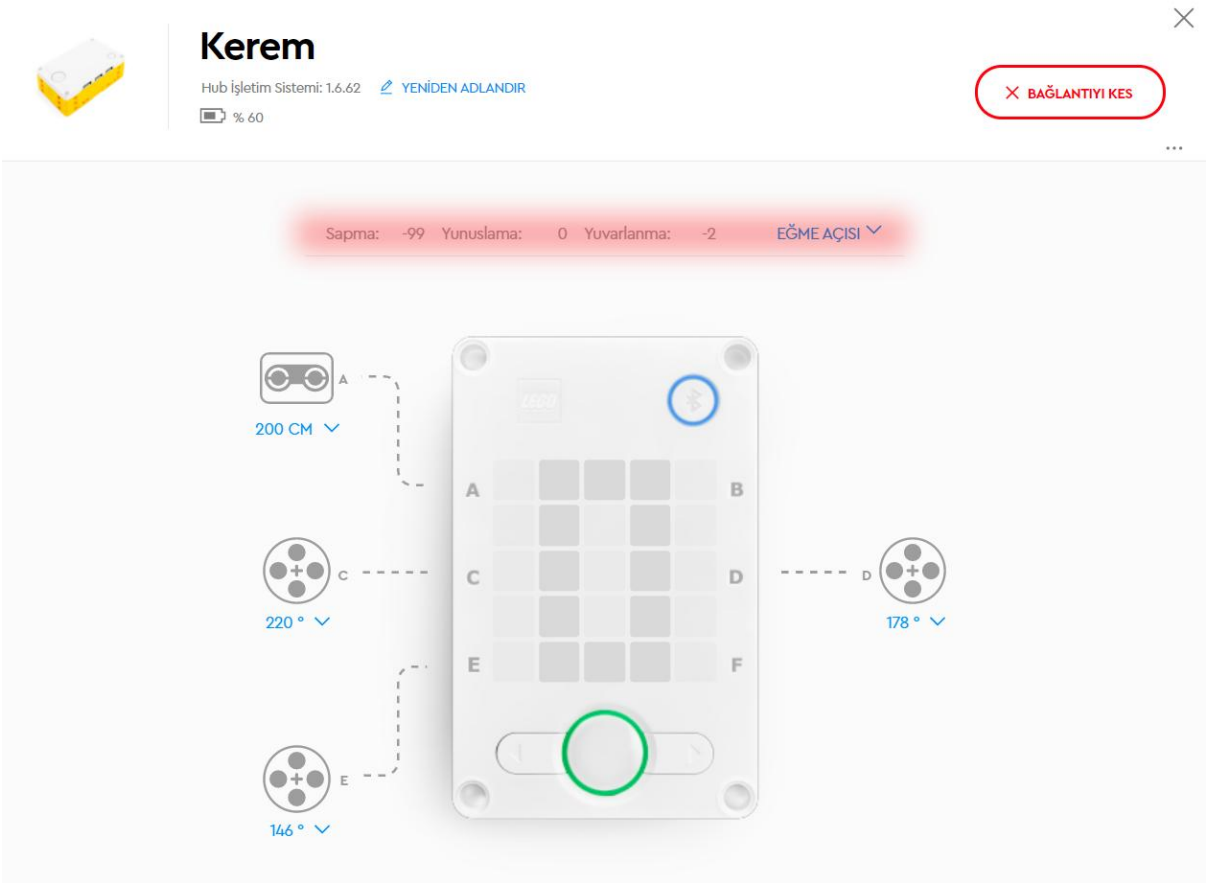
Bu blok, belirlenen durumun (sallanma, tıklatılma veya düşme) gerçekleşmesi durumunda kendisine bağlı olan tüm blokları oynatır. Hub'ın hareketinin değişmemesi durumunda blok yeniden tetiklenmez.

GÖZLE VE UYGULA

Gözle ve Uygula: Robotun 90 Derece Döndürülmesi

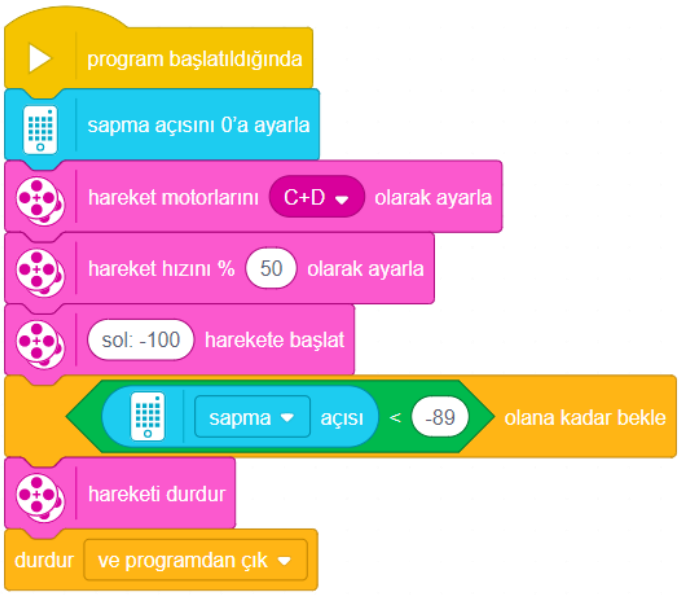
Rehber öğretmen öğrencilere hareket sensörünün kullanıldığı etkinlikler ile devam edeceklerini söyler. Hareket sensörünün Hub'ın içerisinde bütünleşik olduğunu ve diğer sensörler gibi robotlarına birleştirmelerine gerek olmadığını belirtir.

Rehber öğretmen öğrencilerden Spike Prime Uygulamasında robotları Bluetooth veya USB kablosu ile bilgisayara bağlı iken Hub bağlantı penceresini açarak sapma, yunuslama ve yuvarlanma açıların robotun hareketine göre nasıl değiştiğini gözlemlemelerini ister. Rehber öğretmen bu hareketleri açıklamadan öğrencilerin keşfetmelerine olanak sağlar. Resim 3.14'te robotun dönme hareketlerine göre Eğme Açısının (sapma, yunuslama ve yuvarlanma açılarının) değişimi gösterilmiştir.



Resim 3.14 Eğme Açısının Gözlemlenmesi

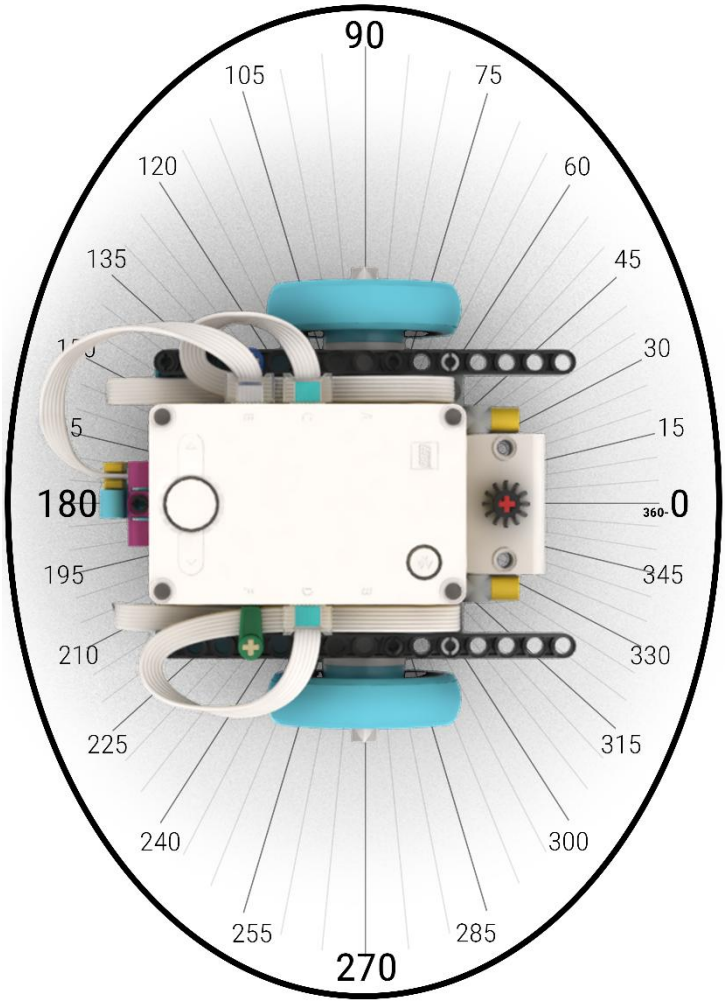
Rehber öğretmen Sürüş Modeli robotunun 90 derece dönmesini sağlayan Resim 3.15'te sunulan programı hazırlayarak Resim 3.16'da gösterildiği gibi ekte sunulan çalışma alanı (açıölçer) üzerinde çalıştırır.



Resim 3.15 Doksan Derece Dönme Örnek Program

Not

Çalışma alanı (açıölçer) öğrencilerde kavram yanılgısına neden olmamak için matematik derslerindeki açılar konusu temel alınarak hazırlanmıştır. Sürüş Modeli tasarımında ise robotun sapma açısı açıölçerin tersi yöndedir. Robotun sapma açısı, saat yönünde artmakta, saatin tersi yönde azalmaktadır. Bu nedenle Resim 3.15'te sunulan program negatif açı değeri ile hazırlanmıştır. Rehber öğretmen programda neden negatif açı değeri kullanıldığını öğrencilere açıklamalıdır.



Resim 3.16 Doksan Derece Dönme Çalışma Alanı

Rehber öğretmen, öğrencilerden aynı programı hazırlayıp çalışma alanı üzerinde robotlarının kaç derece döndüğünü ölçmelerini ve bu ölçümü kaydetmelerini ister. Ayrıca hareket hızını artırarak ve azaltarak farklı ölçümler yapmalarını ister. Sapma açısı değerinin neden 90 dereceden daha büyük olduğu tartışılır. Robotun tam olarak 90 derece dönebilmesi için programda nasıl düzenlemeler yapılabileceği öğrenciler ile tartışılır.

Rehber Öğretmen İçeriği – Robot Neden 90 Dereceden Fazla Döner

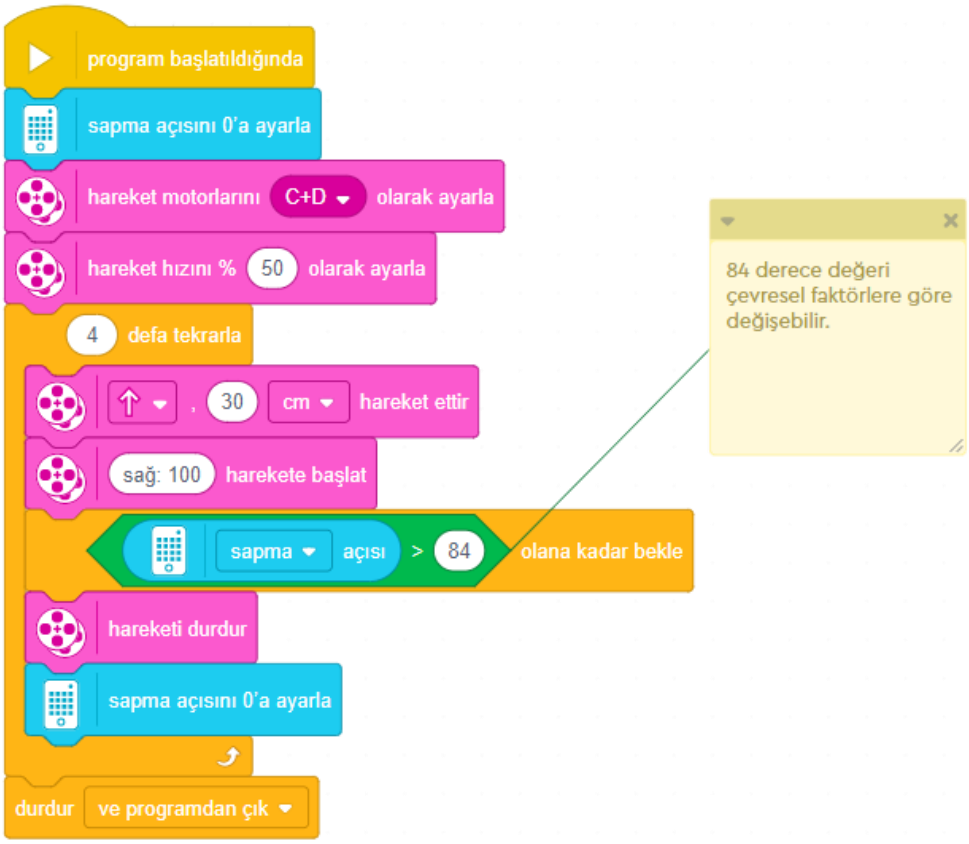
Sensör tam olarak 90 dereceyi ölçtüğü anda bu verinin aktarılması ve motorlara durma komutunun gönderilmesi zaman alır. Bu nedenle robot 90 dereceden fazla döner.

Robot kendi etrafında dönerken oluşan dönme kuvveti, motorlar durdurulduktan sonra da robotun biraz daha dönmesine sebep olur. Dönme kuvvetinin etkisini azaltmak için robotun daha yavaş dönmesi sağlanabilir.

Robotun 90 dereceden daha az bir açı ile dönecek şekilde programlanması bir çözüm olabilir.

Uygula: Kare Şeklinde Hareket

Rehber öğretmen öğrencilerden bir kenarı 30 cm olan bir kare rotasında hareket edecek robotun programını hazırlamalarını ister. Robot başlangıç noktasında, hareketini tamamlamalıdır. Bu nedenle köşelerdeki dönme hareketlerinin tam 90 derece olması gerektiği belirtilir. Örnek program Resim 3.17'de sunulmuştur.



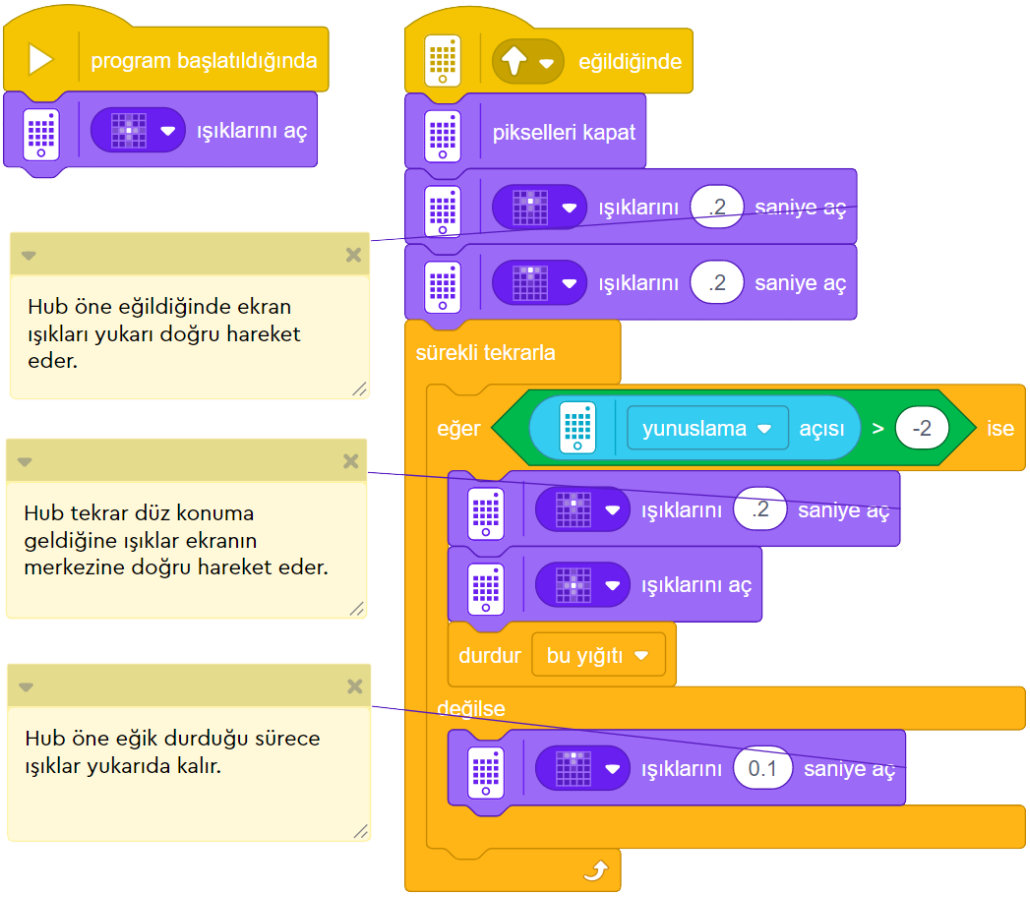
Resim 3.17 Kare Şeklinde Hareket Örnek Program

Gözle ve Uygula: Yunuslama, Yuvarlanma Işıkları

Rehber öğretmen öğrencilere daha öncesinde su terazisi görüp görmediklerini ve su terazisinin ne amaçla kullanıldığını sorar. Tüm öğrencilerin su terazisinin nasıl çalıştığını anladığından emin olduktan sonra benzer şekilde robotun eğimini gösteren bir uygulama gerçekleştireceklerini belirtir.

Resim 3.18 Su Terazisi

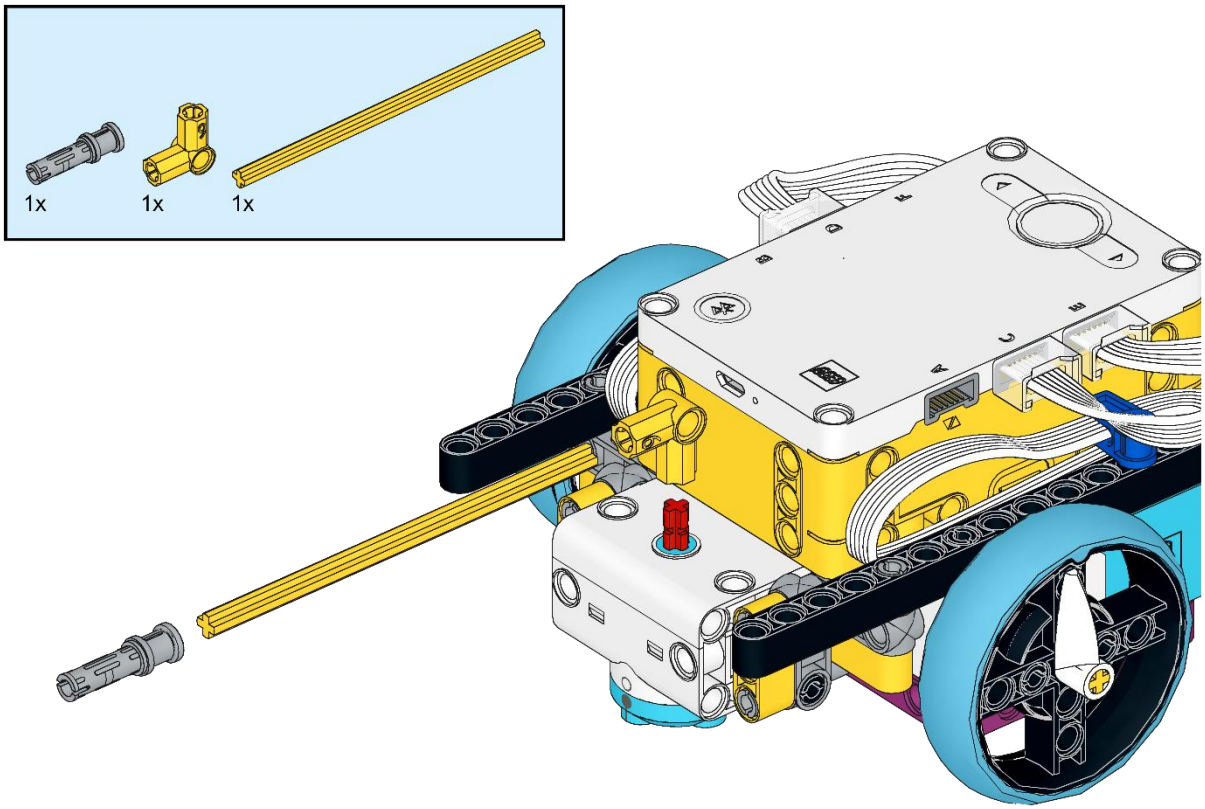
Robotun yunuslama veya yuvarlanma hareketi yapması durumunda bu hareketlerin yönünü ışıkları ile gösteren robot programının nasıl yapılabileceğini öğrencilere açıklar. Şapka bloklarından "*eğildiğinde*" bloğunu öğrencilere anlatır. Yukarı yönlü eğilme gerçekleştiğinde merkezde bulunan ışıkların, yukarıya doğru hareketi ve robot düz konuma geldiğinde tekrar merkeze doğru hareketini rehber öğretmen yaparak öğrencilere anlatır (Resim 3.19). Yapılan programı test eder. Öğrencilerden diğer yönler (aşağı, sağ ve sol) için programı tamamlamalarını ister.



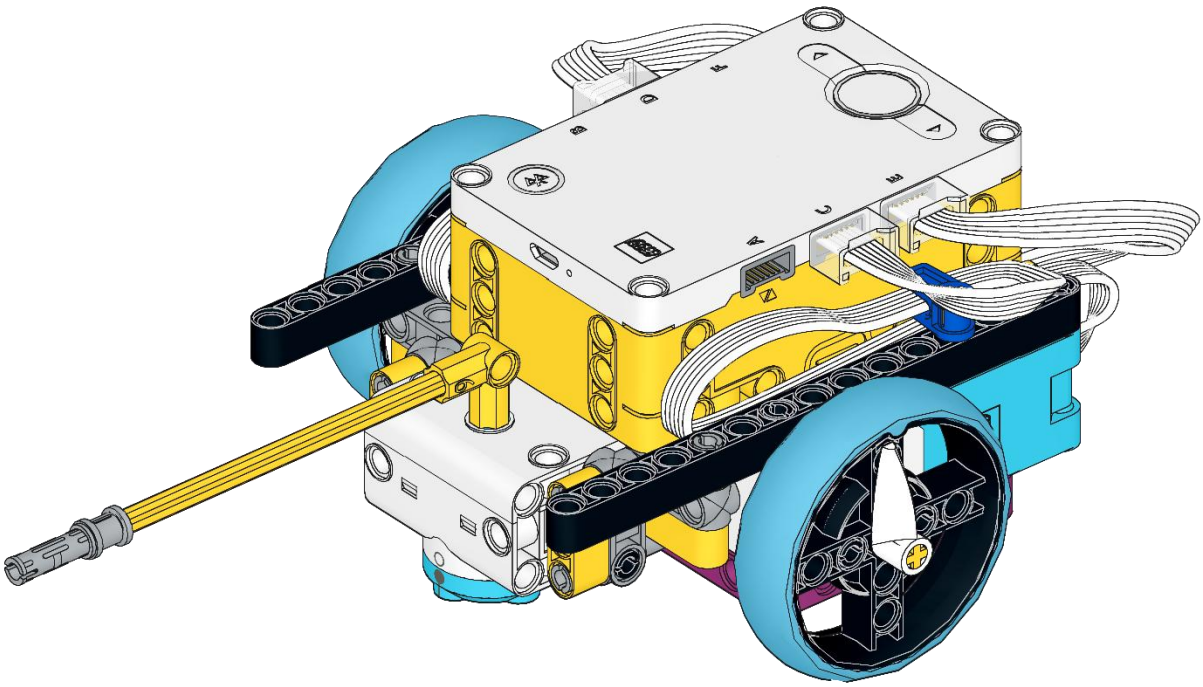
Resim 3.19 Su Terazisi Örnek Programı (Bir Bölümü)

TASARLA

Rehber öğretmen öğrencilerden Resim 3.20 ve 2.21'de görüldüğü gibi sürüş modeline hedef yön kolunu eklemelerini ister.



Resim 3.20 Hedef Yön Kolu Montajı Birinci Adım



Resim 3.21 Hedef Yön Kolu Montajı İkinci Adım

Rehber öğretmen tasarla üret etkinliğinde, büyük motora bağlı hedef yön kolu çevrilerek hedefin yönü, D portuna bağlı motorun (sağ tekerin) döndürülmesi ile robotun ilerleyeceği mesafe ve C portuna bağlı motorun (sol tekerin) döndürülmesi ile hedefe ilerleme hızının belirleneceği bir programın hazırlanacağını belirtir. Robotun D motorunun bir tur (360 derece) çevrilmesi ile robotun hedefe doğru 100 cm, yarım tur (180 derece) çevrilmesi ile 50 cm gitmesi sağlanmalıdır. Benzer şekilde robotun C motorunun bir tur çevrilmesi robotun %100 hızla, çeyrek tur çevrilmesi ise robotun %25 hızla hedefe yaklaşmasını sağlamalıdır. C ve D motorlarının dönme yönünden bağımsız olarak hareket gerçekleşmelidir. Başka bir ifade ile C ve D motorları negatif yönde çevrilmiş olsa bile hareket hedefe doğru olmalıdır, robot hedeften uzaklaşacak yönde hareket etmemelidir.

Robotun üç motoru kullanılarak robota hedefin yönü, hedefin uzaklığı ve hedefe yaklaşma hızı verilerinin girilmesi sağlanmış olacaktır. Program çalıştırıldıktan sonra, bu üç veri girişi Hub üzerindeki sağ veya sol tuşa basılması ile tamamlanacaktır. Sonrasında robot girilen veriler doğrultusunda hareketini gerçekleştirmelidir.

Görevin daha kolay anlaşılabilmesi için, ekte sunulan **Video_3.1_Tasarla_Uret_Gorev.mp4** video dosyası öğrenciler ile paylaşılmalıdır.

Tanımlama: Öğrenciler istenilen programı gerçekleştirebilmek için yapılması gerekenleri tanımlamalı ve işlem adımlarını maddeler halinde sırlamalıdırlar.

Örneğin:

- Program çalıştırıldığında robotun konumu temel alınacak. Robot göreceli bir hareket gerçekleştirecek.
- Hedef kolu sağa veya sola el ile çevrilerek ucu bir hedefi gösterecek şekilde ayarlanacak.
- D portuna bağlı motorun dönme miktarı temel alınarak uzaklık belirlenecek.
- C portuna bağlı motorun dönme miktarı temel alınarak hareket hızı belirlenecek
- Robotun sağ veya sol tuşuna basılacak.
- Robot kendi etrafında dönerek yönünü hedefe yöneltecek.
- Belirlenen uzaklık kadar ileriye doğru hareket edecek.
- Belirlenen hareket hızıyla hareket gerçekleşecek.
- Robot hiçbir zaman hedeften uzaklaşacak yönde bir hareket gerçekleştirmeyecek.

Fikir üretme: Öğrenciler tanımlanan görevleri robotun gerçekleştirebilmesi için, hangi kod bloklarının kullanılacağına ve algoritmanın nasıl olması gerektiğine dair fikirler üretmelidirler.

Örneğin:

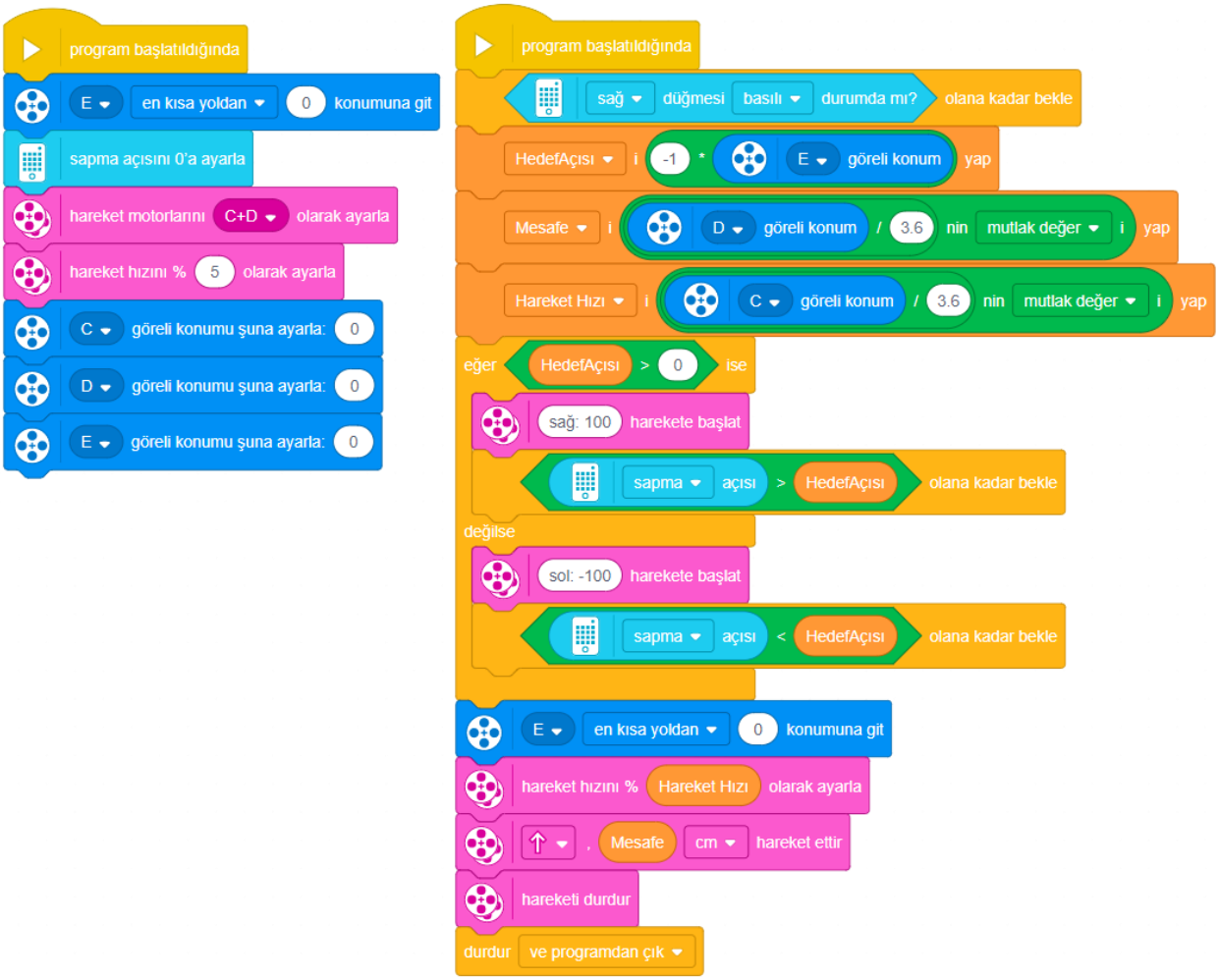
- Hedef kolunun sağa veya sola yönde dönme miktarı nasıl bulunabilir.
- C ve D motorunun dönme miktarları nasıl bulunabilir.
- Bir tur (360 derece) dönme nasıl 100'e eşitlenebilir.
- C ve D motor verileri her koşul pozitif olması nasıl sağlanabilir.
- Robotun sağ veya sol tuşuna basılıncaya kadar beklemesini nasıl sağlanabilir.
- Robotun kendi etrafında dönme miktarı ile hedef kolunun dönme miktarının eşit olması nasıl sağlanabilir.
- Robotun gideceği mesafe nasıl D motorunun dönme miktarı ile belirlenebilir.

- Robotun hızı nasıl C motorunun dönme miktarı ile belirlenebilir.

Öğrenciler verilen probleme farklı çözümler üretebilirler. Tüm grupların aynı veya benzer programı hazırlamaları zorunlu değildir. Rehber öğretmenler öğrencilerin farklı bakış açılarını desteklemeli, onları farklı çözümler üretmeye cesaretlendirmelidir.

ÜRET

Öğrenciler problemin çözümüne yönelik ürettikleri fikirleri bilgisayar ve robotlarla çalışarak test ederler ve istenilen görevleri gerçekleştiren robot programını hazırlarlar. Problemin çözümüne yönelik örnek program Resim 3.22'de sunulmuştur.



Resim 3.22 Hedefe Kilitlen Örnek Programı

Rehber Öğretmen İçeriği – Hedefe Kilitlen Etkinliği Açıklaması

Bu etkinlikte hedef yön kolu ile belirlenen yön ve açı değeri kadar, robotun kendi etrafında dönmesi sağlanacaktır. Hedef yön kolunun program başladığı anda sıfır (0) konumuna alınması

önemlidir. Böylece hedef yön kolunun ne kadar döndüğü daha kolay tespit edilebilir. Temel olarak hedef yön kolu ile robota kaç derece döneceği verisinin girişi yapılmaktadır.

Sürüş modeli tasarımına göre, büyük motorun (E girişine takılı) ters durmasından dolayı, hedef yön kolunun saat ibresi yönünde döndürülmesi büyük motorun göreli konumunu negatif yönde artırırken, aynı yönde robotun döndürülmesi sapma açısını pozitif yönde artırmaktadır. Bu nedenle örnek programda büyük motorun göreli konumu eksi bir (-1) ile çarpılmıştır.

C ve D motorlarının göreli konumları (her turda, 360 derece sonrası tekrar 0 derece olmasından dolayı normal konum kullanılmamıştır. Göreli konum bloğuna "Eklentiler" bölümünden "Daha Fazla Motor" eklentisi aktif edilerek ulaşılabilir) 3.6 değerine bölünerek bir tur dönmenin 100 değerine eşitlenmesi sağlanmıştır. Robotun ters yönde hareketini engellemek amacıyla, bu değerlerin mutlak değeri alınarak Mesafe ve Hareket Hızı değişkenlerine atanmıştır.

DEĞERLENDİR

Rehber öğretmen Resim 3.23'te verilen programı her grubun robotlarına yükleyerek çalıştırmalarını ister. Program robot sallandığında rastgele bir sayı üretip bu sayıyı ekranda 3 saniye göstermektedir. Grup sayısına göre 1- 10 aralığı değiştirilebilir.

Resim 3.23 Sallandığında Rastgele Sayı Üret Programı

Her bir gruba bir sayı verilir. Herhangi bir grup robotlarını sallayarak, cevap verecek grubu seçer. Cevap veren grup robotunu sallayarak bir sonraki grubu seçer. Tüm sorular cevaplanıncaya kadar değerlendir etkinliğine bu şekilde devam edilir.

Rehber öğretmen gruplara aşağıdaki ve benzeri soruları yöneltir.

- Robot setinde bulunan kuvvet sensörünün özellikleri nelerdir?
- Bu gün öğrendiğiniz kod blokları hangileridir? Görevleri nelerdir?
- Kuvvetin nesneler üzerinde nasıl bir etkisi olabilir? Örnekler veriniz?
- Yunuslama hareketi nedir? Açıklayınız.
- Yuvarlanma hareketi nedir? Açıklayınız.
- Sapma hareketi nedir? Açıklayınız.
- Hub içesinde bulunan hareket sensörü ne amaçla kullanılır?
- Hub içesinde bulunan hareket sensörünün özellikleri nelerdir?

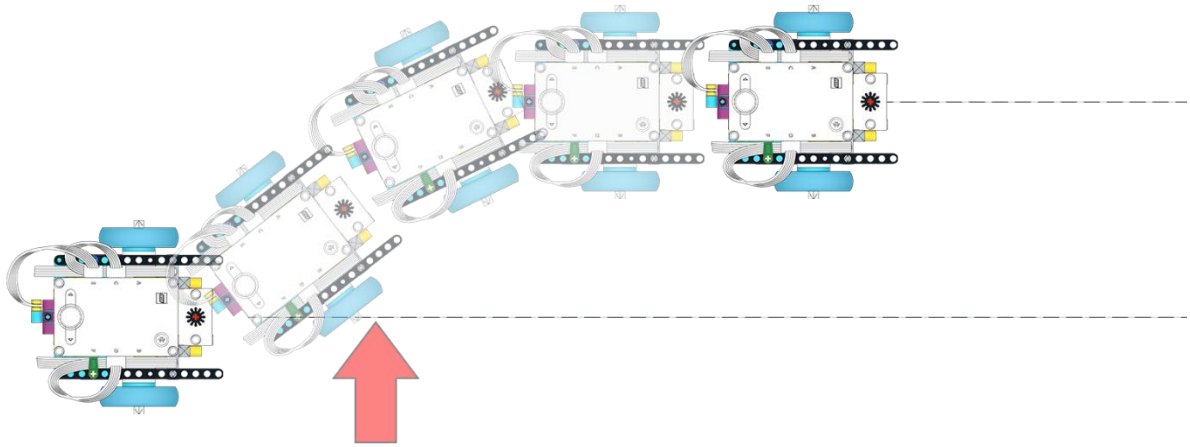
- Hareket sensörünün günlük hayattaki kullanımlarına örnekler veriniz.
- Bu gün öğrendiğiniz kod blokları hangileridir? Görevleri nelerdir?
- Sizce bu haftanın en zor görevi hangisiydi? Zorluğun üstesinden nasıl geldiniz?

İLAVE ETKİNLİK

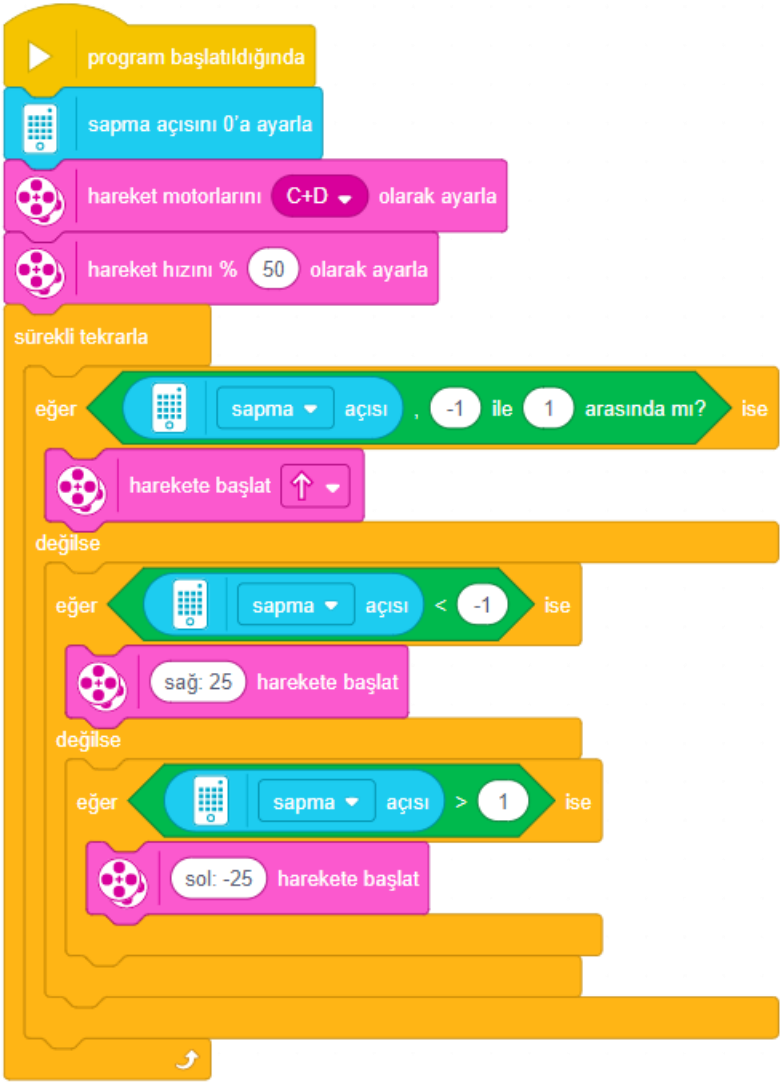
Yönünü Kaybetmeyen Robot

Bu etkinlikte öğrencilerden doğrusal olarak ilerleyen robotun dışarıdan yapılacak bir müdahale sonrasında tekrar önceki doğrultusuna dönmesini sağlayan bir robot tasarlamaları istenir.

Öğrencilerden robotlarını doğrusal ilerleyecek şekilde programlamaları istenir. Fakat doğrusal ilerlerken yönünü sağa veya sola döndürecek bir müdahale sonrasında ilerlediği doğrultuyu tekrar başlangıç doğrultusuna ayarlayabilecek bir robot olması gerektiği belirtilir. Örneğin robot başlangıçta kuzeye doğru doğrusal ilerlerken, rehber öğretmenin müdahalesi ile yönü doğuya çevrilen robotun tekrar yönünü kuzeye çevirmesi gerektiği anlatılır. Resim 3.24'te robotun beklenen hareketi gösterilmiştir. Resim 3.25'te örnek program sunulmuştur.



Resim 3.24 Yönünü Kaybetmeyen Robot



Resim 3.25 Yönünü Kaybetmeyen Robot Örnek Programı

4. Hafta – Renk Sensörü ve Sensörlerin Birlikte Kullanımı

Haftanın Amacı:

Bu haftanın amacı öğrencilere Lego Spike Prime robot setinde yer alan renk sensörünü tanıtmak ve önceki haftalarda öğrenilen diğer sensörler ile eş zamanlı kullanmalarını sağlayacak temel bilgi ve becerileri kazandırmaktır. Öğrenciler renk sensörünün renk okuma ve ışık miktarını ölçme özelliklerini farklı programlarda kullanacaklardır. Bu hafta etkinliklerde öğrenciler birden fazla sensörü eş zamanlı kullanırken "*Ve*", "*Veya*" ve "*Değil*" mantık operatörlerini kullanmaları hedeflenmiştir.

Haftanın Kazanımları:

- Öğrenciler renk sensörünün nasıl çalıştığını ifade edebilirler.
- Öğrenciler renk sensörünün renk algılama ve yansıyan ışığı ölçme özelliklerinin programlarında kullanabilirler.
- Öğrenciler algıladığı renge göre farklı işlemler yapacak robotu programlayabilirler.
- Öğrenciler farklı sensörleri bir arada kullanabilirler.
- Öğrenciler farklı sensörlerden alınan verileri kullanarak çözümler üretebilirler.
- Öğrenciler "*Ve*", "*Veya*" ve "*Değil*" mantık operatörlerini kullanabilirler.
- Öğrenciler mesaj bloklarını kullanarak program akışını düzenleyebilirler.

Kullanılacak Malzemeler:

Robot seti, bilgisayar ve çalışma alanı (mat).

Ekler:

Konu anlatımında ekran görüntüleri verilen örnek programlar, ayrıca dijital formatta rehber öğretmenlere sunulmuştur. Örnek program dosyalarının numaralandırılmasında, konu içerisindeki resim numaraları temel alınmıştır. Bu hafta aşağıda sıralanan programlar, tasarla üret etkinliği için bir video ve çalışma alanı (mat) dosyaları ekte rehber öğretmenlere sunulmuştur. Rehber öğretmen A3 ebatında çıktı alma imkânına sahip ise her grup için bir tane Mat_4_A3.pdf dosyasının, değilse her grup için iki tane Mat_4_A4.pdf dosyasının çıktısını alarak etkinlik öncesinde hazırlamalıdır.

Program_4.3_Kirmizi_Renk_mi.llsp3
Program_4.4_Kirmizi_Renk_Degil_mi.llsp3
Program_4.6_Rengin_Numarasini_Hub_Ekraninda_Goster.llsp3
Program_4.10_Belirli_Bir_Alandan_Cikmayan_Robot.llsp3
Program_4.13_Veya_Blogu.llsp3
Program_4.14_Cizgi_Takibi.llsp3
Program_4.15_Ve_Blogu.llsp3
Program_4.23_Tasarla_Uret_Hedef_Tespit_Kilitlen.llsp3
Video_4.1_Tasarla_Uret_Gorev.mp4
Mat_4_A3.pdf
Mat_4_A4.pdf

Renk Sensörü ve Sensörlerin Birlikte Kullanımı

Rehber Öğretmen İçeriği – Renk Sensörü ve Operatör Blokları

Renk sensörü, rengi, yansımayı veya ortam ışığını algılayabilir. Sensör ayrıca bir ışık kaynağı olarak da kullanılabilir. Sensörün optimum okuma mesafesi 16 mm'dir (nesne boyutuna, rengine ve yüzeyine bağlı olarak değişebilir).

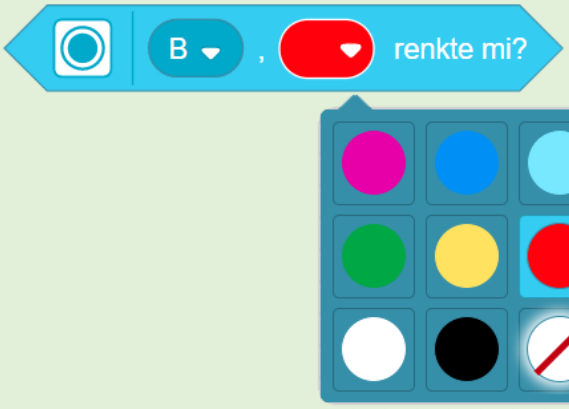
Renk sensörü, renk okuma özelliği kullanıldığında 8 renk (siyah (0), mor (1), mavi (3), açık mavi (4), yeşil (5), sarı (7), kırmızı (9), beyaz (10)) ve renksiz (-1) değerlerini ölçebilmektedir. Ayrıca, yansıyan ışık algılama özelliği sayesinde "0" (yansıtmaz) ile "100" (çok yansıtıcı) değerleri arasında ölçüm yapabilmektedir. Yansıyan ışık yüzdesi sensör üzerindeki 3 LED ile sağlanan beyaz ışık kullanılarak ölçülmektedir.

Renk sensörünün teknik özellikler kitapçığında ortam ışığı yoğunluğunun yüzde olarak ölçülebildiği belirtilmiştir. Ortam ışığı yoğunluğu kullanıldığında renk sensörü herhangi bir ışık göndermez. Doğrudan bulunduğu ortamdan gelen ışığın yoğunluğu ile ilgilenir. Bu yoğunluk değeri yine 0 (en karanlık) ile "100" (en aydınlık) arasındadır. Fakat Spike Prime yazılımında ortam ışığı ile ilgili sözcük bloğu henüz bulunmamaktadır. Ortam ışığı ölçümü Python modunda yapılabilmektedir.

Bu hafta etkinliklerde kullanılacak sözcük blokları ve açıklamaları aşağıda verilmiştir.

Sensörler Blok Paleti

Şu renkte mi?



Siyah – 0
Mor – 1
Mavi – 3
Açık Mavi – 4
Yeşil – 5
Sarı – 7
Kırmızı – 9
Beyaz – 10
Renk yok – -1

Bu "Boole Bloğu" renk sensöründen algılanan renk ile seçilen rengin aynı olması durumunda doğru sonucunu verir. Diğer durumlarda yanlış sonucunu verir.

Renk

Renk sensörü tarafından tespit edilen renk değerini almak için kullanılır. Sensör tarafından tespit edilebilecek renkler ve sayısal değerleri şu şekildedir; siyah (0), mor (1), mavi (3), açık mavi (4), yeşil (5), sarı (7), kırmızı (9), beyaz (10) ve renksiz (-1).

Yansıyan ışık şu mu?

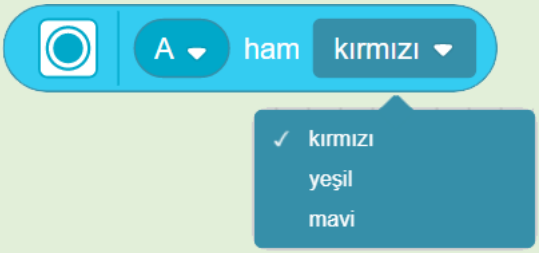
Renk sensörüne geri yansıyan ışığın belirtilen yüzdeden küçük, büyük veya eşit olması durumunda doğru sonucunu verir.

Yansıyan ışık

Renk sensörüne geri yansıyan ışığın mevcut değerini verir.

Daha Fazla Sensör Blok Paleti

Ham renk



Renk sensöründen ham kırmızı, yeşil ya da mavi renk okuması değerini verir. Renk değer aralığı 0-255 olarak ölçülebilir.

Operatörler Blok Paleti

Ve

Bu blok, iki Boole bloğunu Ve (And) işlemi ile birleştirir. Her iki koşulunda doğru olması durumunda Ve işleminin sonucu doğru olur. Aksi takdirde sonuç yanlıştır.

Veya

Bu blok, iki Boole bloğunu Veya (Or) işlemi ile birleştirir. Koşullardan herhangi birinin doğru olması durumunda Veya işleminin sonucu doğru olur. Sonuç yalnızca her iki koşulun da yanlış olması durumunda yanlış olur.

Değil

Bu blok içerisinde yer alan Boole koşulunun tersini verir. Koşul yanlış ise sonuç doğru, doğru ise sonuç yanlıştır.

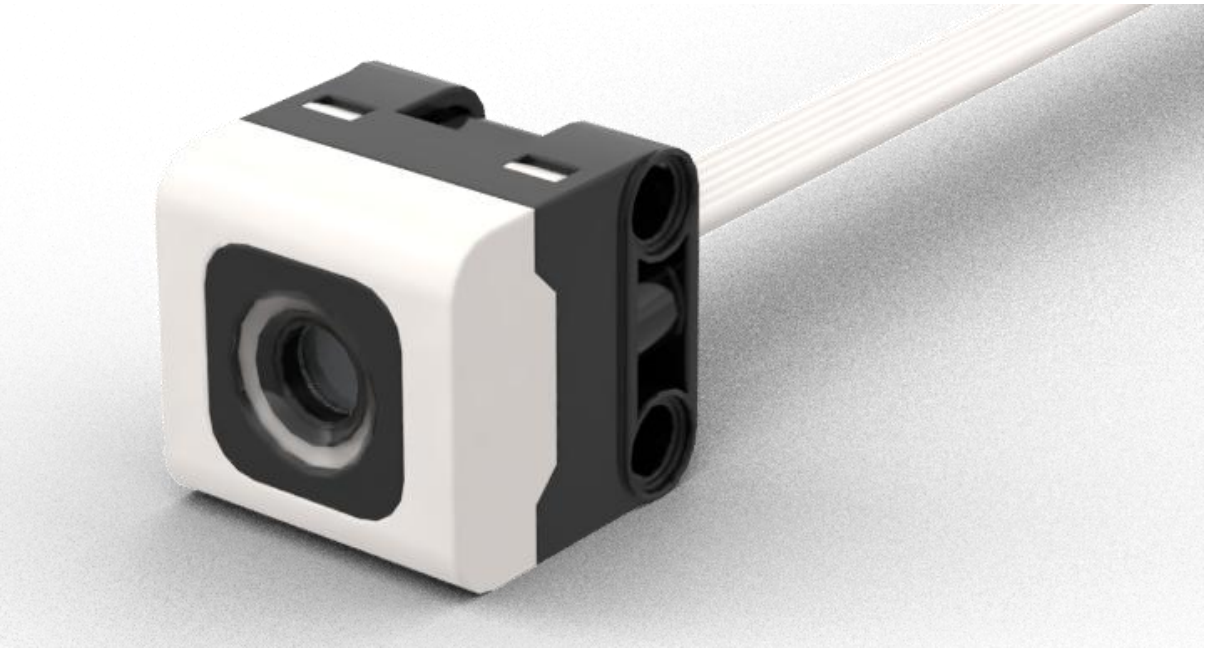
Not

Etkinliklerde hazırlanması planlanan programların ekran görüntüleri, ilgili konu içerisinde sunulmuştur. Ayrıca program dosyaları da rehber öğretmenlerle paylaşılmıştır. Bu paylaşımlar rehber öğretmenlerin etkinliklere hazırlanmasında kolaylık sağlamak içindir. Gözle etkinliklerinde rehber öğretmen ilgili programları öğrencilere açıklayarak adım adım hazırlamalıdır. Tasarla ve üret etkinliklerinde ise rehber öğretmen programları ve ekran görüntülerini öğrencilerle kesinlikle ***paylaşmamalıdır.***

GÖZLE VE UYGULA

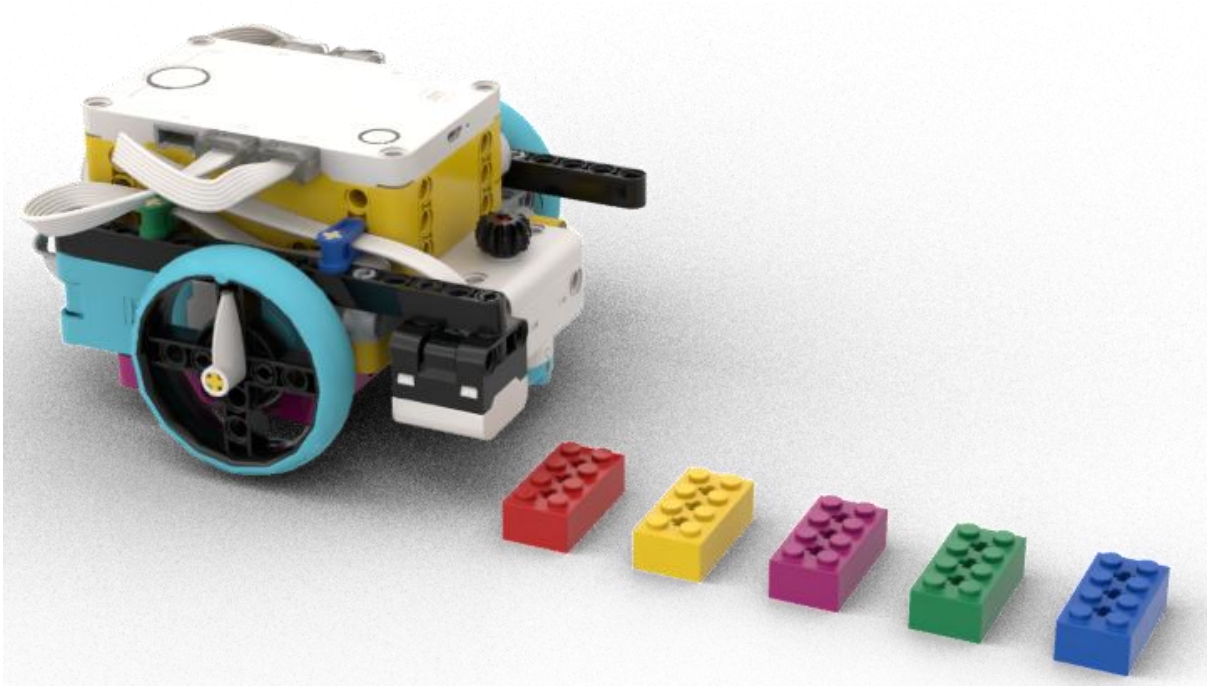
Gözle ve Uygula: Lego Bloğu Kırmızı Renk mi?

Rehber öğretmen bu hafta renk sensörü ve önceki haftalarda öğrenilen diğer sensörlerin birlikte kullanımına yönelik etkinlikler yapacaklarını belirtir. Spike Prime robot setinde yer alan renk sensörünü (Resim 4.1) öğrencilere tanıtır ve renk sensörünün özelliklerini sıralar.



Resim 4.1 Renk Sensörü

Rehber öğretmen, öğrencilerden renk sensörünü Resim 4.2'de gösterildiği gibi Sürüş Modeli tasarımına eklemelerini ve renk sensörünü Hub'ın B portuna bağlamalarını ister.



Resim 4.2 Kırmızı Renk Mi?

Rehber öğretmen, öğrencilerden Resim 4.3'te verilen programı hazırlayıp, Resim 4.2'de görüldüğü gibi farklı renklerdeki Lego bloklarına robotun verdiği tepkiyi gözlemlemelerini ister.

Resim 4.3 Kırmızı Renk Mi Örnek Programı

Gözle ve Uygula: Değil (Not) Mantık Operatörü

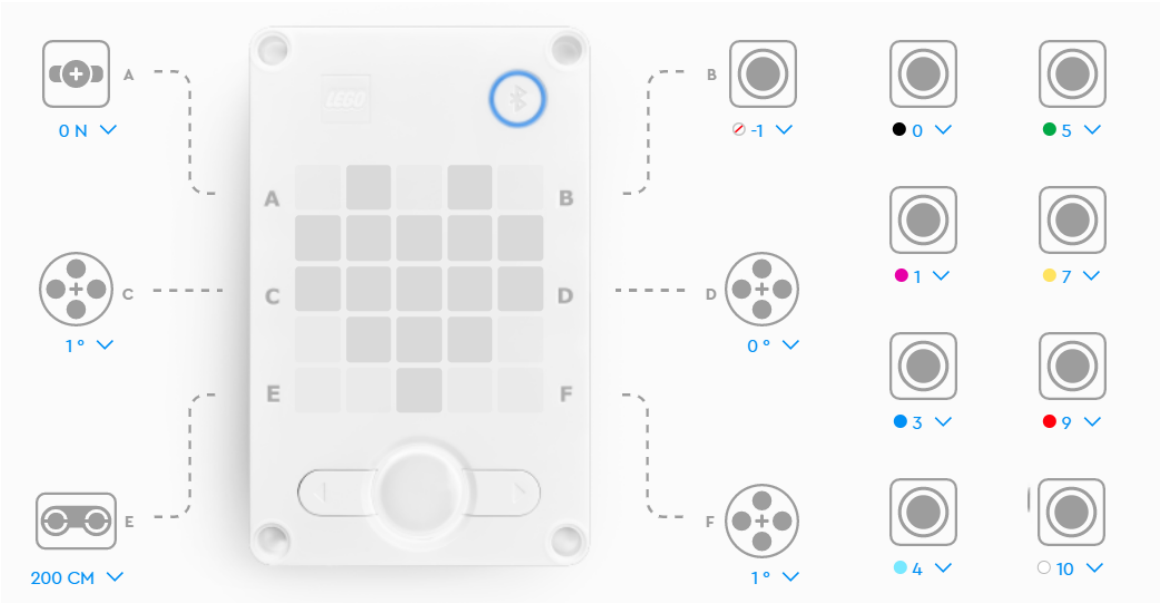
Sonrasında rehber öğretmen Resim 4.4'te gösterildiği gibi "*şu renkte mi*" bloğunu "*değil*" bloğu içerisine alarak aynı programı çalıştırmalarını ve farklı renklerdeki Lego blokları ile programı tekrar test etmelerini ister. Öğrencilerden iki programın sonucunu karşılaştırarak "*değil*" bloğunun görevini keşfetmeleri istenir. İhtiyaç hissedilmesi durumunda, öğrencilerin çıkarımları sonrasında rehber öğretmen, "*değil*" bloğunu ve işlevini özetler.

Resim 4.4 Kırmız Renk Değil Mi Örnek Programı

Gözle ve Uygula: Lego Bloğunun Renk Numarasını Hub Ekranında Gösterme

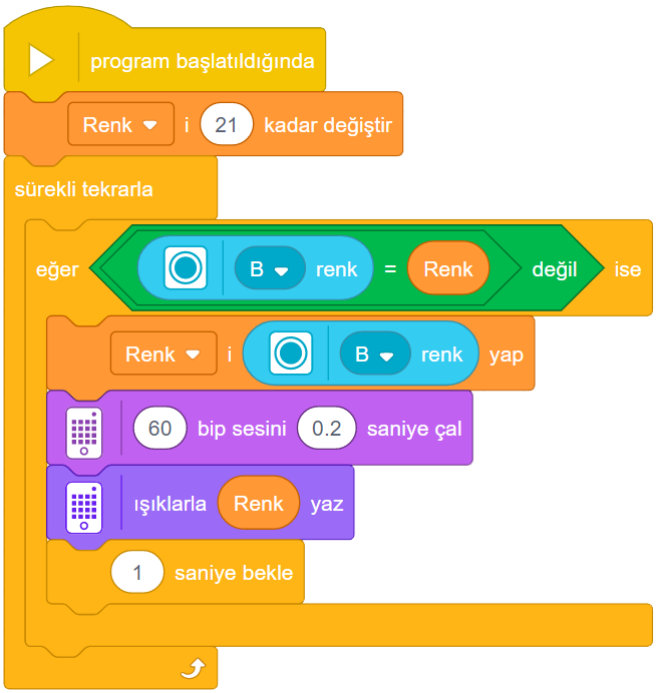
Bir önceki uygulamada robotun renk sensörü ile algılanan rengin, belirlen bir renk ile (kırmızı) kıyaslaması yapılmıştı. Bu uygulamada Resim 4.2'de gösterildiği gibi, renk sensörüne gösterilen Lego bloklarının renk numaralarının Hub ekranında gösterilmesini sağlayan bir program hazırlanacaktır. Robot **farklı bir renk algıladığında** bir ses çıkaracak ve algılanan rengin numarasını Hub ekranında gösterecektir. Renk değişmediği sürece ekranda bir önceki ölçüm değeri görünme devam edecektir (-1 ve 10 hariç, bu iki değer tek seferde ekrana sığmadığında bir kez görülmektedir).

Renk sensörü tarafından algılanan renklerin kodları siyah (0), mor (1), mavi (3), açık mavi (4), yeşil (5), sarı (7), kırmızı (9), beyaz (10) ve renksiz (-1) şeklindedir. Ayrıca Hub bağlantısı ekranında renk sensörü ile ölçülen renkler ve numaraları anlık olarak gösterilmektedir. Resim 4.5'te Hub bağlantısı ekranında farklı renklerdeki Lego blokları ile ölçülen renk değerleri gösterilmiştir. Öğrenciler Hub bağlantısı ekranında robotları bilgisayara bağlı iken, renk sensörüne farklı renklerdeki nesneleri yaklaştırarak renklerin kodlarını keşfedebilirler.



Resim 4.5 Renk Kodları

Rehber öğretmen, Resim 4.6'da ekran görüntüsü sunulan programı adım adım hazırlayarak öğrencilere anlatır. Sonrasında öğrencilerin programı hazırlayarak uygulamalarını ister.



Resim 4.6 Rengin Numarasını Hub Ekranında Gösteren Örnek Program

Rehber Öğretmen İçeriği – Rengin Numarasını Hub Ekranında Gösteren Program Açıklaması

Spike Prime uygulamasında Sözcük Blokları ile uygulama ortamında, koşul belirten yalnızca "Küçüktür", "Eşittir" ve "Büyüktür" operatör blokları mevcuttur. Eşit değildir koşulu için bir blok bulunmamaktadır. Eşit değil koşulu, "Eşittir" bloğu "Değil" bloğunun içerisini yerleştirilerek sağlanır. Ayrıca "eğer öyle değilse şunu yap" bloğu ile koşulun sağlanması ve sağlanmaması durumunda görevler tanımlanabildiğinden, bu bloğun değilse bölümü de kullanılabilir.

Programda renk sensörünün önündeki renk değiştirildiğinde Hub'ın ekranında sadece bir defa renk kodunun yazılması ve bip sesi çıkarılması için Renk adlı bir değişken kullanılmıştır. İlk renkte programın çalışabilmesi için program başladığında Renk değişkenine renk kodlarında olmayan bir değer atanmıştır (örneğin 21). Renk sensörünün önüne her farklı renk geldiğinde programın bir defa çalışması için yeni okunan rengin değeri Renk değişkenine atanmıştır. Bu sayede sensörün önündeki renk değişmediği sürece programın bip sesi çıkarmaması ve önündeki rengin kodunu Hub ekranına yazmaya devam etmesi sağlanmıştır.

Gözle: Yansıyan Işık Şiddeti

Rehber öğretmen öğrencilere renk sensörünün ışık şiddetini yansıyan ışık şiddeti ve ortam ışığı şiddeti olarak da ölçebildiğini belirtir. Yansıyan ışık şiddetini ölçmek için renk sensörünün karşısındaki yüzeye beyaz bir ışık gönderdiğini ve o yüzeyden yansıyan ışığın şiddetini "0" ile "100" arasında bir değer (yüzde) olarak ölçtüğünü anlatır. "0" en karanlık, "100" ise en aydınlık değeri ifade eder. Ölçülen değer sıfıra ne kadar yakınsa yüzeyden yansıyan ışık o kadar azdır, yani

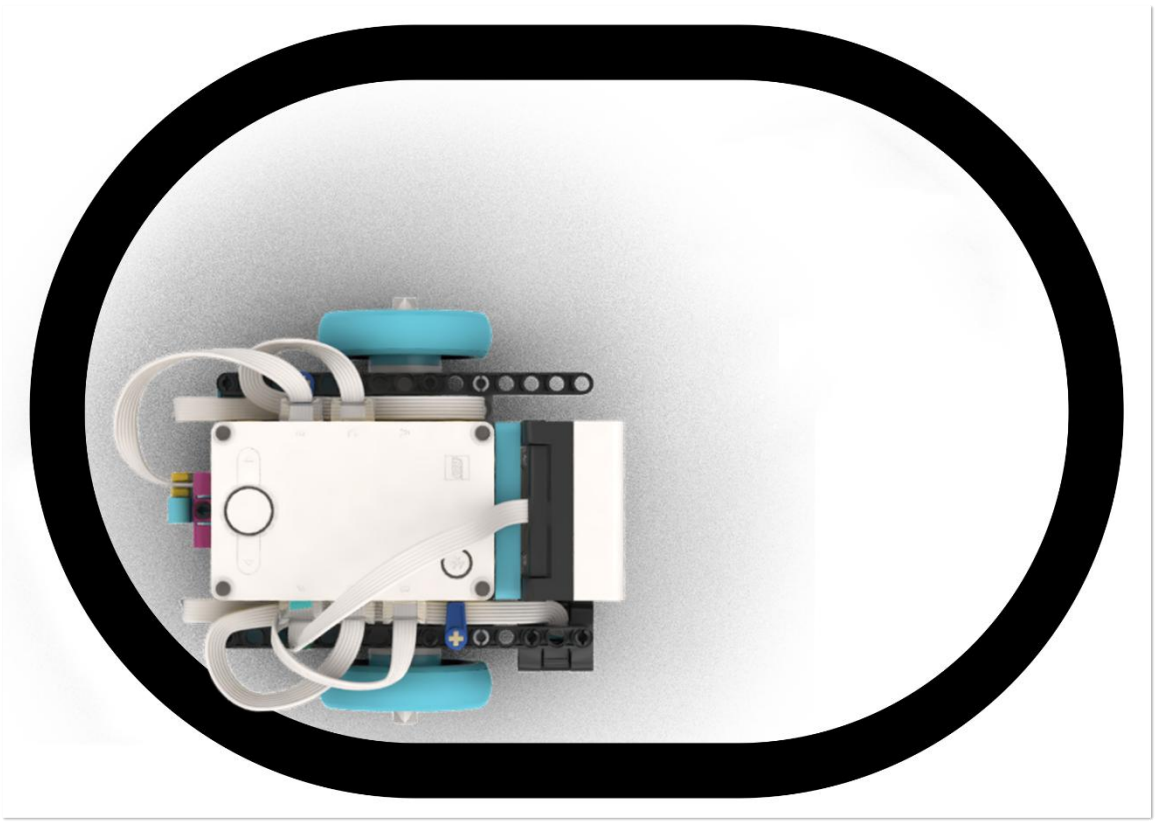
karanlıktır. Ölçülen değer 100'e ne kadar yakınsa yüzeyden yansıyan ışık o kadar fazladır, yani aydınlıktır.

Yansıyan ışık ile işlem yaparken *"yansıyan ışık"* ve *"yansıyan ışık şu mu"* blokları kullanılabilir. *"yansıyan ışık"* bloğu geri yansıyan ışığın değerini renk sensörüne bildirir. *"Yansıyan ışık şu mu"* bloğu ise renk sensörüne geri yansıyan ışık belirtilen yüzdeden büyük, ona eşit veya daha az olduğunda "*doğru*" değerini döndürür.

Uygula: Belirli Bir Alandan Çıkmayan Robot

Rehber öğretmen etkinlik öncesinde çıktısını aldığı çalışma alanlarını (1 adet Mat_4_A3.pdf veya 2 adet Mat_4_A4.pdf) gruplara dağıtır. Robotun Resim 4.7'de görüldüğü gibi siyah alan içerisine yerleştirileceğini ve ileri doğru hareket ederken siyah çizgiye geldiğinde yön değiştirerek alan dışına çıkmadan sürekli hareketine devam edeceğini belirtir. Öğrenciler öncelikli olarak Hub bağlantısı ekranında siyah ve beyaz zeminlerden yansıyan ışık miktarını ölçerek, robotun yön değiştirmesi için kullanılacak kritik değere karar vermelidirler. Rehber öğretmen programın nasıl çalışması gerektiğini öğrencilerle birlikte belirlemeye çalışır. Alan içerisine bırakılan robotun alanın dışına çıkmadan hareket edebilmesi için;

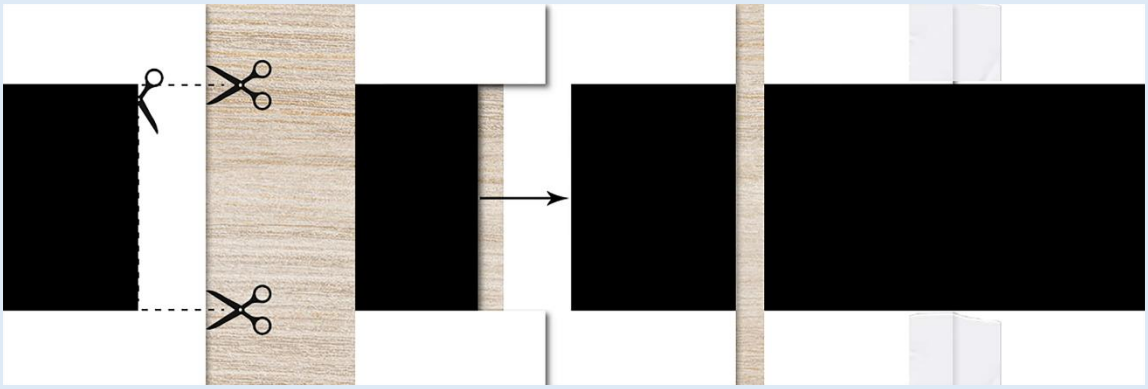
i. yansıyan ışık miktarı azalıncaya kadar (siyah çizgiye kadar) robotun düz ilerlemesi,
ii. siyah renk algıladığında robotun dönmesi,
iii. dönme öncesinde, robot mevcut ivme ile siyah alanı aşmaması için hızının düşük olması,
iv. bu işlemin sürekli tekrar edilmesi gerekir. (Örnek program Resim 4.10'da sunulmuştur)



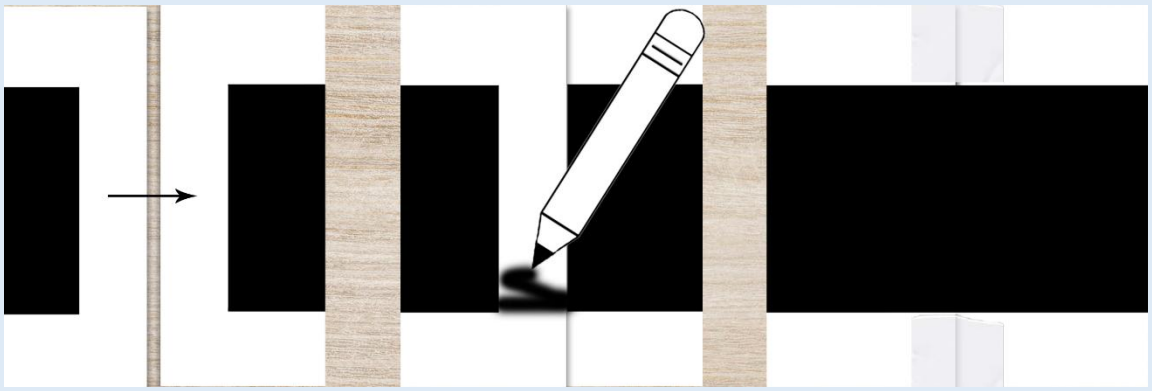
Resim 4.7 Siyah Alandan Çıkmayan Robot Çalışma Alanı

Not

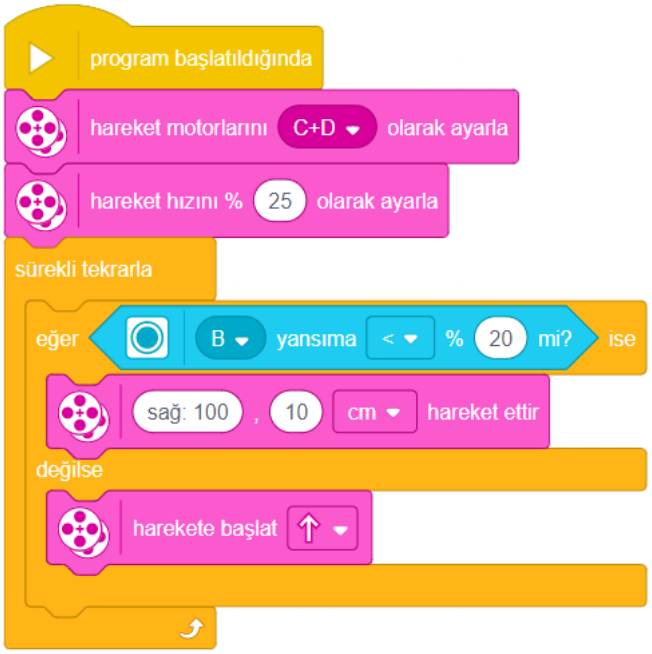
Çalışma alanını iki adet A4 ebatlarındaki çıktıların birleştirilmesi ile oluşturulduğunda, baskı esnasında oluşan kenar boşlukları birleştirilen kenarlardan biri makas ile kesilerek veya siyah kurşun kalemle boyanarak elips şeklindeki rotanın kesintisiz devam etmesi sağlanabilir. Ayrıca iki A4 kâğıdı yapıştırılarak etkinlik esnasında rotanın bozulması önlenebilir. Resim 4.8'de çalışma alanının kenarları kesilerek birleştirme, Resim 4.9'da çalışma alanının kurşun kalemle boyanarak birleştirme adımları gösterilmiştir.



Resim 4.8 Çalışma Alanı Kesilerek Birleştirme Adımları



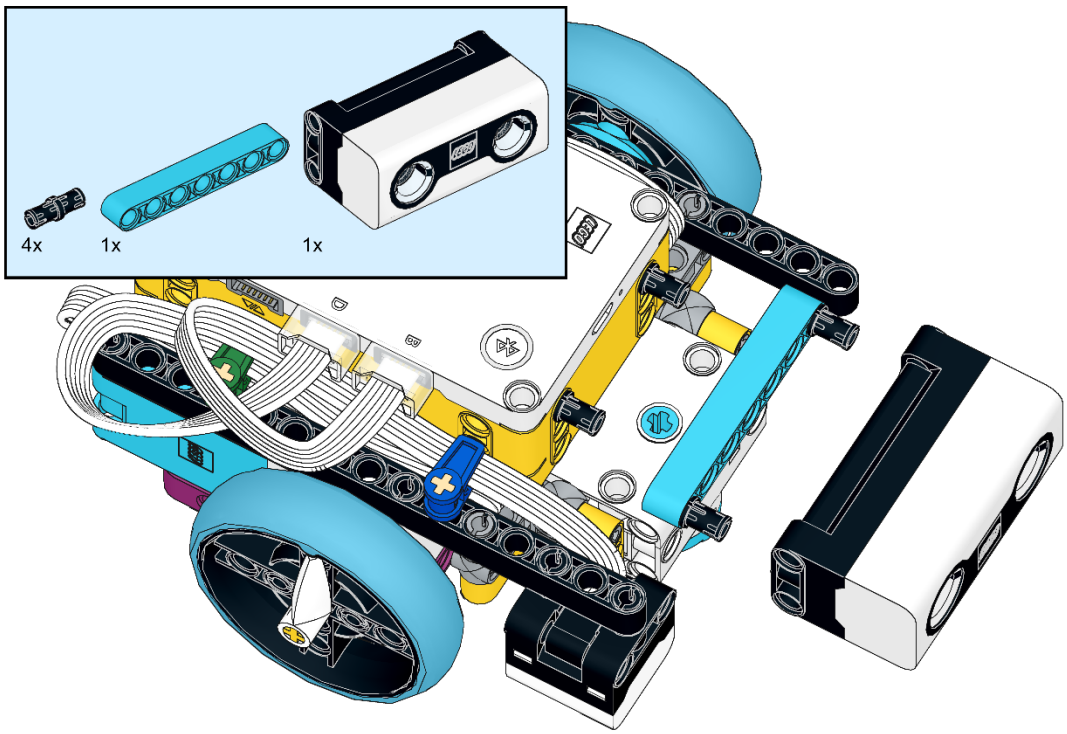
Resim 4.9 Çalışma Alanı Boyanarak Birleştirme Adımları



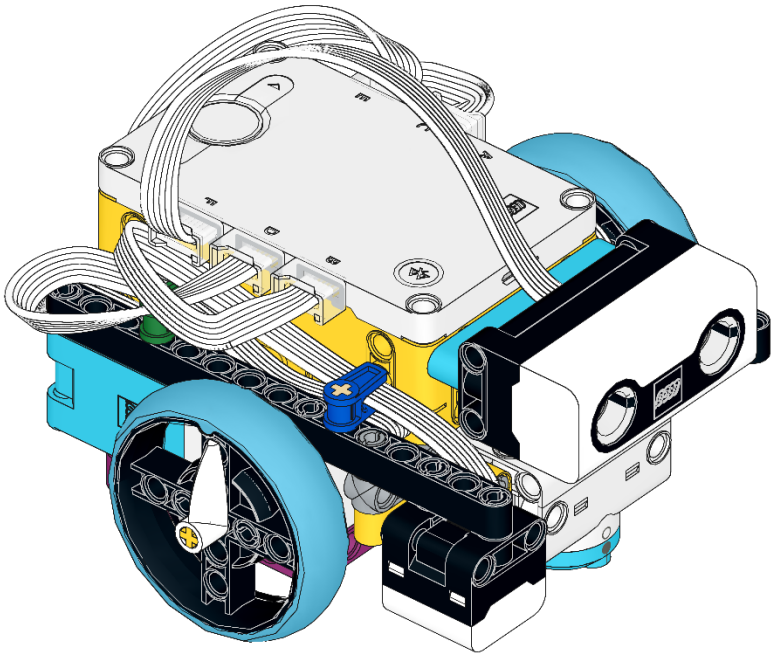
Resim 4.10 Belirli Bir Alandan Çıkmayan Robot Örnek Programı

Gözle ve Uygula: Veya (Or) Mantık Operatörü

Rehber öğretmen öğrencilerden renk sensörü takılı olan Sürüş Modeli tasarımına Resim 4.11 ve Resim 4.12'de gösterildiği gibi mesafe sensörünü kablosunun sıkışmaması için baş aşağı (Lego yazısı ters olacak şekilde) birleştirerek kablosunu F portuna takmalarını ister.



Resim 4.11 Mesafe Sensörünün Takılması Birinci Adım

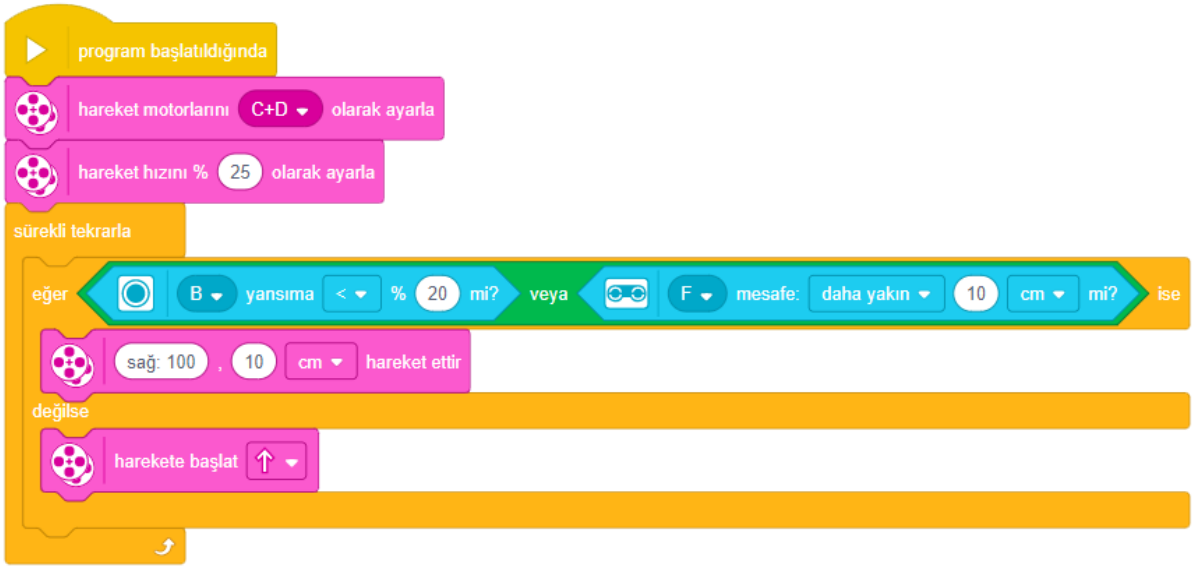


Resim 4.12 Mesafe Sensörünün Takılması İkinci Adım

Rehber öğretmen, bir önceki uygulamaya robotun bir engel ile karşılaşması durumunda koşulunun nasıl eklenebileceğini öğrenciler ile tartışır. Robotlarını siyah renkle belirlenmiş bir alandan dışarı çıkmayacak ve alan içerisindeki bir engele de çarpmayacak şekilde nasıl programlanabileceğini sorar ve öğrencilerin görüşlerini alır.

Rehber öğretmen bir diğer operatör bloğunun "*veya*" bloğu olduğunu belirtir. "*Veya*" bloğunun iki farklı "*Boole Bloğunu*" birleştirdiğini ve herhangi birinin doğru olması durumunda sonucunun doğru, birleştirilen her iki bloğun yanlış olması durumunda sonucunun yanlış olduğunu belirtir. Rehber öğretmen örnek bir cümle ile veya operatörünü somutlaştırır. Örneğin, "Ali **veya** Veli robot yarışmasına katılacak" cümlesinde, Ali'nin robot yarışmasına katılması durumunda, Veli'nin robot yarışmasına katılması durumunda ve Ali ile Veli'nin her ikisinin de robot yarışmasına katılması durumunda cümle doğrudur. Yalnızca Ali ile Veli'nin her ikisinin de robot yarışmasına katılmaması durumunda cümlenin anlamı yanlış olur.

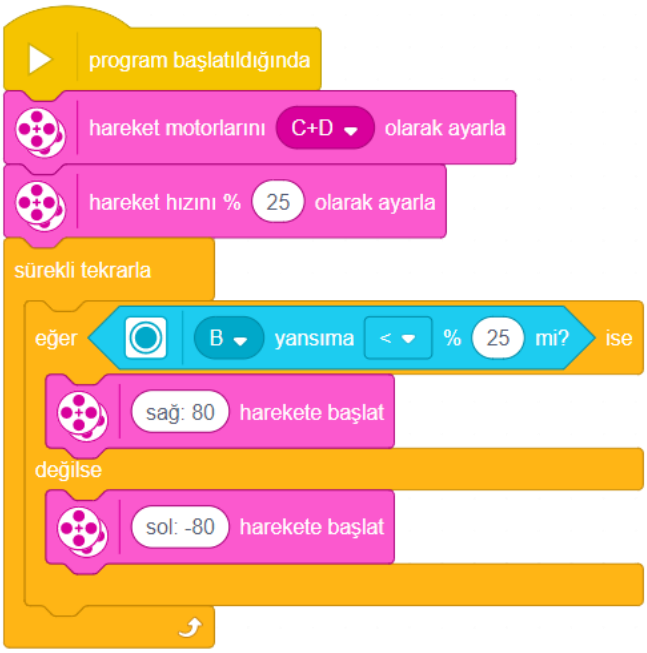
Rehber öğretmen ipucu olarak şu şekilde bir görev tanımı yapabilir; "*robot siyah çizgiyi algıladığında **veya** engele 5 cm'den daha yakın olduğunda yönünü değiştirecek, aksi halde ileri doğru hareketine devam edecek.*" Öğrencilerin Resim 4.13'te verilen programa benzer bir program hazırlamaları beklenmektedir. Sonrasında çalışma alanı üzerinde hazırladıkları programları test ederler. Hazırlanan program çalışırken ellerini mesafe sensörü önüne getirerek engel oluşturabilirler.



Resim 4.13 "Veya" Bloğu Örnek Program

Gözle ve Uygula: Çizgi Takibi

Rehber öğretmen robotu çalışma alanı üzerindeki çizgiyi takip edecek şekilde programlayacaklarını belirtir. Robotun aslında siyah çizginin kendisini değil, siyah ile beyaz alanın sınır çizgisini takip edeceğini ifade eder. Yapılması gereken, robotun renk sensörü siyah alana girdiğinde, robotun beyaz alana; beyaz alana girdiğinde, robotun siyah alana dönmesini sağlamaktır. Böylece robot sürekli alan değiştirerek ilerleyecektir. Resim 4.14'te çalışma alandaki eliptik şekli dış sınır çizgisinden saat ibresinin tersi yönde (veya iç sınır çizgisinden saat ibresi yönünde) takip eden robot programı sunulmuştur. Rehber öğretmen programı adım adım açıklayarak öğrencilerle birlikte hazırlar. Öğrenciler benzer programı hazırlayarak çalışma alanında test ederler.



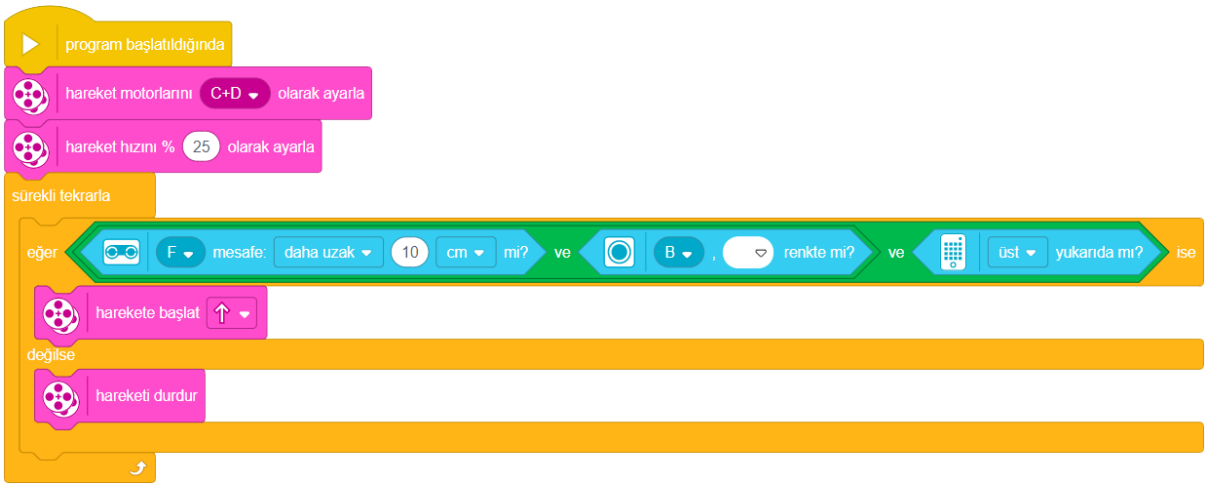
Resim 4.14 Çizgi Takibi Örnek Programı

Sonrasında rehber öğretmen, robotun ters yönde, yani eliptik şekli dış sınır çizgisinden saat ibresi yönünde (veya iç sınır çizgisinden saat ibresinin tersi yönünde) çizgi takibi yapabilmesi için programlarını düzenlemelerini isteyebilir.

Gözle ve Uygula: Ve (And) Mantık Operatörü

Rehber öğretmen "*Veya*" bloğuna benzer şekilde "Ve" bloğunu öğrencilere anlatır. "*Ve*" bloğunun iki farklı "*Boole Bloğunu*" birleştirdiğini ve yalnızca her iki koşulun da doğru olması durumunda sonucunun doğru, diğer durumlarda sonucunun yanlış olduğunu belirtir. Veya operatöründe olduğu gibi örnek bir cümle ile ve operatörünü somutlaştırır. "Ali **ve** Veli robot yarışmasına katılacak" cümlesinde, yalnızca Ali'nin robot yarışmasına katılması durumunda, yalnızca Veli'nin robot yarışmasına katılması durumunda ve Ali ile Veli'nin her ikisinin de robot yarışmasına katılmaması durumunda cümle yanlış olacaktır. Yalnızca Ali ile Veli'nin her ikisinin de robot yarışmasına katılması durumunda cümle anlam bakımından doğru bir cümle olur.

Rehber öğretmen öğrencilerden robotun yalnızca karşısında 10 cm mesafede içerisinde bir engel olmadığında **ve** robot beyaz renkte bir zemindeyken **ve** robot sağa veya sola devrilmemişse (Hub'ın üst yüzü yukarıda ise) ileri doğru hareket etmesini sağlayacak programı hazırlamalarını ister. Bu koşullardan herhangi biri gerçekleşmemiş ise robotun hareket etmemesi gerekmektedir. Bu koşullarda hareket eden robotun örnek programı Resim 4.15'te sunulmuştur.

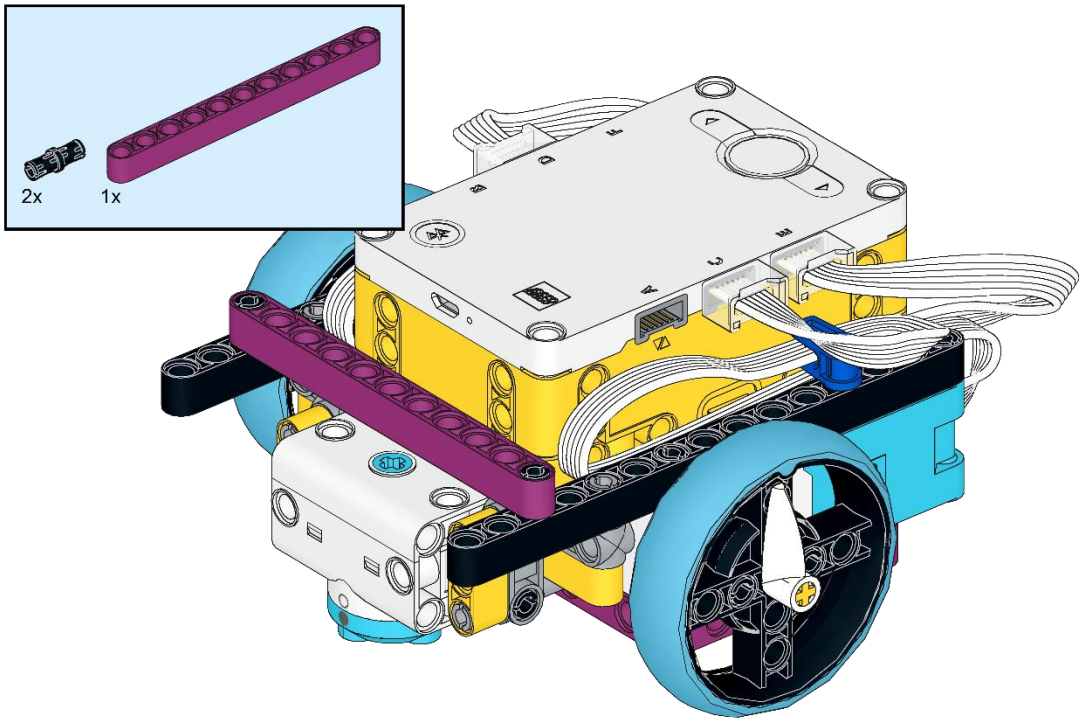


Resim 4.15 "Ve" Bloğu Örnek Program

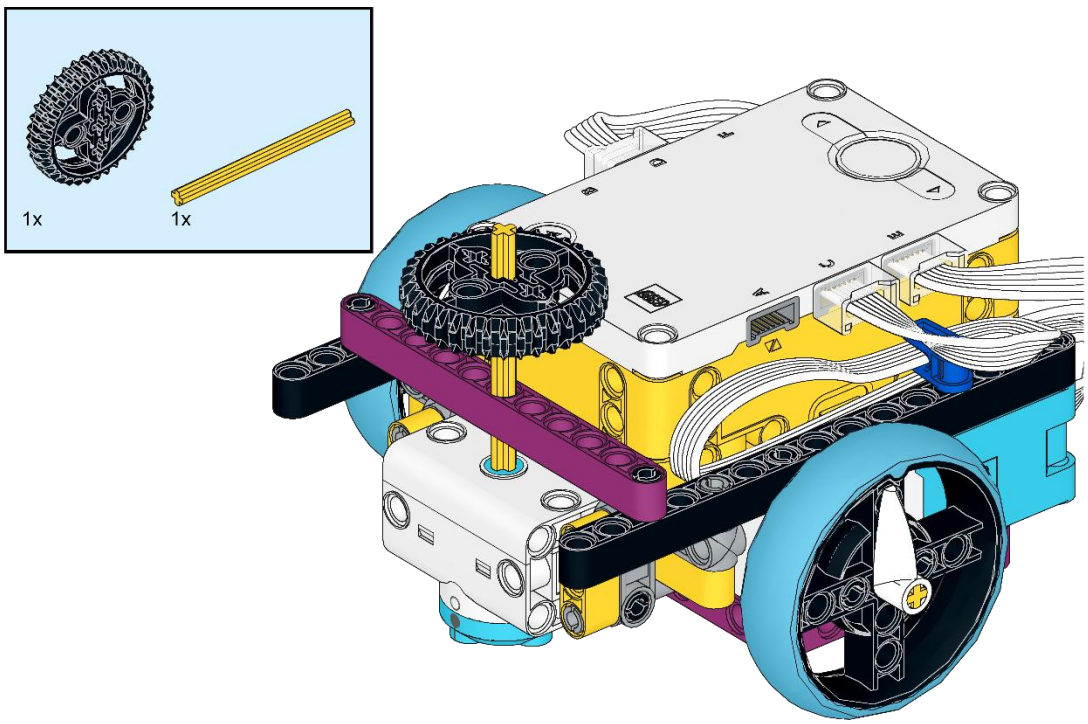
TASARLA

Tasarla etkinliğinde rehber öğretmen öğrencilere 2. haftada yaptıkları hedefe kilitlenen robot etkinliğini hatırlatır. O etkinlikte, robotun hareket sensörünü kullanarak hedef yön kolu ile gösterilen hedefe yöneldiğini hatırlatır. Rehber öğretmen, bu etkinlikte de robotun hedefe yöneleceğini fakat önceki etkinlikten farklı olarak, hedefi kendisinin tespit edeceğini ifade eder.

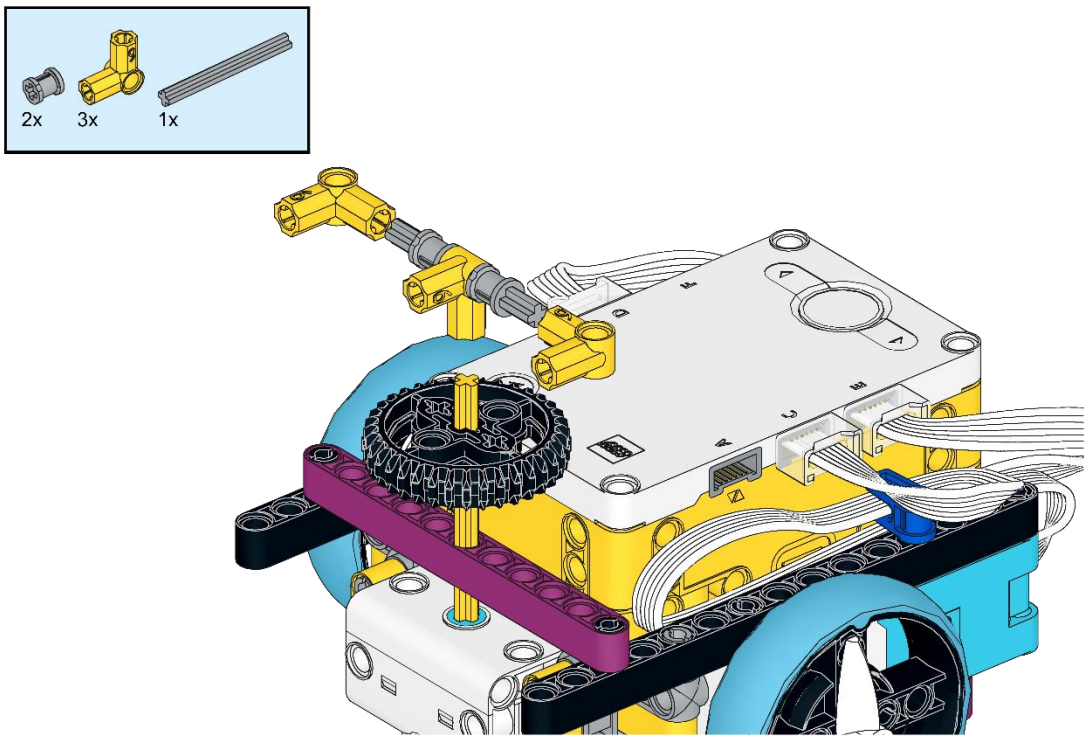
Rehber öğretmen öncelikle takılı olan mesafe sensörünü Resim 4.16–4.21 adımlarında gösterildiği şekilde robotlarına takmalarını ister. Rehber öğretmen mesafe sensörünü takmadan önce büyük motorun Resim 4.16'da olduğu gibi sıfır konumunda olması gerektiğini vurgular.



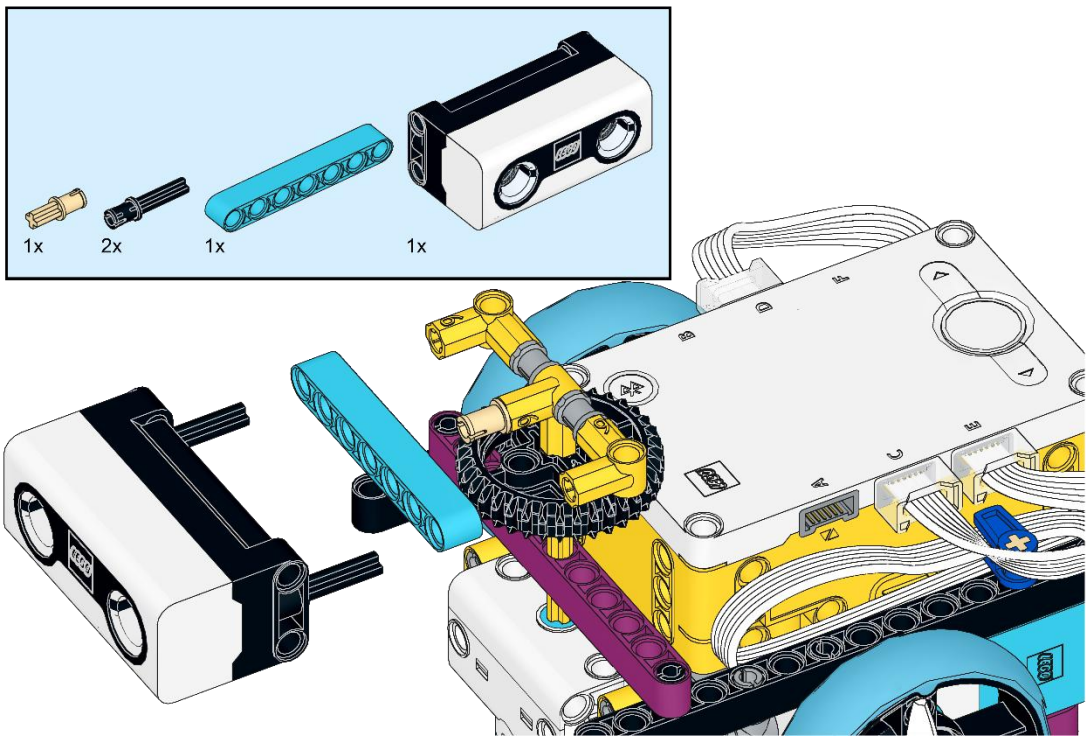
Resim 4.16 Hedef Tespit Robotu Birinci Adım



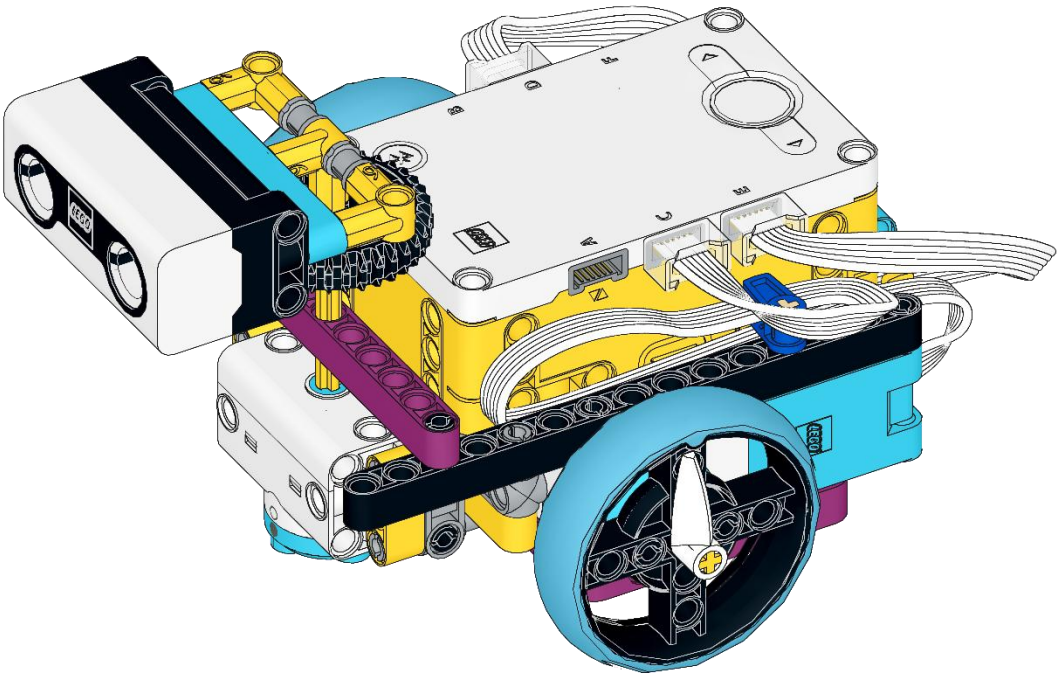
Resim 4.17 Hedef Tespit Robotu İkinci Adım



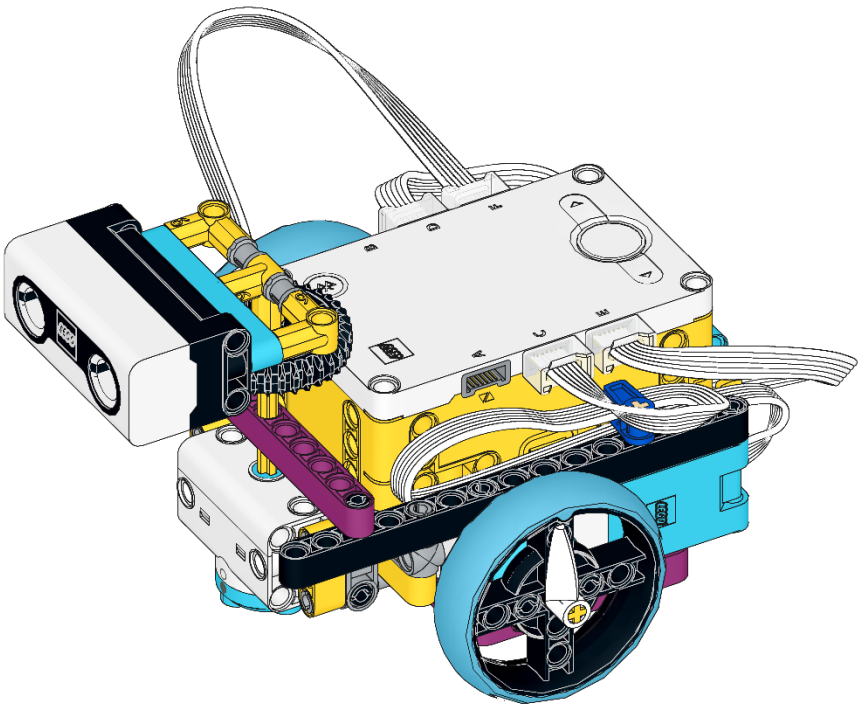
Resim 4.18 Hedef Tespit Robotu Üçüncü Adım



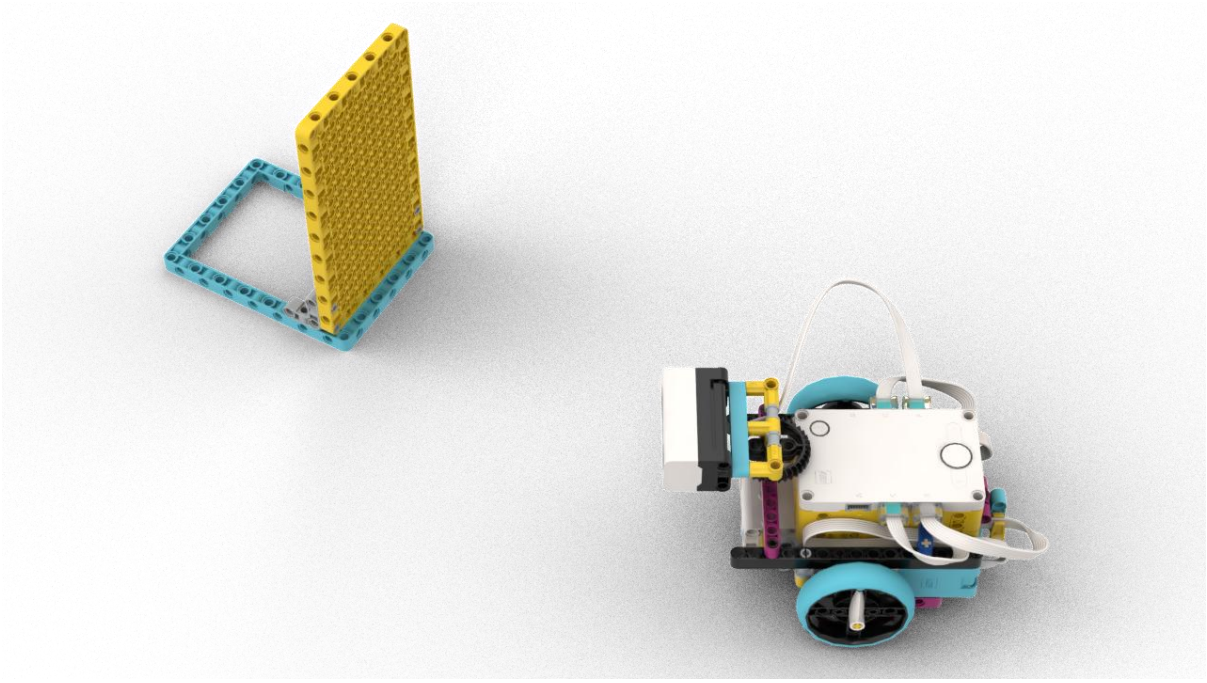
Resim 4.19 Hedef Tespit Robotu Dördüncü Adım



Resim 4.20 Hedef Tespit Robotu Beşinci Adım



Resim 4.21 Hedef Tespit Robotu Altıncı Adım



Resim 4.22 Hedef Tespit Robotu Görev Alanı

Öğrenciler mesafe sensörünü robotlarına taktıktan sonra, rehber öğretmen ayrıca Resim 4.22'de olduğu gibi Lego parçaları ile bir hedef hazırlamalarını ister. Robotun görevi mesafe sensörü ile karşısındaki 180 derecelik alanı 10 derece artan açılar ile dönerek taramaktır. Robot, eğer 50 cm'den daha yakın mesafede herhangi bir hedef tespit edemez ise 10 cm ileri giderek tekrar tarama işlemi yapmalıdır. Robot 50 cm'den daha yakın bir mesafede hedef tespit ettiğinde, robot hedefe doğru hareket ederek hedefe yaklaşmalı ve hedefe 10 cm uzaklıkta durmalıdır. Rehber öğretmen görevin daha kolay anlaşılması için ekte sunulan **Video_4.1_Tasarla_Uret.mp4** video dosyasını öğrencilere izletmelidir.

Tanımlama: Öğrenciler öncelikli olarak problemi tanımlamalıdır. Tanımlama süreci kesinlikle hızlıca geçilmemeli, robotun gerçekleştirmesi istenilen görevler kalem kâğıt kullanılarak yazılmalıdır.

Örneğin:

- Mesafe sensörü sürekli 180 derecelik bir alanı taramalıdır.
- Tarama esnasında mesafe sensörü 10 derecelik açılarla dönmelidir.
- Herhangi bir hedef tespit edilemediğinde, robot 10 cm ileri hareket etmelidir.
- Mesafe sensörü bir hedef algıladığında, robot yönünü hedefe çevirmelidir.
- Mesafe sensörünün hedefi algıladığı andaki yönü, robota aktarılmalıdır.
- Robotun belirlenen açı değeri kadar dönmesi sağlanmalıdır.
- Robot hedefe doğru hareket etmelidir.
- Robot hedeften 10 cm uzaklıkta durmalıdır.

Fikir üretme: Öğrenciler tanımlanan görevleri robotun gerçekleştirebilmesi için, hangi bloklarının kullanılacağına ve algoritmanın nasıl olması gerektiğine dair fikirler üretmelidirler.

Örneğin:

- Mesafe sensörü sürekli 180 derecelik bir alanı tarayacağından sonsuz döngü kullanılmalıdır.
- Mesafe sensörü büyük motora direk bağlı olduğundan motorun dönme açısı mesafe sensörünün hedefi algıladığı açı olacaktır.
- Büyük motorun dönme açısı 10 derece artarak 0 ile 180 (veya -90 ile 90) aralığında olmalıdır. Bu nedenle "*olana kadar bekle*" veya "*olana kadar tekrarla*" blokları kullanılabilir.
- Robotun kendi etrafında, büyük motorun dönme açısı kadar dönmesi hareket sensörü sapma açısı kullanılarak yapılabilir (2. haftada benzer bir uygulama yapılmıştı).
- Robotun hedefe 10 cm mesafede durması için yine mesafe sensörü kullanılmalıdır.

Öğrenciler bu süreçte farklı çözümler gerçekleştirebilirler. Örneğin mesafe sensörü 180 derecelik bir alanı tararken, bazı gruplar 0'dan onar onar artarak 180'e ulaşınca tekrar 0 dereceye dönüp aynı işlemi tekrar ederken, bazı gruplar 180'den onar onar azalarak 0'a ulaşmayı veya -90 ile +90 aralığında programlarını hazırlamayı seçebilir. Tüm gruplar aynı veya benzer programı hazırlamak zorunlu değildir. Ayrıca, öğrencilerin farklı fikirleri test etmeleri rehber öğretmenler tarafında desteklenmelidir.

ÜRET

Öğrenciler problemi tanımlayıp, çözüm için fikirler ürettikten sonra bilgisayar ve robotlarla çalışarak fikirlerini test etmelidirler. İstenilen görevi gerçekleştirilen programı bu sürecin sonunda üretmelidirler. Bu etkinlik için örnek program kodları Resim 4.23'te sunulmuştur.

Rehber Öğretmen İçeriği – Mesaj Blokları

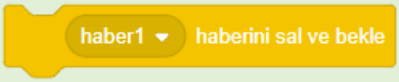
Resim 4.23'te sunulan programda "mesaj yayınla" (haberini sal) ve "mesaj aldığımda" (haberini aldığımda) blokları kullanılmıştır. Öğrenciler kendi programlarını hazırlarken, aynı programı hazırlamak zorunda olmamakla birlikte, rehber öğretmenin yeni blokların kullanılmasını teşvik edebilir. Tasarla ve üret etkinliği öncesinde bu blokların öğrencilere anlatması önerilmektedir.

Olay Blok Paleti

Mesaj Yayınla

Bu blok belirli bir haber yayınlar. Haber yayınlandıktan sonra, tüm "mesaj aldığımda" şapka blokları çalıştırılır. Program akışı mesaj gönderildikten sonra bir sonraki blok ile devam eder. Böylece program akışında istenilen noktadan itibaren diğer yığıt veya yığıtların çalıştırılması sağlanır.

Mesajı Yayınla ve Bekle

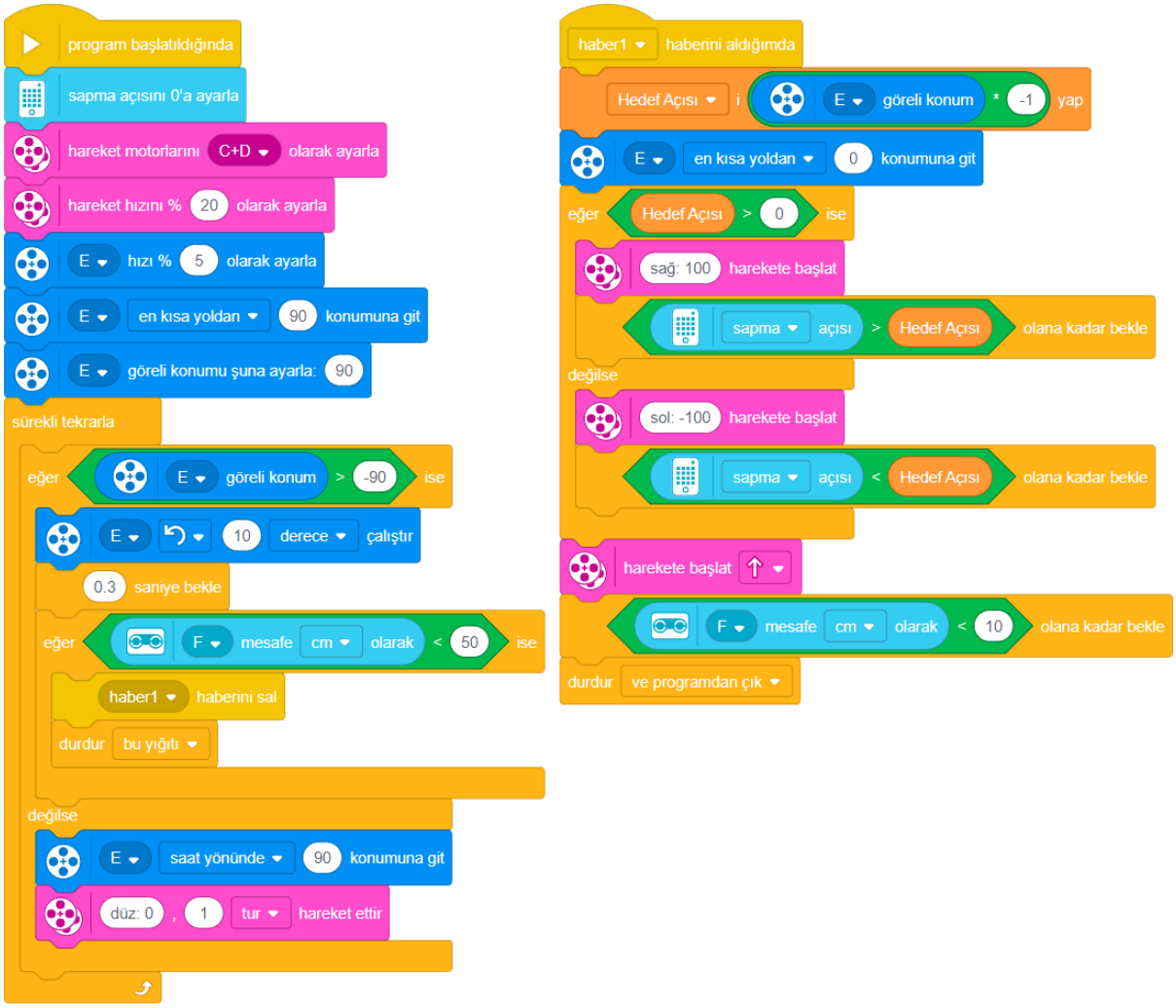


Bu blok belirli bir haber yayınlar. Haber yayınlandıktan sonra, tüm "mesaj aldığımda" şapka blokları çalıştırılır. "mesaj yayınla" bloğundan farklı olarak program akışı durdurulur ve tüm

"mesaj aldığımda" şapka bloğu ile başlayan yığıtlar tamamlandıktan sonra, program akışı kaldığı yerden devam eder.

Mesaj Aldığımda

Bu blok "mesaj yayınla" ve "mesajı yayınla ve bekle" blokları tarafından gönderilen haberi aldığında kendisine bağlı bulunan blokları çalıştırır.



Resim 4.23 Hedef Tespit Robotu Örnek Programı

Rehber Öğretmen İçeriği – Hedef Tespit Robotu Etkinliği Açıklaması

Sürüş modeli tasarımında büyük motor (E girişine takılı) ters durduğu için, büyük motorun göreli konumu, robotun sapma açısı ile ters yönlü olup, biri artarken diğeri azalmaktadır. Bu nedenle örnek programda Hedef Açısı değişkeni (robotun sapma açısı miktarı) büyük motorun göreli konumunun eksi bir (-1) ile çarpımına eşitlenmiştir. Benzer durum 2. haftada gerçekleştirilen

tasarla üret etkinliğinde de anlatılmıştı. Rehber öğretmenin bu durumu öğrencilere hatırlatması faydalı olacaktır.

Ayrıca robotun hedefin konumunu daha kusursuz bir şekilde tespit etmesi isteniyorsa, mesafe sensörünün 10'ar derecelik artışları daha küçük derecelere, örneğin 5 veya 3 dereceye ayarlanabilir. Fakat böyle bir durumda robotun hedefi tespit etme süresi aratacaktır.

DEĞERLENDİR

Gün sonunda öğrencilerle halka oluşturulur. Sensör isimleri kâğıtlara yazılır ve tombala oyunu misali her grubun bir kâğıt seçmesi istenir:

- Öğrencilerden seçilen sensörlerin özelliklerini anlatmaları ve bu sensörlerin hangi projelerde, nasıl kullanılabileceğini açıklamaları istenir.
- Sonrasında iki farklı sensörü olan gruplar bir araya gelirler. Öğrencilerden kâğıtlarındaki sensörlerin ikili olarak hangi projelerde, nasıl kullanılabileceğini örneklendirmeleri istenir.
- Bütün sensörlerin bir arada kullanılabileceği projeler tartışılır.
- Gün içerisinde öğrenilen kod blokları ve görevleri öğrenciler tarafından özetlenir.
- Bu haftanın en zor görevi ve öğrencilerin bu zorluğun üstesinden nasıl geldikleri konuşulur.

5. Hafta – Python Programlama ile Robotu Hareket Ettirme

Haftanın Amacı:

Bu haftanın amacı öğrencilere robot setini ve robot setini programlamak için kullanılacak olan Python ortamını tanıtmaktır. Öğrenciler robotun belirli bir mesafe boyunca hareket etmesi, viraj alma, kendi ekseninde dönme, bir teker etrafında dönme gibi manevraları yapması ve motorlar ile keskin hareketler gerçekleştirmesi beklenmektedir.

Haftanın Kazanımları:

- Öğrenciler orta ve büyük motoru tanır.
- Robotu programlamak için Python ortamını kullanır.
- Robotun keskin manevra ve hareketler yapabilmesi için ilgili komutlar ile günlük yaşam verisini birleştirir.
- Problem çözümü için hareket komutlarını bir arada kullanır.

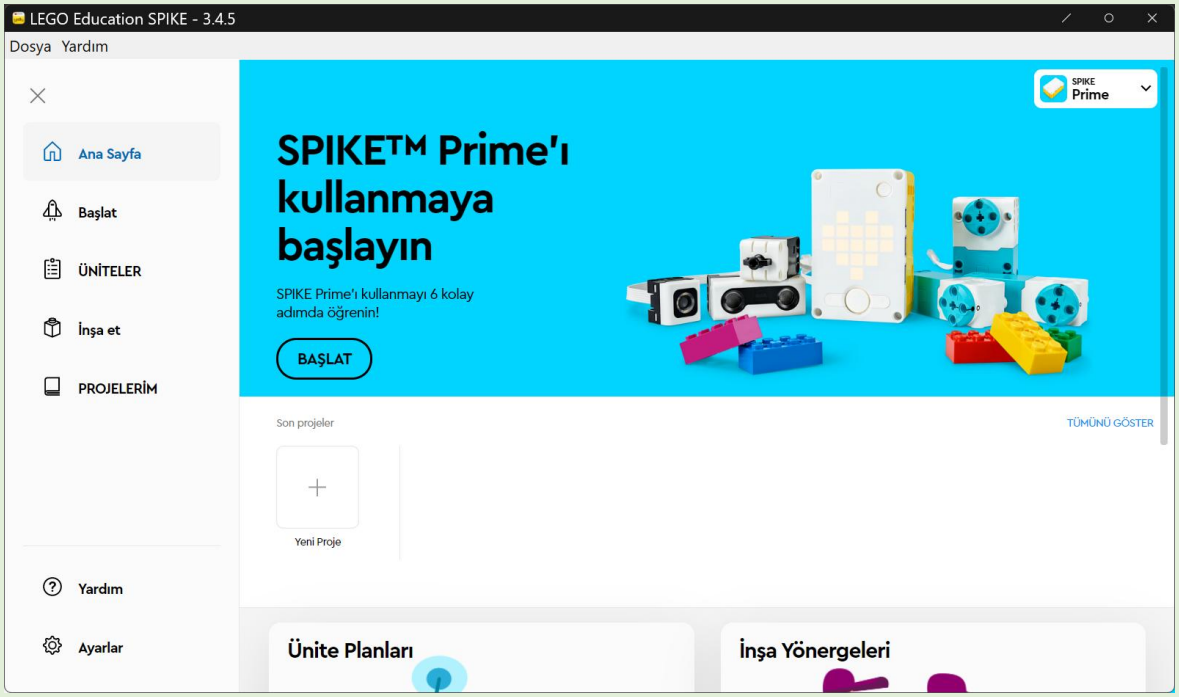
Kullanılacak Malzemeler:

Robot seti, bilgisayar.

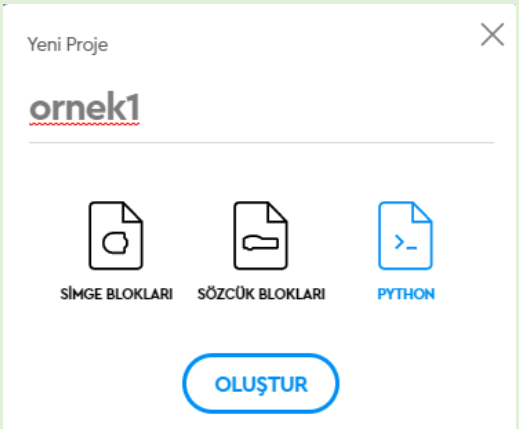
GÖZLE VE UYGULA

Rehber Öğretmen İçeriği – Python programlamaya diline giriş

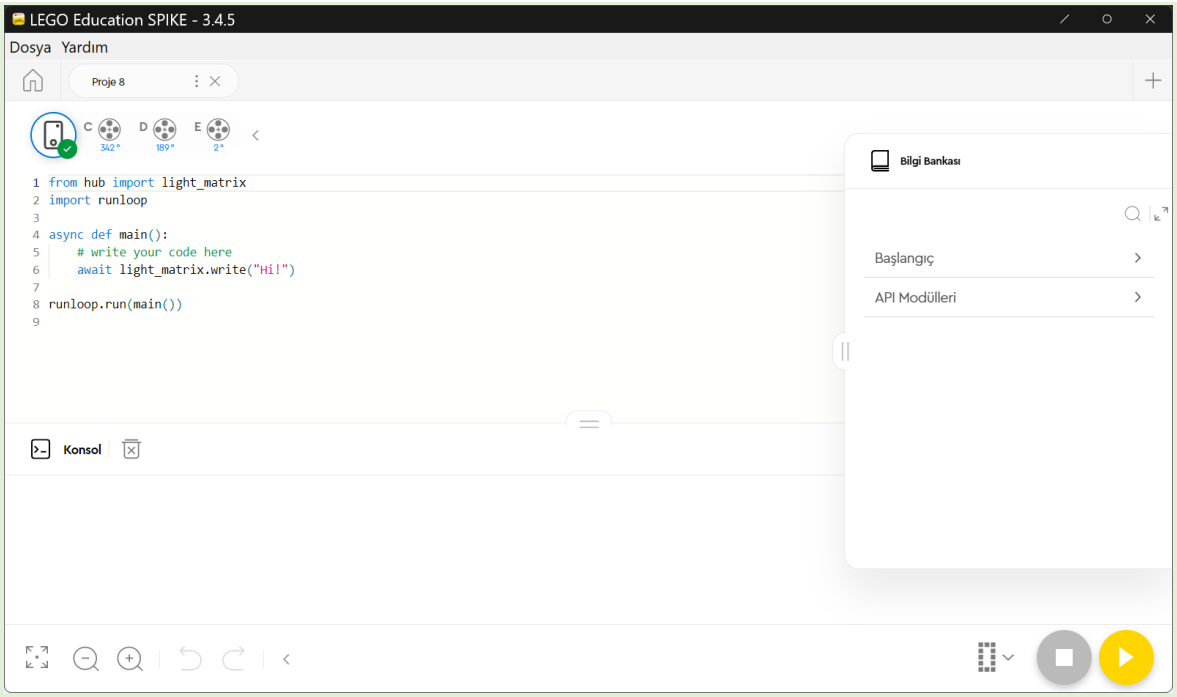
Bu hafta metin tabanlı programlama dillerinden olan Python programla dilini kullanarak örnekler yapmaya başlanacaktır. Bunun için Spike Prime Yazılımı arayüzünden "Yeni Proje" seçeneği ile ilk örnek oluşturulur.



Gelen ekrandan oluşturulacak projenin adı verilir. Örneğin projenin adı "ornek1" olarak giriş yapılır ve PYTHON seçilerek programlama arayüzüne geçilir.



Yeni proje ile metinsel programlama arayüzü açılır ve arayüzde neler olduğu gösterilir. Blok tabanlı programlama arayüzü ile benzerlikler ve farklılıklar vurgulanır. Daha sonra yazılımının bölümleri tanıtılır.



Arayüzün sağ tarafından "Bilgi Bankası" bölümünden bu ders boyunca da işleyeceğimiz birçok konu hakkında özet bilgilere erişilebileceği görülür. Arayüzün programlama alanına kodlar yazılarak sağ altındaki oynat düğmesi (sarı düğme) ile yazdığımız programları Hub'a gönderilerek çalıştırılabilir. Ekrandaki ilk kodların çalıştırılması durumunda Hub'ın ekranında "Hi!" metni görülür.

Not

Spike 3 sürümüyle gelen yenilik gereği, çalışması beklenen kodlar 'main' fonksiyonu içerisine yazılmalıdır.

Gözle: Sürüş Modelinde Direksiyon Hareketi

Bu etkinliğin amacı, robotun hareketini sağlamaktır. Bunun için Spike uygulamasında yer alan sürüş modeli tasarımı kullanılacaktır. Öğrenciler yönergeyi takip ederek Sürüş Modeli 2 yönergesinin 26. adımına kadar olan montajı tamamlamalıdırlar. Öncelikle, robotun tanıtılması gerekmektedir. Sürüş modelinin kullanılabilmesi için, motor_pair ve port kütüphanelerinin projeye dâhil edilmesi önemlidir.

```
import motor_pair
from hub import port
```

Sürüş modelinin tanımlanması için bir *motor_pair* nesnesi oluşturulmalıdır. *MotorPair* nesnesi oluşturulurken iki parametre belirlenmelidir. İlk parametre sol motorun bağlı olduğu portu, ikinci parametre ise sağ motorun bağlı olduğu portu belirlemek için kullanılır. Sürüş Modelinin sol ve

sağ motorları sırasıyla C ve D portlarına bağlanmıştır. Sürüş Modeli aşağıdaki kod kullanılarak oluşturulur.

```
motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
```

Sürüş Modeli oluşturulduktan sonra ileri doğru doğrusal hareketi sağlamak için move metodu kullanılabilir. Move metodunun parametreleri detaylıca incelenmeden önce bir örnek vermek yerinde olacaktır. Aşağıdaki kod ile sürüş modeli oluşturulmuş ve ileriye doğru 2 saniye boyunca hareket etmesi sağlanmıştır. Rehber öğretmen kodu tahtaya yansıtır, programı çalıştırır ve öğrencilere kodu ve kodun görevini açıklar.

```
motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
motor_pair.move(motor_pair.PAIR_1,0)
await runloop.sleep_ms(2000)
motor_pair.stop(motor_pair.PAIR_1)
```

Ardından aşağıdaki kodu tahtaya yansıtır ve çalıştırır. Bu kodun görevinin sürüş modelini 3 saniye boyunca ileriye doğru 500 derece/sn hızında hareket ettirmek olduğunu belirtir. Rehber öğretmen her iki kodu da öğrencilerin yazıp çalıştırmasını sağlar.

```
motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
motor_pair.move(motor_pair.PAIR_1, 0, velocity=500)
await runloop.sleep_ms(3000)
motor_pair.stop(motor_pair.PAIR_1)
```

Uygula: Direksiyon Hareketini Keşfediyorum

Rehber öğretmen öğrencilere aşağıdaki soruyu sorar:

Move metodunun kullanımını keşfetmek için bir önceki etkinlikte yazılan move metodu üzerinde değişiklikler yaparak robotu ileri, geri, sağa ve sola hareket ettirmeye çalışınız.

Rehber öğretmen öğrencilere yeteri kadar süre verir. Bu sürenin sonunda bazı gruplardan soru ile ilgili bulgularını sınıf önünde açıklamalarını ister.

Gözle: Direksiyon Hareketinin Detaylarını Öğreniyorum

Move metodu 4 parametre almaktadır:

```
motor_pair.move(motor_pair.PAIR_1, 0, velocity=500, acceleration=1000)
```

İlk parametre olan motor_pair.PAIR_1, kontrol edilmek istenen motor çiftini belirtir. Spike 3'te motorlar çiftler halinde gruplandırılır ve her çift bir yuva numarasıyla temsil edilir. Bu durumda, PAIR_1, birinci motor çiftini ifade eder.

İkinci parametre olan 0, direksiyon kontrolü için kullanılır ve robotun hareket yönünü belirler. Bu parametre -100 ile 100 arasında bir değer alabilir. 0 değeri, motor çiftinin düz bir çizgide hareket edeceğini belirtir. Pozitif bir değer sağa dönüşe, negatif bir değer sola dönüşe yol açar. Örneğin,

50 değeri, robotun sağa doğru sağ tekeri üzerinde dönmesini, 25 değeri sağa doğru daha geniş bir dönme hareketi sağlar.

Velocity parametresi, motorların hızını belirler ve birim olarak derece/saniye kullanılır. Hız, motor tipine göre belirli sınırlar içinde olmalıdır: Küçük motorlar için -660 ile 660, orta motorlar için -1110 ile 1110, büyük motorlar için -1050 ile 1050 arasında bir değer alabilir. Bu örnekte, velocity=500, motorların saniyede 500 derece hızla ileri hareket etmesini sağlar. Eğer negatif bir değer verilirse, motorlar geri yönde hareket eder.

Son parametre olan acceleration, motorların ivmesini belirler ve derece/saniye² birimiyle ifade edilir. İvme, motorların maksimum hıza ne kadar hızlı ulaşacağını kontrol eder. Bu değer 0 ile 10000 arasında bir sayı olmalıdır. Örnekte, acceleration=1000, motorların 1000 derece/saniye² ivme ile hızlanacağını ifade eder. Yüksek ivme değerleri, robotun daha hızlı bir şekilde maksimum hıza ulaşmasını sağlar.

```
motor_pair.move_for_degrees(motor_pair.PAIR_1, 360, 0, velocity=500)
```

Yukarıdaki kod, belirli bir motor çiftini, belirtilen bir açı boyunca hareket ettirir. *move* komutundan farklı olarak ikinci parametrede yazılan 360, motorların toplamda kaç derece döneceğini belirtir ve bu örnekte motorlar bir tam tur (360 derece) hareket edecektir.

```
motor_pair.move_for_time(motor_pair.PAIR_1, 2000, 0, velocity=500)
```

Yukarıdaki kod, belirli bir motor çiftini, belirli bir süre boyunca hareket ettirir. *move* komutundan farklı olarak ikinci parametre olan 2000, motorların ne kadar süre çalışacağını belirtir ve bu süre milisaniye cinsindedir. Bu örnekte motorlar 2 saniye boyunca hareket edecektir.

Rehber öğretmen öğrencilere *move* metodunun parametrelerini örnekleyerek anlatır. Her bir parametre için ayrı ayrı örnek verilir.

Uygula: İlerleyip Aynı Yere Gelen Robot

Öğrencilerden

(i) robotun 3 tur ileri gitmesini,
(ii) robotun geri dönmesini,
(iii) geri gelerek başlangıç noktasına ulaşmasını
(iv) ve yüzünü ilk başlangıç haline dönmesini sağlayan programı yazmaları istenir.

Bu görev için aşağıdaki kod kullanılabilir. Burada robotun 180 derece dönmesini sağlamak için tekerlerin 2 tur (360x2) boyunca kendi ekseni etrafında sola doğru bir dönüş kullanılmıştır. Öğrenciler farklı şekillerde ve değerler kullanarak dönüş işlemlerini gerçekleştirebilir. Rehber öğretmen öğrencilerden birebir aşağıdaki kodu yazmasını beklememelidir. Öğrencilerin kendi yollarından çözüme ulaşması önemlidir.

```
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 360*3, 0,
velocity=500)
    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 360*2,-50,
velocity=500)
```

```
    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 360*3, 0,
velocity=500)
    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 360*2, -50,
velocity=500)
```

Not

Await komutu, bir işlemi duraklatarak, onun tamamlanmasını bekler ve ardından programın çalışması devam eder.

Uygula: Tekerin Yarıçapını Hesaplıyorum

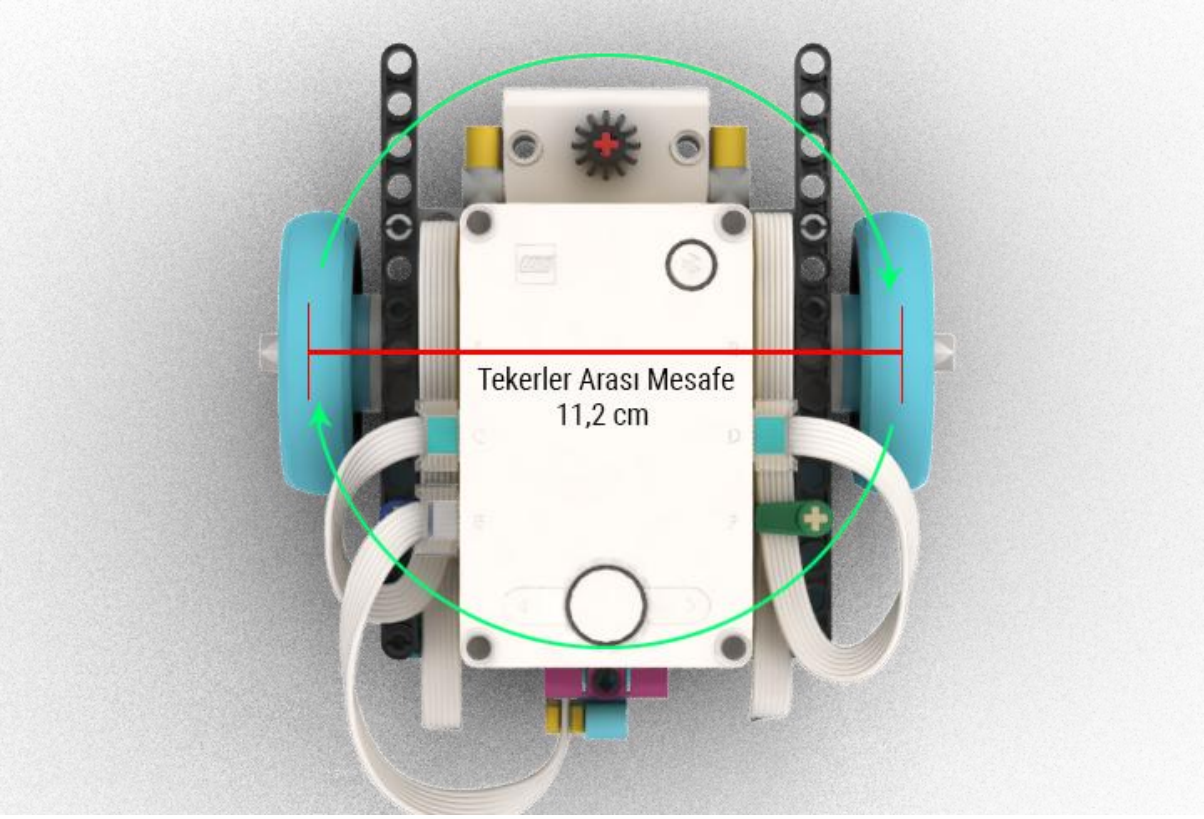
Bu etkinlikte öğrencilerden tekerlerin bir tur döndürüldüğünde robotun aldığı mesafeyi cm cinsinden bulmaları istenir. Ardından öğrencilerden tekerin yarıçapını hesaplamaları istenir. Spike Prime ile gelen tekerler standart küçük teker olarak adlandırılır ve bu tekerin çapı 5.6 cm'dir. Tam tur teker dönüşünde robot 17.6 cm yol alır. Öğrencilerin kendilerinin hesaplaması yapması beklenir ve hesaplamalarının ardından bu değer onlara söylenir.

Uygula: 360 Derece Dönen Robot

Öğrencilere iki tur kendi ekseni etrafında dönen robotun yaklaşık olarak kendi ekseni etrafında 360 derece döndüğü söylenir ve aşağıdaki kodu kullanarak bu durumu gözlemeleri beklenir. Öğrencilerden bu durumun neden böyle olduğunu açıklamaları istenir.

```
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    motor_pair.move_for_degrees(motor_pair.PAIR_1, 360*2, -100,
velocity=500)
```

Sol teker ve sağ teker arasında yaklaşık olarak 11.2 cm mesafe (teker çapının iki katı) bulunmaktadır. Robot kendi ekseni etrafında tam bir dönüş yaptığında çapı yaklaşık olarak 11.2 cm olan dairesel bir hareket yapar. Tam dönüşün elde edilebilmesi için tekerlerin 2 tam tur (35.2 cm) dönmesi gerekmektedir.



Resim 5.1 Kendi Ekseninde Dönme

Gözle: Palet Hareketi

Direksiyon hareketinde dönülecek yön ve dönüş miktarı bildirildikten sonra her bir tekere verilecek hız büyüklüğünü robotun algoritması belirler. Palet hareketi direksiyon hareketinden farklı olarak sol ve sağ motorlara verilecek hız miktarını programcının belirlemesine imkân sağlar. Palet hareketi için *move_tank* metodu kullanılır. Bu metotta dört parametre bulunmaktadır:

```
motor_pair.move_tank(motor_pair.PAIR_1,500,1000)
```

Yukarıdaki kod belirli bir motor çiftini tank tarzı hareketle kontrol eder, yani her motorun hızını bağımsız olarak belirler. İlk parametre olan motor_pair.PAIR_1, kontrol edilecek motor çiftini ifade eder; burada birinci motor çifti seçilmiştir. İkinci parametre olan 500, sol motora uygulanacak hız değerini derece/saniye biriminde belirtir. Üçüncü parametre olan 1000, sağ motora uygulanacak hız değerini derece/saniye biriminde ifade eder. Bu örnekte sol motor, saniyede 500 derece hızla hareket ederken sağ motor, saniyede 1000 derece hızla hareket edecektir. Bu hız farkı, robotun sola doğru bir dönüş yapmasını sağlar. Bu komut, iki motorun hızlarının bağımsız olarak kontrol edilebildiği durumlarda, dönüşler veya yönlendirmeler için kullanılır.

Uygula: Sürüş Modelinde Palet Hareketi

Bu etkinlikte amaç öğrencilerin palet hareketinin sağladığı manevra çeşitliliğini görmelerini sağlamaktır. Bu iş için öğrencilerden aşağıdaki üç farklı hareket için kod yazmaları istenir:

i) Virajı alır şekilde dönme:

```
motor_pair.move_tank_for_degrees(motor_pair.PAIR_1, 360*2, 400, 800)
```

ii) Kendi ekseninde dönme:

```
motor_pair.move_tank_for_degrees(motor_pair.PAIR_1, 360*2, 400, -400)
```

(iii) Bir tekerlek (sol veya sağ) etrafında dönme:

```
motor_pair.move_tank_for_degrees(motor_pair.PAIR_1, 360*2, 0, 800)
```

Gözle: Robotun Özelliklerini Ayarlıyorum

Sürüş modelinde bahsedilmesi gereken son özellik durma şeklidir. Sürüş modelinde üç farklı durma şekli bulunmaktadır. Bunlar *"hold", "brake" ve coast'ur*. *Hold* seçildiğinde tekerler ani bir şekilde bulunduğu yerde durur ve dışarıdan uygulanan kuvvet (yer çekimi, el ile itme gibi) ile tekerleri hareket ettirmek mümkün olmaz. Başka bir değişle sürüş modeli bulunduğu yere sabitlenir. *Brake* seçildiğinde tekerler hızlıca durur fakat durma sonrasında dışarıdan uygulanan bir kuvvet ile hareket edebilir. *Coast* seçildiğinde tekerler görece yavaş bir şekilde durur ve dışarıdan uygulanan bir kuvvet ile hareket edebilir. Aşağıdaki komutlar kullanılarak istenilen durma şekli ayarlanabilir (`import motor` ile motor kütüphanesi projeye dahil edilmelidir.):

```
motor_pair.stop(motor_pair.PAIR_1, stop=motor.HOLD)
motor_pair.stop(motor_pair.PAIR_1, stop=motor.BRAKE)
motor_pair.stop(motor_pair.PAIR_1, stop=motor.COAST)
```

Rehber öğretmen aşağıdaki komutu farklı durma şekilleri belirlendikten sonra çalıştırır ve her bir şekil için durma mesafesi ve durma sonrası hareket edip etmeme durumlarını öğrencilere gösterir. Durma sonrası hareketi göstermek için robot durduktan sonra yavaşça itilerek hareket edip etmediği belirlenebilir.

```
motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
motor_pair.move_tank(motor_pair.PAIR_1,500,500)
await runloop.sleep_ms(1000)
motor_pair.stop(motor_pair.PAIR_1, stop=motor.HOLD)
```

Gözle: Motoru Belirli Bir Açıya Getirme

Sürüş Modeli gibi robot araba benzeri cihazların hareket ettirilmesi için motor çiftine yönelik komutlara ihtiyaç duyulur. Fakat bazı durumlarda motor çiftinden ziyade motorların ayrı ayrı hareketinin sağlanması gerekebilir. Bu durumda öncelikle aşağıdaki komutlar kullanılarak motor ve port kütüphaneleri projeye dahil edilmelidir.

```
import motor
from hub import port
```

Motoru istenen bir açıya getirmek için *run_to_absolute_position* isimli metot kullanılabilir. Bu metot ile motorun pozisyonu bulunduğu açı değerinden hedef açı değerine doğru hareket ettirilir.

```
    motor.run_to_absolute_position(port, hedef motor açısı, hız, yön)
    motor.run_to_absolute_position(port.E,100,200,direction=
motor.CLOCKWISE)
```

Port parametresine motorun bağlı bulunduğu port bilgisi yazılır. Hedef motor açısı parametresine derece cinsinden motorun ulaşması istendiği açı değeri girilir. Bu değerler 0 ile 359 arasında değişir. Hız parametresi orta motorlar için -1110 ile 1110 ve büyük motorlar için -1050 ile 1050 arasında değişen değerler alabilir. Yön parametresi ile istenen açıya hangi yönden gidilebileceği belirtilir. Yön dört farklı değer alabilir. Bunlar "CLOCKWISE", "COUNTERCLOCKWISE", "SHORTEST_PATH" *ve* "LONGEST_PATH" değerleridir. CLOCKWISE değeri Resim 5.2'de görüldüğü gibi motoru saat yönünde çevirir. CLOCKWISE ile sürüş modeli önünde bulunan hareketli kol yukarı doğru hareket eder. COUNTERCLOCKWISE saat yönünün tersine hareket sağlar. Sürüş modelindeki hareketli kol aşağı doğru hareket edecektir. SHORTEST_PATH bulunan açı ile hedef açı arasındaki en kısa yolu seçer. LONGEST_PATH ise bulunan açı ile hedef açı arasındaki en uzun yolu tercih eder. Örneğin motor 300 derecede duruyor olsun. 340 dereceye gitmek için motor en kısa yol olan saat yönünde (clockwise) hareketi tercih edecektir.

Resim 5.2 Motor Dönüş Yönü ve Sıfır Noktası

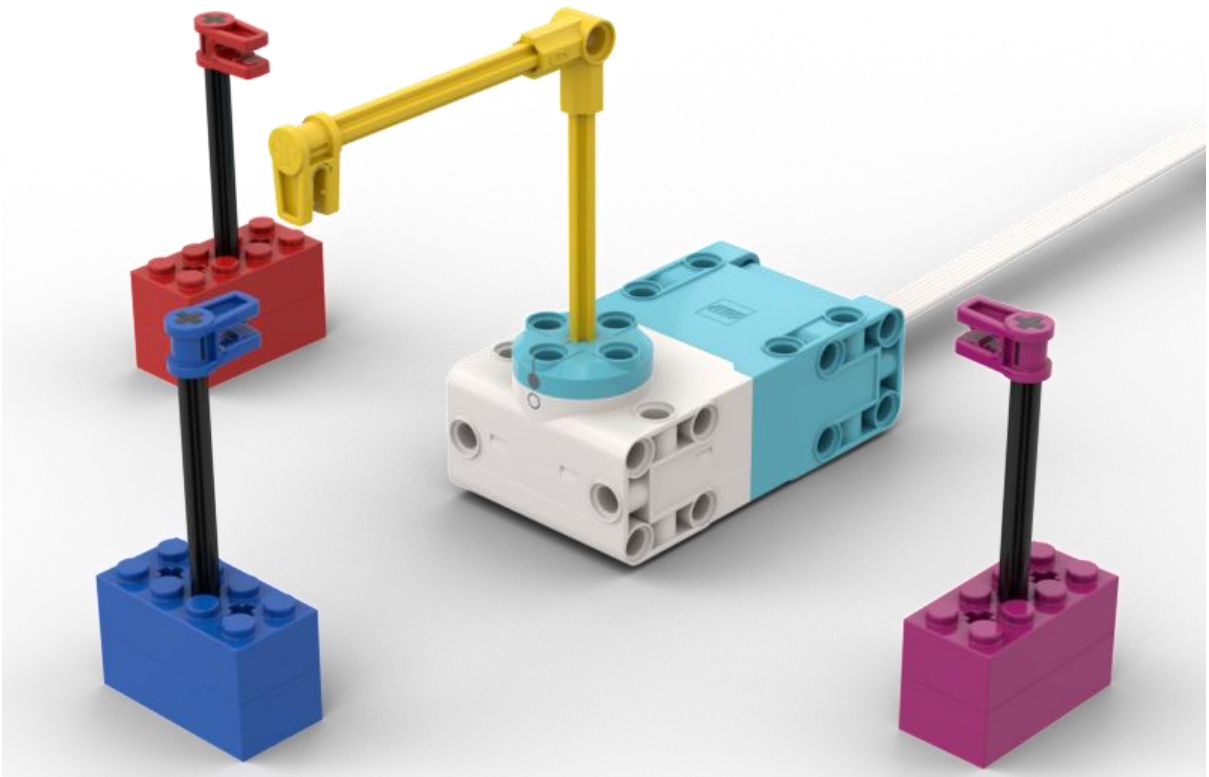
Resim 5.2'de gösterildiği gibi motorlar üzerinde sıfır dereceyi gösteren bir ibare bulunmaktadır (0 Noktası). Motorun işaretli kısmı bu noktaya geldiğinde motor sıfır derecede olacaktır. Bunun dışında motorun açısını bulmak için kod yazılan ekranın üst kısmındaki panelde bulunan motor açı değeri kullanılabilir. Resim 5.3'te örnek açı değeri gösterilmiştir.

Resim 5.3. Motor Açısı

Rehber öğretmen sürüş modeline bağlı olmayan bir büyük motoru Hub'a bağlayarak run_to_absolute_position metodunun farklı kullanımlarını onlara gösterir. Bunu yaparken farklı açı, yön ve hız değerlerini kullanır.

Uygula: İşaretçi Direk Eşleştirmesi

Rehber öğretmen öğrencilerden Resim 5.4'te gösterildiği gibi bir düzenek kurmalarını ister. Düzenek kurulduktan sonra geniş motor etrafındaki üçlü direği her bir öğrenci grubu için geniş motor etrafında rasgele bölgelere yerleştirir. Öğrencilerden, geniş motora bağlı işaretçinin sadece direklerin karşısında geldiğinde duracak şekilde, robotun kodlanmasını ister.



Resim 5.4 Etkinlik Düzeneği

Gözle: Motoru Belirlenen Başlangıç Noktasına Göre İstenen Bir Açıya Getirme

run_to_absolute_position metodu ile motor 0-359 arasındaki mutlak açı değerlerine getirilebilir. Fakat bazı durumlarda motor mutlak açı değerleri yerine göreceli bir açı değerine getirilmek istenebilir. Göreceli bir açı değerinden bahsedebilmek için öncelikle başlangıç değerinin belirlenmesi gerekmektedir. Robot ilk çalıştırıldığında motorun mutlak açı değeri ne olursa olsun,

bulunan yer için göreceli açı değeri 0 olarak alınır. Bunun yanında istendiği zaman başlangıç açı değeri reset_relative_position metodu ile belirlenebilir. Bu metodun kullanımı aşağıda gösterilmiştir

```
motor.reset_relative_position(port, başlangıç açı değeri)
motor.reset_relative_position(port.E, 50)
```

Bu metot ile motor herhangi bir mutlak açı değerinde duruyor iken (örneğin 180 derecede durduğunu düşünelim) motor.reset_relative_position(port.E, 0) komutu ile bulunan yerin göreceli açı değerinin 0 olması sağlanır. Mutlak açı değerinde herhangi bir değişiklik olmamıştır (hala mutlak açı değeri 180 derecedir). Fakat bulunan yerin göreceli değeri artık 0 olarak kabul edilir. Bu metoda 0 değeri girmek zorunlu değildir. İstenen tam sayı değeri verilebilir. Verilen tam sayı değeri bulunan pozisyonun göreceli açı değeri olarak kabul edilecektir. Göreceli açı değerine göre belirli bir açıya hareket etmek için run_to_relative_position metodu kullanılabilir. Bu metodun kullanımı aşağıdaki şekildedir.

```
motor.run_to_relative_position(port, hedef açı, hız)
motor.run_to_relative_position(port.E,50,350)
```

Hedef açı değeri sayesinde göreceli açı değeri hedef değere ulaşana kadar motor hareket eder. Hedef açı bütün tam sayı değerlerini alabilir. Hız ise istenen hedef açıya hangi hız ile gidileceğini belirtir. Örnek vermek gerekirse bulunan pozisyonun açı değeri 50 olarak belirlenmişse hedef açı 100 girildiğinde, göreceli açı değerini 100 yapabilmek için motor saat yönünde 50 derece dönecektir. Bunun ardından ikinci bir komut ile hedef açı 80 olarak belirlenirse motor göreceli açı değerini 80 yapabilmek için saat yönünün tersine 20 dere dönecektir.

Rehber öğretmen öncelikle motorun mutlak açı değerini eli ile 0 derece yapar. Ardından aşağıdaki kodu çalıştırır ve motorun bulunduğu yerin göreceli açı değerini 50 derece olarak belirlediğini ve 40 derece saat yönünde döndüğünü anlatır. Motorun göreceli açı değeri 90 iken mutlak açı değeri 40 olacaktır.

```
motor.reset_relative_position(port.E,50)
motor.run_to_relative_position(port.E,90,350)
```

Rehber öğretmen aşağıdaki kodu çalıştırır ve motorun önce saat yönünde 90 derece hareket ettiğini daha sonra 180 dereceye ilerleyebilmek için saat yönünde doksan derece daha hareket ettiğini öğrencilere anlatır.

```
motor.reset_relative_position(port.E,0)
motor.run_to_relative_position(port.E,90,350)
motor.run_to_relative_position(port.E,180,350)
```

Rehber öğretmen aşağıdaki kodu çalıştırır ve motorun önce saat yönünde 90 derece hareket ettiğini ardından değerin -90'a gelebilmesi için motorun saat yönünün tersine 180 derece döndüğünü anlatır.

```
motor.reset_relative_position(port.E,0)
await motor.run_to_relative_position(port.E,90,350)
await motor.run_to_relative_position(port.E,-90,350)
```

Rehber öğretmen aşağıdaki kodu çalıştırır ve öğrencilere açı değerinin 720 olabilmesi için motorun yaklaşık olarak iki tur döndüğünü açıklar.

```
motor.reset_relative_position(port.E,0)
motor.run_to_relative_position(port.E,720,350)
```

Uygula: Üç Turda İşaretçi Direk Eşleştirmesi

Bu etkinlikte, bir önceki etkinlikte yapılan işaretçi direk eşleştirmesinin bir benzeri yapılacaktır. Fakat bu kez işaretçi üç tur atacaktır. Her bir turda da işaretçinin sadece direkler karşısında durması gerekmektedir. Rehber öğretmen öğrencilerden gerekli kodu yazmalarını ister.

Gözle: Belirli Bir Derece/Tur/Saniye Hareket Etme

Bazı durumlarda motorun mutlak veya göreceli istenen açı değerine hareket etmesi yerine, bulunduğu açı değerinden bağımsız olarak belirli bir açı, süre ve tur dönmesi istenebilir. Bu durumlarda aşağıdaki komutlar kullanılır.

```
run_for_degrees(port, miktar, hız) # Belirtilen açı miktarı kadar dönme
run_for_time(port, miktar, hız) # Belirtilen süre miktarı kadar dönme
```

Örneğin motorun saat yönünde 90 derece 500 derece/sn hızında hareket etmesi isteniyorsa aşağıdaki komut kullanılır.

```
motor.run_for_degrees(port.E,90,500)
```

Başka bir örnek vermek gerekirse, motorun 1.5 saniye saat yönünün tersine 500 derece/sn hızında hareket etmesi isteniyorsa aşağıdaki komut kullanılır.

```
motor.run_for_time(port.E,1500,-500)
```

Not

Motorun bir tur hareketi için özel bir metot yoktur. Ancak motorun 360 derece dönmesi bir tur olacağından 360 ile pozitif bir tamsayıyı çarptığımız zaman o kadar tur motor hareket edebilir. Örnek:

```
motor.run_for_degrees(port.E,360*3,500) # motor üç tur döner.
```

Açı miktarı tam sayı değerleri alabilir. Süre milisaniye cinsindendir. Rehber öğretmen yukarıda bulunan üç komutu da öğrencilerin deneyimlemesini sağlar ve gerekli açıklamaları yapar.

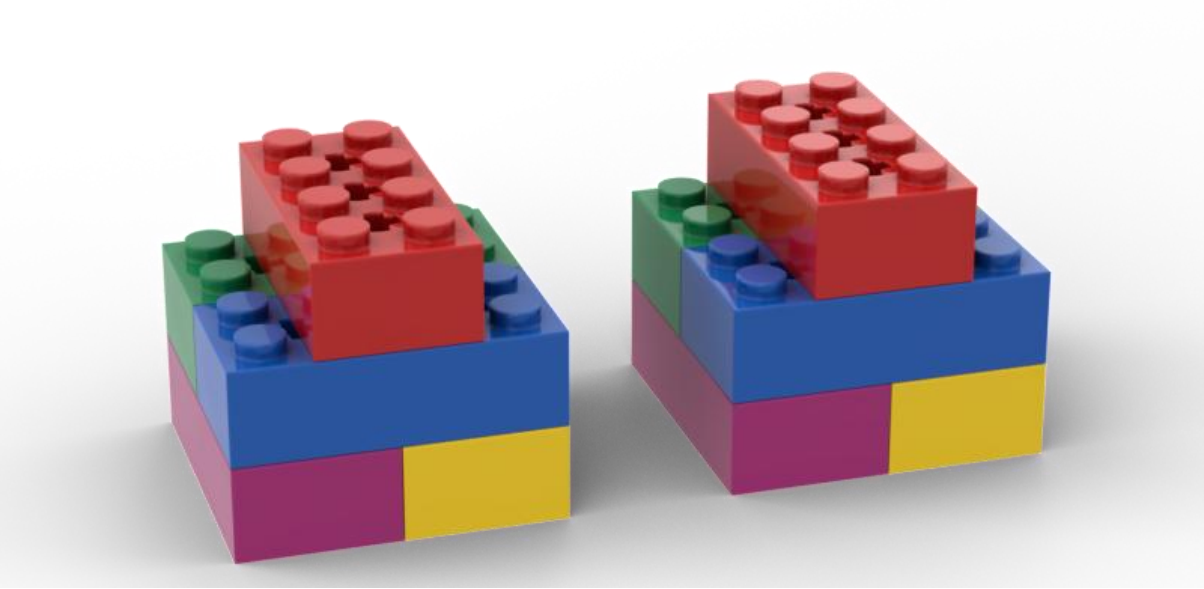
Uygula: Belirli Bir Derece/Tur/Saniye Hareket Etme

Öğrenciler sürüş modeline hareketli kolu inşa eder. Ardından rehber öğretmen, öğrencilerden sürüş modelindeki hareket eden kolu büyük motor kullanılarak yukarı aşağı hareket ettirmeleri istenir. Bu hareket ritmik dans hareketleri şeklinde olmalıdır.

TASARLA

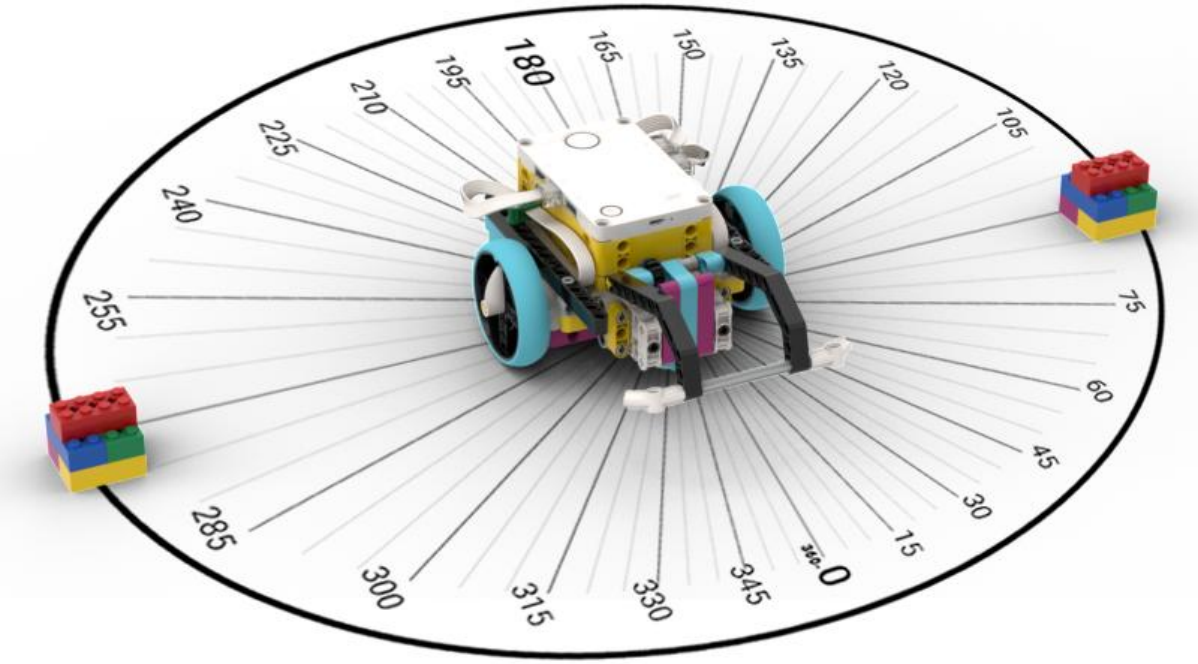
Tasarla: Cisimleri Kenardan Ortaya Taşıyorum

Rehber öğretmen, öğrencilerden Sürüş Modeli 2 yönergesini tamamlayarak büyük motoru robotlarına monte etmelerini ve 'Araçlar ve Aksesuarlar' yönergesindeki ilk 18. adımı gerçekleştirerek büyük motor ile kontrol edilecek kolu robotlarına eklemelerini ister. Bu etkinlik için, aşağıda gösterildiği gibi Sürüş Modeli tarafından taşınmak üzere iki cisme ihtiyaç duyulacaktır. Öğrenciler, robot tarafından taşınabilir nitelikte olmak kaydıyla farklı cisimler inşa edebilirler.



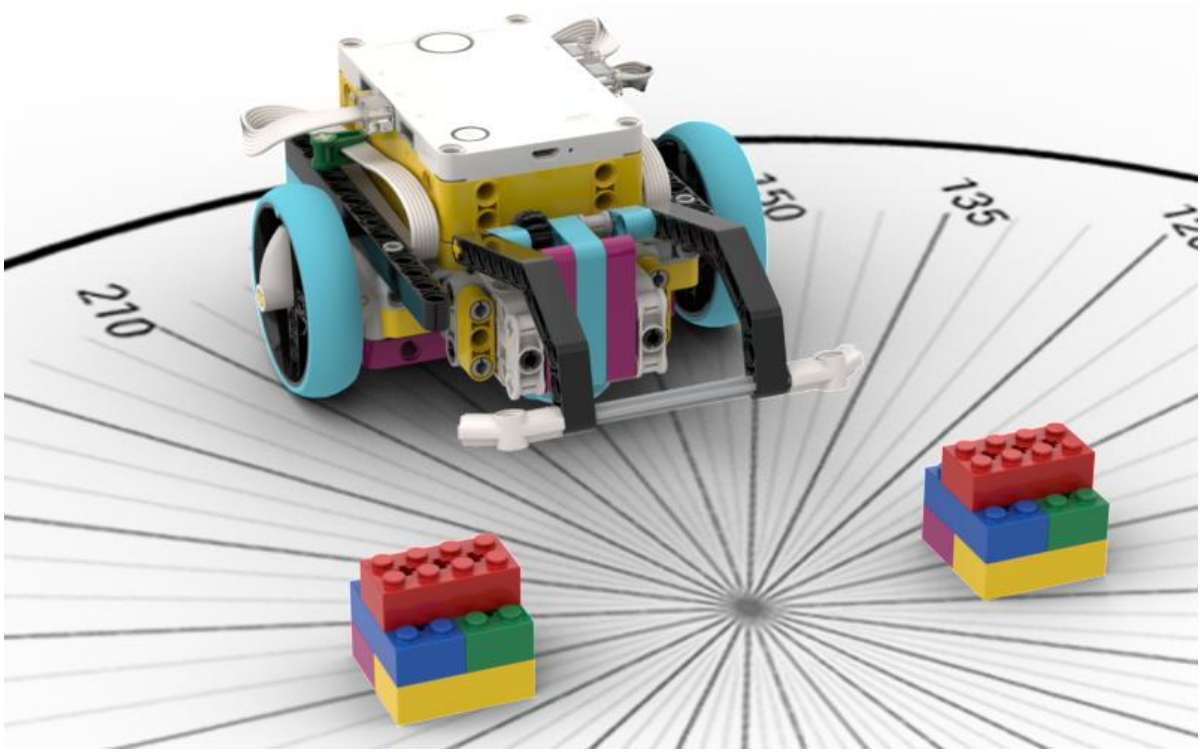
Resim 5.5 Taşınacak Cisimler

Sürüş Modeli robotu ve taşınacak cisimler Resim 5.6'da gösterildiği gibi yerleştirilir. Çalışma alanı örnek olarak gösterilmiştir. Bu şekilde eliptik bir yüzey kullanılmak zorunda değildir. Şekilde görülen, robotun ve cisimlerin başlangıç duruşudur. Bu etkinlikte robot kenarlardaki cisimleri orta kısımda toplayacaktır.



Resim 5.6 Etkinlik Başlangıç

Resim 5.7'de görüldüğü gibi robot görevini bitirdiğinde cisimler orta kısımda olacaktır. Robot ise cisimlerin arka kısmında şekildeki gibi durmalıdır.



Resim 5.7 Etkinlik Bitiş

Rehber öğretmen öğrencilerden öncelikle çözümleri için bir tasarım hazırlamaları gerektiğini söyler. Öğrenciler doğrudan çözümü inşa etmeye başlamamalıdır. Öncelikle çözüm üzerine düşünüp bir tasarısını oluşturmalıdır. Tasarım için aşağıda gösterilen iki adımlı bir süreç kullanılabilir.

Tanımlama: Bu adımda öğrenciler problemi ve bileşenlerini tanımlamalıdır. Kâğıt kalem alarak robotun yapması gerektiği hareketin krokisini çizebilirler. Ardından robot için bu hareketlerin ne anlama geldiğini ve hangi adımlardan geçerek bu görevi yerine getirebileceğini tanımlarlar.

Bu problemin çözümü için:

- Robot 90 derece sağa dönecek,
- Cisme kadar ilerleyecek ve cismi alıp geri geri gelerek orta kısma taşıyacak,
- Cismi bıraktıktan sonra 90 derece sağa dönecek ve yeterli miktarda geri gelecek,
- Robot yeniden 90 derece sola dönecek ve diğer cismi alıp bir miktar ileriye taşıyacak,
- Daha sonra 90 derece sağa dönerek cismi orta kısma taşıyıp bırakacak,
- Cismi bıraktıktan sonra robot bir miktar geri gelecek 90 derece sola dönecek,
- Cisim bir miktar daha geriye gidecek ve 90 derece sağa dönecek,
- Robot bir miktar ilerledikten sonra 90 derece sağa dönecek ve cisimlere yaklaşacak.

Dikkat edilmesi gereken noktalar:

1. Robotun sağa ve sola 180 derece dönmesi gerekecek, bu dönüşler olabildiğince keskin bir şekilde ayarlanmalı,
2. Robotun cisimlere olan uzaklığı ölçülüp gerekli hareketler olabildiğince keskin şekilde yapılmalı,
3. Robot kolun aşağı ve yukarı hareketi cisimlerin boyutlarına göre ayarlanmalı ve bu hareket olabildiğince keskin şekilde yapılmalı.

Fikir üretme: Bu aşamada öğrencilerden tanımlamada belirlenen işlemlerin nasıl yapılacağına dair fikir üretmeleri beklenmektedir.

- 90 derece sağa ve sola dönüşler için robotun kendi merkezinde dönmesi sağlanmalıdır,
- 180 derecelik bir tekerlek dönüşü 90 derecelik robotun dönüşüne denk gelebilir. Bu değer denenerek keskin dönüş için uyarlanmalıdır.
- Robot kolun yukarı aşağı hareketi için gerekli açı değişim miktarları 90 dereceden küçük olmalıdır. Bu açılar denenerek bulunmalıdır. Bu iş için *move_to_position* ve *run_for_degrees* metotları kullanılabilir.
- Robotun cisme olan uzaklığı ölçülüp bulunmalıdır. Move komutu kullanılarak bu işlem yapılabilir. Bu işlemin keskin bir şekilde yapılması için denemeler yapılmalıdır.
- Diğer manevralar için de gerekli denemeler yapılmalıdır.

Tasarla aşamasında öğrenciler yukarıda anlatılan adımları birebir gerçekleştirmek zorunda değildir. Bu aşamadaki amaç problemin sınırlarının iyice çizilmesinin ardından, çözüm üzerine düşünmek onun bir tasarısını çıkarmaktır. İyi programcıların en fazla vakit ayırdığı etkinlik tasarıdır. Giriş seviyesindeki programcılar genel olarak çözüme doğrudan başlar ve deneme yanılma süreciyle ilerlemeye çalışır. Bu yeterince verimli bir yaklaşım değildir. Tüm bunlarla birlikte gerçek yaşamda program yazılırken tasarı ve üretim aşamaları zaman zaman iç içe geçebilir. Burada gösterildiği kadar ayrık bir şekilde ele alınmayabilir. Buna benzer şekilde tasarlama adımlarının her birinde öğrencinin mutlaka tanımlama ve fikir üretme adımlarını yapmasını beklemek gerçekçi olmayabilir. Fakat gene de problem çözümünden önce tasarımı

hakkında düşünmesi öğrenciler için faydalı olacaktır. Rehber öğretmen tasarla adımını öğrencileri sıkmayacak ve aynı zamanda verimli olacak şekilde uygulamalıdır.

ÜRET

Üret aşamasında öğrenciler aktif bir şekilde robotu ve programlama ortamını kullanarak çözüme ulaşmaya çalışırlar. Öğrenciler bu aşamada grup arkadaşı ile işbirlikli çalışmalıdır. Rehber öğretmen öğrencilerin sorularına doğrudan yanıtlar vermemelidir. Öğrencilerin çözümü kendi bulmaları ana hedeftir. Fakat onları çözüm konusunda düşündürecek şekilde yönlendirebilir. Bu aşamada rehber öğretmen kesinlikle tahta karşısına geçip çözümü öğrencilere göstermemelidir. Üret etkinliklerindeki amaç öğrencilerin üst seviye bilişsel etkinlikler içerisinde olmalarıdır. Çözüme ulaşmak ikincildir. Bu yüzden çözüm yerine çözüm sürecindeki öğrenci etkinlikleri gözlenmelidir. Rehber öğretmen öğrencileri ileri seviye düşünme becerilerini tatbik edecek şekilde yönlendirmelidir.

DEĞERLENDİR

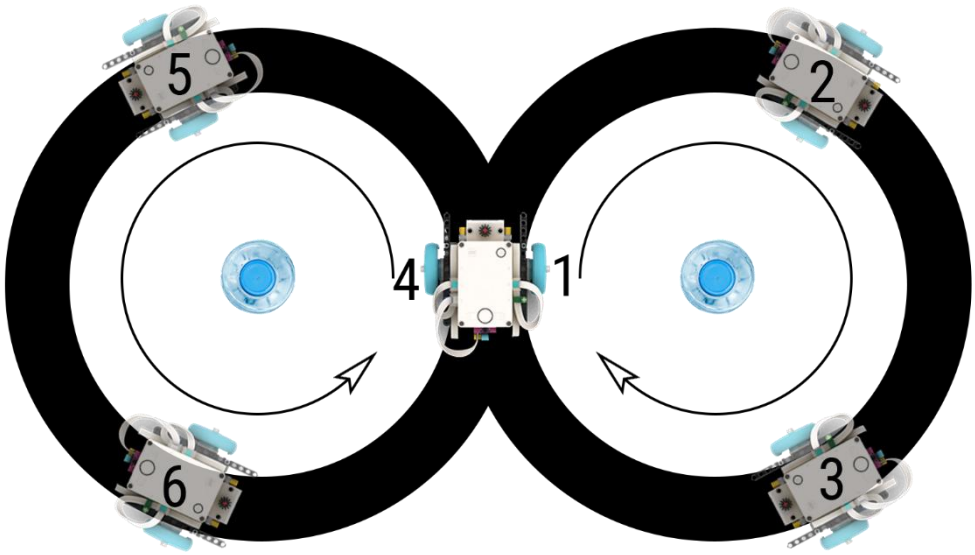
Günün sonunda öğrenciler ile halka oluşturulur ve aşağıdaki sorular tartışılır:

- Sürüş modelinde tekerlekler arasındaki mesafe artırılırsa move ve move_tank komutları eski durumda ürettiği sonuçları aynen üretir mi?
- Verilen görevleri göz önünde bulundurduğunuzda en çok hangi görevde zorlandınız? Bu zorlukların üstesinden nasıl geldiniz? (Problemin çözümü için hangi stratejileri kullandınız ve neden bu stratejileri seçtiniz?) Yeteri kadar tartışma ortamı oluşmazsa, rehber öğretmen aşağıdaki soruları kullanarak tartışma ortamı yaratmaya çalışır.
    - Robot hareketlerini keskin bir şekilde yerine getirmekte zorlandınız mı?
    - Komutları akılda tutmakta veya doğru yazmakta zorlandınız mı?
    - Cisimleri Kenardan Orta Kısma Taşıyorum etkinliğinde hangi komutları kullandınız. Bunları kullanma sebepleriniz nelerdir?
- Kullandığınız yöntemler, bu sıkıntıları gidermekte başarılı oldu mu?
- Grup arkadaşınızla fikir ayrılığına düştüğünüz durumlar oldu mu ve bunların üstesinden gelmek için neler yaptınız?
- Grup arkadaşınızdan ne öğrendiniz?
- Bir sonraki derse daha verimli bir öğrenme için neleri yapıp neleri yapmamanız gerekir?

İLAVE ETKİNLİK

Sonsuz Üzerinde Hareket

Zaman kalması durumunda bu etkinlik gerçekleştirilebilir. Bu etkinlikte robotun sonsuz benzeri bir şekilde hareket etmesi istenmektedir. Bunun için rehber öğretmen tarafından belirlenen iki nesne kullanılacaktır. Bu nesneler sonsuz işaretinin göz kısımlarını (beyaz renkli boşluklar) temsil etmektedir. Nesnelerin arasındaki mesafe 40-50 cm olabilir. Robotun bu nesnelerin etrafından sonsuz şekli çizerek ve dokunmadan hareket etmesi sağlanmalıdır. Bunun için örnek bir yol aşağıdaki resimde verilmiştir.



Resim 5.8 Robot Yolu

Programı oluşturmaya başlamadan önce grupların tasarlama adımı için yukarıda bir örneği verilen tanımlama ve fikir üretme süreçlerini gerçekleştirmeleri gerekir. İlave etkinlik bittiğinde fazladan zaman kalırsa iki nesne arasındaki mesafe artırılabilir veya azaltılabilir. Öğrenciler bu yeni durumlar için kodlarını değiştirmelidir.

6. Hafta – Işık Matrisi, Hoparlör ve Hub Düğmeleri

Haftanın Amacı:

Bu haftanın amacı öğrencilere ışık matrisini, hoparlörü ve Hub düğmelerini kullanmayı öğretmektir. Öğrencilerin ışık matrisini kullanarak hareketli yazı yazdırmaları, pikselleri ayrı ayrı kullanmaları, ışık matrisine şekil çizdirmeleri, hoparlörden farklı notalarda bip sesi çıkartmaları ve Hub düğmelerini kullanmaları beklenmektedir. Bunun yanında öğrenciler temel düzeyde değişken, döngü ve liste komutlarını kullanacaktır.

Haftanın Kazanımları:

- Öğrenciler ışık matrisini ve ilgili komutları tanır.
- Öğrenciler hoparlörü ve ilgili komutları tanır.
- Öğrenciler Hub düğmelerini ve ilgili komutları kullanır.
- Öğrenciler değişkenleri, döngüleri ve listeleri basit problemlerin çözümünde kullanır.
- Öğrenciler şimdiye kadar öğrendiği komutları bir araya getirerek problemler çözebilir.

Kullanılacak Malzemeler:

Robot seti, bilgisayar ve çalışma alanları.

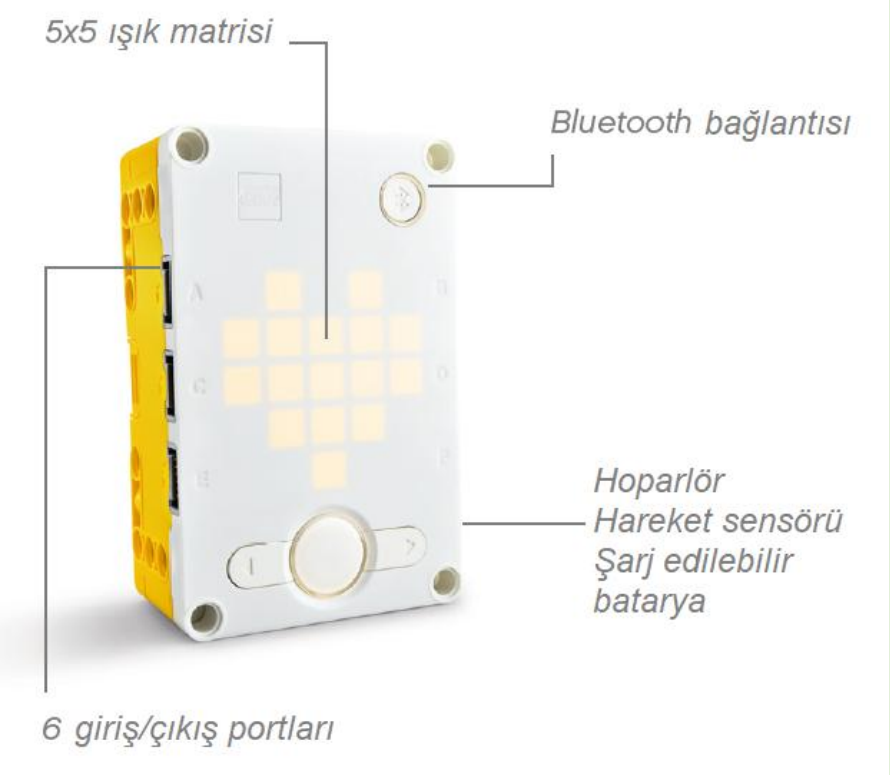
Ekler:

Resimler Listesi.pdf

GÖZLE VE UYGULA

Rehber Öğretmen İçeriği – Hub Hakkında

Lego Spike Prime'da robotun programlanabilir kısmına Hub (hab diye okunur) ismi verilmiştir. Motorlar ve sensörler Hub üzerinde bulunan 6 farklı port üzerinden bağlanabilir. Hub'da bulunan altı portun tamamı hem girdi hem de çıktı işlemleri için kullanılabilir. Hub üzerinde 5x5 ışık (LED) matris, üç buton, bir durum ışığı, bir hoparlör, USB ve Bluetooth bağlantı noktaları ve altı eksenli hareket sensörü bulunmaktadır.



Hub üzerinde Python programlarının çalıştırılması için MicroPython işletim sistemi yüklüdür. Python programlama dilinde, projeler oluştururken genellikle bir kütüphane (library) içe aktarmanız gerekir. Kütüphane, programınızda kullanabileceğiniz tüm olası "malzemeleri" içeren bir koleksiyondur. Yani, bir kütüphane, tarifinizi oluşturmak için kullanabileceğiniz tüm "malzemeleri" temsil eder.

Örneğin, "hub" Python modülü, SPIKE Prime hub'ı ile etkileşimde bulunmak için gerekli olan tüm ana fonksiyonları ve nesneleri içerir. Bu modül, robotun çeşitli bileşenleriyle (LED matris, butonlar, ses gibi) etkileşim kurmanıza olanak tanır.

Kütüphaneleri içe aktarmak için Python'da `import` ifadesini kullanırız. Örneğin, "hub" modülünden belirli bileşenleri içe aktarmak için şu şekilde yazabiliriz:

```
from hub import light_matrix    # LED matrisini kontrol etmek için
from hub import button         # Butonları kontrol etmek için
from hub import sound         # Ses çıkarmak için
```

Gözle: Işık Matrisinde Yazı Yazdırmak

Bu etkinlikte amaç Hub üzerinde bulunan ışık matrisinde sağdan sola doğru hareket eder şekilde yazılar yazdırmaktır. Bu iş için ışık matrisinin *light_matrix.write* metodu kullanılır. Işık matrisi ile ekranda her defasında bir karakter gösterilir ve karakterler sağdan sola doğru kayarak kelimeler gösterilir. Bu metot metin ve sayı cinsinden parametre alabilir. Ekrana bir metin, örneğin TÜBİTAK yazdırmak için aşağıdaki şekilde kod yazılmalıdır.

```
await light_matrix.write("TUBITAK")
```

Gözle: Değişkenlere Giriş

Işık matrisi ile ekrana sadece metinler yazdırılabilir. Python programlama dilinde çift veya tek tırnak içerisinde yazılan veriye metin cinsinden veri denilir. Bu veri tipi Python'da *str* (string'in kısaltması) şeklinde ifade edilir. Buna örnek olarak "Prime" verilebilir. Python'da sayısal veri tipleri de bulunmaktadır. Tam sayı cinsinden olan veriye *int*, ondalıklı veriye ise *float* isimleri verilmiştir. Int veya float cinsinden bir veriyi *write* metodu ile ışık matrisinde yazdırmak için öncelikle bu veri türleri metin haline dönüştürülmelidir. Örneğin aşağıdaki ifade ile tam sayı olan 5, metin olarak ışık matrisinde yazdırılır.

```
light_matrix.write("5") #ışık matrisine 5 yazdırır.
```

Python'da bazen program boyunca veya programın bir kısmında çeşitli verilere ihtiyaç duyulur. Bu verilerin değişme ihtimali bulunur. Python'da bu tip verilerin saklandığı programlama yapısına değişken ismi verilir. Günlük atılan adım sayısı, hava sıcaklığı ve yenen yemek miktarı değişkenlere günlük yaşamdan örnek olarak verilebilir. Programlamada değişkenler verimli bir şekilde kullanmak için isimlendirilirler. Örneğin aşağıdaki kod ile *str* cinsinden metin isimli bir değişken oluşturulmuştur ve bu değişkene değer olarak "TUBITAK" metni atanmıştır.

```
metin="TUBITAK"
```

Bu değişkenin içerisindeki değer değiştirilmek isteniyorsa ona yeni bir değer atanmalıdır. Örneğin aşağıdaki kod ile metin isimli değişkene "Kodlama" metni atanmıştır.

```
metin="Kodlama"
```

Aşağıdaki kod ile her defasında aynı komutla *light_matrix.write(metin)* ışık matrisinde metin yazdırılsa da ekrana önce TUBITAK ardından Kodlama yazdırılacaktır. Çünkü metin isimli değişkenin değeri değiştirilmiştir.

```
metin="TUBITAK"
await light_matrix.write(metin)
metin="Kodlama"
await light_matrix.write(metin)
```

Değişkenler sayısal değerler de alabilir. Aşağıda görüldüğü üzere *sinif_mevcudu* değişkenine tam sayı (*int*) değeri olan 20 atanmıştır. Değişkenlere 10.5 gibi ondalık (*float*) değerler de atanabilir.

```
sinif_mevcudu=20
```

Benzer şekilde int veya float veri türünde bir n değişkeni ışık matrisine yazdırılmak istenildiğinde aşağıdaki şekilde yazdırılabilir.

```
light_matrix.write(str(sinif_mevcudu)) # değişkeninin sayısal değeri
                                       ekrana yazdırılır
```

Gözle ve Uygula: 0'dan 5'e kadar sayıyorum

Rehber öğretmen öğrencilerden ışık matrisinde sırayla 0, 1, 2, 3, 4 ve 5 sayılarını gösteren kodu yazmalarını ister. Öğrenciler büyük ihtimalle aşağıdakine benzer bir kod yazacaktır.

```
light_matrix.write(0)
light_matrix.write(1)
light_matrix.write(2)
light_matrix.write(3)
light_matrix.write(4)
light_matrix.write(5)
```

Bu kod istenen görevi yerine getirmez. Çalıştırıldığında aralarda bekleme olmadığı için, hızlıca ilerler ve ışık matrisinde sadece 5 yazıldığı görülür. Rehber öğretmen öğrencilere problemin üzerinde uğraşması için yeteri kadar süre verir. Ardından, her bir yaz (*write*) komutundan sonra bekleme süresi konulması gerektiğini söyler. Python'da bulunan *time* modülü de bu iş için kullanılabilir. Time modülünün kullanılabilmesi için öncelikle yazdığımız programa dahil edilmesi gerekir. Bunun için "`import time`" komutu programın üst kısmına diğer *import* satırlarının hemen altına yazılmalıdır. Bundan sonra aşağıdaki komut ile istenen süre kadar işlemler duraklatılabilir.

```
time.sleep_ms (milisaniye cinsinden beklenecek süre)
```

Rehber öğretmen sonuç olarak aşağıdaki kodu yazar ve öğrencilerin de aynı kodu yazıp çalıştırmalarını sağlar.

```
from hub import light_matrix
import runloop
import time

async def main():
    light_matrix.write("0")
    time.sleep_ms(500) # 500 milisaniye (0.5 saniye) bekleme.
    light_matrix.write("1")
    time.sleep_ms(500)
    light_matrix.write("2")
    time.sleep_ms(500)
    light_matrix.write("3")
    time.sleep_ms(500)
    light_matrix.write("4")
    time.sleep_ms(500)
    light_matrix.write("5")
```

```
    time.sleep_ms(500)

runloop.run(main())
```

Gözle ve Uygula: 0'dan 5'e kadar sayıyorum

Bir önceki örneğimiz tekrar incelendiğinde altı defa aynı görev tekrarlanmaktadır: ekrana n sayısını yaz ve 500 milisaniye bekle. Programcılar bu tip tekrarlı durumlarda aynı kodu 6 farklı satırda yazmak yerine döngüleri kullanırlar. Döngüler isminden de anlaşılacağı üzere belirli bir sayıda tekrarlanması istenen veya belirli bir koşula bağlı olacak şekilde tekrar etmesi istenen komutlar için kullanılır. Python'da iki çeşit döngü bulunur. Bunlardan ilki *for* döngüsüdür. Aşağıdaki kod ışık matrisine 0 ile 5 arasındaki sayıları 0,5 saniye ara ile yazar.

```
for n in range(6):
    light_matrix.write(str(n))
    time.sleep_ms(500)
```

Rehber öğretmen bu kodu ekrana yansıtır ve öğrencilerden bu kod üzerinde çeşitli değişiklikler yaparak kodun kullanım kurallarını olabildiğince kendilerinin keşfetmelerini sağlar. Öğrencilere yeteri kadar süre verdikten sonra aşağıda verilen bilgileri kullanarak konuyu açıklar.

Range metodu ile belirli bir aralıkta sayı üretilmesi sağlanır. Örneğin, range(6) ifadesi ile 0-5 arasındaki tam sayılar üretilir. 0-6 arasındaki tam sayılar için range(7) ifadesi kullanılmalıdır. Görüldüğü gibi *range* metodu ile sayı üretme işlemi sıfırdan başlar ve metoda parametre olarak verilen sayının bir eksiğine kadar devam eder. Böylece istenen büyüklükte sayı üretilmiş olur. Aşağıdaki ifade ile ekrana üç kere TUBITAK yazılacaktır.

```
for n in range(3):
    await light_matrix.write("TUBITAK")
```

Burada dikkat edilmesi gereken önemli noktalar bulunmaktadır. Döngünün içerisinde bulunan ve döngü gövdesi olarak adlandırılan, tekrar etmesi istenen, komutları belirtmek için Python'da öncelikle *for* satırının sonuna iki nokta üst üste işareti ifadesi konulmuştur. Bu ifade döngü gövdesinin başlangıcını belirtir. Döngü gövdesindeki komutların belirlenmesi için ise *Tab* tuşu ile bir kere girinti yapılmalıdır. Yukarıdaki kodda görüldüğü üzere ışık matrisine TUBITAK yazılması için kullanılan komut *Tab* ile girintili olacak şekilde yapılmıştır. Çoğu zaman programcının girinti yapmasına gerek yoktur. Girinti işlemi Enter tuşuna basıldığında otomatik olarak Python programlama ortamları tarafından yapılmaktadır. Döngü n=0 için başlar döngü gövdesindeki komutlar, yani ışık matrisine TUBITAK yazılması, çalıştırılır. Ardından n=1 için döngü gövdesindeki komutlar çalıştırılır. Son olarak n=2 için döngü gövdesindeki komutlar çalıştırılarak döngüden çıkılır. Buradaki n bir değişkendir range(3) komutu ile sıralı olarak oluşturulan değerler her defasında n'ye atanır. Burada n yerine farklı bir isim de verilebilir. Örneğin aşağıdaki komut çalıştırıldığında konsol ekranına sırasıyla 0, 1 ve 2 basılır.

```
for i in range(3):
    print(i)
```

Range komutu ile 0'dan n'ye kadar ardışık sayılar üretilebildiği gibi 0'dan n'ye kadar belirli adım miktarında atlayarak sayılar da sayı üretilebilir. *Range* fonksiyonunun genel kullanımı aşağıdaki gibidir.

```
range(başlangıç, bitiş, adım)
```

Örneğin aşağıdaki kod çalıştırıldığında ışık matrisinde 0, 2, 4, 6, 8 sayıları görünecektir. Başlangıçtan itibaren teker teker değil adım sayısı olan ikişer ikişer ilerlenmiştir. Üçer üçer ilerlenmesi isteniyorsa adım parametresi yerine 3 girilmelidir. Burada başlangıcın dâhil olup bitişin dâhil olmadığını yeniden belirtmekte fayda bulunmaktadır. Aşağıdaki kod ile 10 değerinin de yazılması isteniyorsa bitiş parametresi yerine 11 yazılmalıdır.

```
for n in range(0,10,2):
    light_matrix.write(str(n))
    time.sleep_ms(500)
```

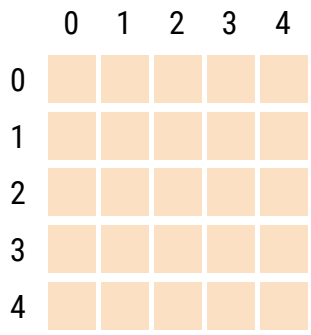
Uygula: Geri Sayım Yaptırıyorum

Rehber öğretmen öğrencilerden 5'ten 1'e kadar geri sayan ve bu sayıları ışık matrisinde gösteren programı yazmalarını ister. Bu iş için aşağıdaki kod kullanılabilir.

```
for n in range(5,0,-1):
    light_matrix.write(str(n))
    time.sleep_ms(1000)
```

Gözle: Piksel Kontrolü ile Işık Matrisi

Işık matrisi 5x5 boyutundadır. Yani 5 kolunu ve 5 satırı olmak üzere toplamda 25 tane hücresi bulunur. Buradaki hücrelerden her biri piksel (pixel) olarak adlandırılır. Bu durum aşağıda temsili olarak gösterilmiştir. Satırlar x eksenini sütunlar ise y esenini göstermektedir. Bu iki değer kullanılarak istenen piksele ulaşılabilir. Örneğin birinci satırın en sonundaki piksel için (0,4) değeri kullanılır.



Python ile ışık matrisinde bulunan piksellerin her birinin yanıp söndürülmesi ve parlaklığının ayarlanması mümkündür. Bunun için kullanılan metot aşağıda gösterilmiştir.

```
light_matrix.set_pixel(x, y, parlaklık(intensity))
```

Buradaki x değeri ulaşılmak istenen pikselin hangi satırda olduğunu, y değeri ise ulaşılmak istenen pikselin hangi sütunda olduğunu belirtmektedir. Bu yüzden her ikisi de 0-4 arasında değer alır. Parlaklık değeri ise 0-100 arasında değişebilir. 100 azami ölçüde parlaklık anlamına gelirken 0 hiç ışık verilmemesini temsil eder. Parlaklık değeri girilmediğinde ön tanımlı değer olarak 100 kabul edilir.

Işık matrisindeki bütün piksellerin bir anda söndürülmesi için *off* metodu bulunmaktadır. Aşağıdaki komut kullanılarak ışık matrisindeki bütün pikseller söndürülebilir.

```
light_matrix.off()
```

Rehber öğretmen ışık matrisindeki piksellerin nasıl yakıldığını anlattıktan sonra, öğrencilere aşağıdaki kodu gösterir ve açıklar. Kodu açıklarken iç içe döngüler konusuna değinir. Öğrencilerin iç içe döngüler konusunu anlaması önemlidir. Rehber öğretmen konu anlatımına gerekli süreyi ayırır. Bu kod ile bütün pikseller sırası ile teker teker yakılmış olur. İstenilirse 0.25 saniyelik bekleme süresi kaldırılarak tüm piksellerin birden yakılması sağlanabilir. Rehber öğretmen bu kodu öğrencilere açıklayarak anlatır.

```
    for x in range(5):
        for y in range(5):
            light_matrix.set_pixel(y,x,100)
            time.sleep_ms(250)
```

Uygula: Geri Sayım Öncesi Yanıp Sönme

Rehber öğretmen öğrencilerden 5'ten 1'e kadar geri sayım yapan programın geliştirilmesini ister. Geri sayıma başlanmadan önce bir defa bütün piksellerin yanıp sönmesi istenir. Bu görev için aşağıdaki kod kullanabilir.

```
from hub import light_matrix
import runloop
import time

async def main():
    for x in range(5):
        for y in range(5):
            light_matrix.set_pixel(x,y,100)
    time.sleep_ms(1000)
    for i in range(5,0,-1):
        light_matrix.write(str(i))
        time.sleep_ms(1000)

runloop.run(main())
```

Rehber öğretmen öğrencilere kodu tamamlaması için yeteri kadar süre verir. Ardından bu kodu biraz daha geliştirmelerini ister. Sayma öncesinde bütün pikseller bir kere değil üç kere yanıp sönmelidir. Bütün etkinlik bittikten sonra öğrencilere bütün pikselleri yakıp söndürme işlemini nasıl yaptıkları sorulur. Eğer bu iş için döngü kullanmamışlarsa tekrarlanan komutların programlamada döngüler vasıtası ile yapıldığını hatırlatır ve döngü içeren komutu yazar.

Gözle ve Uygula: Ekrana Resim Çiziyorum

Işık matrisinin her bir pikseli ayrı ayrı kullanılarak gülen yüz gibi şekiller yapılabileceği gibi daha önceden tanımlanan şekillerin ışık matrisinde gösterilmesi de sağlanabilir. Bu iş için *show_image* metodu kullanılır. Örneğin, daha önce de gösterildiği gibi, aşağıdaki komut kullanılarak ışık matrisinde gülen yüz gösterilir.

```
light_matrix.show_image(light_matrix.IMAGE_HAPPY)   #Resim dosyası adı
light_matrix.show_image(3)  #Resim dosyası numarası
```

Işık matrisi için gülen yüz dışında birçok şekil tanımlanmıştır. Bu şekillerin isimlerine ve numaralarına bu haftanın ekinde sunulan Resimler Listesi dokümanından veya program arayüzünde Bilgi Bankası > Api Modülleri > Hub > Işık Matrisi adımları ile ulaşılabilir.

Rehber öğretmen değişik şekillerin ışık matrisinde gösterilmesini sağlar ve kodu öğrencilere anlatır. Bunun ardından, rehber öğretmen öğrencilere Resimler Listesi dokümanını açarak ışık matrisinin *show_image()* metodu ile kullanabilecekleri resim/şekil listesinden farklı şekilleri keşfetmelerini ister. Onlara bu iş için yeterli süreyi verir.

Uygula: Ok Animasyonu

Rehber öğretmen öğrencilerden sırayla kuzey, doğu, güney ve batı yönlerini gösteren bir animasyon yapmalarını ister. Bu görev için aşağıdaki kod kullanılabilir.

```
from hub import light_matrix
import runloop
import time

async def main():
    light_matrix.show_image(light_matrix.IMAGE_ARROW_N)#Kuzey
    time.sleep_ms(500)
    light_matrix.show_image(light_matrix.IMAGE_ARROW_E)#Doğu
    time.sleep_ms(500)
    light_matrix.show_image(light_matrix.IMAGE_ARROW_S)#Güney
    time.sleep_ms(500)
    light_matrix.show_image(light_matrix.IMAGE_ARROW_W)#Batı
    time.sleep_ms(500)

runloop.run(main())
```

Gözle ve Uygula: Sürekli Tekrarlanan Görevler (While Döngüsü)

Python programlama dilinde ikinci döngü çeşidi *while* tipi döngüdür. Bu döngü tipinde işlemler koşula bağlı olarak tekrar edilir. *While* döngüsünün genel kullanımı daha sonra anlatılacaktır. Burada *while* döngüsünün özel bir durumu gösterilecektir. Bu özel durum sonsuz döngüdür. Hub açık olduğu müddetçe yapılması istenen komutlar sonsuz döngü içerisinde yaptırılır. Başka bir şekilde isimlendirmek gerekirse sonsuz döngü sürekli tekrarlanacak işlemleri gerçekleştirir. Blok tabanlı programlama araçlarında buna sürekli tekrarla tipi döngü denilir. Yukarıda yapılan ok

animasyonu bir kereye mahsustur. Bu animasyonun sürekli olması için aşağıdaki kod kullanılabilir.

```
from hub import light_matrix
import runloop
import time

async def main():
    while True:
        light_matrix.show_image(light_matrix.IMAGE_ARROW_N)
        time.sleep_ms(500)
        light_matrix.show_image(light_matrix.IMAGE_ARROW_E)
        time.sleep_ms(500)
        light_matrix.show_image(light_matrix.IMAGE_ARROW_S)
        time.sleep_ms(500)
        light_matrix.show_image(light_matrix.IMAGE_ARROW_W)
        time.sleep_ms(500)

runloop.run(main())
```

While döngüsünde de döngü gövdesinin başlangıcı için iki nokta üst üste ve döngü gövdesindeki komutları belirtmek için bir *Tab* girinti kullanılır. While tipi döngülerde *while* ifadesinden sonra koşul yazılır ve bu koşul doğru olduğu müddetçe döngü gövdesindeki işlemler yapılır. Burada *while*'dan sonra *True (Doğru)* yazıldığı için sürekli doğru olan bir koşul olduğu düşünülerek gövdede bulunan komutlar sürekli çalıştırılacaktır.

Rehber öğretmen sürekli tekrarlanan ok animasyonunu kullanarak öğrencilere *while* döngüsünü anlatır. Ardından öğrencilerden sürekli tekrarlanan başka bir animasyonu yapmalarını ister.

Gözle ve Uygula: Listeler

Bu derste daha önce anlatıldığı gibi değişkenler içerisinde bir defasında sadece bir değer tutulabilir. Python programlama dilinde birden farklı veriyi bir arada tutmak için liste ismi verilen veri yapıları kullanılır. Listeler içerisinde her türlü veri aynı anda tutulabilir. N elemanlı bir liste tanımlamak için aşağıdaki komut kullanılabilir. Görüldüğü üzere listelerde indeksleme 0'dan başlamaktadır. Rehber öğretmen listeler konusunu öğrencilere anlatıdır. Bunun için append ve clear gibi temel liste metotlarından da bahseder. Bu metotlar ilerideki etkinliklerde kullanılacaktır.

```
liste_ismi=[eleman0, eleman2…, elemanN-1]
```

Sürekli tekrarlanan görevler etkinliğinde dört farklı yönü gösteren ok kullanılmıştır. Bu ok'ların ismini tutmak için liste kullanılabilir.

```
oklar=[light_matrix.IMAGE_ARROW_N,light_matrix.IMAGE_ARROW_E,light_matrix.IMAG
E_ARROW_S,light_matrix.IMAGE_ARROW_W]
```

Bu kod bloğu ile kod isimleri aynı liste içerisinde tutulmaktadır ve bu listeye oklar ismi verilmiştir. Oklar isimli liste ve *for* tipi döngü ile kullanılarak daha gelişmiş kodlar yazılabilir. Örneğin sürekli

tekrarlanan görevler etkinliğinde okların gösterilmesi için ayrı ayrı dört *show_image* metodu kullanılmıştır. Bunun yerine *liste ve for* döngüsü birlikte kullanılabilir.

```
from hub import light_matrix
import runloop
import time

async def main():
    oklar=[light_matrix.IMAGE_ARROW_N,light_matrix.IMAGE_ARROW_E,light_mat
rix.IMAGE_ARROW_S,light_matrix.IMAGE_ARROW_W]
    while True:
        for ok in oklar:
            light_matrix.show_image(ok)
            time.sleep_ms(500)

runloop.run(main())
```

Rehber öğretmen yukarıdaki kodu öğrencilere açıklayarak anlatır. Ardından bir önceki etkinlikte kendilerinin geliştirdiği animasyonu liste kullanarak yeniden yapmalarını ister.

Gözle: Robotum Ses Veriyor

Hub üzerinde 12 Bit 16KHz (mono) ses kalitesine sahip bir hoparlör bulunmaktadır. Bu hoparlör ile farklı frekanslarda bip sesi çalmak mümkündür. Hub üzerindeki hoparlörü kullanabilmek için programa;

```
from hub import sound
```

komutu ile sound modülünün eklenmesi gereklidir. Bip sesi çalmak için iki metot bulunmaktadır. Bunlardan ilki *sound_beep* metodudur. Kullanımı aşağıdaki gibidir.

```
sound.beep(frekans, süre, ses yüksekliği(0-100))
```

stop() komutu ile *sound_beep* metodu ile başlanan bip sesi çıkarma işlemi sonlandırılabilir. Aşağıdaki komut ile bir saniye boyunca bip sesi çalınır. Rehber öğretmen öğrencilere kodu açıklayarak anlatır.

```
import runloop
from hub import sound

async def main():
    await sound.beep(440,1000,100) #frekans, çalma süresi, yükseklik

runloop.run(main())
```

Uygula: Kare Şeklinde Hareket Ederken Ses Çıkarıp Yön Gösteren Robot

Bu etkinlikte robot kare şeklindeki bir yol üzerinde hareket edecektir. Hareket ederken gittiği yönü ok işareti ile gösterecek ve köşelere geldiğinde bip sesi çalacaktır. Bu görev için aşağıdaki kod

kullanılabilir. Rehber öğretmen öğrencilerin bu görev için nasıl bir kod yazdığını gözlemler. Kodu yazarken döngü kullanmayan öğrenci gruplarını döngü kullanmaları konusunda yönlendirir.

```
from hub import light_matrix, sound, port
import motor_pair, runloop, time

async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    for i in range(4):
        light_matrix.show_image(light_matrix.IMAGE_ARROW_N)
        await motor_pair.move_for_degrees(motor_pair.PAIR_1, 360 , 0 ,
velocity=500)
        sound.beep(600)
        light_matrix.show_image(light_matrix.IMAGE_ARROW_W)
        time.sleep_ms(500)
        await motor_pair.move_for_degrees(motor_pair.PAIR_1, 180 , -100 ,
velocity=250)

runloop.run(main())
```

Gözle: Hub'ın Sol ve Sağ Düğmeleri

Hub üzerinde sağ ve sol düğme olarak adlandırılan iki buton bulunmaktadır. Bu butonların programlama dili alanında kullanılabilmesi için aşağıdaki gibi import edilmesi gerekmektedir.

```
from hub import button
```

Python kodu ile Hub üzerinde bulunan sol ve sağ düğmelerin basılıp bırakılma durumu algılanabilir ve bu durumlarda istenen işlemler gerçekleştirilebilir. Düğmelerden birine basılana kadar beklemek için aşağıdaki olay kullanılabilir.

```
button.pressed()
```

Hangi düğmenin basıldığının anlaşılabilmesi için ilgili düğmenin belirtilmesi gereklidir. Bu ifadeyi sol düğme ile kullanmak için *button.LEFT*, sağ düğme için kullanmak için *button.RIGHT* ifadesi kullanılır.

```
button.LEFT / button.RIGHT
```

Aşağıdaki kod çalıştırıldığında Hub üzerinde bulunan sol düğmeye her basıldığında robot 90 derece sola dönecektir. Rehber öğretmen kodu öğrencilere açıklayarak anlatır.

```
from hub import port, button
import runloop, motor_pair

async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    while True:
        if button.pressed(button.LEFT) == True:
            motor_pair.move_for_degrees(motor_pair.PAIR_1, 180,-100,
velocity=250)
```

```
runloop.run(main())
```

Uygula: Düğmeye Basılma Süresi

"Kare Şeklinde Hareket Ederken Ses Çıkarıp Yön Gösteren Robot" etkinliğinde sürüş modeli program robota yüklenir yüklenmez çalışmıştır. Burada öğrencilerden bu programı geliştirmeleri istenmektedir. Program yüklenir yüklenmez çalışmayacak bunun yerine Hub'ın sağ düğmesine basıldığında çalışacaktır. Rehber öğretmen bir olay gerçekleşinceye kadar programın nasıl bekletilebileceğini aşağıdaki örnek kodu vererek açıklar. Öğrencilere programlarında gerekli düzenlemeleri yapmaları için zaman tanır.

```
while not button.pressed(button.RIGHT):
    pass
```

Yukarıdaki kod içerisinde ilk defa kullanılan iki programlama yapısı bulunmaktadır. Bunlar *not ve pass* isimli yapılardır. Not mantıksal bir operatördür. İfadenin mantıksal olarak değilini alır. Yani eğer bir ifade *doğru (True*) ise değili *yanlış (False) yanlış ise değili* doğru olacaktır. Aşağıda verilen ifade kod içerisinde kullanılmıştır.

```
not button.pressed(button.RIGHT)
```

Burada *not* sonrasında bulunan ifade daha önce anlatıldığı gibi sağ düğmeye basıldığında doğru, aksi durumda yanlış değerini üretecektir. Bu ifadenin başına *not* getirilirse sağ düğmeye basıldığında yanlış basılmadığında ise doğru değerini üretecektir. Böylece sağ düğmeye basılmadığı müddetçe döngü gövdesindeki komutlar çalıştırılacaktır. Sağ düğmeye basıldığında döngüden çıkılır. Döngü gövdesinde sadece *pass* ifadesi bulunmaktadır. Bu ifade içinde bulunduğu blokta hiçbir işlem yapılmayacağı zaman kullanılır. Yani döngü çalıştığı müddetçe hiçbir işlem gerçekleştirme denilmektedir. Tüm döngü yeniden gözden geçirildiğinde sağ düğmeye basılmadığı müddetçe hiçbir işlem yapılmayacaktır, yani sağ düğmeye basılana kadar bekleyecektir.

TASARLA

Tasarla: Dans Eden Robot

Bu etkinlikte robotun dans edecek şekilde programlanması istenmektedir. Rehber öğretmen öğrencilere dans içerisinde hareket, müzik ve ışık matrisinin şekillerinin arka arkaya gelecek şekilde kullanılması gerektiğini belirtir. Öğrenciler robot kodunun nasıl olması gerektiği hakkında düşünür ve grup olarak olası çözümler hakkında tartışırlar. Tasarlama sürecinde öğrencilerden aşağıda örneklenen tanımlama ve fikir üretme adımlarına benzer bir süreci gerçekleştirmeleri beklenir. Rehber öğretmen burada sorumluluğu öğrencilere bırakarak öğretici değil yönlendirici bir yaklaşım sergilemelidir.

Tanımlama: Bu adımda dans işleminin neler gerektirdiği belirlenmelidir. Öğrenciler bu adımda bildikleri veya İnternetten araştırıp buldukları bir dans stilini kullanabilirler. Dans ederken yapılacak hareketler, çıkarılacak ses ve ışık matrisinde gösterilecek ifade belirlenebilir. Öğrenciler tüm bunları maddeler hâlinde detaylı olarak yazabilir.

- İstenen bir ses çalınır,
- Ses ile uyumlu dans figürleri yapılır (bir dans stili kullanılabilir),
- Işık matrisinde ses ve dansa bağlı olarak uygun ifadelerin belirmesi sağlanır,
- Dansın, müziğin ve ışık matrisindeki görüntülerin tekrarı sağlanır.

Fikir üretme: Bu aşamada öğrencilerden problem çözümüne yönelik daha somut öneriler ortaya koyması beklenmektedir.

- Ses çalmak için Hub hoparlörü kullanılabilir. Notalar yardımı ile istenen melodiler çalınabilir. Buna ek olarak bilgisayar da müzik çalmak için kullanabilir,
- Zeybek figürleri robotun programlanması için kullanılabilir. Motor komutları ile ilgili figürler oluşturulabilir,
- Gülen yüz, kalp ve ok resimleri dans hareketine bağlı olarak ışık matrisinde gösterilebilir,
- Dans hareketi, müzik ve ışık matrisi görüntüleri uyumlu olmalıdır,
- Bu görevler sonsuz döngü (while True:) vasıtası ile sürekli hale getirilebilir.

ÜRET

Öğrenciler bu aşamada tasarlama adımında ortaya koydukları bilgiler doğrultusunda programlarını yazmaya başlarlar. Öğrenciler problem çözümünde aktif rol üstlenmelidir. Rehber öğretmen öğretici rolünde değildir. Öğrencilerin sorularına doğrudan yanıtlar vermeyip yönlendirici olması yeterlidir. Öğrenciler problem çözerken yinelemeli çalışabilirler. Önce basit bir şekilde dans eden robot kodu geliştirler ve denerler. Ardından yazdıkları kodu biraz daha geliştirip gerçeğe benzer dans bileşenlerini oluştururlar. Son olarak robotun tam olarak dans etmesini sağlarlar. Rehber öğretmen yinelemeli çalışma konusunda öğrencileri yönlendirebilir. Bu etkinlikte kullanılacak dans bileşenleri öğrencilerin yaratıcılığına kalmıştır. Öğrenciler çok farklı dans bileşenleri kullanacağı gibi geleneksel halk danslarına yönelik program da oluşturabilirler. Rehber öğretmen öğrencileri kendi istedikleri yönde ilerlemesi için cesaretlendirmelidir.

DEĞERLENDİR

Günün sonunda öğrenciler ile halka oluşturulur ve ilk olarak aşağıdaki sorular tahtaya yansıtılarak tartışılır:

1. Aşağıdaki programda yanlış yazılan satırları bulup düzeltiniz.

```
from hub import sound
import runloop

async def main():
    for i in rang(4):
        sound.beep(440,1000,100)

runloop.run(main())
```

2. Aşağıdaki komut çalıştırıldığında kaç adet *bip* sesi duyulur?

```
from hub import sound
import runloop

async def main():
    # write your code here
    for i in range(4):
        sound.beep(440)

runloop.run(main())
```

Daha sonra rehber öğretmen aşağıdaki maddeleri kullanarak öğrencilerin sınıfça tartışmasını sağlar.

- Verilen görevleri göz önünde bulundurduğunuzda en çok hangi görevde zorlandınız? Bu zorlukların üstesinden nasıl geldiniz? (Problemin çözümü için hangi stratejileri kullandınız ve neden bu stratejileri seçtiniz?) Yeteri kadar tartışma ortamı oluşmazsa, rehber öğretmen aşağıdaki soruları kullanarak tartışma ortamı yaratmaya çalışır.
    - Döngüleri kullanmada zorlandınız mı?
    - Listeleri kullanmada zorlandınız mı?
    - Komutları akılda tutmakta veya doğru yazmakta zorlandınız mı?
    - Şarkı çaldırırken notaları bulmakta zorlandınız mı?

- Kullandığınız yöntemler, bu sıkıntıları gidermekte başarılı oldu mu?
- Grup arkadaşınızla fikir ayrılığına düştüğünüz durumlar oldu mu ve bunların üstesinden gelmek için neler yaptınız?
- Grup arkadaşınızdan ne öğrendiniz?
- Bir sonraki derse daha verimli bir öğrenme için neleri yapıp neleri yapmamanız gerekir?

7. Hafta – Kuvvet Sensörü

Haftanın Amacı:

Bu hafta, Lego Spike Prime robot setinde yer alan kuvvet sensörü öğrencilere tanıtılarak, kuvvet sensörünün; basılma ve bırakılma parametreleri ve ölçülen kuvvet miktarının farklı programlarda kullanılması sağlanacaktır.

Haftanın Kazanımları:

- Öğrenciler kuvvet sensörünün basılma ve bırakılma durumlarını, program akışında kullanabilir.
- Öğrenciler Boolean değişkenleri bilir.
- Öğrenciler koşul ifadelerini kullanabilir.
- Öğrenciler while döngüsünü kullanabilir.
- Öğrenciler kuvvet sensörü kullanarak, kuvvet ölçümü yapabilir.

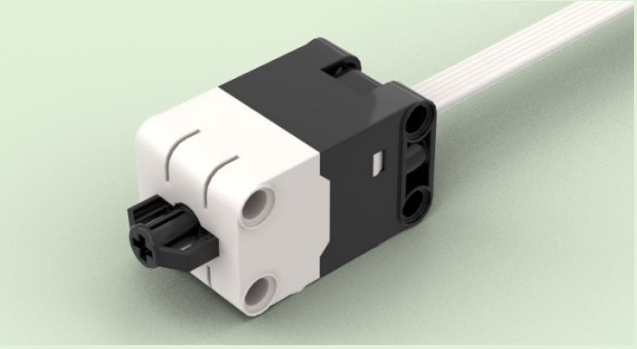
Kullanılacak Malzemeler:

Robot seti ve bilgisayar.

GÖZLE VE UYGULA

Rehber Öğretmen İçeriği – Kuvvet Sensörü Hakkında

Kuvvet sensörü, EV3 robot setinde bulunan dokunma sensörüne birçok yönden benzer. Ancak kuvvet sensörü, dokunma sensörün yapabileceğinden daha fazlasını yapar. Bu sensör, kullanıcının 10 Newton'a kadar kuvveti ölçmesini sağlar. Ayrıca bir dokunma sensörü gibi basılı olma durumunu algılayabilir. Kuvvet sensörü Resim 7.1'de gösterilmiştir. Kuvvet birimi Newton olup simgesi N'dir. Kütlesi 1 kg olan bir cismin hızını, 1 saniye içerisinde 1 m/s arttırmak için o cisme uygulanması gereken kuvvet, Newton cinsinden 1N = 1kg.m/s^2 olarak tanımlanır.



Resim 7. 1 Kuvvet sensörü

force_sensor modülü, kuvvet sensörünü kullanmak için gerekli olan tüm fonksiyonları ve sabitleri içerir. Kuvvet sensörünün kullanılabilmesi için projeye aşağıdaki gibi kuvvet sensörü modülünün eklenmesi gerekir.

```
import force_sensor
```

Kuvvet sensörünün pressed isimli bir metodu bulunur. Bu metot belirtilen porta bağlı kuvvet sensörünün butonuna basılıp basılmadığını kontrol eder. Butona basılı ise doğru (True), basılı değilse yanlış (False) değeri döndürür. Bu metodun kullanımı aşağıda gösterilmiştir.

```
force_sensor.pressed(port.F)
```

Sözcük Blokları programlama ortamında aşağıdaki resimde gösterildiği gibi basılı, sert basılı, bırakılmış ve basınç değişti gibi dört farklı durum vardır. Spike Prime Python bunlardan sadece basılı olma durumunu kontrol eder. Diğer durumlar ise yazılacak bir kod ile ele alınabilir.

Gözle: Kuvvet Sensörünü Tanımlayıp Kullanıyorum

Bu etkinlikte amaç kuvvet sensörünü kullanmaya yöneliktir. Aşağıdaki kod kuvvet sensörüne basılana kadar bip sesini çalmaktadır. Rehber öğretmen aşağıdaki kodu öğrencilere açıklayarak anlatır.

```
from hub import port, sound
import runloop
import force_sensor

async def main():
    while force_sensor.pressed(port.F)==False:
        sound.beep()

runloop.run(main())
```

Bu kod ile öncelikle hub ve kuvvet sönsörü oluşturulmuştur. While döngüsünde while ifadesi ile iki nokta üst üste ifadesi arasında kalan kısım (`force_sensor.pressed(port.F)==False`) döngünün koşul ifadesidir. Buna test ifadesi de denilebilir. Döngünün devam edip etmeyeceği buradaki test ifadesinin sonuçlarına bakılarak karar verilir. Test ifadesi doğru olduğu müddetçe döngü tekrar eder ve test ifadesi yanlış olduğu an döngüden çıkılır. Bu test ifadesi ile kuvvet sensörüne basılmama durumu kontrol edilmektedir. Kuvvet sensörüne basılmadığı müddetçe döngü gövdesini tekrar etme işlemi sürdürülecektir. Döngü gövdesini aşağıdaki komut oluşturmaktadır.

```
sound.beep()
```

Döngü gövdesini oluşturan komutlar bir veya birden fazla satırda olabilir. Gövdede bulunan komutları belirlemek için bir Tab büyüklüğünde girinti yapılır. Sonuç olarak kuvvet sensörüne basılmadığı süre boyunca arka arkaya bip sesi çalınacaktır. Kuvvet sensörüne basıldığında ise gövdenin tekrarı biter ve bip sesi kesilir.

Rehber öğretmen yukarıdaki kodu çalıştırıp öğrencilere anlattıktan sonra “for” döngüsü ile “while” döngüsü arasındaki farkları sorar. Grupların görüşlerini aldıktan sonra “for” döngüsünde range komutunda belirtilen sayıda döngü gövdesinin tekrar edildiğini, while döngüsünde ise koşul ifadesi doğru olduğu müddetçe döngü gövdesindeki komutların tekrar edildiğini vurgular.

Gözle: Koşul İfadeleri

Koşul ifadeleri ile belirli bir koşul gerçekleştiğinde çalışan programlama yapısından bahsedilir. Örnek vermek gerekirse ders notu 60 üzerinde olan öğrencilerin geçip altında olan öğrencilerin kaldığı bir puan sistemi için öğrencinin toplam notunu değerlendirerek, onun geçip geçmediğini bulan bir program yazmak koşul ifadeleri kullanılabilir. Aşağıda bu görev için sözde kod verilmiştir.

```
eğer toplam_puan>60:
    print(“Geçti”)
değilse:
```

```
    print("kaldı")
```

Spike Prime'dan bir örnek vermek gerekirse kuvvet sensörüne basıldığında ışık matrisinde kuzey yönlü ok basılmadığında ise güney yönlü ok gösterilmesi istenen program için aşağıdaki sözde kod kullanılabilir.

```
eğer force_sensor.pressed():
    light_matrix.show_image(light_matrix.IMAGE_ARROW_N)
değilse:
    light_matrix.show_image(light_matrix.IMAGE_ARROW_S)
```

Python'da bir koşulun gerçekleşip gerçekleşmediğini kontrol etmek için if-else programlama yapısı kullanılır. Koşulun gerçekleşme durumu *if* kısmında ele alınırken gerçekleşmeme durumu *else* kısmında ele alınır. *If-else* programlama yapısının kullanım şekli aşağıda gösterilmiştir.

```
if koşul:
    Koşul doğruysa çalıştırılacak komutlar
else:
    Koşul doğru değilse çalıştırılacak komutlar
```

Rehber öğretmen öğrencilere yukarıda anlatılanları kullanarak *if-else yapısını* açıklar. Koşul ifadelerinde de bir *tab* girintisi yapıldığını vurgular. Ardından aşağıdaki kodu öğrencilerin yazıp çalıştırmasını ister. Bu kod ile kuvvet sensörüne basıldığında ışık matrisinde kuzey yönlü ok basılmadığında ise güney yönlü ok gösteren programın yapılması tasarlanmıştır. Fakat kod bir kerelik çalıştığı için istenen görevi yerine getirmez. Öğrencilerin kodun neden görevi yerine getirmediğine dair tartışması sağlanır. Tartışma sonucunda rehber öğretmen sebebi öğrencilere açıklar.

```
from hub import port, light_matrix
import runloop, force_sensor

async def main():
    if force_sensor.pressed(port.F):
        light_matrix.show_image(light_matrix.IMAGE_ARROW_N)
    else:
        light_matrix.show_image(light_matrix.IMAGE_ARROW_S)

runloop.run(main())
```

Uygula: Sürekli Tekrar ve Koşul İfadesinin Birlikte Kullanımı

Yukarıdaki örnekte koşul ifadesi sürekli tekrarlanmadığı için istenen görevi gerçekleştirmez. İstenen görevin gerçekleştirilmesi için *while döngüsü* ile sürekli tekrarlama işlemi gerçekleştirilmelidir. Rehber öğretmen bu ipucunu öğrencilere verdikten sonra onlardan kuvvet sensörüne basıldığında ışık matrisinde kuzey yönlü ok, basılmadığında ise güney yönlü ok gösterilmesi istenen programı yazmalarını ister. Bu işe için örnek kod aşağıda verilmiştir.

```
    while True:
        if force_sensor.pressed(port.F):
            light_matrix.show_image(light_matrix.IMAGE_ARROW_N)
        else:
            light_matrix.show_image(light_matrix.IMAGE_ARROW_S)
```

Rehber öğretmen daha sonra öğrencilerden kuvvet sensörüne basıldığında aynı zamanda bip sesi çıkartmalarını ister. Bu iş için aşağıdaki kod örnek olarak verilebilir.

```
from hub import port, light_matrix, sound
import runloop, force_sensor

async def main():
    while True:
        if force_sensor.pressed(port.F):
            light_matrix.show_image(light_matrix.IMAGE_ARROW_N)
            sound.beep(440,100)
        else:
            light_matrix.show_image(light_matrix.IMAGE_ARROW_S)

runloop.run(main())
```

Uygula: Döngü, Koşul ve Değişkenin Bir Arada Kullanımı

Yukarıdaki kod ile kuvvet sensörü basılı olduğu durumda sürekli bip sesi çalmaktadır. Öğrenciler bu programı geliştirmelidir. Kuvvet sensörüne her basıldığında sadece bir kere bip sesi çalınmalıdır. Rehber öğretmen öğrencilerden gerekli kodu yazmalarını ister. Kuvvet sensörüne basılı olup olmadığını bulmak için değişkenler kullanılabilir. Rehber öğretmen gerektiğinde öğrencilere değişkenlerin kullanımına yönelik olarak ipucu verebilir.

Gözle: Miktar Belirtmeden Sürüş Modeli ve Motorları Hareket Ettirme

Daha önceki derslerde sürüş modelindeki motorların veya tek tek motorların nasıl hareket ettirileceği anlatılmıştır. Bu hareket belirli bir süre veya miktarda yapılmaktadır. Hareket miktarını sensörlerden gelen veriye bağlı olarak belirlemek için hareket işlemini süre veya miktardan bağımsız olarak yapmak gerekmektedir. Sürüş modelinin veya bir motorun ne kadar süre veya uzunlukta hareket edeceği belirtilmeden de hareket etmesi sağlanabilir. Aşağıda gösterilen *move* metodu ile sürüş modeli istenilen yön ve hızda hareket ettirilebilir.

```
motor_pair.move(motor_pair.PAIR_1, 0, velocity=500, acceleration=1000)
```

Görüldüğü üzere *move* metodunda hareket miktarına yönelik parametre bulunmamaktadır. Öncesinde tanımlanan motor çifti (motor_pair.PAIR_1), direksiyon yönü (0), hız ve ivme parametreleri bulunur. Örneğin aşağıdaki program çalıştırılırsa sürüş modeli varsayılan hızda sürekli ileriye doğru hareket edecektir.

```
from hub import port
import runloop, motor_pair

async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    motor_pair.move(motor_pair.PAIR_1,0)

runloop.run(main())
```

Sadece bir motorun istenen hızda harekete başlaması isteniyorsa yön parametresini kullanmaya gerek yoktur. Aşağıdaki kod kullanılırsa C portuna bağlı olan motor sürekli 200 derece/saniye hızında dönecektir.

```
from hub import port
import runloop, motor

async def main():
    motor.run(port.C,200)

runloop.run(main())
```

Sürüş modeli için sol ve sağ motorlara ayrı ayrı hız verilmesi de sağlanabilir. Bu iş için move_tank metodu kullanılır. Bu metodun kullanımı aşağıdaki gibidir.

```
motor_pair.move_tank(motor_pair.PAIR_1, 1000, 1000)
```

Sürüş modelinin veya bir motorun durdurulması için *stop()* metodunun (`motor_pair.stop(motor_pair.PAIR_1)` veya `motor.stop(port.A)`) kullanılması gerekir. Fakat *start* metodunun hemen arkasından stop kullanılırsa motorların hareketi gerçekleşmez. Durdurma işlemini belirli bir sensörden gelen değere veya bir süreye (`time.sleep_ms(500)` gibi) bağlamak yerinde olacaktır.

Aşağıdaki kod ile sürüş modeli 400 derece/saniye hızla ileriye doğru ilerler. Kuvvet sensörüne basıldığında robot duracaktır. Rehber öğretmen bu kodu öğrencilere açıklayarak anlatır.

```
from hub import port
import runloop, motor_pair, force_sensor

async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    motor_pair.move(motor_pair.PAIR_1,0, velocity=400)
    while not force_sensor.pressed(port.F):
        pass
    motor_pair.stop(motor_pair.PAIR_1)
```

```
runloop.run(main())
```

Yukarıdaki kod içerisinde geçen hafta da kullanılan düğmeye basılıncaya kadar programın bekletilmesine benzer şekilde, kuvvet sensörüne basılıncaya kadar programın bekletilmesi sağlanmıştır. *Pass* komutu yerine `time.sleep()` veya `runloop.sleep_ms()` komutları da kullanılabilir.

Gözle ve Uygula: Engele Çarpınca Geri Giden Robot

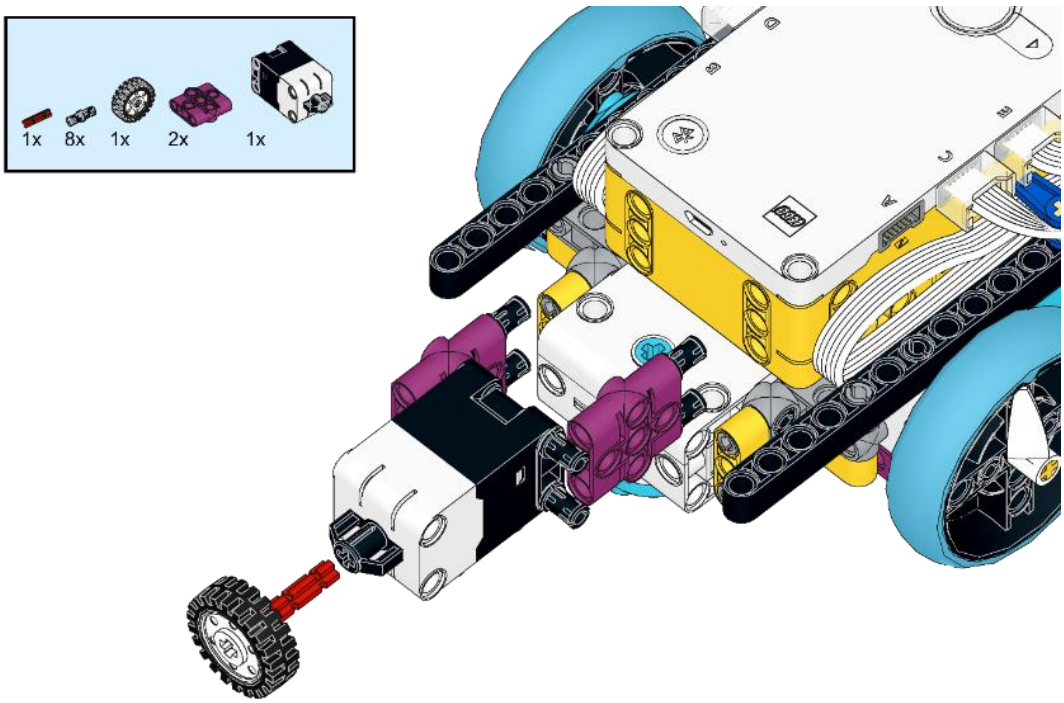
Öğrencilerden engele çarpınca geri dönen bir robot programlamaları istenir. Robot bu iş için aşağıdaki adımları takip etmelidir:

(i) Robotun bir engele çarpıncaya kadar düz gider.

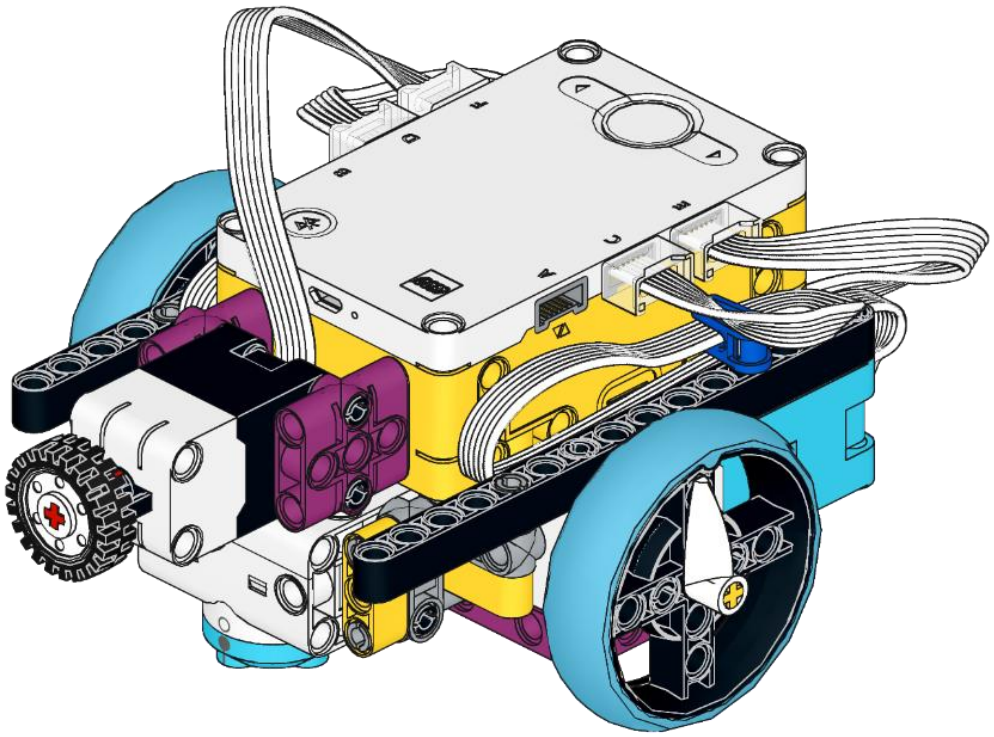
(ii) Robot engele çarpınca bir miktar geri gider.

(iii) Robot son olarak kendi etrafında dönme hareketini yaparak durmalıdır.

Bu görev için sürüş modeli Resim 7.2 ve 7.3' de gösterildiği gibi düzenlenmelidir.



Resim 7.2 Kuvvet sensörü montajı 1. adım



Resim 7.3 Kuvvet Sensörü Montajı 2. Adım

Bu görev için aşağıdaki kod kullanılabilir.

```
from hub import port
import runloop, motor_pair, force_sensor

async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    motor_pair.move(motor_pair.PAIR_1, 0, velocity=250)
    while True:
        while not force_sensor.pressed(port.F):
            pass
        await motor_pair.move_for_degrees(motor_pair.PAIR_1, 180, 0,
velocity=-250)
        await motor_pair.move_for_degrees(motor_pair.PAIR_1, 360, -100,
velocity=250)

runloop.run(main())
```

Uygula: Sürekli olarak Engele Çarptığında Geri Dönen Robot

Bu etkinlikte öğrencilerden önceki etkinliğe benzer bir robot programını yazmaları istenir. Fakat bu defa robot engele çarptığı her seferde geriye dönme işlemini yapmalıdır. Yani önceki görevde bir kez gerçekleştirilen görev sürekli gerçekleştirilmelidir. Bunun için öğrencilere sürekli tekrarla döngüsünün kullanılabileceği söylenebilir. Bu görev için örnek kod aşağıdadır.

```
from hub import port
import runloop, motor_pair, force_sensor

async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    while True:
        motor_pair.move(motor_pair.PAIR_1, 0, velocity=250)
        if force_sensor.pressed(port.F)==True:
            motor_pair.stop(motor_pair.PAIR_1)
            await motor_pair.move_for_degrees(motor_pair.PAIR_1, 180, 0,
velocity=-250)
            await motor_pair.move_for_degrees(motor_pair.PAIR_1, 360, -
100, velocity=250)

runloop.run(main())
```

Gözle: Kuvvet Sensörüne Her Basıldığında Bir Kere Bip Sesi Çalan Program

Bu etkinlikte öğrencilerden kuvvet sensörüne her basıldığında, basılma süresinden bağımsız olarak, sadece bir kere bip sesi çıkaran programı yazmaları istenmektedir. Bu programın yazılabilmesi için değişkenler kullanılarak kuvvet sensörünün önceki ve şimdiki durumu modellenebilir. Programda sadece sensöre basılı olup olmadığı kontrol edilirse, sensöre basılı olduğu müddetçe bip sesi çalınır. Fakat soruda sensöre her basıldığında bir kere bip sesinin çalınması istenmektedir. Bu durumda basılmama durumundan basılma durumuna geçiş kontrol edilmeli ve bu durum oluştuğunda bip sesi çıkarılmalıdır. Sensörün önceki durumunun basılı değil (*False*) ve şimdiki durumunun ise basılı (True) olması bize bu geçişi göstermektedir. Aşağıdaki kod bu düşünce temel alınarak yazılmıştır. Rehber öğretmen verilen kodu öğrencilere açıklayarak anlatır. Kodu açıklamadan önce öğrencilere *Boolean* yani sadece *True ve False* değeri alabilen değişkenleri anlatır. Bunun yanında aşağıdaki kod içerisinde *and* operatörü kullanılmıştır. Rehber öğretmen etkinlik öncesinde and operatörün kullanımını öğrencilere anlatır arından etkinliğe geçer.

```
from hub import port, sound
import runloop, force_sensor

async def main():
    onceki_durum = False #sensöre henüz basılmamış
    simdiki_durum = True #sensöre henüz basılmamış
    basili = False
    while True:
        simdiki_durum = force_sensor.pressed(port.F) #sensör değeri
okunuyor
```

```
        if onceki_durum == False and simdiki_durum == True: #sensöre
basıldı
            basili = True
        onceki_durum = simdiki_durum #durum güncellendi
        if basili:
            sound.beep()
        basili = False #yeniden basılma işlemi için gerekli

runloop.run(main())
```

Uygula: Kuvvet Sensörüne Her Basılıp Çekildiğinde Bir Kere Bip Sesi Çalan Program

Önceki etkinlikte sensöre her basıldığında sadece bir kere bip sesi çalınmıştır. Bu etkinlikte bip sesinin çalışma koşulu değiştirilmiştir. Kuvvet sensörüne basıldığında değil, sensöre basılıp çekildiğinde bip sesi çıkaran program yazılmalıdır. Sensörün önceki ve şimdiki hâli farklı şekilde modellenerek problem çözülebilir. Rehber öğretmen öğrencilerden gerekli kodu yazmalarını ister. Bu etkinlik için örnek kod bir sonraki sayfada verilmiştir.

```
from hub import port, sound
import runloop, force_sensor

async def main():
    onceki_durum = False
    simdiki_durum = False
    basilip_cekildi = False
    while True:
        simdiki_durum = force_sensor.pressed(port.F)
        if onceki_durum == True and simdiki_durum == False:
            basilip_cekildi = True
        onceki_durum = simdiki_durum
        if basilip_cekildi:
            sound.beep()
        basilip_cekildi = False

runloop.run(main())
```

Gözle ve Uygula: Uygulanan Kuvveti Ölçüp Işık Matrisinde Yazdırıyorum

Kuvvet sensörü, uygulanan kuvvet miktarını yüzde olarak ölçebilir ve bu değer 0 ile 100 arasında değişir. 0N için 0, 10 N için 100 ve 0-10 N arasında kalan kuvvet değerleri için orantısal değerler üretilir. Kuvvet ölçümü için kullanılan *force_sensor.force()* metodu, uygulanan kuvvet miktarını yüzde olarak döndürmektedir. Metodun kullanımı aşağıda verilmiştir.

```
force_sensor.force(port.F)
```

Sensör, maksimum 10 Newton'a kadar kuvvet ölçümü yapabilir; bu değerin üzerindeki kuvvetler ölçülemez. Ölçülen yüzde değeri, 10'a bölünerek Newton cinsinden kuvvet değeri elde edilebilir. Rehber öğretmen aşağıdaki kodu öğrencilere açıklayarak anlatır. Ardından öğrencilerden Newton cinsiden elde edilen değeri ışık matrisinde yazdırmaları istenir.

```
from hub import port, light_matrix
import runloop, force_sensor, time

async def main():
    while True:
        kuvvet_miktari = force_sensor.force(port.F)
        light_matrix.write(str(kuvvet_miktari))
        time.sleep_ms(1000)

runloop.run(main())
```

Uygula: Kuvvet Miktarına Göre Değişen Bip Sesi

Bu etkinlikte amaç kuvvet miktarı değiştikçe değişik frekanslar ile bip sesi çalan programı yazmaktır. Öğrenciler bu etkinlikte 0-100 arasında elde edilen kuvvet yüzdesi değerini 10 ile çarparak sound.beep metodunun frekans parametresine gönderebilirler. Buradaki 10 katsayı değeri değiştirilebilir.

```
from hub import port, sound
import runloop, force_sensor, time

async def main():
    while True:
        kuvvet=force_sensor.force(port.F)
        print(kuvvet)
        time.sleep_ms(100)
        sound.beep(10*kuvvet)

runloop.run(main())
```

Gözle: İki Olay Arasında Geçen Süreyi Saniye Cinsinden Hesaplıyorum

Python'da time modülü içerisinde time isimli bir fonksiyon bulunmaktadır. Bu fonksiyon 1 Ocak 1970 gece yarısından itibaren şimdiye kadar geçen süreyi saniye cinsinden vermektedir. Bu süreyi bulmak üzere aşağıdaki kod kullanılabilir.

```
import runloop, time

async def main():
    print(time.time())

runloop.run(main())
```

Görüldüğü üzere time metodunu kullanmak için öncelikle time modülü import komutu ile programa dâhil edilmelidir.

Aşağıdaki program sürüş modelinin tekerlerinin harekete başlamasından itibaren kuvvet sensörüne basılıp tekerlerin durdurulması arasında geçen süreyi milisaniye cinsinden ölçerek konsola yazmaktadır. *time.ticks_ms()* fonksiyonu kullanılarak milisaniye cinsiden geçen süre hesaplanmıştır. Süre farkını saniye cinsiden bulmak için elde edilen sonuç 1000 ile bölünmüştür. *round()* fonksiyonu ile gecen_sure değişkeni yuvarlanmıştır. Rehber öğretmen aşağıdaki kodu öğrencilere açıklayarak anlatır.

```
from hub import port
import runloop, force_sensor, time, motor_pair

async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    motor_pair.move(motor_pair.PAIR_1,0)
    baslama_zamani = time.ticks_ms()
    while force_sensor.pressed(port.F)==False:
        pass
    bitis_zamani = time.ticks_ms()
    motor_pair.stop(motor_pair.PAIR_1)
    gecen_sure = (bitis_zamani-baslama_zamani)/1000
    print(round(gecen_sure,1))
    time.sleep_ms(1000)

runloop.run(main())
```

Uygula: Sürüş Modelinin Maksimum Hızını Buluyorum

Sürüş modeli robotun hareketi için motorlara verilen hız miktarı sürüş modelinin gerçek hızını vermemektedir. Bu etkinlikte sürüş modeline maksimum güç verildiğinde (100) ve doğrusal olarak ileriye doğru hareket ederken gerçek hızının **cm/saniye** cinsinden bulunması istenmektedir. Rehber öğretmen öğrencilere sürüş modelinin bir tur ilerlediğinde teorik olarak 17,5 cm yol aldığını (zemine göre değişebilir) hatırlatır. Rehber öğretmen öğrencilerden gerekli kodu yazmalarını ister. Rehber öğretmen için örnek bir kod aşağıda verilmiştir.

```
from hub import port
import runloop, force_sensor, time, motor_pair, motor
async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    birinci = time.ticks_ms()
    await motor_pair.move_for_degrees(motor_pair.PAIR_1,360*4,0,
velocity=1050, acceleration=10000)
    ikinci = time.ticks_ms()
    gecen_sure = (ikinci-birinci) / 1000
    hiz = 70 / gecen_sure
    print(round(hiz,1))
runloop.run(main())
```

TASARLA

Kuvvet Sensörü Oyunu

Bu etkinlikte kuvvet sensörüne basılma miktarı ile orantılı olarak ilerleyen bir robot kodu yazılmalıdır. Oyunun oynanabilmesi için her bir öğrenci grubu kendi kodunu yazmalıdır. Bu kod çalıştırıldığında:

(i) Işık matrisinde gülen yüz şeklinde ışıklar yanacaktır.
(ii) Sürüş modeline sabitlenmiş kuvvet sensörüne basıldığında ışık matrisinde 3 ten geriye doğru sayılmaya başlanacaktır.
(iii) Bu süre zarfında sensöre basılı tutulmalıdır. Sıfır anında kuvvet sensöründen ölçüm yapılacaktır.
(iv) Ölçümün ardından hoparlörden bip sesi çalınacaktır. Bip sesi çalındığında kuvvet sensörü serbest bırakılmalıdır.
(v) Bip sesi bittiği anda sürüş modeli kuvvet sensöründen ölçülen basınç miktarı kadar cm ileri gitmelidir.

Rehber öğretmen bir hedef uzaklık belirleyip (0-100 cm arasında) belirlenen uzaklığa bir Lego parçası yerleştirecektir. Öğrenciler ise programı çalıştırıp kuvvet sensörüne uygulayacakları kuvvet ile sürüş modelinin istenen miktarda giderek hedefe ulaşmasını sağlayacaklardır. Robotu hedefe en çok yaklaştıran grup yarışmayı kazanır. Yarışma öncesinde gruplar hedefe yaklaşma denemeleri yapabilirler fakat yarışma esnasında her grubun sadece bir hakkı bulunmaktadır. İki grup arasında eşitlik olması durumunda hedefe daha hızlı varan grup birinci sayılacaktır. Hızlı grup sürüş modellerinin hareket hızına bakılarak karar verilebilir.

Tanımlama: Bu aşamada istenilenler belirlenmeli ve robotun kod ile neler yapılacağı tanımlamalı ve bunlar maddeler halinde sıralamalıdırlar. Rehber öğretmen için aşağıda bir örnek verilmiştir.

- Geri sayım için kuvvet sensörüne basılması beklenmeli,
- Bip sesi ile birlikte kuvvet sensöründen ölçüm alınmalı,
- Alınan ölçüm büyüklüğü kullanılarak sürüş modeli hareket ettirilmeli,
- Sürüş modelinin ani bir şekilde durması sağlanmalı,
- Sürüş modelinin hızı olabildiğince fazla olacak şekilde belirlenmeli. Fakat gereğinden fazla belirlenecek hız miktarının robotun görevi yerine getirmesini engelleyebileceği göz ardı edilmemelidir.

Fikir üretme: Bu aşamada öğrencilerden istenen kodun yazılabilmesi için, kullanılacak programlama yapılarına ve algoritmaya dair fikir üretmeleri beklenmektedir. Rehber öğretmen için aşağıda örnek bir liste verilmiştir.

- Kuvvet sensörüne basılana kadar bekle mantığı kullanılabilir,
- *for* döngüsü kullanılarak geri sayım işlemi gerçekleştirilebilir,
- force_sensor.force metodu ile kuvvet sensöründen ölçüm yapılabilir. Ölçüm değerleri 0-100 desinewton arasındadır. Elde edilen sonuç bir fonksiyon yazılarak cm cinsine çevrilebilir.
- Sürüş modeli keskin bir duruş için *hold* ile durdurulabilir,

ÜRET

Üret bölümünde öğrenciler tasarla bölümünde ortaya koydukları fikirleri kullanarak ilgili programı yazmalıdırlar. Diğer üret etkinliklerinde olduğu gibi kod yazma sorumluluğu öğrencilere aittir ve öğrenciler aktif rol üstlenmelidir. Rehber öğretmen doğrudan anlatan ve problem çözümünde aktif rol üstlenen bir yöntem izlememelidir. Bunun yerine öğrencileri yönlendirerek onlara rehberlik etmelidir. Öğrenciler grup olarak bilgisayar ve robot başında çalışarak gerekli kodu geliştirirler. Aşağıda rehber öğretmen için örnek bir kod verilmiştir.

```
from hub import port, light_matrix, sound
import runloop, force_sensor, time, motor, motor_pair

async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    motor_pair.stop(motor_pair.PAIR_1, stop=motor.HOLD)
    light_matrix.show_image(light_matrix.IMAGE_HAPPY)
    while force_sensor.pressed(port.F)==False:
        pass
    for i in range(3,-1,-1):
        light_matrix.write(str(i))
        time.sleep_ms(1000)
    kuvvet = force_sensor.force(port.F)
    sound.beep()
    light_matrix.write(str(kuvvet))
    tur = round(kuvvet/17.5)
    await motor_pair.move_for_degrees(motor_pair.PAIR_1,tur*360,0)

runloop.run(main())
```

DEĞERLENDİR

Günün sonunda rehber öğretmen öğrencilere aşağıdaki soruları yönelterek öğrenilen konuların birlikte özetlenmesini sağlar.

- Robot setinde bulunan kuvvet sensörünün özellikleri nelerdir?
- Günlük hayatta kullanılan dokunma/kuvvet/ağırlık sensörlerine örnekler veriniz? (Örneğin arabanın kapısı açık kaldığında uyarı amacıyla ses vermesi, asansörlerde yük fazla olması durumunda uyarı vermesi, arabalarda lastik basıncının ölçülmesi).
- Bugün öğrendiğiniz komutlar hangileridir? Görevleri nelerdir?
- Sizce bu hafta en zor görev hangisidir? Zorluğun üstesinden nasıl geldiniz?

İLAVE ETKİNLİK

Rehber öğretmen öğrencilere kuvvet sensörü oyununda uygulanan kuvvet oranında hareket sağlandığını belirtir. Bu oyunda ise sürüş modeli uygulanan kuvvet ve kuvvet sensörü bırakılıncaya kadar saniye cinsinden geçen süre kadar ileri gidecek şekilde geliştirmelidir. Takip edilmesi gereken adımlar aşağıda verilmiştir.

- Kuvvet sensörüne basılınca geri sayım işlemi başlayacaktır. Kuvvet sensörü serbest bırakılmamalıdır.
- Geri sayım işlemi bitince ışık matrisinde YES (✓) şekli belirmelidir. Bu şekil belirdiği andan itibaren süre ölçülmeye başlanacaktır.
- Kuvvet sensörü serbest bırakılınca toplam süre hesaplanacaktır (bunun için kronometre kullanılmamalıdır).
- Bip sesinin çalmasının ardından sürüş modeli geçen süre kadar ileri gitmelidir (kuvvet Newton ve süre saniye cinsinden).

Rehber öğretmen bir hedef uzaklık belirleyip örneğin 75 cm belirlenen uzaklığa bir Lego parçası yerleştirecektir. Öğrenciler ise programı çalıştırıp kuvvet sensörüne uygulayacakları kuvvet ve belirledikleri süre ile sürüş modelinin istenen miktarda giderek hedefe ulaşmasını sağlayacaklardır. Robotu hedefe en çok yaklaştıran grup yarışmayı kazanır. Gruplar arasında eşitlik olması durumunda hızlı olan grubun robotu yarışmayı kazanır. Yarışma öncesinde gruplar hedefe yaklaşma denemeleri yapabilirler fakat yarışma esnasında her grubun sadece bir hakkı bulunmaktadır. Rehber öğretmen için aşağıda örnek bir kod verilmiştir.

```
from hub import port, light_matrix, sound
import runloop, force_sensor, time, motor, motor_pair
async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    motor_pair.stop(motor_pair.PAIR_1, stop=motor.HOLD)
    light_matrix.show_image(light_matrix.IMAGE_HAPPY)
    while force_sensor.pressed(port.F)==False:
        pass
    for i in range(3,-1,-1):
        light_matrix.write(str(i))
        time.sleep_ms(1000)
    light_matrix.show_image(light_matrix.IMAGE_YES)
    birinci=time.ticks_ms()
    kuvvet = force_sensor.force(port.F)
    while force_sensor.pressed(port.F)==True:
        pass
    ikinci=time.ticks_ms()
    gecen_sure= ikinci-birinci
    sound.beep()
    light_matrix.write(str(gecen_sure//1000))
    await
motor_pair.move_for_time(motor_pair.PAIR_1,gecen_sure,0,velocity=200)
runloop.run(main())
```

8. Hafta – Hareket (Gyro) Sensörü

Haftanın Amacı:

Bu haftanın amacı öğrencilerin hareket (gyro) sensörü hakkındaki temel bilgi ve becerileri kazanmasını sağlamaktır. Hareket sensörünün üç eksenli dönme hareketi kullanılarak çeşitli etkinlikler yapılacaktır. Hareket sensörünün yön özelliği kullanılarak farklı programlar yazılacaktır. Hub hareketleri temel alınarak robot kodları yazılacaktır. Üç eksenli açı, yön ve hareket özellikleri Python programlama dili ile birlikte problem çözümünde kullanılacaktır. Öğrencilerden bu özellikleri kullanarak orta-ileri seviye algoritmalar geliştirmeleri ve kod yazmaları beklenmektedir.

Haftanın Kazanımları:

- Öğrenciler hareket sensörünü ve sensörün nasıl çalıştığını bilirler.
- Öğrenciler hareket sensörünün üç eksenli dönme açısını Python komutlarını kullanarak tespit eder ve program içerisinde kullanabilir.
- Öğrenciler sapma açısını kullanarak robotun çeşitli dönme hareketleri yapmasını sağlarlar.
- Öğrenciler Hub'ın yön bilgisini kullanarak orta-ileri seviyede programlar yazabilirler.
- Öğrenciler Hub'ın sallanama, tıklatılma ve düşme durumlarını Python programlama dilini kullanarak saptar ve bu bilgiyi problem çözümünde kullanabilir.

Kullanılacak Malzemeler:

Robot seti ve bilgisayar

GÖZLE VE UYGULA

Rehber Öğretmen İçeriği – Hareket (Gyro) Sensörü Hakkında

Gyro sensörü olarak da isimlendirilen hareket sensörü üç eksenli açısal hız ölçümü (jiroskop) ve üç eksenli ivme ölçümü olmak üzere toplamda altı çeşit ölçüm yapabilen bir sensördür. Ayrıca hareket sensörü Hub'ın sallandığı, düştüğü veya Hub'a dokunulduğu bilgisini Python komutları aracılığı ile programcıya sağlayabilmektedir. Bu tip sensörler günümüzde akıllı telefonlar, navigasyon sistemleri, oyun kumandaları ve robotlar gibi birçok teknolojik üründe kullanılmaktadır. Spike Prime robot setinde hareket sensörü Hub ile bütünleşiktir.

Hareket sensörü elektronik programlama bağlamında farklı bir anlama gelmektedir. Elektronik programlamada hareket sensörü, merdiven otomatlarında olduğu gibi, etrafta canlı veya cansız bir nesnenin hareket edip etmediğini kontrol etmek için kullanılır. Prime robot setinde ise hareket sensörü üç eksenli jiroskop ve üç eksenli ivme ölçer barındırmaktadır. Rehber öğretmen öğrencilere bu terminolojiden bahseder ve öğrencilere Elektronik Programlama ve Nesnelerin İnterneti dersinde kullanılacak olan hareket sensörünün buradakinden farklı olduğunu vurgular.

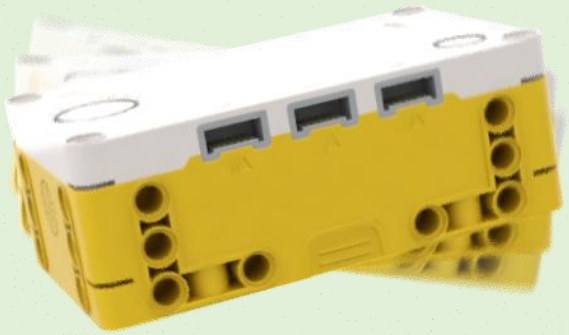
Spike Prime robot setinde Hub'ın üç eksenli açısal hareketi ve bu eksenlerde yaptığı açı değişim miktarı ile ilgili ölçümler yapılabilir. Üç eksende yapılan açısal hareket için sapma (yaw), yuvarlanma (roll) ve yunuslama (pitch) isimleri kullanılmıştır. Sapma (yaw) ismi sürüş modelinin tekerleri yerde iken sağa veya sola doğru yaptığı hareket için kullanılır bu hareket aşağıdaki resimde gösterilmiştir.

Resim 8.1 Sapma Açısı

Yuvarlanma bir merdanenin yuvarlanmasına benzetilebilir. Yuvarlanma sürüş modelinin sağ veya sol yanına yatmasını ifade etmek için kullanılır. Bu hareket için aşağıdaki resim örnek olarak verilebilir.

Resim 8.2 Yuvarlanma Açısı

Son hareket şekli ise yunuslama olarak adlandırılmıştır. Yunusların su yüzüne inip çıkarken yaptıkları harekete benzemektedir. Bu hareket şeklinde sürüş modelinin ön veya arkası kaldırılmaktadır. Yunuslama hareketi için aşağıdaki şekil örnek olarak gösterilebilir.



Resim 8.3 Yunuslama Açısı

Bu üç hareket şekli aslında üç boyutlu düzlemin her birindeki açısal hareketleri tarif etmek için kullanılmaktadır. Bunlar x, y ve z eksenlerindeki dönme hareketi olarak da adlandırılabilir. Öğrenciler x, y, z eksenlerini tam olarak gözünde canlandıramayabilir. Rehber öğretmen konuyu anlattıktan sonra https://www.youtube.com/watch?v=pQ24NtnaLI8 bağlantısındaki videoyu kullanarak öğrencilerin üç eksenli hareketi somutlaştırmasına yardımcı olabilir.

Gözle: Hub'ın Açısal Hareketlerini Öğreniyorum

Bu etkinlikte amaç Hub'ın açısal hareketlerini öğretmeye yöneliktir. Sapma (yaw), yuvarlanma (roll) ve yunuslama (pitch) hareketinin açısının alınması için tilt_angles metodu kullanılır. Açı değeri derece cinsindendir. Bu metodun kullanımı aşağıda gösterilmiştir.

```
motion_sensor.tilt_angles(sapma, yunuslama, yuvarlanma)
```

Rehber öğretmen öğrencilere sapma, yuvarlanma ve yunuslama hareketlerini göstererek anlatır ve gerektiğinde video ile öğrencilerin somut olarak bu hareketleri gözlemlemelerini sağlar. Bu hareketler sonucunda oluşan açıların değerlerinin alınması için kullanılması gereken komutları anlatır. Rehber öğretmen aşağıdaki kodu tahtaya yansıtır ve öğrencilerin bu kodu yazıp çalıştırmalarını ister. Öğrencilerin Hub'ı ilgili eksenlerde döndürerek onların bu kod ile sapma, yuvarlanma ve yunuslama açılarındaki değişimi gözlemlemelerini sağlar.

```
from hub import motion_sensor
import runloop, time

async def main():
    while True:
        egme_acilari = motion_sensor.tilt_angles()
        print("Sapma, Yunuslama, Yuvarlanma Açısı: ", egme_acilari)
        time.sleep_ms(500)

runloop.run(main())
```

Not

Sapma, yunuslama ve yuvarlama açılarını bireysel olarak öğrenmek için tilt_angles()[0], tilt_angles()[1], tilt_angles[2] komutları kullanılmaktadır.

motion_sensor.tilt_angles () metodu ile okunan açı değerleri desi derece (desidegrees) cinsindendir. 10 desi derece 1 derecedir.

Rehber öğretmen, yukarıda verilen koda ek olarak, aşağıda sapma açısını için örneği sunulan program gibi teker teker sapma, yunuslama ve yuvarlanma açılarını veren bir programı kullanabilir.

```
from hub import motion_sensor
import runloop, time

async def main():
    while True:
        sapma = motion_sensor.tilt_angles() [0]
        print("Sapma Açısı: ", sapma)
        time.sleep_ms(500)

runloop.run(main())
```

Bu üç farklı hareket şeklinden sadece sapma hareketinde açı değerinin sıfırlanması sağlanabilir. Sürüş modeli ilk çalıştırıldığında sapma açısı 0 derece olarak belirlenir. Daha sonraki hareketlerinde burayı başlangıç noktası olarak kabul eder ve sapma açısını ona göre belirler. Bazı durumlarda sapma açısının sıfırlanması gerekebilir. Bunun için *reset_yaw(angle: int)* metodu

kullanılır. Rehber öğretmen aşağıdaki kodu öğrencilere açıklayarak anlatır. *Reset_yaw* metodu sayesinde son verilen açı miktarının bir anda değişiklik gösterdiğini vurgular.

```
from hub import motion_sensor, port
import runloop, motor_pair, time

async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    while motion_sensor.tilt_angles() [0] <= 1600:
        motor_pair.move_tank(motor_pair.PAIR_1, 0, 100)
        print(motion_sensor.tilt_angles() [0])
    motor_pair.stop(motor_pair.PAIR_1)
    motion_sensor.reset_yaw(0)
    time.sleep_ms(500)
    print(motion_sensor.tilt_angles() [0])

runloop.run(main())
```

Gözle ve Uygula: Sürüş Modelini 90 Derece Döndürüyorum

Bu etkinlikte sürüş modelinin 1 tur düz gittikten sonra 90 derece sağa dönmesini gerekmektedir. Fakat dönme işlemi için hareket sensörünün sapma açısı kullanılmalıdır. Aşağıdaki kod bu iş için kullanılabilir.

```
from hub import motion_sensor, port
import runloop, motor_pair, time

async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)# motorları tanıtma
    motion_sensor.reset_yaw(0) #sıfırlama
    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 360, 0,
velocity=200) #bir tur ileri
    while motion_sensor.tilt_angles()[0]<900: # 90 dereceyi ifade eder.
        motor_pair.move(motor_pair.PAIR_1, -100) #sağa döndürme
    motor_pair.stop(motor_pair.PAIR_1) #motoru durdurma

runloop.run(main())
```

Rehber öğretmen bu kodu çalıştırır ve sapma açısının dönme sonucunda 90 dereceden biraz daha büyük bir değere sahip olduğunu konsol ekranını ekrana yansıtarak gösterir. Öğrenciler ile bu değerin neden 90 dereceden büyük olduğu ve nasıl 90 değerine getirebileceği konusunda tartışılır.

Not

Sensör tam olarak 90 dereceyi ölçtüğü anda ölçülen verinin aktarılması ve ardından motorlara durma komutunun gönderilmesi biraz zaman alır. Ayrıca robot kendi etrafında dönerken oluşan dönme kuvveti, motorlar durdurulduktan sonra da robotun biraz daha dönmesine sebep olur. Robotun 90 dereceden biraz daha az bir açı ile dönecek şekilde programlanması bir çözüm olabilir.

Uygula: Kare şeklinde hareket

Bu etkinlikte bir kare şeklinde hareket edecek olan robotun programı yazılacaktır. Bu görev için for döngüsünün içerisinde while döngüsü kullanılacaktır. Bu da bir iç içe döngüdür. Rehber öğretmen öğrencileri iç içe döngü kullanma konusunda yönlendirir. Rehber öğretmen için örnek bir program aşağıda sunulmuştur.

```
from hub import motion_sensor, port
import runloop, motor_pair

async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D) # motorları tanıtma
    for i in range(4):
        motion_sensor.reset_yaw(0)
        await motor_pair.move_for_degrees(motor_pair.PAIR_1, 360*2, 0,
velocity=200) #bir tur kadar motoru 200 hızda döndürme
        while motion_sensor.tilt_angles()[0]<900: # 90 derece için
            motor_pair.move(motor_pair.PAIR_1, -90)#sola döndürme
        motor_pair.stop(motor_pair.PAIR_1) #motoru durdurma

runloop.run(main())
```

Rehber Öğretmen İçeriği –Hub'ın Yönleri

Hub'ın ön, arka, sağ, sol, üst ve alt olmak üzere altı farklı yönü bulunmaktadır. Hub'ın yönleri aşağıdaki resimde gösterilmiştir.

Resim 8.4 Hub'ın Yönleri

Gözle: Hub Yönünün Algılanması

Hub'un yukarı bakan yüzeyi için kullanılan metot up_face'dir. Bu metodun kullanımı aşağıda gösterilmiştir.

```
motion_sensor.up_face()
```

Bu metot altı farklı değer üretir.

- motion_sensor.TOP: Işık Matrisinin bulunduğu hub'un üst yüzeyi. – 0
- motion_sensor.FRONT: Hoparlörün bulunduğu hub'un ön yüzeyi. – 1
- motion_sensor.RIGHT: Ön yüzeye bakıldığında hub'un sağ tarafı. – 2
- motion_sensor.BOTTOM: SPIKE Prime hub'un bataryanın bulunduğu alt tarafı. – 3
- motion_sensor.BACK: USB şarj portunun bulunduğu hub'un arka yüzeyi. – 4
- motion_sensor.LEFT: Ön yüzeye bakıldığında hub'un sol tarafı. – 5

Rehber öğretmen öğrencilerden aşağıdaki kodu yazıp çalıştırmalarını ister. Öğrencilerin Hub yönünü çalışan program yardımıyla anlamlandırmalarını sağlar.

```
from hub import motion_sensor
import runloop, time

async def main():
    while True:
        print(motion_sensor.up_face()) #konsola sayı cinsinden yönü yazar
        time.sleep_ms(1000)

runloop.run(main())
```

Uygula: Sürekli Yukarıyı Gösteren Ok Resimleri

Bu etkinlikte Hub'ın ekranında sürekli yukarıyı (sınıfın tavanını) gösterecek bir ok sembolü için gerekli program yazılacaktır. Robotun duruş şekli değiştikçe okun yönü de değiştirilerek robotun her konum için yukarıyı (sınıfın tavanını) göstermesi sağlanmalıdır. Rehber öğretmen için aşağıda örnek bir program verilmiştir.

```
from hub import motion_sensor, light_matrix
import runloop, time

async def main():
    while True:
        yon=motion_sensor.up_face()
        if yon == 4: # 4 yerine motion_sensor.BACK de yazılabilir
            light_matrix.show_image(light_matrix.IMAGE_ARROW_N)
        if yon == 1: # 1 yerine motion_sensor.FRONT da yazılabilir
            light_matrix.show_image(light_matrix.IMAGE_ARROW_S)
        if yon == 2:
            light_matrix.show_image(light_matrix.IMAGE_ARROW_E)
        if yon == 5:
            light_matrix.show_image(light_matrix.IMAGE_ARROW_W)
        if yon == 3:
            light_matrix.set_pixel(2,2,100) #Orta Nokta, ok arka görünüm
        if yon == 0:
            for i in range(5): #Çapraz, önden görünüm temsili
                for j in range(5):
                    if i==j or i+j==4:
                        light_matrix.set_pixel(i,j, 100)
        time.sleep_ms(200)
        light_matrix.clear()

runloop.run(main())
```

Uygula: Yunuslama Açısına Göre İleri Geri Dönen Tekerlekler

Bu etkinlikte öğrencilerden yunuslama açısına göre tekerleklerin dönüş yönünü belirleyen ve onları hızlandırıp yavaşlatan bir program yazmaları istenmektedir. Yunuslama açısı 0 ise tekerlekler duracaktır, pozitif yönlü ise ileri doğru gidecektir, negatif yönlü ise geri doğru gidecektir ve hareket hızı yunuslama açısının büyüklüğü ile orantılı olacak şekilde değişecektir. Robotun hareketi, yunuslama açısının konsolda yazan değerin pozitif veya negatif olmasına göre değişmektedir. Rehber öğretmen için aşağıda örnek bir kod verilmiştir.

```
from hub import motion_sensor, port
import runloop, motor_pair, time
```

```
async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    while True:
        yunuslama_acisi = motion_sensor.tilt_angles()[1]
        motor_pair.move(motor_pair.PAIR_1, 0, velocity=yunuslama_acisi)
        print(yunuslama_acisi)
        time.sleep_ms(100)
runloop.run(main())
```

Not

Bu etkinlikte öğrenciler robotlarını ellerinde tutarak farklı yunuslama açıları oluşturacaktır.

Gözle: Hub ile Yön Değişimine Bağlı İşlem Yapma

Bazen yapılmak istenen işlemler Hub'ın yönü yerine, yön değişimine bağlı olarak gerçekleştirilmek istenebilir. Bu durumlarda Hub'ın yönü değişene kadar beklemeyi sağlayacak bir yöntem faydalı olacaktır. İstenilen olay gerçekleşinceye kadar program akışının bekletilmesi gerekecektir.

Rehber öğretmen aşağıdaki kodu öğrencilere vermeden önce *if-elif-else* seçim ifadesini öğrencilere anlatır. Ardından kodu tahtaya yansıtır ve öğrencilerden kodu yazıp, çalıştırıp sonuçlarını deneyimlemelerini ister. Kodun ne iş yaptığı sınıfça tartışılır.

```
from hub import motion_sensor
import runloop

async def main():
    while True:
        onceki_yon = motion_sensor.up_face()
        while onceki_yon == motion_sensor.up_face():
            await runloop.sleep_ms(10)
        yeni_yon=motion_sensor.up_face()
        print("Yön değişti")
        if yeni_yon== 0: # 0 yerine motion_sensor.TOP da yazılabilir
            print("Üst yukarıda")
        elif yeni_yon== 1:
            print("Ön yukarıda")
        elif yeni_yon== 2:
            print("Sağ yukarıda")
        elif yeni_yon== 3:
            print("Alt yukarıda")
        elif yeni_yon== 4:
            print("Arka yukarıda")
        elif yeni_yon== 5:
            print("Sol yukarıda")
        onceki_yon = yeni_yon
runloop.run(main())
```

Uygula: Sürekli Yukarıyı Gösteren Ok Resimleri - Yeni Versiyon

Sürekli yukarıyı gösteren ok resimleri isimli etkinlikte öğrencilerden Hub'ın ekranında sürekli yukarıyı (sınıfın tavanını) gösterecek bir ok sembolü bulunan programı yazmaları istenmiştir. Işık matrisinde ok gösterme işlemi 0,2 saniye aralıklarla gerçekleştirilmiştir. Bu etkinlikte 0,2 saniye gibi bir süre beklemeden Hub'ın yönü değiştiğinde yeni ok yönü ışık matrisinde gösterilmelidir. Bu görev için önceki etkinlikte kullanılan programın bekletilmesi yöntemi kullanılabilir. Rehber öğretmen için örnek kod aşağıda sunulmuştur.

```
from hub import motion_sensor, light_matrix
import runloop

async def main():
    yon=motion_sensor.up_face()
    while True:
        while True:
            if yon!=motion_sensor.up_face():
                break
        light_matrix.clear()
        yon=motion_sensor.up_face()
        if yon == 4: # 4 yerine motion_sensor.BACK de yazılabilir
            light_matrix.show_image(light_matrix.IMAGE_ARROW_N)
        elif yon == 1: # 1 yerine motion_sensor.FRONT da yazılabilir
            light_matrix.show_image(light_matrix.IMAGE_ARROW_S)
        elif yon == 2:
            light_matrix.show_image(light_matrix.IMAGE_ARROW_E)
        elif yon == 5:
            light_matrix.show_image(light_matrix.IMAGE_ARROW_W)
        elif yon == 3:
            light_matrix.set_pixel(2,2,100) #Orta Nokta, ok arka görünüm
        elif yon == 0:
            for i in range(5): #Çapraz, ok önden görünüm temsili
                for j in range(5):
                    if i==j or i+j==4:
                        light_matrix.set_pixel(i,j, 100)
runloop.run(main())
```

Gözle: Hub'a Tıklama, Hub'ı Sallama ve Düşürme Hareketleri

Hareket sensörü Hub üzerine tıklama, çift tıklama, Hub'ın sallanması ve düşürülmesi gibi hareketleri algılayabilmektedir. Hub hareketlerinin algılanabilmesi için *gesture* metodu kullanılabilir. Bu metodun kullanımı aşağıda örneklendirilmiştir.

```
hub.motion_sensor.gesture()
```

gesture metodu aşağıda sırlanan değerleri döndürür.

- motion_sensor.TAPPED : tıklandı – 0
- motion_sensor.DOUBLE_TAPPED : çift tıklandı – 1

- motion_sensor.SHAKEN : sallandı – 2
- motion_sensor.FALLING : düşüyor – 3
- motion_sensor.UNKNOWN : hareket yapılmadı veya algılanmadı – -1

Rehber öğretmen öğrencilerin aşağıdaki kodu yazıp çalıştırmalarını ve sonuçlarını deneyimlemelerini sağlar.

```
from hub import motion_sensor
import runloop, time

async def main():
    while True:
        hub_hareket = motion_sensor.gesture()
        if hub_hareket != motion_sensor.UNKNOWN:
            print(hub_hareket)
            time.sleep_ms(100)
runloop.run(main())
```

Öğrenciler kodun çalışmasını deneyimlerken şu hususlara dikkat etmelidirler:

- Tıklama için biraz sert vurmak gerekmektedir. Tıklamayı (“tapped”) deneyimlemek için yumuşak bir dokunuş ile başlanıp yavaş yavaş sert vurmaya doğru ilerlenerek eşik değeri bulunabilir.
- Çift tıklama (“doubletapped) için daha önce kullanılan tıklama sertliğinde ve hızlı bir şekilde iki tıklama yapılmalıdır. Aksi takdirde çift tıklama gözlenemeyebilir.
- Düşme hareketi için öğrenciler robotu **yumuşak** bir zemine 20 cm gibi bir mesafeden bırakmalıdırlar. Daha yüksek değerler robota zarar verebilir.

Uygula: Hub Hareketine Göre Ses Çıkaran Robot

Bu etkinlikte Hub’ın farklı hareketlerine göre hoparlörden farklı frekanslarda ses çıkaran robot programı yapılacaktır. Öğrenciler tıklama, çift tıklama, sallama ve düşme hareketleri için farklı frekanslarda sesler belirlemelidir. Bu görev için aşağıda örnek kod verilmiştir.

```
from hub import motion_sensor, sound
import runloop

async def main():
    while True:
        hub_hareketi = motion_sensor.gesture()
        if hub_hareketi==motion_sensor.TAPPED:
            sound.beep(200)
        if hub_hareketi==motion_sensor.DOUBLE_TAPPED:
            sound.beep(400)
        if hub_hareketi==motion_sensor.SHAKEN:
            sound.beep(600)
        if hub_hareketi==motion_sensor.FALLING:
            sound.beep(800)
runloop.run(main())
```

Gözle: Tıklama Sayısı

Bir program başlatıldıktan sonra algılanan her tıklanma, hub tarafından sayılır ve `tap_count()` metodu ile toplam tıklanma sayısı elde edilebilir. Program içerisinde istenildiğinde `reset_tap_count()` metodu ile *top_count* fonksiyonunun döndürdüğü tıklanma sayısı sıfırlanabilir. Bu iki metodun kullanımı şu şekildedir;

```
motion_sensor.tap_count()
motion_sensor.reset_tap_count()
```

Uygula: Tıklanma Sayısı Kadar İleri Giden Robot

Bu etkinlikte Hub istenilen sayıda tıklanılacak ve sol tuşa basıldığında, robot tıklanma sayısı kadar tur ileri gidecektir. Her tıklamada, robot ekranında tıklanma sayısı yazdırılarak sayı takip edilebilecektir. Sürüş Modeli hareketini tamamladıktan sonra, tekrar tıklanıp sol tuşuna basıldığında tekrar tıklanma sayısı kadar ileri gidebilecektir. Rehber öğretmen öğrencilere çözülmesi istenen problemi anlatır. Rehber öğretmen için örnek program bir sonraki sayfada verilmiştir.

```
from hub import motion_sensor, light_matrix, port, button
import runloop, motor_pair, time

async def main():
    tiklanma_sayisi=0
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    motion_sensor.reset_tap_count()
    while True:
        tiklanma_sayisi=motion_sensor.tap_count()
        if button.pressed(button.LEFT)==False:
            await runloop.sleep_ms(10)
        else:
            light_matrix.write(str(tiklanma_sayisi))
            tiklanma_sayisi=motion_sensor.tap_count()
            await
motor_pair.move_for_degrees(motor_pair.PAIR_1,360*tiklanma_sayisi,0)
            motion_sensor.reset_tap_count()

runloop.run(main())
```

TASARLA

Hareket Sensörü ile Programlanabilen Robot

Bu etkinlikte öğrenciler programlanabilen bir robot geliştireceklerdir. Robota komutları Hub'ın yönleri kullanılarak verilecektir ve robotun bu komutları otomatik olarak yerine getirmesi sağlanacaktır. Bu işlemler için gerekli yönerge aşağıda verilmiştir:

- Sağ yön ("RIGHT") robotun sağ tarafa doğru kendi ekseni etrafında 90 derece dönmesini sağlar,

- Sol yön (“LEFT”) robotun sol tarafa doğru kendi ekseni etrafında 90 derece dönmesini sağlar,
- Ön yön (“FRONT”) robotun 250 derece/sn hızıyla doğrusal bir şekilde 1 tur geri gitmesini sağlar,
- Alt yön (“BOTTOM”) robotun 250 derece/sn hızıyla doğrusal bir şekilde 1 tur ileri gitmesini sağlar,
- Her yeni komuttan önce robot üst yönüne (“TOP”) doğru çevrilmelidir,
- Robotun verilen komutları yerine getirmesi için yavaşça arka yüzü (“BACK”) çevrilmeli, ardından robot tekerleri üzerinde masaya bırakılmalı ve robotun üzerine çift tıklanmalıdır (“DOUBLE_TAPPED”).

Örneğin robota iki kere ileri, bir kere sağa ve bir kere geri komutları şu şekilde verilir: üst, alt, üst, alt, üst, sağ, üst, ön, (yavaşça) alt, robotu masaya bırak, çift tıkla. Robot çift tıklama işleminin ardından iki kere ileri gidecek, bir kere sağa dönecek ve bir kere geri gidecektir.

Programı yazmaya başlamadan önce grupların tasarlama adımı için, daha önce örnekleri verilen, tanımlama ve fikir üretme sürecini gerçekleştirmeleri gerekir.

ÜRET

Öğrenciler tasarımlarını yaptıktan sonra bilgisayar ve robot başında çalışarak istenen görevi yerine getirmelidir. Aşağıdakine benzer bir program hazırlayabilirler. Öğrenciler bu programın aynısını yapmak zorunda değildir. Aynı problem farklı yollardan çözülebilir.

```
from hub import motion_sensor, port
import runloop, motor_pair, time

async def main():
    komut_listesi=[]
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    while True:
        kontrol=motion_sensor.up_face()
        while True:
            if kontrol!=motion_sensor.up_face():
                break
        if kontrol!=0 and kontrol!=3:
            komut_listesi.append(kontrol)
        elif kontrol==3:
            while True:
                if motion_sensor.gesture()==1:
                    break
            if len(komut_listesi) != 0:
                del komut_listesi[-1]
            for kontrol in komut_listesi:
                if kontrol==2:
                    await motor_pair.move_for_degrees(motor_pair.PAIR_1,
180, 100, velocity=250)
                if kontrol==5:
```

```
                    await motor_pair.move_for_degrees(motor_pair.PAIR_1,
180, -100, velocity=250)
                if kontrol==1:
                    await motor_pair.move_for_degrees(motor_pair.PAIR_1,
360, 0, velocity=-250)
                if kontrol==4:
                    await motor_pair.move_for_degrees(motor_pair.PAIR_1,
360, 0, velocity=250)
            komut_listesi.clear()
runloop.run(main())
```

Not

Sürüş modeli robotunda hub'ın ters olduğunu unutulmamalıdır. Robotun ön yönü, hub'ın arka yönüdür. Bu program hub'a göre hazırlanmıştır.

Bu programda çeşitli iyileştirmeler yapılabilir. Örneğin robotun verilen komutları yerine getirmesi için arka yüzü çevrilmeli fakat bu işlem yavaşça yapılmalıdır. Eğer hızlı bir şekilde yapılırsa verilen son komut çalıştırılmayabilir. Başka bir örnek olarak program çalışırken ışık matrisinde hareket edilen yöne dair ok işaretleri gösterilebilir ve hareketin çeşidine göre hoparlörden farklı uyarı sesleri çalınabilir. Program üzerinden bu ve benzer alanlarda iyileştirmeler yapılabilir fakat pedagojik olarak bu seviyedeki program yeterli görüldüğü için bu iyileştirmeler yapılmamıştır. Vakit kalması durumunda sınıfça bu iyileştirmelerin nasıl yapılabileceği tartışılır ve yapılır.

DEĞERLENDİR

Günün sonunda rehber öğretmen öğrencilere aşağıdaki soruları yönelterek öğrenilenlerin üzerine düşünmesini sağlamalıdır.

- Aşağıdaki kodun çıktısı ne olur (Öğrenciler programı çalıştırmadan çıktıyı tahmin etmelidir.)?

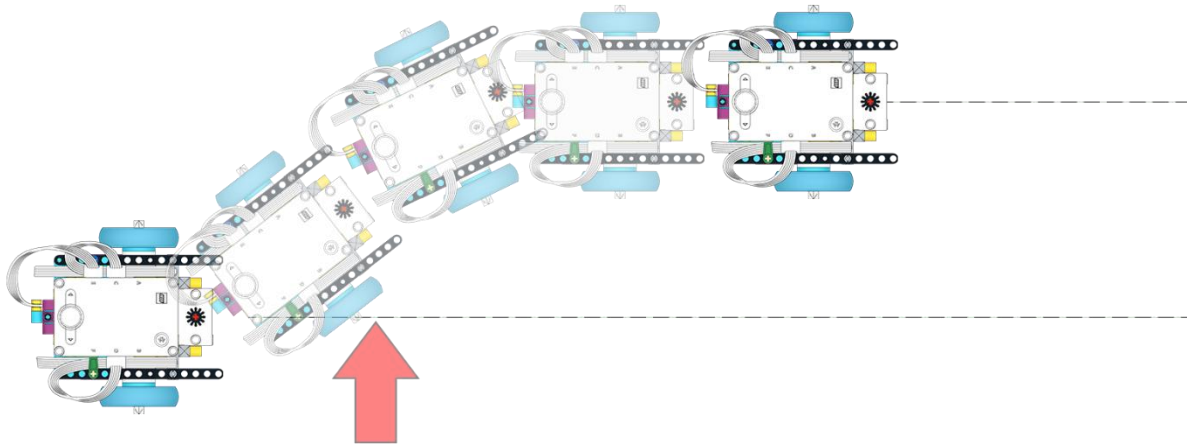
```
liste=[]
for i in range(5):
    ic_liste=[]
    for j in range(5):
        ic_liste.append(i+j)
    liste.append(ic_liste)
for i in range(5):
    for j in range(5):
        if i==j:
            print(liste[i][j])
```

- Bugün gerçekleştirdiğiniz programlama etkinliklerinde en çok zorlandığınız şeyler (örn. konu, kavram, işbirlikli çalışma ve yöntem) neler oldu? Bunları aşmak için neler yaptınız?
- Bir sonraki derse daha verimli bir öğrenme deneyimi elde etmek için neleri yapıp neleri yapmamanız gerekir?
- Hareket sensörü günlük hayatta nerelerde kullanılabilir?

İLAVE ETKİNLİKLER

Yönünü Kaybetmeyen Robot

Bu etkinlikte öğrencilerden belirli bir yönde giderken dışarıdan yapılan müdahalelere tepki göstererek yeniden başlangıç yönüne dönen robot kodu yazılacaktır. Robot belirli bir yönde doğrusal gidecek şekilde kodlanacaktır. Ardından dışarıdan yapılan bir müdahale ile robotun yönü değiştirildiğinde Resim 8.5'te gösterildiği gibi robot kendisini toparlayıp eski yönüne yönelmelidir. Örneğin robot başlangıçta güneye doğru doğrusal ilerlerken, rehber öğretmenin müdahalesi ile yönü doğuya çevrilen robot yönünü tekrar güneye çevirmelidir. Robotun sapma öncesi hareket çizgisine dönmesi beklenmemektedir; sapma öncesi yöne dönmesi yeterlidir.



Resim 8.5 Yönünü Kaybetmeyen Robot

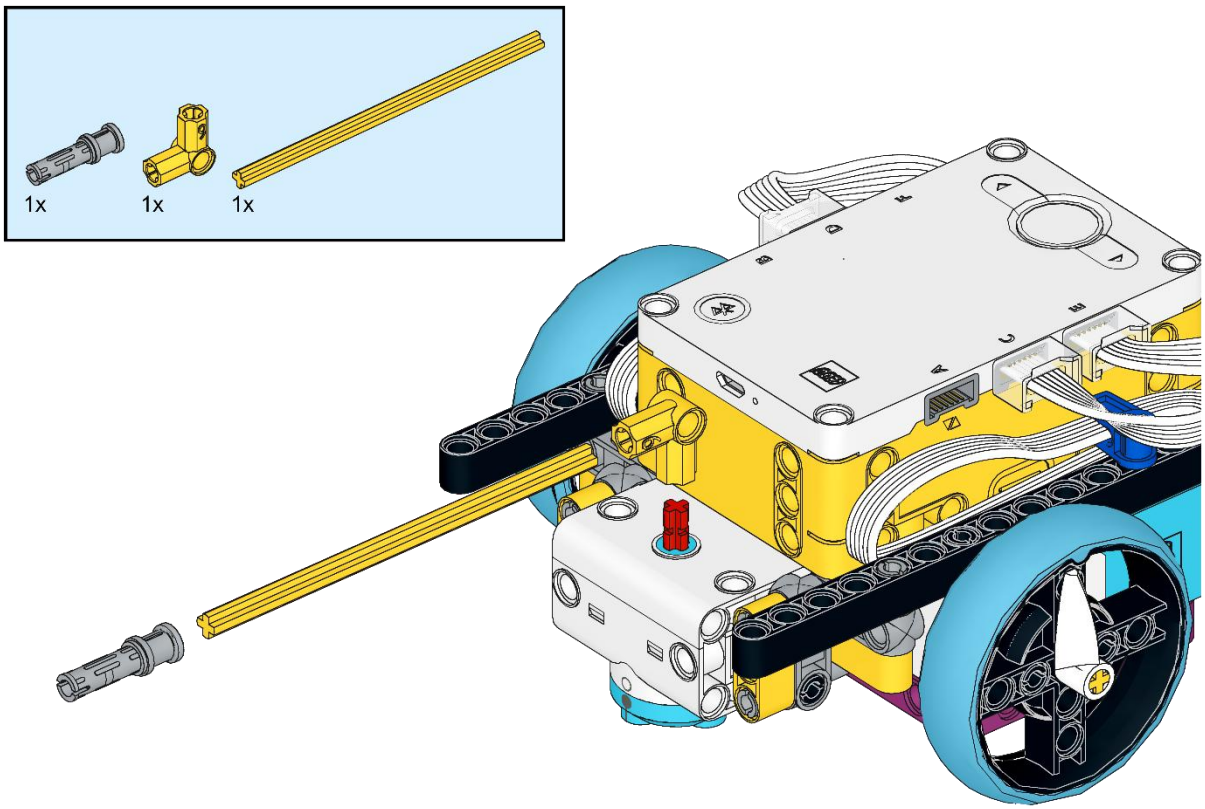
Aşağıda bu görev için kullanılabilecek örnek bir program verilmiştir. Sapma açısı -1800 ile +1800 arasında değer almaktadır. Sapma açsını 18'e kalansız bölerek direksiyon hareketinin -100 ile 100 arasında kalması sağlanmıştır. Öğrenciler bu değerleri kendi koşullarına göre uyarlayabilirler.

```
from hub import motion_sensor, port
import runloop, motor_pair
async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    motion_sensor.reset_yaw(0)
    while True:
        sapma_acisi=motion_sensor.tilt_angles()[0]
        motor_pair.move(motor_pair.PAIR_1, sapma_acisi//18, velocity=200)
runloop.run(main())
```

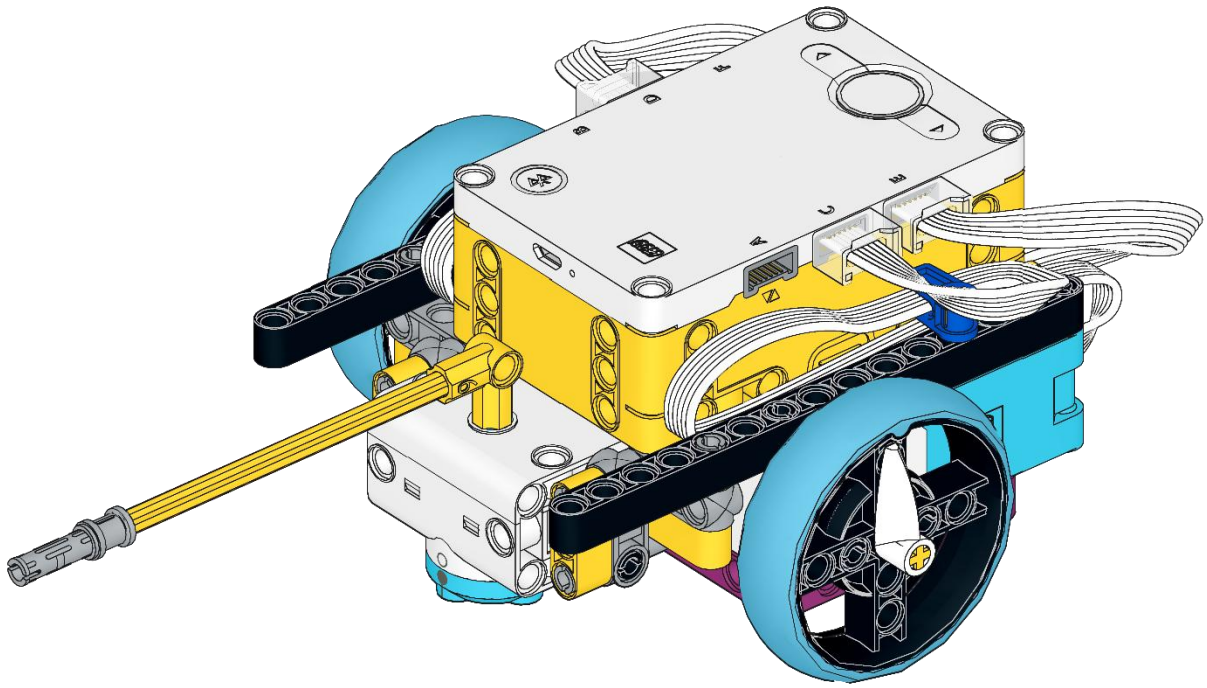
Hedef Yön Koluna Yönelen Robot

Bu etkinlikte durmakta olan robot için hedef yön kolu kullanılarak bir hedef belirlenecektir. Hedefin yönü gösterildikten sonra Hub'a çift tıklanmasının ardından robotun yönünü hedefe doğru çevirmesi sağlanacaktır. Robot bu işlemi sürekli tekrarlamalıdır. Etkinliğin gerçekleştirilebilmesi için sürüş modelinde değişiklik yapılmalıdır. Rehber öğretmen

öğrencilerden Resim 8.6 ve 8.7'de görüldüğü gibi sürüş modeli robotuna hedef yön kolunu eklemelerini ister. Etkinlik sonrasında öğrenciler sürüş modelini eski haline getirmelidir.



Resim 8.6 Hedef Yön Kolu Montajı Birinci Adım



Resim 8.7 Hedef Yön Kolu Montajı İkinci Adım

Rehber öğretmen için örnek bir program aşağıda verilmiştir. Programda dönülecek açı için 4 derecelik bir düzeltme yapılmıştır. Bu değer farklı koşullarda değişiklik gösterebilir. Öğrenciler bu değeri kendi durumlarına göre düzenlemelidir. Sapma açısının sol tarafa doğru pozitif ve sağ taraf doğru negatif olması ve büyük motorun sola dönme hareketi ile aldığı değerin pozitif , sağa dönme hareketi ile aldığı değerin negatif yönde değerler alması göz önünde bulundurularak dönülecek açı değeri üzerinde de düzenleme yapılmıştır. Aynı kod farklı şekillerde de yazılabilir.

```
from hub import motion_sensor, port
import runloop, motor_pair, motor
async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    motor.run_to_absolute_position(port.E,0,100)
    while True:
        motion_sensor.reset_yaw(0)
        while True:
                if motion_sensor.gesture()==1:
                    break
        donulecek_aci = motor.absolute_position(port.E)
        if donulecek_aci > 0:
            while motion_sensor.tilt_angles()[0] < donulecek_aci*10-40:
                motor_pair.move_tank(motor_pair.PAIR_1, -75, 75)
        else:
            while motion_sensor.tilt_angles()[0] > donulecek_aci*10+40:
                motor_pair.move_tank(motor_pair.PAIR_1, 75, -75)
        motor_pair.stop(motor_pair.PAIR_1)
        motor.run_to_absolute_position(port.E,0,100)
runloop.run(main())
```

Proje Hazırlıyorum- Hazırlık Aşaması

Bu hafta itibari ile öğrenciler projeleri üzerinde çalışmaya başlayacaktır. Projelerin geliştirme süreci için dört haftalık bir süre gereklidir. Proje geliştirme sürecine öğrenciler gruplar halinde çalışacaktır. Proje geliştirme sürecinde tasarım ve üretim dersinde bir benzeri kullanılan *hazırlık, tanımlama, empati, fikir üretme, geliştirme-test etme ve sunum* adımlarına yer verilecektir. Öğrenciler 4 hafta boyunca rehber öğretmen gözetiminde ders dışı etkinlikler şeklinde çalışacaktır. Bu süre boyunca gerektiğinde atölyelerden faydalanabilirler. Bu hafta projeye hazırlık aşaması gerçekleştirilecektir.

Açıklama: Proje süresinin ilk iki haftasında gruplar öncelikle ürünlerini tanımlamaya ve tasarlamaya yönelik çalışmalar yapacaktır. Kalan haftalarda, tanımlayıp tasarladıkları ürün için gerekli mekanik ve yazılım bileşenlerini geliştireceklerdir. Öğrencilere sabit bir proje konusu verilmeyecektir. Öğrenciler proje konusunu öğretmenin rehberliğinde kendileri belirleyecektir. Bu amaç doğrultusunda öğrenciler günlük hayatta karşılaştıkları bir probleme veya bilimsel bir probleme çözüm getirmek üzere bir konu belirleyecektir. Ürettikleri ürün ile bu problemi çözmeleri beklenmektedir. Öğrenciler problemlerini tanımlayıp tasarladıktan sonra ürünlerini oluşturacaklardır.

Sonraki Haftaya Hazırlık: Öğrencilerin bir sonraki haftaya kadar bir proje konusu belirlemeleri gerekmektedir. Proje ile bir probleme çözüm getirilecektir. Bu problem günlük

hayattan olabileceği gibi bilimsel, sanayiye yönelik veya sağlığa yönelik bir problem olabilir. Geliştirilecek ürün özgün olmalıdır ve bir kullanım değeri bulunmalıdır. Uygun bir proje konusu belirlemenin ilk basamağı onu anlamaktan geçer. Bu sayede projenin hedefleri tanımlanabilir ve zorlukları belirlenebilir. Bu iş için probleme yönelik araştırma ve gözlemlerin yapılması gerekebilir. Rehber öğretmen öğrencilere gerekli bilgilendirmeyi yaparak ve bir sonraki hafta uygun bir proje önerisi ile gelmeleri gerektiğini söyler. Bu hafta boyunca gerekli ayarlamaları yaparak atölyelerden faydalanabileceklerini ve rehber öğretmenlerden yardım isteyebileceklerini hatırlatır.

9. Hafta – Mesafe Sensörü

Haftanın Amacı:

Bu haftanın amacı, öğrencilerin mesafe (distance) sensörünün kullanımı ile ilgili temel bilgi ve becerileri kazanmasıdır. Öğrenciler bu temel bilgi ve becerileri kullanarak görece gelişmiş programlar oluşturacaktır. Öğrenciler bu amaç doğrultusunda mesafe sensörünün ışıklarını düzenleme, önde bulunan aracı takip etme, duvar takibi yapma, etrafındaki nesneleri tespit edip ona doğru yönelme gibi problemlere çözüm üretecektir. Buna ek olarak öğrenciler Python'da fonksiyon kavramını öğrenecek ve bu kavramı problem çözümünde kullanacaktır.

Haftanın Kazanımları:

- Öğrenciler mesafe sensörünün görevini ve nasıl çalıştığını anlatabilirler.
- Öğrenciler robotun mesafe sensöründen gelen değere göre hareket etmesini sağlayabilirler.
- Öğrenciler araç takip programı yazabilirler.
- Öğrenciler mesafe sensörünün ışıklarını programlayabilirler.
- Öğrenciler Python fonksiyonları oluşturup bu fonksiyonları programlarında kullanabilirler.
- Öğrenciler duvar takip programı yazabilirler.
- Öğrenciler nesne tespit ve ulaşma programı oluşturabilirler.

Kullanılacak Malzemeler:

Robot seti, bilgisayar ve çalışma alanı

GÖZLE VE UYGULA

Rehber Öğretmen İçeriği – Mesafe Sensörü Hakkında

Mesafe sensörü isminden de anlaşılacağı üzere mesafe ölçmeye yarayan bir sensördür. 5 cm ile 2 metre arılığındaki (bu değerlerden 2 cm fazla veya eksik değerler de elde edilebilir) nesne ve yüzeyleri algılayarak bu cisim ve yüzeylerin mesafe sensörüne olan uzaklıklarını ölçebilir. Ölçüm yaparken karşısına doğru insan kulağı ile algılanamayan ses dalgaları gönderir ve bu ses dalgalarının karşıdaki nesneye çarpıp sensöre geri dönme süresini kullanarak cismin uzaklığını hesaplar. Algılanması istenen cisim veya yüzey mümkün mertebe mesafe sensörünün tam karşısında bulunmalıdır. Yani cismin veya yüzeyin mesafe sensörüne olan açısı sıfıra yakın olmalıdır. Bu açı arttıkça ölçüm değerlerindeki hata payı artacaktır ve ölçüm yapılamaz hale gelinecektir. Mesafe sensöründe kısa menzil ölçüm özelliği bulunmaktadır. Kısa menzil ölçüm özelliği modunda mesafe sensörü 5-30 cm arasındaki cisim ve yüzeyleri algılayıp ölçüm yapabilir. Kısa menzil modunda yapılan ölçümün doğruluğu artar. Mesafe sensörünün "gözlerinin" etrafında dört adet programlanabilir ışık bulunmaktadır. Bu ışıklar birbirinden bağımsız olarak programlanabilir.

Mesafe sensörünün kullanılabilmesi için öncelikle distance_sensor kütüphanesi projeye dahil edilmelidir. Daha sonra distance metodu kullanılarak mesafe sensörünün bağlı olduğu port bilgisi yazılır ve mesafe sensöründen milimetre cinsinden ölçüm yapılır. Eğer mesafe sensörü geçerli bir mesafe ölçümü gerçekleştiremez ise -1 değerini döndürür.

```
import distance_sensor

distance_sensor.distance(port.F)
```

***Hatırlatma**: "from hub import port" komutu ile port kütüphanesi de projeye dâhil edilmelidir.*

Burada mesafe sensörü F portuna bağlandığı için bu parametre girilmiştir. Başka bir porta bağlanan sensör için o portun ismi kullanılmalıdır. Sensörün "gözleri" etrafındaki ışıkları ayrı ayrı yakmak için show metodu kullanılır. Bu metodun kullanımı aşağıda verilmiştir.

```
distance_sensor.show(port, [sağ üst, sol üst, sağ alt, sol alt])
```

Show metodunun ilk parametresi sağ üst ışık kaynağının parlaklığını belirlemek için kullanılır. Benzer şekilde parlaklık miktarı 0-100 arasındaki tam sayılar aracılığı ile belirlenir. Bu metodun ikinci parametresi sol üst, üçüncü parametresi sağ alt ve son parametresi sol alt ışık kaynağının parlaklık miktarlarını belirlemek için kullanılır.

Parlaklık parametresi 0 ile 100 arasında tam sayı değerleri alabilir. 0 kapalı, 100 tam parlaklık anlamına gelir. 0-100 arasındaki değerler ise ara parlaklık miktarlarını göstermek için kullanılır. Eğer herhangi bir parlaklık değeri girilmezse varsayılan değer olarak 100 kullanılır.

Gözle: Sol Gözünü Kırpan Robot

Bu etkinlikte mesafe sensörünün sol gözünün ışıklarını açıp kapamak suretiyle kırpılması sağlanacaktır. Göz kırpma işlemi sürekli gerçekleştirilmelidir. Bu görev için aşağıdaki kod kullanılabilir.

```
from hub import port
```

```
import runloop, distance_sensor
async def main():
    while True:
        distance_sensor.show(port.F, [0, 100, 0, 100])
        await runloop.sleep_ms(500)
        distance_sensor.show(port.F, [0, 0, 0, 0])
        await runloop.sleep_ms(500)
runloop.run(main())
```

Uygula: Gözlerini Sırayla Kırpan Robot

Bu etkinlikte mesafe sensörünün gözleri sırayla kırpılacaktır. Önce sol göz kırpılacak ardından sağ göz kırpılacak ve bu işlem sürekli tekrarlanacaktır. Bu görev için aşağıdaki kod kullanılabilir.

```
from hub import port
import runloop, distance_sensor
async def main():
    while True:
        distance_sensor.show(port.F, [0, 100, 0, 100])
        await runloop.sleep_ms(500)
        distance_sensor.show(port.F, [0, 0, 0, 0])
        await runloop.sleep_ms(500)
        distance_sensor.show(port.F, [100, 0, 100, 0])
        await runloop.sleep_ms(500)
        distance_sensor.show(port.F, [0, 0, 0, 0])
        await runloop.sleep_ms(500)
runloop.run(main())
```

Rehber Öğretmen İçeriği – Fonksiyonlar

Python'da tekrar tekrar kullanılacak kodlar için fonksiyonlar tanımlanır. Örneğin daha önce anlatılan *runloop.sleep_ms* fonksiyonu programın istenen süre için duraklatılmasını sağlar veya *range* fonksiyonu istenen aralıkta tam sayı üretmek için kullanılır. Bu fonksiyonlar Python'un yazarları tarafından oluşturulup Python kütüphanesine eklenmiştir. Programcılar bu fonksiyonları istedikleri zaman çağırıp kullanırlar. Programcılar da kendi fonksiyonlarını tanımlayabilir. Fonksiyon tanımlamak için def anahtar kelimesi kullanılır. Bundan sonra fonksiyonun ismi yazılır. Ardından parantez açıp kapatılarak parametre alınacaksa parametreler virgül ile ayrılarak parantezin içerisine yazılır. Eğer parametre almayacaksa parantezin içi boş bırakılır. Parantezden sonra iki nokta üst üste konulur. Buradan sonra fonksiyonun gövdesindeki komutlar yazılacaktır. Bir tab büyüklüğünde girinti yapılarak yazılan bütün komutlar, fonksiyon gövdesinde sayılacaktır ve fonksiyon çağrıldığında gövdedeki bütün komutlar çalıştırılacaktır. Aşağıda bir fonksiyonun nasıl tanımlanacağı gösterilmiştir. Eğer tanımlanan fonksiyon parametre almazsa parantezin içerisi boş bırakılmalıdır.

```
def fonksiyon_ismi(parametre_1, parametre_,...,parametre_n):
    komut_1
    komut_2
    ...
    komut_n
```

Fonksiyon tanımlandıktan sonra çağrılarak kullanılabilir. Fonksiyonu çağırırken fonksiyon ismi ve var ise gerekli parametreler yazılır. Yukarıda yazılan fonksiyonun çağrılıp çalıştırılması için aşağıdaki komut kullanılabilir.

```
fonksiyon_ismi(parametre_1, parametre_,...,parametre_n):
```

Programda fonksiyon tanımı fonksiyon çağrısından önce olmalıdır. Fonksiyon çağrısından sonra tanımlanmış veya tanımı yapılmamış bir fonksiyonun kullanımı hataya sebep olacaktır.

Uygula: Fonksiyon Kullanarak Göz Kırpma

Gözlerini Sırayla Kırpan Robot etkinliğinde sağ ve sol gözler ayrı ayrı kırpılmaktadır. Burada göz kırpma işlemi için arka arkaya benzer komutlar yazılmıştır. Bu durum gereksiz kod yazılmasına sebep olmaktadır. Bunun yerine bu etkinlikte goz_kirp fonksiyonu tanımlanarak kullanılacaktır. Aşağıda bu fonksiyonun tanımı verilmiştir. Rehber öğretmen gerekli kodları yazarak öğrencilere açıklar ve fonksiyonların faydalarından bahseder.

```
def goz_kirp(goz):
    if goz == "sol":
        distance_sensor.show(port.F, [0, 100, 0, 100])
    elif goz == "sag":
        distance_sensor.show(port.F, [100, 0, 100, 0])
    await runloop.sleep_ms(500)
    distance_sensor.show(port.F, [0, 0, 0, 0])
    await runloop.sleep_ms(500)
```

Hangi gözün kırpılması gerektiği goz_kirp fonksiyonuna parametre olarak gönderilmelidir. Fonksiyon içerisinde koşul ifadesi aracılığı ile gelen parametreye göre sol veya sağ gözün kırpılması sağlanır. Bu fonksiyon farklı şekillerde de yazılabilir. Örneğin mesafe_sensoru değeri bu fonksiyona parametre olarak gönderilebilir. Bu haliyle mesafe sensörünün ismi değiştirildiğinde program çalışmayabilir. Fonksiyon tanımlandıktan sonra artık program içerisinde çağrısı yapılarak kullanılabilir.

```
    while True:
        await goz_kirp("sol")
        await goz_kirp("sag")
```

Yukarıdaki kod ile sürekli olmak kaydıyla, sırayla sol ve sağ gözlerinin kırpılması sağlanmaktadır. Görüldüğü üzere fonksiyon bir kere tanımlandıktan sonra onu kullanmak için sadece çağrılması yeterlidir. Böylece fonksiyonlar bize daha kısa ve daha kolay yönetilebilen programlar yazmada yardımcı olur.

Uygula: Hub Düğmelerine Göre Göz Kırpan Robot

Hub düğmelerini kullanabilmek için button kütüphanesinin projeye dâhil edilmesi gerekir.

```
from hub import button
```

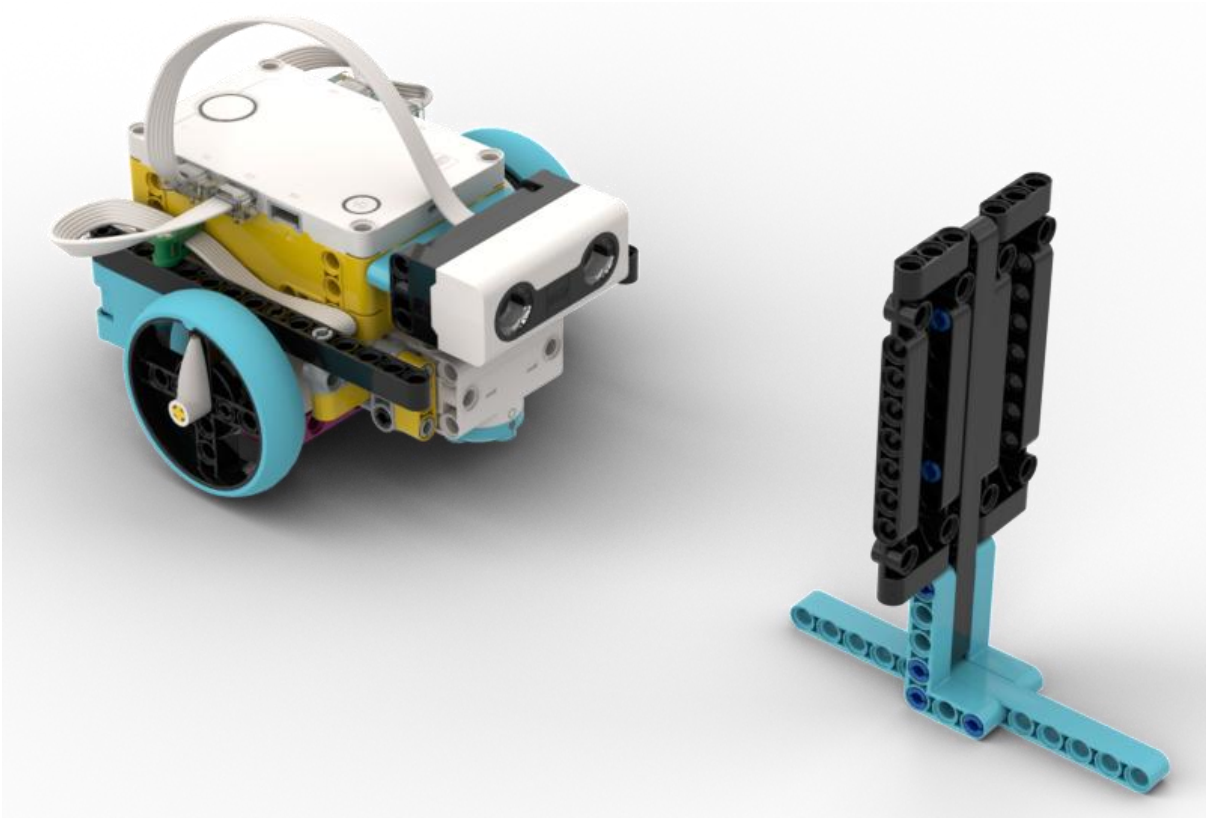
Bu etkinlikte sol Hub düğmesi tıklandığı süre boyunca sol gözünü, sağ Hub düğmesi tıklandığı süre boyunca sağ gözünü ve herhangi bir düğme tıklanmadığında sırayla sol ve sağ gözlerini sürekli kırpan bir robot programı yazılmalıdır. Bu görev için aşağıdaki kod kullanılabilir.

```
from hub import port, button
import runloop, distance_sensor
def goz_kirp(goz):
    if goz == "sol":
        distance_sensor.show(port.F, [0, 100, 0, 100])
    elif goz == "sag":
        distance_sensor.show(port.F, [100, 0, 100, 0])
    await runloop.sleep_ms(500)
    distance_sensor.show(port.F, [0, 0, 0, 0])
    runloop.sleep_ms(500)
async def main():
    while True:
        if button.pressed(button.LEFT):
            await goz_kirp("sol")
        elif button.pressed(button.RIGHT):
            await goz_kirp("sag")
        else:
            await goz_kirp("sol")
            await goz_kirp("sag")
runloop.run(main())
```

Gözle: Belirli Bir Mesafeye Kadar Giden Robot

Bu etkinlikte robotun karşısında bulunan engele 10 cm yaklaşıp burada durmasını sağlayan bir program yazılacaktır. Rehber öğretmen kodu açıklayarak anlatır. Bu görev için aşağıdaki kod kullanılabilir. Etkinlikte kullanılacak robot ve engel Resim 9.1'de gösterilmiştir.

```
from hub import port
import runloop, distance_sensor, motor_pair
async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    while distance_sensor.distance(port.F) > 100: #mesafe 100 mm ise
        motor_pair.move(motor_pair.PAIR_1,0,velocity=500) # hareket et
    motor_pair.stop(motor_pair.PAIR_1) #döngü bitince dur
runloop.run(main())
```



Resim 9.1 Bir Mesafeye Kadar Giden Robot

Uygula: Fonksiyon Kullanarak Belirli Bir Mesafeye Kadar Giden Robot

Bu etkinlikte öğrencilerden cm_ye_kadar_ilerle isimli bir fonksiyon yazmaları istenecektir. Bu fonksiyona gönderilen parametre robotun engele kaç cm kalana kadar ilerleyeceğini belirleyecektir. Örneğin cm_kadar_ilerle(20) komutu ile sürüş modeli engele 20 cm kalana kadar ilerleyecektir. Rehber öğretmen için gerekli kod aşağıda verilmiştir.

```
from hub import port
import runloop, distance_sensor, motor_pair

def cm_kadar_ilerle(mesafe):
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    while distance_sensor.distance(port.F) > mesafe*10:
        motor_pair.move(motor_pair.PAIR_1,0,velocity=500)
    motor_pair.stop(motor_pair.PAIR_1)
async def main():
    cm_kadar_ilerle(15)
runloop.run(main())
```

Rehber Öğretmen İçeriği – Mesafe Sönsörü Olayları

Mesafe Sensöründe belirli bir mesafeden daha yakın olana kadar bekle olayı oluşturulabilir. Belirli bir mesafeden yakın olana kadar bekle olayının fonksiyon olarak yazımı ve kullanımı aşağıda verilmiştir. Burada hedef_mesafe istenen mesafedir. Birimi ise santimetre olarak programlanmıştır.

```
def cm_az_olana_kadar(hedef_mesafe):
    motor_pair.pair(motor_pair.PAIR_1, port.E, port.F)
    while True:
        mesafe=distance_sensor.distance(port.D)/10
        if mesafe is not None and mesafe < hedef_mesafe:
            motor_pair.stop(motor_pair.PAIR_1)
        else:
            motor_pair.move(motor_pair.PAIR_1,0,velocity=500)
async def main():
   cm_az_olana_kadar(20)
runloop.run(main())
```

Gözle ve Uygula: Öndeki Aracı Takip Eden Robot

Bu etkinlikte amaç öndeki aracı 20 cm uzaklıktan takip eden robot programını yazmaktır. Öndeki araç ilerlerse aradaki mesafe 20 cm kalacak şekilde robot da ilerlemelidir, öndeki araç durduğunda robot da durmalıdır ve öndeki araç geri gelirse robot da 20 cm takip mesafesini koruyarak geriye doğru gitmelidir. Etkinlikte önde bulunan araç yerine kitap benzeri bir cisim kullanılabilir. Bu görev için aşağıdaki kod kullanılabilir.

```
from hub import port
import runloop, distance_sensor, motor_pair
async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    takip_mesafesi=200 #mm cinsinden yazılmıştır.
    tepki_gucu=10
    while True:
        aradaki_mesafe=distance_sensor.distance(port.F)
        if aradaki_mesafe!=None:
            tepki=(aradaki_mesafe-takip_mesafesi)*tepki_gucu
        motor_pair.move(motor_pair.PAIR_1,0,velocity=tepki)
runloop.run(main())
```

Rehber öğretmen programı öğrencilere anlatır ve onların da programı yazmalarını sağlar. Konuyu öğrencilere anlatırken düzenli olarak öndeki araç ile robotun arasındaki mesafenin (aradaki_mesafe) ölçüldüğünü, bu mesafenin 20 cm (takip_mesafesi) dışına çıktığında robotun aradaki mesafe ölçüsünde karşı tepki verdiğini, verilen tepkinin gücünün de tepki_gucu değişkeniyle ayarlandığını vurgular. Farklı tepki güçleri için (örn, 3, 7 ve 10) programı çalıştırıp öğrencilere deneterek tepki gücünün pratik olarak ne anlama geldiğini öğrencilerin anlamasını sağlar.

Not

`distance_sensor.distance(port.F)`

komutunda mesafe sensörü geçerli bir ölçüm yapamadığında -1 değerinin oluşturulduğu unutulmamalıdır. Program başladığı anda mesafe sensörünün algılayacağı bir nesne olmaması durumunda robot geri gidecektir.

Uygula: Öndeki Aracı Takip Eden Gelişmiş Robot

Bu etkinlikte yine öndeki aracı 20 cm mesafe ile takip eden bir robot yapılacaktır. Fakat bu sefer robota birkaç özellik daha eklenecektir. Ek olarak

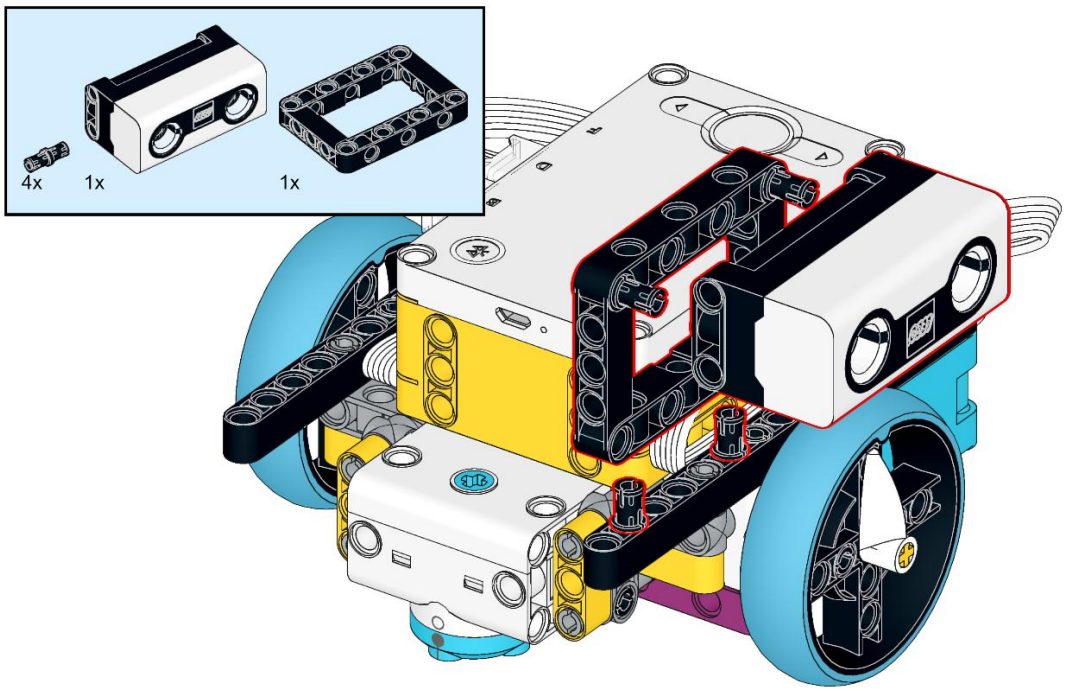
i. 0,5 saniyede bir konsol ekranında aradaki_mesafe değişkeninin değerini yazdırmalıdır,
ii. Aradaki mesafe 20'den büyük ise ışık matrisinde ileri ok, 20'den küçük ise geri ok ve aradaki mesafe 0 ise gülen yüz görünmelidir,
iii. Aradaki mesafe 20'den büyük ise hoparlörden uygun bir bip sesi çalınmalıdır, 20'den küçük ise geriye gittiğini gösteren bir bip sesi çalınmalıdır.
iv. Robot kuvvet sensörüne dokunulduğunda durmalıdır ve ses vermeyi bırakmalıdır.

Bu görev için aşağıdaki kod kullanılabilir. Rehber öğretmen öğrencilerin kodu kendilerinin yazmalarını istemelidir. Öğrencilerin yazdıkları kod verilen koddan farklılıklar gösterebilir.

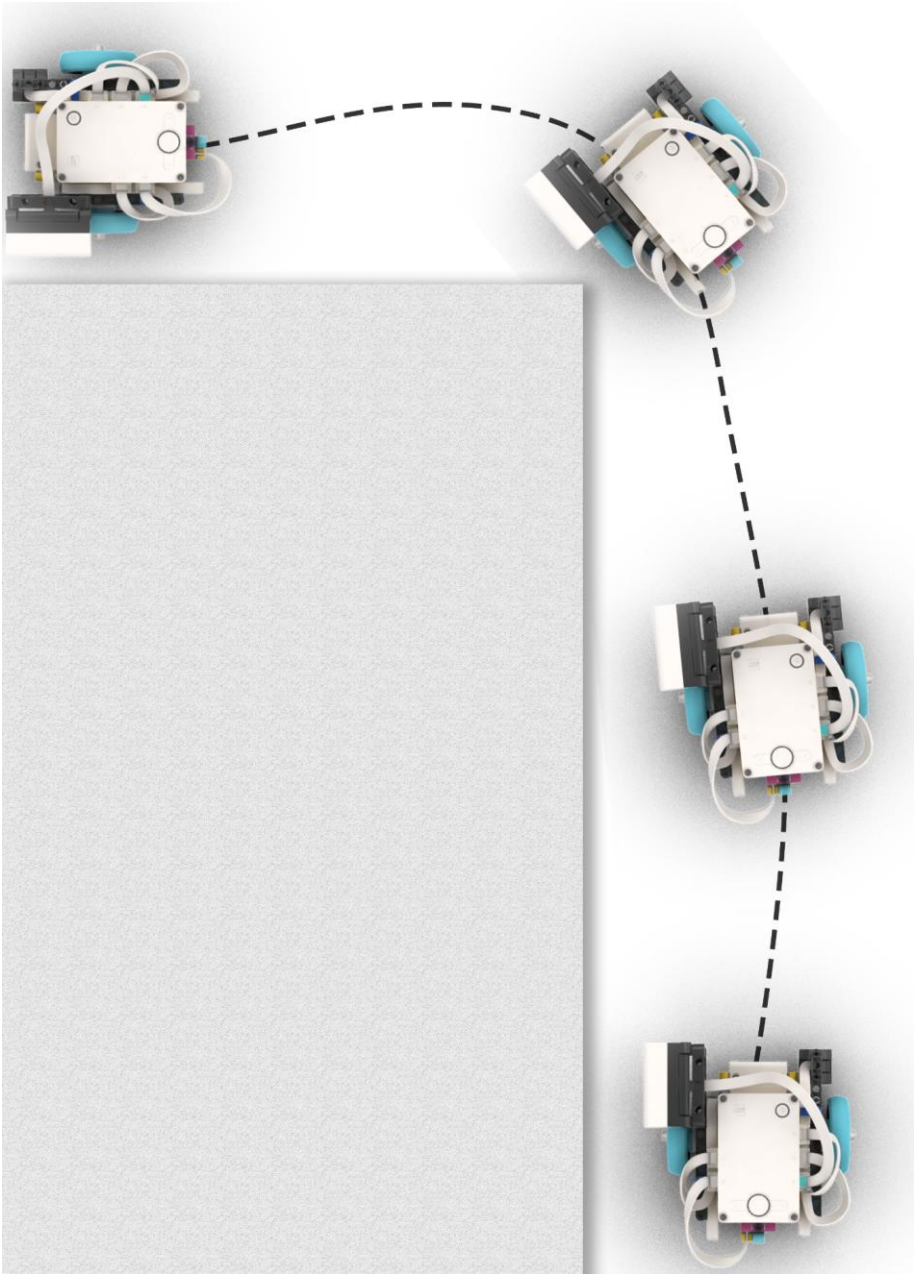
```
from hub import port, light_matrix, sound
import runloop, distance_sensor, motor_pair, force_sensor
async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    takip_mesafesi=200
    tepki_gucu=2
    while not force_sensor.pressed(port.A):
        aradaki_mesafe=distance_sensor.distance(port.F)
        if aradaki_mesafe!=None:
            tepki=(aradaki_mesafe-takip_mesafesi)*tepki_gucu
            if aradaki_mesafe > 209:
                light_matrix.show_image(light_matrix.IMAGE_ARROW_N)
                sound.beep(440,50,100)
            elif aradaki_mesafe < 199:
                light_matrix.show_image(light_matrix.IMAGE_ARROW_S)
                sound.beep(380,50,100)
            else:
                light_matrix.show_image(light_matrix.IMAGE_HAPPY)
                sound.stop
        motor_pair.move(motor_pair.PAIR_1,0,velocity=tepki)
    motor_pair.stop(motor_pair.PAIR_1)
runloop.run(main())
```

Gözle: Duvar Takibi Yapan Robot

Bu etkinlikte robotun duvar takibi yapması istenmektedir. Bu görev için robota mesafe sensörü Resim 9.2'de gösterildiği gibi sola bakacak şekilde monte edilmeli ve F portuna bağlanmalıdır.



Resim 9.2 Mesafe Sensörü Montajı



Resim 9.3 Duvar Takibi

Robot, sol tarafında bulunan düz duvarı takip edeceği için mesafe sensörü sol tarafa takılmıştır. Robot, duvara paralel bir şekilde yerleştirilecek ve bırakılma uzaklığı çok fazla olmamalıdır. Robot, duvar ile arasındaki mesafeyi 20 cm olacak şekilde duvarı takip ederek ileriye doğru hareket etmelidir. Ayrıca, duvarın sonunda robotun sola doğru dönüş yapabilmesi de gerekmektedir. Görev, Resim 9.3'te gösterilmektedir. Bu görev için aşağıdaki kod kullanılabilir. Rehber öğretmen, kodu yazarak öğrencilere açıklamalıdır.

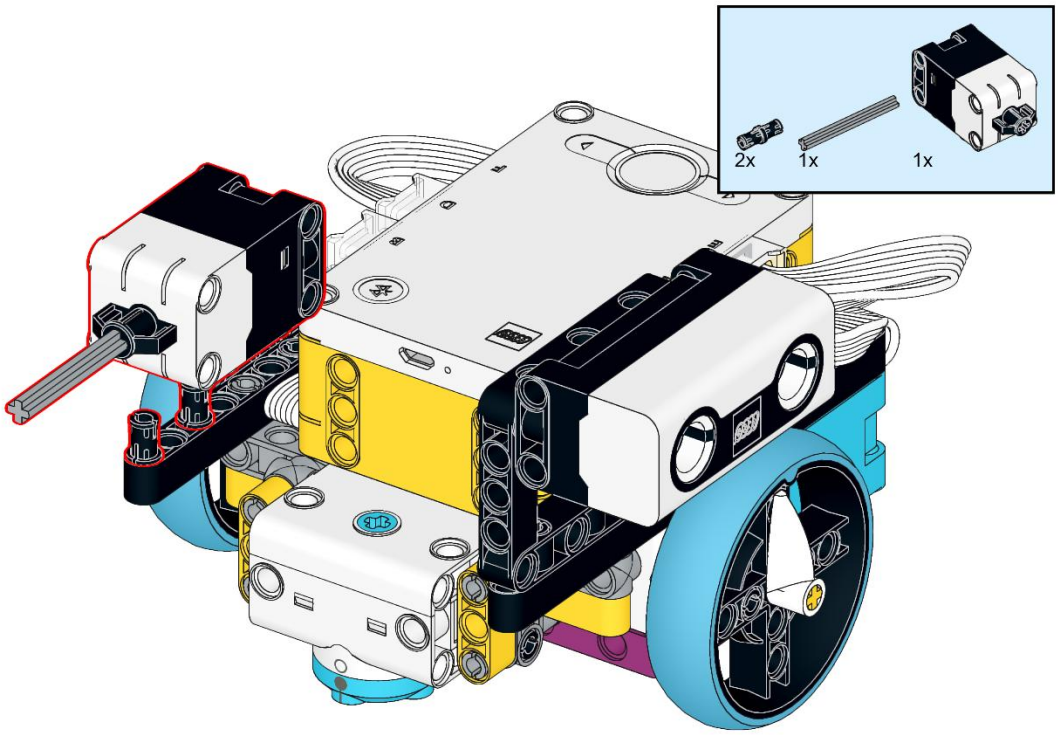
```
from hub import port
import runloop, distance_sensor, motor_pair
async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    istenen_mesafe=200
```

```
    duzeltme_gucu=0.4
    sapma=0
    while True:
        gercek_mesafe=distance_sensor.distance(port.F)
        if gercek_mesafe!=-1:
            sapma=gercek_mesafe-istenen_mesafe
        tepki=-sapma*duzeltme_gucu
        motor_pair.move_tank(motor_pair.PAIR_1,int(300+tepki),int(300-
tepki))
runloop.run(main())
```

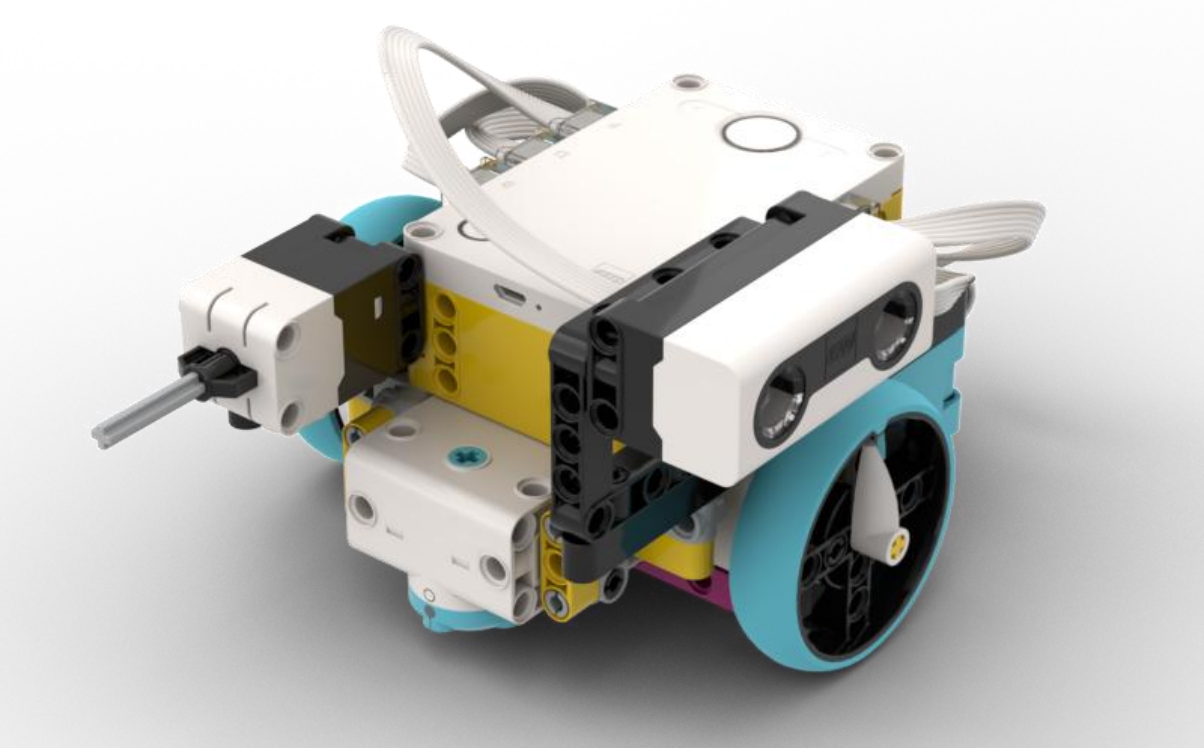
Mesafe sensörü bazen gerçek mesafeden bağımsız olarak beklenmedik değerler üretebilmektedir. Bu durumun sıklığı pürüzlü ve eğik yüzeylerde artmaktadır. Bu yüzden etkinlikte kullanılan duvar düz ve pürüzsüz olmalıdır.

Uygula: Duvar Takibi ve Sağa Dönüş

Bir önceki etkinlikte robot, sol tarafında bulunan duvarı takip ederek dönüş gerçekleştirebiliyordu. Ancak dönüş sonrasında yine sol tarafındaki duvarı takip etmeye devam ettiği için karşısına çıkacak bir engeli tespit edemeyecektir. Bu etkinlikte, robotun duvar takibi yaparken bir nesneye (veya karşı duvara) dokunması durumunda 90 derece sağa dönmesini sağlayan bir program hazırlanacaktır. Öncelikle, Resim 9.4'te gösterildiği şekilde kuvvet sensörünün sürüş modeli robotuna monte edilmesi ve B portuna bağlanması gerekmektedir.



Resim 9.4 Kuvvet Sensörü Montajı



Resim 9.5 Duvar Takibi Robotu

Resim 9.5'te gösterildiği gibi montajı gerçekleştirilen robot, önceki etkinlikte olduğu gibi duvar takibi yapacak ve kuvvet sensörüne basıldığında 90 derece sağa dönecek şekilde programlanmalıdır. Öğrenciler, programı kendileri hazırlamalıdır. Rehber öğretmen için aşağıda bir kod örneği sunulmuştur.

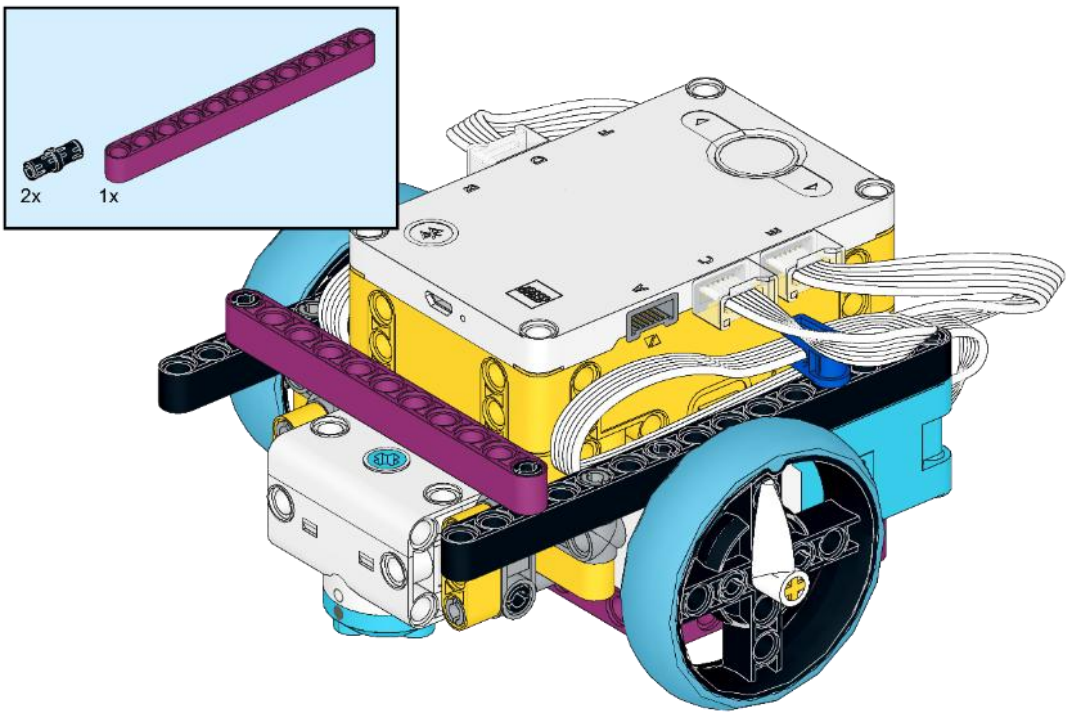
```
from hub import port
import runloop, distance_sensor, motor_pair, force_sensor
async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    istenen_mesafe=200
    duzeltme_gucu=0.4
    sapma=0
    while True:
        gercek_mesafe=distance_sensor.distance(port.F)
        if force_sensor.pressed(port.B):
            motor_pair.stop
            await motor_pair.move_for_degrees(motor_pair.PAIR_1,-360,0)
            await motor_pair.move_for_degrees(motor_pair.PAIR_1,180,100)
        elif gercek_mesafe!=-1:
            sapma=gercek_mesafe-istenen_mesafe
        tepki=-sapma*duzeltme_gucu
        motor_pair.move_tank(motor_pair.PAIR_1,int(300+tepki),int(300-
tepki))
runloop.run(main())
```

Öğrencilere bu etkinliği yapmaları için gerekli süre verilmelidir. Bu süre içerisinde rehber öğretmen onlara sadece ipuçları vererek öğrencileri yönlendirmelidir. Verilen süre bittiğinde öğrenciler başarılı bir şekilde kodu yazamamışlarsa rehber öğretmen verilen kodu onlara açıklayarak anlatır.

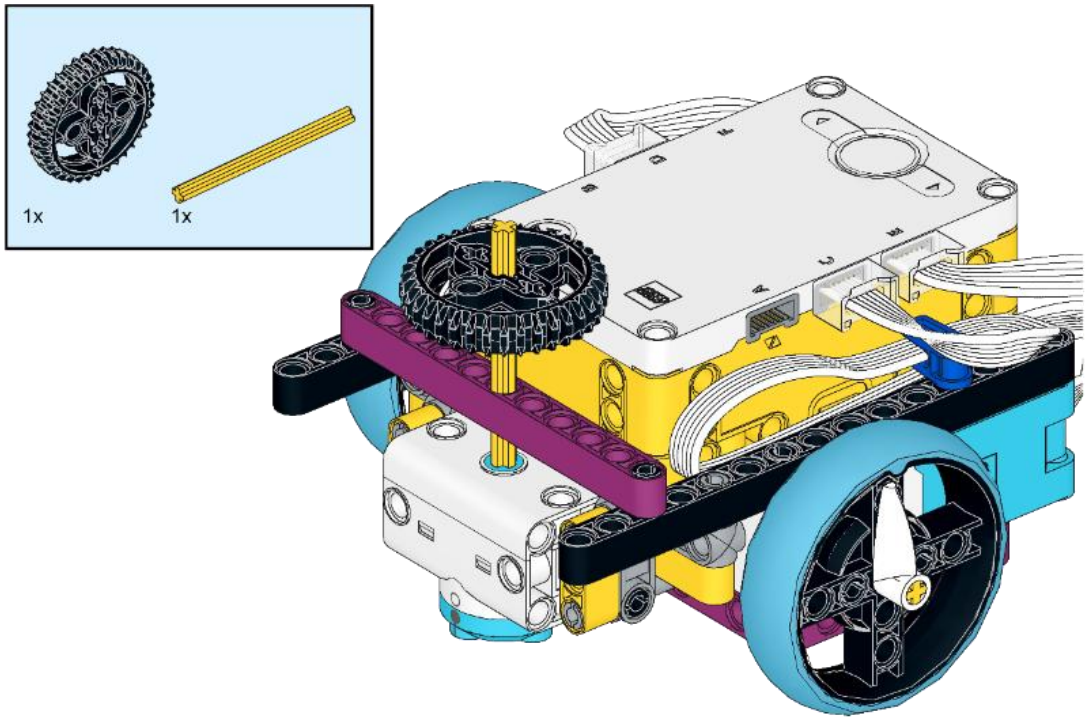
TASARLA

Hedef Tespit Robotu

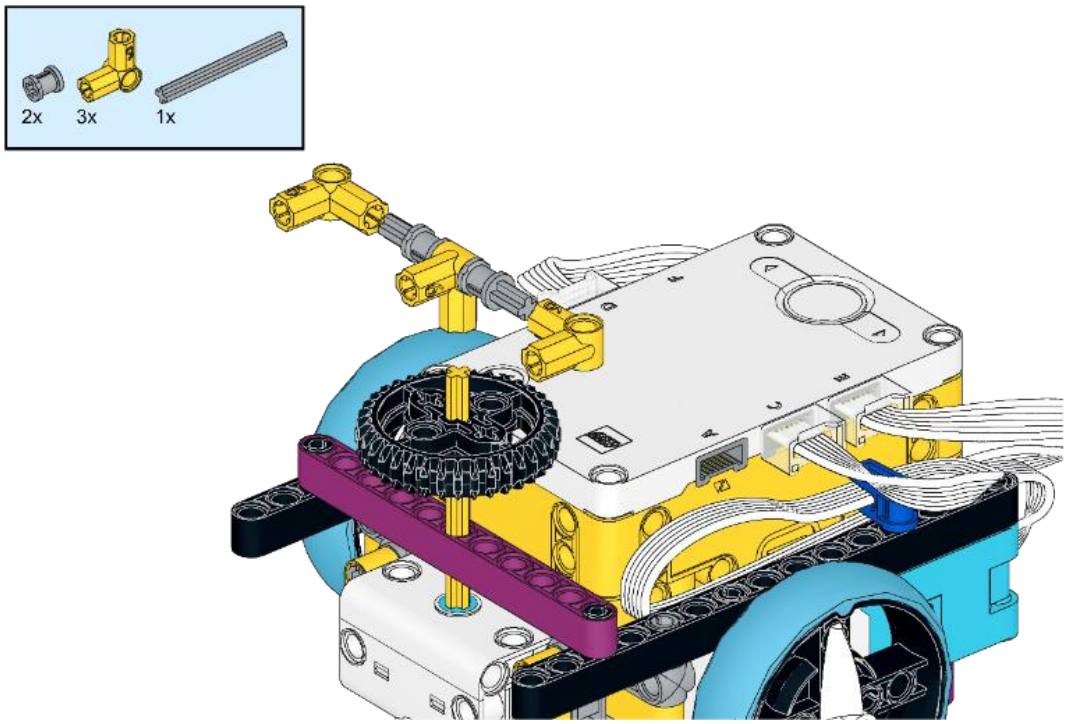
Bu etkinlikte öğrencilerden önünde bir yerde bulunan hedefi tespit edip, onun yanına giden robotu programlamaları beklenmektedir. Bu görev için mesafe sensörü ve hareket sensörü birlikte kullanılmalıdır. Uzaklık sensörü 180 derece etrafını tarayıp hedef bulunup bulunmadığını kontrol edecektir. Hedefi bulduğunda, bulduğu yönde duracaktır. Hareket sensörü ise robotun bu yöne doğru hareket etmesi için kullanılacaktır. Bu görev için robot tasarımının değiştirilmesi gerekmektedir. Aşağıda robotun inşa adımları verilmiştir. Robot inşa adımları öğrenciler ile paylaşılmalıdır.



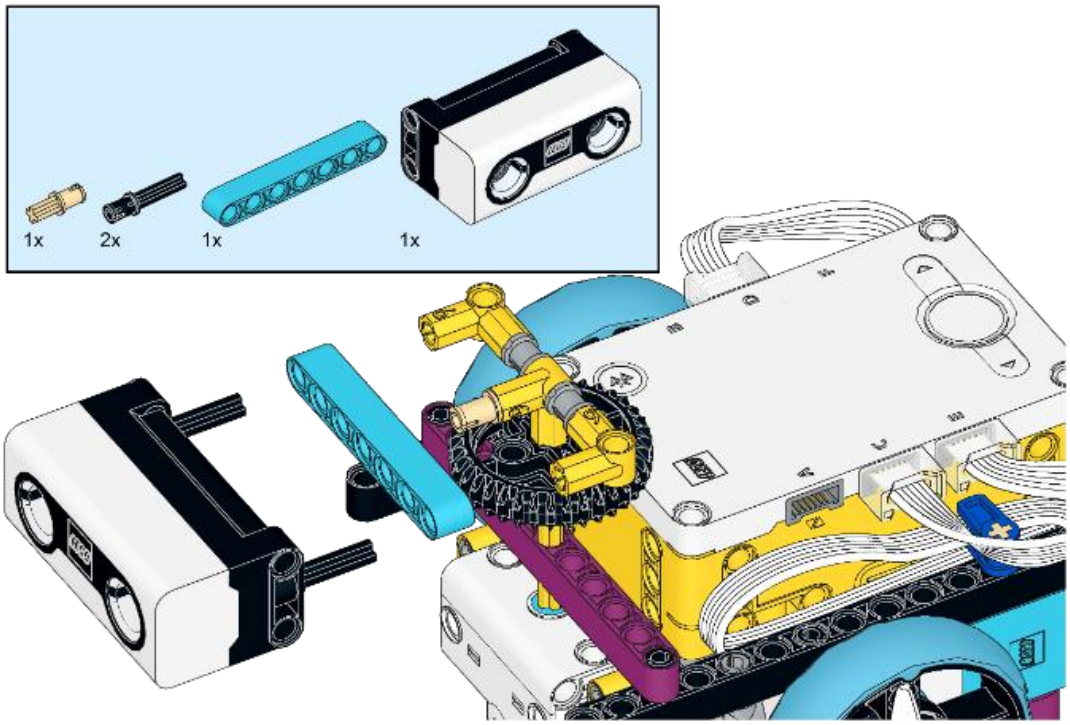
Resim 9.6 Hedef Tespit Robotu Birinci Adım



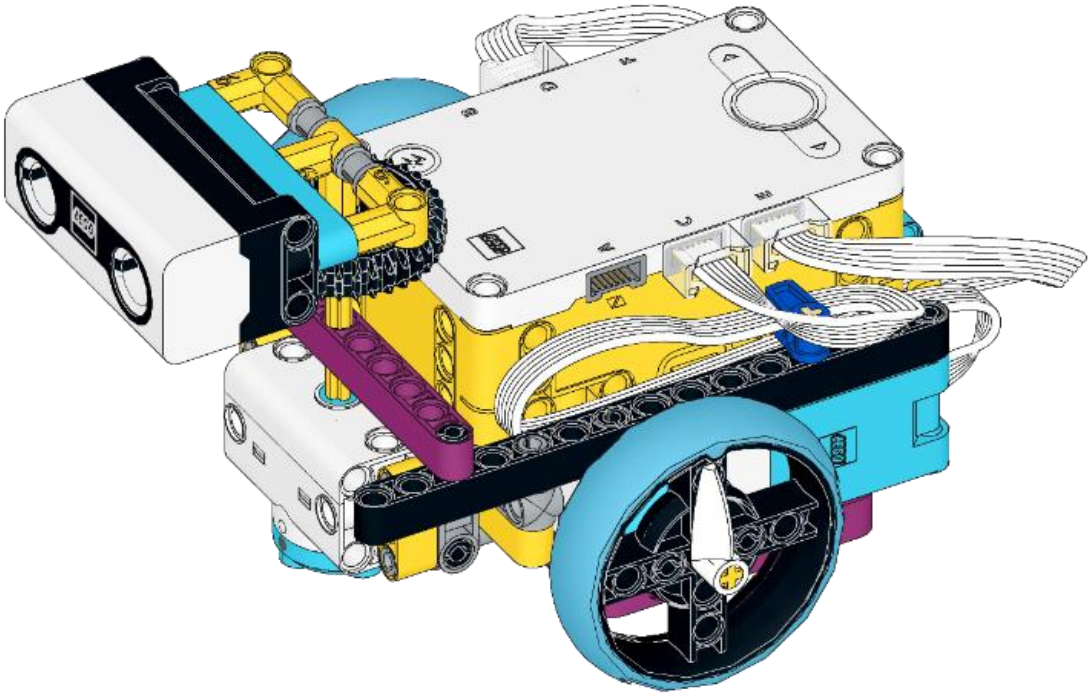
Resim 9.7 Hedef Tespit Robotu İkinci Adım



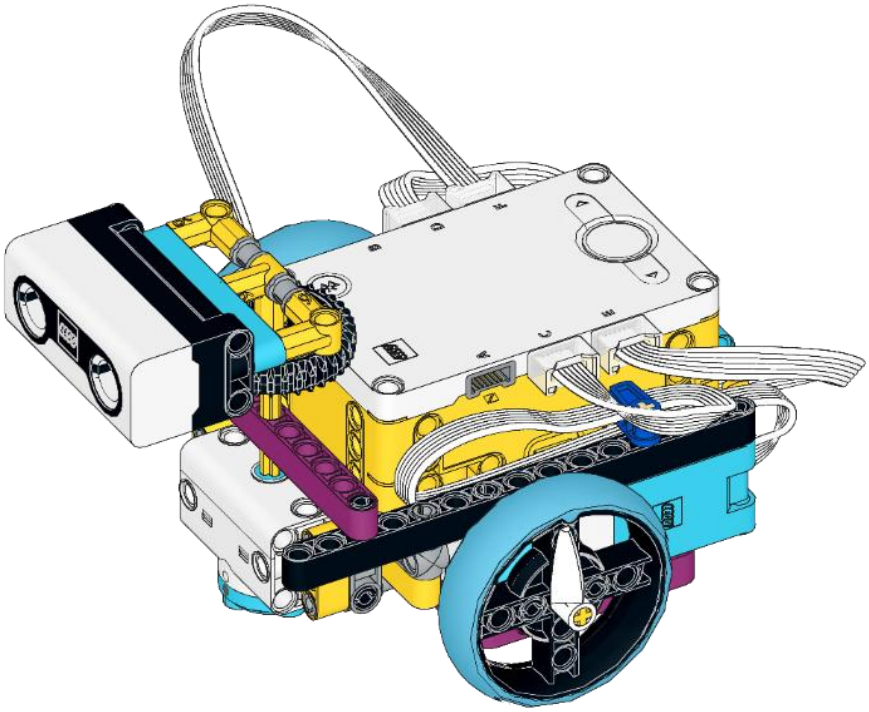
Resim 9.8 Hedef Tespit Robotu Üçüncü Adım



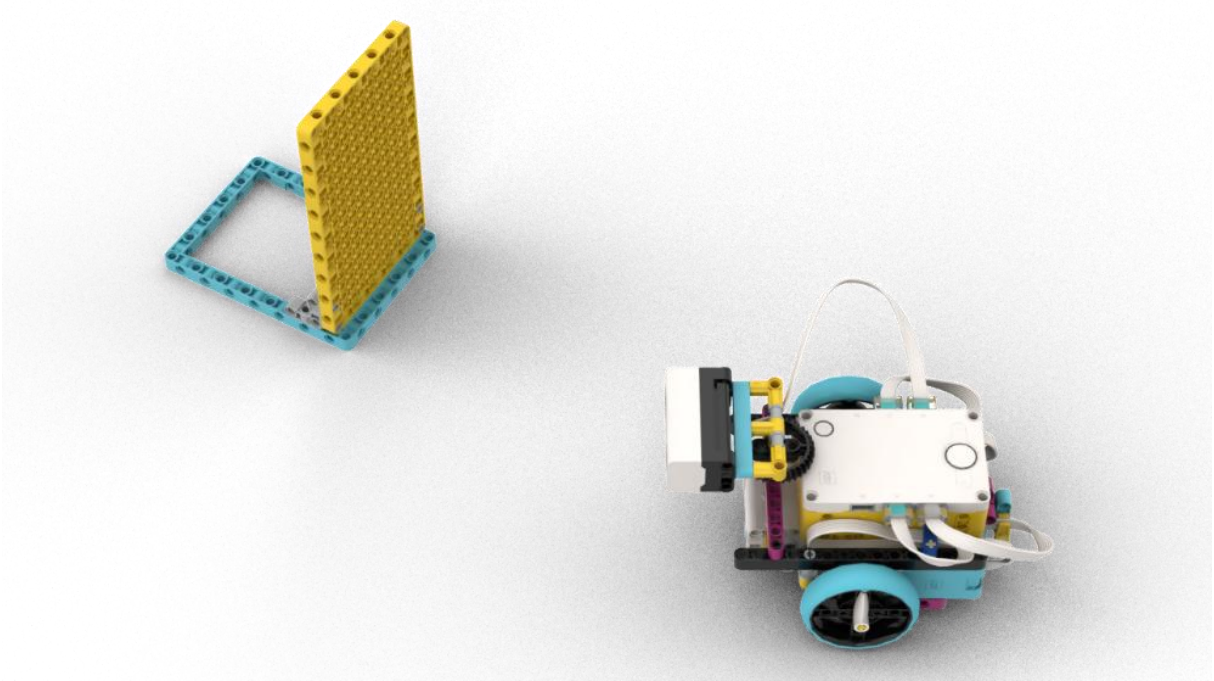
Resim 9.9 Hedef Tespit Robotu Dördüncü Adım



Resim 9.10 Hedef Tespit Robotu Beşinci Adım



Resim 9.11 Hedef Tespit Robotu Altıncı Adım



Resim 9.12 Hedef Tespit Robotu Görev Alanı

Öğrenciler robot üzerinde gerekli değişikliği yapıp hedefi hazırladıktan sonra rehber öğretmen robotun çalışmasını biraz daha detaylandırır. Mesafe sensörü 10'ar derecelik açılar ile dönerek önündeki alanı 180 derece tarar. Taradığı alan içerisindeki 50 cm mesafede bir hedef varsa robot mesafe sensörü aracılığı ile birkaç değer okur ve bu değerlerin ortalamasını alarak hedef tespiti yapar. Robot hedefe yönelerek ona doğru hareket eder ve ona 10 cm kala durur. 180 derece tamamlandığında bir hedef bulamazsa 10 cm ileriye gider ve yeniden mesafe sensörü ile tarama işlemine başlar ve bu işlemleri hedefi bulana kadar tekrar eder.

Tanımlama: Öğrenciler kodu yazmaya başlamadan önce problemi kâğıt kalem kullanarak tanımlamalıdır. Öğrenciler doğrudan kod yazmaya geçmek isteyebilir. Rehber öğretmen onları tanımlama adımına yönlendirmelidir. Öğrencilerin problemi çözmeden önce onun tasarımı hakkında düşünmeleri önemlidir. Aşağıda örnek tanımlama adımları verilmiştir. Bu adımlar öğrenciler ile paylaşılmamalıdır. Öğrenciler kendi adımlarını oluşturmalıdır.

i. Hedef, robotun önünde bulunacağı için 180 derecelik tarama yeterlidir.
ii. 180 derece belirli adımlara bölünerek, adım adım taranmalıdır.
iii. Hedefin tespiti mesafe sensöründen gelen bir değer üzerinden yapılmamalıdır. Yer tespiti için mesafe sensöründen gelen birkaç değerin ortalaması alınmalıdır.
iv. Tarama esnasında bir hedef tespit edilirse robot hedefe doğru yönelip hareket etmeli ve hedefe 10 cm kadar yaklaşmalıdır.
v. Hedef tespit edilemezse robot tespit alanında 10 cm ilerlemelidir.
vi. Robot hedefi tespit edene kadar ii-iv adımları tekrarlanmalıdır.

Fikir Üretme: Öğrenciler bu adımda probleme dair daha somut çözüm önerileri sunmalıdır. Algoritmayı detaylandırma, hangi programlama yapılarını kullanacağına, algoritmanın gerçekleştirilmesi için ne gibi programlama tekniklerinin kullanacağına karar verip tanımlama bu adımda gerçekleştirilir. Aşağıda örnek adımlar verilmiştir. Bu adımlar rehber öğretmen için örnek niteliğindedir. Öğrenciler kendi adımlarını oluşturmalıdır.

i. Mesafe sensörü adımlarla 180 derece tarama yapacaktır. Adımlar 3, 5 veya 10 derece olarak belirlenebilir. Adım derecesine kod uygulandıktan sonra hedefe yaklaşma hata payı ve geçen süre göz önünde bulundurularak karar verilebilir.
ii. Robotun hedefe yönelme açısı mesafe sensörünün hedefi tespit ettiği açı olmalıdır. Bu açı mesafe sensörünün bağlı olduğu motordan edinilebilir. Bu işlemin benzeri daha önce *"Motor Açısına Göre Hareket Eden Robot (Hafta 8)"* etkinliğinde gerçekleştirilmiştir. Orada öğrenilen deneyimler burada kullanılabilir.
iii. Büyük motorun başlangıç açısı 90 derece olarak ayarlanabilir. Başlangıcın ardından her bir taramada büyük motor -90 ile 90 derece arasında tarama işlemi yapmalıdır.
iv. Tespit işlemi, ilk tespit anında bitirilmek yerine üst üste birkaç tane tespit yapılarak bunların ortalaması hedef açı olarak alınabilir.
v. Hedefe 10 cm yaklaşmak için mesafe sensörü kullanılacaktır. Bunun için mesafe sensörünün olayları kullanılabilir.

ÜRET

Hedef Tespit Robotu

Öğrenciler tasarlama adımının ardından, ürettikleri fikirleri uygulamaya geçirmek için robot başına geçerler ve gerekli kodları yazmaya başlarlar. Rehber öğretmen bu adımda öğrencilerin sorduğu sorulara doğrudan yanıtlar vermemelidir. Problemi çözüp çözmeme sorumluluğu öğrencilere aittir. Rehber öğretmen bu adımda sadece yönlendirme yapmalıdır. Kesinlikle sorunun çözümünü öğrenciler ile paylaşmamalıdır. Rehber öğretmen için örnek bir çözüm aşağıda paylaşılmıştır.

Not

Örnek çözümde mesafe sensörü sağdan sola doğru tarama yapacak şekilde programlanmıştır. Bu yüzden mesafe sensörü en sağda olduğunda üzerinde bulunan motorun açı değeri sıfır olarak ayarlanmalıdır.

```
from hub import port, motion_sensor
import runloop, distance_sensor, motor_pair, motor
async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    await motor.run_to_absolute_position(port.E,10,200)
    mesafe= distance_sensor.distance(port.F)
    bulunmadi=True
    hedef_acilari=[]
    hedef_acisi=0
    while bulunmadi:
        if motor.absolute_position(port.E) < 180 and
motor.absolute_position(port.E) > 0 :
            motor.run_for_degrees(port.E, 10, 50)
            await runloop.sleep_ms(200)
            mesafe= distance_sensor.distance(port.F)
            if mesafe != -1 and mesafe < 300:
                sayac=1
                while mesafe != -1 and mesafe < 300 and sayac <=5:
                    hedef_acilari.append(motor.absolute_position(port.E))
                    print(hedef_acilari[sayac-1])
                    await motor.run_for_degrees(port.E, 5, 50)
                    await runloop.sleep_ms(200)
                    mesafe= distance_sensor.distance(port.F)
                    sayac+=1
                bulunmadi=False
        else:
            await motor.run_to_absolute_position(port.E,10,200)
            await motor_pair.move_for_degrees(motor_pair.PAIR_1,200,0)
    for aci in hedef_acilari:
        hedef_acisi+=aci
    hedef_acisi=hedef_acisi/len(hedef_acilari)
    await motor.run_to_absolute_position(port.E, 90,200)
    motion_sensor.reset_yaw(0)
    if hedef_acisi<90:
        print("hedef acisi:", hedef_acisi)
        while (motion_sensor.tilt_angles()[0] > (90-hedef_acisi)*-10):
```

```
            print("sapma", motion_sensor.tilt_angles()[0])
            motor_pair.move(motor_pair.PAIR_1, 100, velocity=100)
        motor_pair.stop(motor_pair.PAIR_1)
    else:
        while (motion_sensor.tilt_angles()[0] < (90-hedef_acisi)*-10):
            motor_pair.move(motor_pair.PAIR_1, -100, velocity=100)
        motor_pair.stop(motor_pair.PAIR_1)
    while distance_sensor.distance(port.F) > 100:
        motor_pair.move(motor_pair.PAIR_1, 0, velocity=100)
    motor_pair.stop(motor_pair.PAIR_1)
runloop.run(main())
```

DEĞERLENDİR

Günün sonunda rehber öğretmen öğrencilere aşağıdaki soruları yönelterek öğrenilenlerin sorular üzerine sınıfça düşünmesini sağlar.

- Fonksiyon nedir?
- Fonksiyon ne zaman kullanılır?
- Fonksiyonları kullanmanın avantajları nelerdir?
- Hedef Tespit Robotu etkinliğinde fonksiyonlar kullanılabilir mi? Yanıtınızı açıklayınız.
- Bugün gerçekleştirdiğiniz programlama etkinliklerinde en çok zorlandığınız şeyler (örn. konu, kavram, işbirlikli çalışma ve yöntem) neler oldu? Bunları aşmak için neler yaptınız?
- Mesafe sensörü ile bir proje yapsaydınız nasıl bir proje yapardınız?
- Bir sonraki derse daha verimli bir öğrenme deneyimi elde etmek için neleri yapıp neleri yapmamanız gerekir?

İLAVE ETKİNLİKLER

Kaç Duvar Olduğunu Sayan Robot

Bu etkinlikte aşağıdaki resimde gösterildiği gibi duvarlar aralıklarla yan yana dizilecektir. Duvarlar arasında 10-15 cm mesafe bulunacaktır. Duvar için kitap ve mukavva benzeri bir cisim kullanılabilir. Resimde sadece üç tane duvar gösterilmiştir. Duvar sayısı öğrenciler için belirsiz olacaktır. Öğretmen istediği kadar duvar koyabilir. Öğrenciler robotu duvardan istedikleri bir uzaklığa duvara paralel olacak şekilde koyabilirler.

Resim 9.13 Duvar Sayan Robot

Öğrenciler bir kod yazarak öğretmenin kaç adet duvar koyduğunu buldurmalıdır. Duvar sayısı bulunduktan sonra robot durmalı, bir adet bip sesi çalmalı ve ışık matrisinde kaç adet duvar bulunduğunu yazmalıdır. Bu etkinlik yapılırken öğrenciler doğrudan kod yazmaya başlamamalı, öncelikle yukarıda bir örneği verilen tasarlama adımlarını (tanımlama ve fikir üretme) gerçekleştirmelidir.

LEGO Kutusu Etrafında Dönen Robot Yarışması

Bu etkinlikte öğrencilerden Spike Prime kutusu etrafında dönen robot kodunu yazmaları istenir. Öğrenciler kutunun kenar uzunluklarını tahmini veya gerçek ölçüm olarak kullanamazlar. Robot bir duvar takip algoritması ile solundaki duvarı takip ederken boşluk ile karşılaştığında dönmelidir. Duvar takip etme ve dönme algoritmaları sürekli tekrar edildiğinde robot kutunun etrafında dönmüş olacaktır. Rehber öğretmen öğrencilere gerekli açıklamaları yapar, onlara makul bir süre verir. Süre sonunda bütün gruplardan kutu etrafında robotlarını kullanarak görevi gerçekleştirmelerini ister.

Yarışmada amaç hızlı ve hatasız bir şekilde görevi yerine getirmektir. Kazanan grubu öğretmen ve grup puanları belirleyecektir. Her grup diğer gruplara puan verecektir. Gruplar kendilerini puanlayamaz. Öğretmen de her gruba bir puan verecektir. Verilen puanlar toplandığında en yüksek puanı alan grup yarışmayı kazanır. Grupların puanlarında eşitlik olması durumunda kazananı öğretmen belirleyecektir.

Proje Hazırlıyorum- Tanımlama ve Empati Aşaması

Öğrenciler bir önceki hafta, hafta boyunca bilimsel veya günlük yaşam problemi belirleme üzerine çalışmışlardır. Problemin belirlenmesi önemli bir adımdır. Öğrenciler belirledikleri problemleri rehber öğretmenlere sunmalıdır. Rehber öğretmenler belirlenen problemleri değerlendirmelidir. Rehber öğretmen projeleri değerlendirirken aşağıdaki sorulardan yararlanır:

- Belirlenen proje konusu ne kadar özgündür?

- Öğrenciler bu problemi çözmek için yeterli bilgi ve birikime sahip midir?
- Problem çözümü için gerekli malzemeler mevcut mudur?
- Proje ile gerçekten bir probleme çözüm bulunacak mıdır?

Rehber öğretmen bu ve bunun gibi kriterleri hesaba katarak projeleri değerlendirir. Projelerin özgün, çözülebilir ve kullanım değeri olması konusunda öğrencileri yönlendirmelidir. Öğrenciler çözemeyecekleri kadar büyük bir problem veya kullanım değeri olmayan bir konu belirlemiş olabilir. Rehber öğretmen bu gibi durumlara müdahale ederek proje konularının istenen şekilde olmasını sağlar. Fakat bunu yaparken fazla kısıtlayıcı olmamalıdır. Öğrenciler proje konusunu belirlemede sorumluluk sahibi olmalıdır.

Sonraki Haftaya Hazırlık:

Bu hafta itibariyle öğrenciler proje konularını belirlemiş olmalıdır. Öğrenciler bir sonraki haftaya belirledikleri problem için empati ve tanımlama çalışması yapacaktır. Öğrenciler tasarım ve üretim dersinde empati ve tanımlama adımlarını öğrenmiştir. Empati adımında geliştirilen üründen faydalanacak kullanıcıların gözünden ürünün nasıl olması gerektiği belirlenir. Ürün nasıl olursa kullanıcılar daha fazla yararlanır, konfor elde eder ve verimleri artar gibi sorular sorularak bunlara kullanıcıların gözünden yanıtlar aranır. Tanımlama adımında ise empati adımında elde edilen bilgiler bir bütün haline getirilir. Geliştirilecek ürünün başarılı olabilmesi için ihtiyaçların iyi tanımlanmış olması gereklidir. Buna ek olarak geliştirilecek ürün bu ihtiyaçları karşılamakta uygun bir role sahip olmalıdır. Süreçte ortaya çıkabilecek problemlerin çözümü için yapılabilecekler üzerine düşünülmelidir. Öğrenciler aşağıdaki sorular üzerine düşünebilir:

- Problem cümlesi: Robotun ne yapması isteniyor?
- Robotun kullanılması planlanan ortamdaki ihtiyaçlar nelerdir?
- Robotun kullanılması planlanan ortamdaki beklentiler nelerdir?
- Robot gerçekten belirlenen ihtiyaç ve beklentileri karşılayabilir mi?

Öğrenciler bir sonraki haftaya kadar empati ve tanımlama adımlarının gereklerini yerine getirmelidir. Öğrenciler bu süreç boyunca gerekli koşulları yaratarak laboratuvarları kullanabilir ve rehber öğretmen ile iletişim kurup yardım isteyebilir.

10. Hafta – Renk Sensörü

Haftanın Amacı:

Bu haftanın amacı, öğrencilerin renk (color) sensörünün çeşitli kullanım şekillerini kavrayarak sensörü farklı görevler için programlarken gerekli düzenlemeleri yapabilmelerini sağlamaktır. Ayrıca, öğrencilerin PID algoritması ile robotun farklı görevlerde daha verimli çalışmasını sağlamak için gerekli düzenlemeleri yapabilmelerini sağlamak da amaçlanmıştır.

Haftanın Kazanımları:

- Öğrenciler renk sensörünün nasıl çalıştığını ifade edebilirler.
- Öğrenciler renk sensörünün ölçtüğü renk değerine göre robotun davranışını düzenleyebilirler.
- Öğrenciler renk sensörünün ölçtüğü ortam ışığı değerine göre robotun davranışını düzenleyebilirler.
- Öğrenciler renk sensörünün ölçtüğü yansıyan ışık değerine göre robotun davranışını düzenleyebilirler.
- Öğrenciler PID algoritma kavramını açıklayabilirler.
- Öğrenciler oluşturdukları programların daha verimli çalışması için robotun PID algoritması içeren program kodlarını oluşturabilirler.

Kullanılacak Malzemeler:

Robot seti, bilgisayar ve çalışma alanı

Ekler:

Spike_lise_Hafta.10_PID_Dortgen_Mat .pdf
Spike_lise_Hafta.10_PID_Dortgen_Mat _A3.pdf
Spike_lise_Hafta.10_PID_Egri_Mat.pdf

GÖZLE VE UYGULA

Rehber Öğretmen İçeriği – Renk Sensörü Hakkında

Renk sensörünün üç temel görevi bulunmaktadır. Bunlardan ilki gördüğü cismin rengini belirlemektir. Kırmızı (red), Yeşil (green), Mavi (blue), Fuşya (magenta), Sarı (yellow), Turuncu (orange) Gök mavisi (azure), Siyah (black) ve Beyaz (white) renkleri algılayabilir. Bunun yanında tanımlanan renklerden herhangi birinin olmadığını da (unknown) algılayabilir.

Renk sensörünün ikinci görevi yansıyan ışığın yoğunluğunu ölçmektir. Sensörde üç adet ışık kaynağı bulunmaktadır. Bu ışık kaynaklarından beyaz ışık gönderir ve karşıdan yansıyan ışığın yoğunluğunu ölçebilir. Bu değer 0 ile 100 arasında değişir. 0 hiç ışık yansımadığı anlamına gelir 100 ise en yüksek yoğunlukta ışık yansıdığını belirtir. Yansıyan ışık yoğunluğu yansıtıcı yüzeyin rengine ve mat veya parlak oluşuna göre değişir. Siyah renk ışığı büyük oranda soğurur ve yansıtmaz, beyaz renk ise ışığı büyük oranda yansıtır. Mat yüzeyler ışığı dağıtırken parlak yüzeyler ışığı yansıtır. Karşıdaki cismin rengini belirlerken veya karşıdaki cisimden yansıyan ışığın yoğunluğunu belirlerken sensörün cisme olan ideal uzaklığı 16 mm'dir. En iyi ölçümleri bu mesafede yapacaktır.

Renk sensörünün son görevi ortam ışığının yoğunluğunu algılamaktır. Ortam ışığının yoğunluğu 0 ile 100 arasında değerler alabilir. 0 ortamda ışık olmadığı, 100 ise en yüksek yoğunluklu ışık olduğu anlamına gelir. Dolayısı ile 0 en karanlık ve 100 en aydınlık değerdir.

Bu görevlere ek olarak sensörde bulunan ışıklardan da bahsetmek yerinde olacaktır. Sensörde üç adet ışık kaynağı bulunmaktadır. Bu ışık kaynaklarının parlaklığı ayrı ayrı programlanabilir.

Gözle: Cismin Rengini Belirliyorum

Bu etkinlikte yerinde duran robotun renk sensörüne gösterilen cismin rengi belirlenecektir. Gösterilecek cisimlerin rengi sensörün tanıdığı renkler olan siyah, mor, mavi, açık mavi, yeşil, sarı, kırmızı ve beyaz renklerden biri olmalıdır. Bunun dışındaki cisimler şu aşamada tanımlanamayabilir.

Diğer sensörlerde olduğu gibi, öncelikle renk sensörünün kütüphanesi tanımlanmalıdır. color_sensor ve color kütüphaneleri, renk sensörünün çalışması ve algılanan renklerin işlenmesi için kullanılmalıdır. Zamanlama ve gecikme işlemleri için time modülü, asenkron görevlerin yönetimi için ise runloop kütüphanesi tanımlanmalıdır. Hub üzerindeki LED ışık matrisini kontrol etmek amacıyla light_matrix, sensör ve motorların bağlı olduğu portlarla iletişim kurabilmek için port kütüphanesi kullanılmalıdır. Bu kütüphaneler, programın donanım bileşenleriyle uyum içinde çalışabilmesi için tanımlanmalıdır. Aşağıdaki kodlarda tanımlamalar yapılmıştır.

```
import color_sensor
import color
import time
import runloop
from hub import light_matrix
from hub import port
```

Kütüphane tanımlamaları yapıldıktan sonra, aşağıdaki kodlar renk sensörü tarafından algılanan renkleri kontrol ederek LED ışık matrisinde ilgili renk adını yazdırmaya yarıyor. while True döngüsü, sürekli olarak renk sensöründen alınan veriyi kontrol eder. color_sensor.color(port.B), bağlı renk sensörünün algıladığı rengi kontrol eder. Eğer sensör algılayabildiği 9 renkten birini algılarsa, LED ışık matrisinde algıladığı rengin Türkçe ismini yazar. Eğer algılanan renk yoksa veya sensör herhangi bir renk tespit edemezse, matris üzerinde "Renksiz" yazısı görüntülenir. Kod, her kontrol işleminden sonra bir saniyelik bir gecikme ile çalışmaya devam eder.

```
async def main():
    while True:
        if color_sensor.color(port.B) is color.RED:
            await light_matrix.write("Kirmizi")
        elif color_sensor.color(port.B) is color.GREEN:
            await light_matrix.write("Yeşil")
        elif color_sensor.color(port.B) is color.BLUE:
            await light_matrix.write("Mavi")
        elif color_sensor.color(port.B) is color.MAGENTA:
            await light_matrix.write("Fusya")
        elif color_sensor.color(port.B) is color.YELLOW:
            await light_matrix.write("Sari")
        elif color_sensor.color(port.B) is color.ORANGE:
            await light_matrix.write("Turuncu")
        elif color_sensor.color(port.B) is color.AZURE:
            await light_matrix.write("Gok Mavisi")
        elif color_sensor.color(port.B) is color.BLACK:
            await light_matrix.write("Siyah")
        elif color_sensor.color(port.B) is color.WHITE:
            await light_matrix.write("Beyaz")
        else: # None değerini kontrol eder
            await light_matrix.write("Renksiz")
        time.sleep_ms(1000)
```

Gözle: İleri ve Geri Komutunu Renkler ile Veriyorum

Bu etkinlikte robota renkler aracılığı ile ileri ve geri 2 tur gitme komutları verilecektir. Robota mavi renk gösterildiğinde 2 tur ileri gidip duracaktır. Robota kırmızı renk gösterildiğinde 2 tur geri gidip duracaktır. Yeni ileri veya geri komutu robot durduktan sonra verilmelidir. Robot her yeni komutta (mavi veya kırmızı) gerekli görevi yerine getirmelidir.

color_sensor.color(port.B) metodu renk sensörünün okuduğu renk değerini anlık olarak döndürür. Bazı durumlarda anlık renk değeri okumak yerine yeni bir renk algılanana kadar bekleme işlemi yapılması gerekebilir. Aşağıda etkinliğin çözümüne yönelik örnek bir kod verilmiştir. Rehber öğretmen kodu öğrencilere açıklayarak anlatır.

```
from hub import port
import runloop, color_sensor, color, motor_pair
async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    onceki_renk = -1
    while True:
        mevcut_renk = color_sensor.color(port.B)
        if mevcut_renk != onceki_renk:
            onceki_renk=mevcut_renk
            if mevcut_renk == color.BLUE:
                await motor_pair.move_for_degrees(motor_pair.PAIR_1,
360*2,0, velocity=1000)
            elif mevcut_renk == color.RED:
                await motor_pair.move_for_degrees(motor_pair.PAIR_1,
360*2,0, velocity=-1000)
runloop.run(main())
```

Uygula: Renk Sensörü ile Programlanabilen Robot

Bu etkinlikte de öğrenciler renkler ile robotun nasıl hareket edeceğini programlayacaktır. Sarı renk robotun 1 tur ileri gitmesini sağlar, kırmızı renk robotun 1 tur geri gitmesini sağlar, yeşil renk robotun kendi ekseninde sola dönmesini sağlar ve mavi renk robotun kendi ekseni etrafında sağa dönmesini sağlar. Fakat bu sefer robot her bir hareket sonunda kodlanmayacaktır. Öncelikle robota hangi hareketi yapması gerektiğini söyleyen komutlar verilecektir. Siyah renk gösterildiğinde komut verme işleminin bittiği anlaşılacaktır ve robot verilen komutları yerine getirecektir. Siyah renk ile verilen komutlar bittikten sonra robot tekrar tekrar yeni hareket komutlarını kabul ederek siyah renk ile yeniden harekete başlamalıdır. Rehber öğretmen etkinliği açıkladıktan sonra öğrencilerin gerekli çözümü yapmalarını sağlar. Rehber öğretmen için örnek bir kod aşağıda verilmiştir.

```
from hub import port
import runloop, color_sensor, color, motor_pair, time
async def main():
    hareketler=[]
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    son_renk = None
    while True:
        mevcut_renk = color_sensor.color(port.B)
        if mevcut_renk != son_renk:
            if mevcut_renk == color.YELLOW:
                hareketler.append("ileri")
            elif mevcut_renk == color.RED:
                hareketler.append("geri")
```

```
            elif mevcut_renk == color.GREEN:
                hareketler.append("sol")
            elif mevcut_renk == color.BLUE:
                hareketler.append("sag")
            elif mevcut_renk == color.BLACK:
                for hareket in hareketler:
                    if hareket == "ileri":
                        await
motor_pair.move_for_degrees(motor_pair.PAIR_1, 360, 0, velocity=200)
                    elif hareket == "geri":
                        await
motor_pair.move_for_degrees(motor_pair.PAIR_1, -360, 0, velocity=200)
                    elif hareket == "sol":
                        await
motor_pair.move_tank_for_degrees(motor_pair.PAIR_1, 180, -200, 200)
                    elif hareket == "sag":
                        await
motor_pair.move_tank_for_degrees(motor_pair.PAIR_1, 180, 200, -200)
                hareketler.clear()
            son_renk = mevcut_renk
        else:
            time.sleep_ms(100)
runloop.run(main())
```

Not

Bu kod ile aynı hareket komutu iki kere verilmek isteniyorsa istenen renk gösterilip çekilmeli ardından bir daha gösterilmelidir.

Gözle: Siyah Rengi Görene Kadar İlerleyen Robot

Bu etkinlikte robot, renk sensöründe siyah rengi görene kadar hareket etmektedir. Kodlarda ilk olarak, motor çiftleri tanımlanır ve robot hareket etmeye başlar. Sürekli çalışan bir döngü ile renk sensöründen gelen veriler kontrol edilir. Sensör tarafından algılanan renk siyah olmadığında robot ilerlemeye devam eder. Siyah renk algılandığında döngü sonlanır ve motorlar durdurularak robotun hareketi tamamlanır. Bu süreç, robotun renk algılamasıyla çevresine duyarlı bir şekilde hareket etmesini sağlar.

```
from hub import port
import runloop, color_sensor, color, motor_pair
async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
```

```
    motor_pair.move(motor_pair.PAIR_1, 0)
    while True:
        if color_sensor.color(port.B)==color.BLACK:
            motor_pair.stop(motor_pair.PAIR_1)
            break
runloop.run(main())
```

Uygula: Renk Kodları ile Harekete Başlayan Robot

Bu etkinlikte de robot siyah rengi görene kadar ilerleyecektir. Fakat robot program çalışır çalışmaz ilerlemeye başlamamalıdır. Robotun hareketine başlayabilmesi için öncelikle renk kilidinin açılması gerekir. Renk kilidi mavi, yeşil ve sarıdır. Bu renkler sırasıyla renk sensörü tarafından algılanmadığı müddetçe hareket başlamaz. Rehber öğretmen öğrencilerin gerekli kodu yazmalarını sağlar. Aşağıda rehber öğretmen için hazırlanmış örnek bir kod bulunmaktadır.

```
from hub import port
import runloop, color_sensor, color, motor_pair
async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    while True:
        if color_sensor.color(port.B)==color.BLUE:
            break
    while True:
        if color_sensor.color(port.B)==color.GREEN:
            break
    while True:
        if color_sensor.color(port.B)==color.YELLOW:
            break
    motor_pair.move(motor_pair.PAIR_1, 0)
    while True:
        if color_sensor.color(port.B)==color.BLACK:
            break
    motor_pair.stop(motor_pair.PAIR_1)
runloop.run(main())
```

Gözle: Ham Renkleri Öğreniyorum

Doğadaki bütün renkler kırmızı (Red), yeşil (Green) ve mavi (Blue) rengin karışımı şeklinde belirlenebilir. Örneğin kırmızı ve yeşil renk karıştırıldığında sarı renk elde edilir. Turuncu ise kırmızı ve sarı rengin karışımıdır. Buna RGB renk modeli denilir. MicroPython ile bir nesnenin renginde hangi yoğunlukta kırmızı, yeşil ve mavi renklerin bulunduğu öğrenilebilir. Bunun için color_sensor.rgbi(port)[n] kullanılır. Kırmızı için n yerine 0, yeşil için n yerine 1 ve mavi için ne yerine 2 yazılır. Bu dizilerin her biri 0-1024 arasında değer üretir. 0 değeri ilgili renk yoğunluğunun bulunmadığını gösterirken 1024 değeri o rengin yoğunluğunun en üst seviyede olduğunu gösterir.

Örneğin tam olarak kırmızı renkteki bir nesne için dizinin birinci elemanı yani (n=0) 1024 değeri döndürürken, dizinin ikinci (n=1) ve üçüncü (n=2) elemanları 0 değerini döndürecektir. Uluslararası RGB sisteminde ve Spike Prime sözcük blokları kullanıldığında bu değerler 0-255 aralığındadır. Fakat MicroPython için bu değerler 0-1024 olacak şekilde standart dışı tanımlanmıştır

Rehber öğretmen aşağıdaki kodu öğrenciler ile paylaşır. Öğrencilerin farklı renkte cisimler kullanarak programın çıktılarını gözlemelerini sağlar. Bu iş için Spike Prime seti ile gelen Lego tuğlaları kullanılabilir.

```
from hub import port
import runloop, color_sensor, time
async def main():
    while True:
        kirmizi = color_sensor.rgbi(port.B)[0]
        yesil = color_sensor.rgbi(port.B)[1]
        mavi = color_sensor.rgbi(port.B)[2]
        print(kirmizi,yesil,mavi)
        time.sleep_ms(500)
runloop.run(main())
```

Kod çalıştırıldığında tuğlaların renklerinin mutlak kırmızı, mutlak yeşil ve mutlak mavi (get_blue olmadıkları anlaşılacaktır. Kırmızı renkli tuğlada, ana renk olan kırmızının değeri diğerlerine göre daha yoğun olmakla birlikte, yeşil ve mavinin değerleri 0 değildir.

Uygula: Turuncu Rengi Algıladığında Bip Sesi Çalan Robot

Bu etkinlikte öğrencilerden turuncu rengi algıladığında bip sesi çalan bir robot programı yazmaları beklenmektedir. Renk sensörü turuncu rengi doğrudan algılayamamaktadır. Öğrenciler get_red, get_green ve get_blue metotlarını kullanarak turuncu rengi tanımlayabilirler. Bunun için turuncu renkli bir cisim alınmalıdır ve bu cismin kırmızı, yeşil ve mavi yoğunluk değerleri bulunmalıdır. Ardından bu değerlere yakın bir aralıkta bulunan değerler turuncu olarak tanımlanmalıdır. Programın çalıştırıldığı koşullarda kullanılan turuncu nesne içi kırmızı=700, yeşil=320 ve mavi=300 yaklaşık yoğunluk değerleri bulunmuştur. Bu sebeple turuncu için 690<kırmızı<710, 310<yeşil<330, ve 290<mavi<310 aralıkları kullanılmıştır. Bu değerler uygulama koşullarında öğrenciler için farklılık gösterecektir. Öğrenciler kendi değerlerini bulmalıdır. Bulunan aralıklar için yazılmış örnek kod aşağıda verilmiştir.

```
from hub import port, sound
import runloop, color_sensor
async def main():
    while True:
        kirmizi = color_sensor.rgbi(port.B)[0]
        yesil = color_sensor.rgbi(port.B)[1]
        mavi = color_sensor.rgbi(port.B)[2]
```

```
        print(kirmizi,yesil,mavi)
        if 690<kirmizi<710 and 310<yesil<330 and 290<mavi<310:
            sound.beep(440,300,volume=100)
runloop.run(main())
```

Not

RGB renk modeli renk algılanması işlemi için verimli değildir. Bu iş için HSV adında başka bir renk modeli daha verimli sonuçlar üretir. Fakat ders kapsamında bu modelin bilinmesi gerekli değildir. Öğrencilerin RGB modeli ile renk algılama kodlarının verimsiz olduğunu bilmesi yeterlidir.

Gözle ve Uygula: Yansıyan Işık Yoğunluğunu Belirleyen Faktörler

Yansıyan ışık ölçülürken renk sensörü üzerindeki üç ışık kaynağından beyaz ışık gönderir. Karşıdaki cisme (eğer varsa) çarpan ışık yansıyarak renk sensörüne geri döner. Renk sensörü yansıyan ışığın yoğunluğunu ölçebilir. Yansıyan ışığın yoğunluğunu ölçmek için *color_sensor.reflection* metodu kullanılır. Bu metot 0 ile 100 arasında değerler üretir. 0 yansıyan ışığın yoğunluğunun olmadığı ve 100 ise yansıyan ışığın yoğunluğunun en yüksek seviyede olduğunu söyler.

Yansıyan ışığın yoğunluğunu belirleyen çeşitli parametreler vardır. Bunlardan ilki mesafedir. Yansıyan ışık yoğunluğu mesafe ile ters orantılıdır. Renk sensörü karşısındaki cisme ne kadar uzaksa yansıyan ışık yoğunluğu o kadar azalır. İkinci parametre yüzeyin cinsidir. Mat ve pürüzlü yüzeyler ışığı daha az yansıtırken pürüzsüz ve parlak yüzeyler ışığı daha iyi yansıtır. Son parametre ise cismin rengidir. Siyah gibi rengi daha çok soğuran (emen) renkler ışığı daha az yansıtırken beyaz gibi ışığı fazla soğurmayan renkler ışığı daha iyi yansıtır. Işık daha az yansıtıldığında ölçülen ışık yoğunluğu daha az olurken iyi yansıtılan yüzeylerde ışık yoğunluğu daha fazla olacaktır.

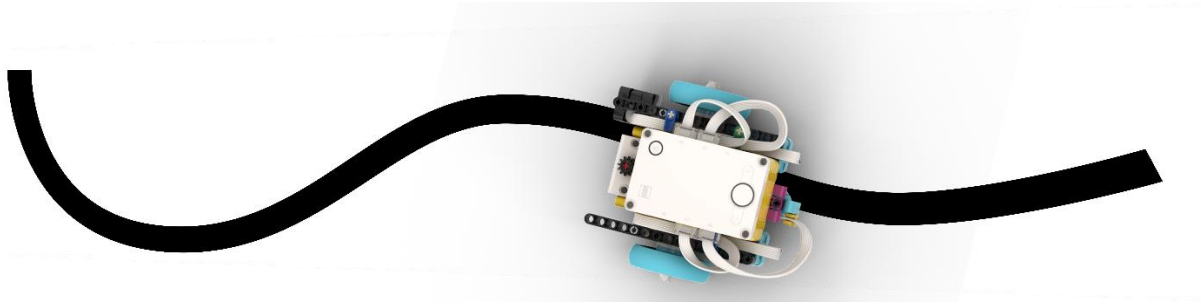
Bu etkinlikteki amaç öğrencilerin farklı yüzeyler ve mesafeler için yansıyan ışık yoğunluğunu deneyimlemeleridir. Rehber öğretmen aşağıdaki kodu öğrencilerle paylaşır ve farklı mesafeler ve yüzeyler için kodun çıktısını gözlemelerini sağlar. Bunun ardından öğrencilerden yansıyan ışığın şiddetini belirleyen faktörleri sınıfça tartışmalarını sağlar. Tartışma sonucunda açık noktalar bulunursa rehber öğretmen gerekli bilgilendirmeyi yapar.

```
from hub import port
import runloop, color_sensor, time
async def main():
    while True:
        print(color_sensor.reflection(port.B))
        time.sleep_ms(250)
runloop.run(main())
```

Bu görev için (i) farklı mesafelerden, (ii) aynı renkteki Lego parçalarından delikli ve düz olanlarından, (iii) siyah ve beyaz yüzeylerden, (iv) farklı renkteki yüzeylerden ve (v) mat ve parlak yüzeylerden yansıyan ışık yoğunluğu ölçülebilir.

Gözle ve Uygula: PID Algoritması ile Çizgi Takip Eden Robot

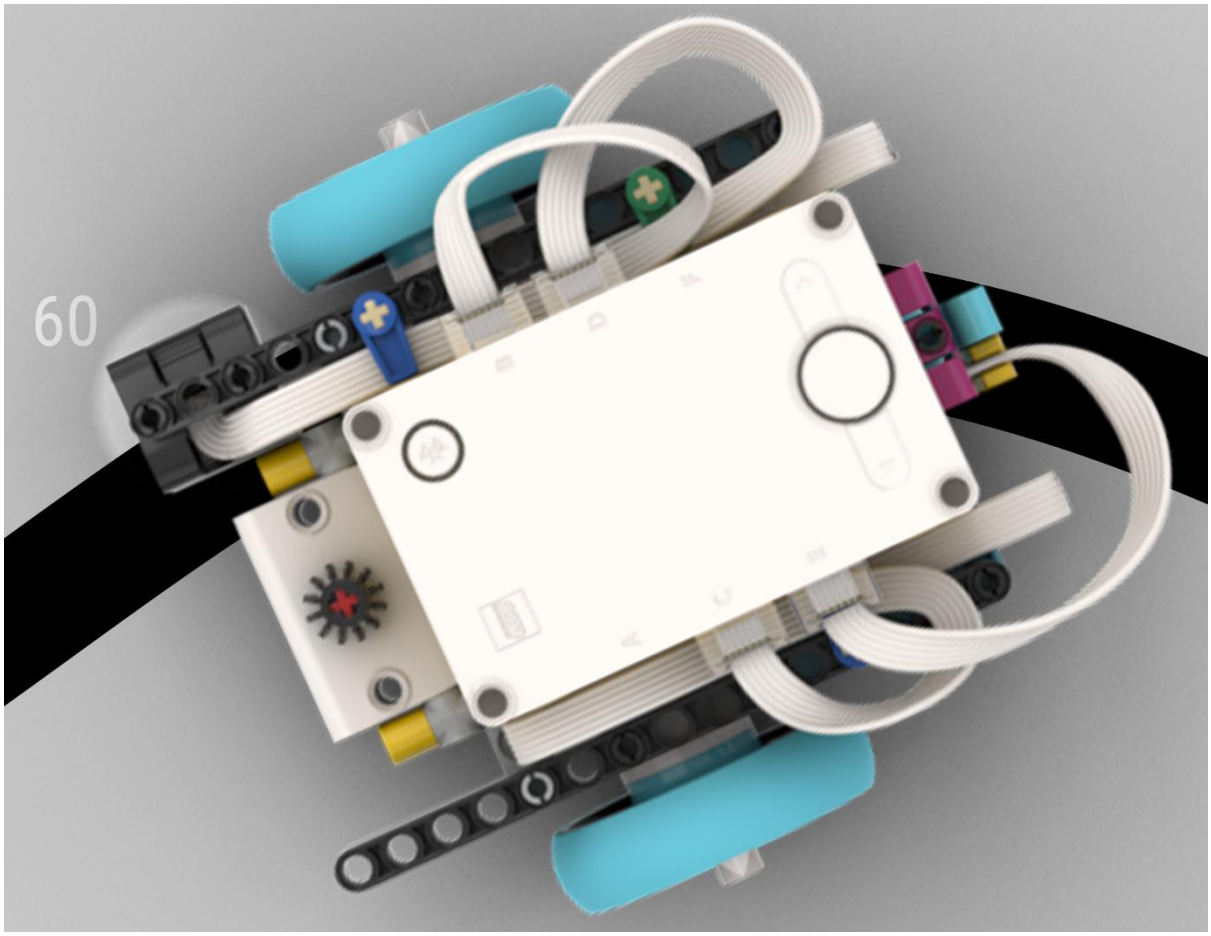
Bu etkinlikte PID algoritması ile çizgi takip eden robot programı yazılacaktır. Bu ders kapsamında dördüncü haftada çizgi takip programı yapılmıştır. Bu program fazla zig zag yapacak şekilde hareket eder ve eğimli çizgileri takip etmekte zorlanır. Yani program yeterince verimli çalışmaz. Bu etkinlikte robot aşağıdaki şekilde gösterilen eğri bir çiziyi takip edecektir.



Resim 10.1 Çizgi Takip Eden Robot Etkinliği

Günlük yaşamda birçok alanda gerçekte var olan bir değerin belirlenen hedef değere ulaşıp burada olabildiğince kalmasını sağlamak önemlidir. Örneğin kışın soğuk havalarda termostat sıcaklığı 23 dereceye ayarlanan bir kombi oda sıcaklığını 23 derece civarında tutmaya çabalar. Oda sıcaklığı 23 derecenin altında ise kombi çalışır. Oda sıcaklığı 23 derece olunca termostat devreye girer ve kombiyi kapatır. Fakat peteklerde sıcak su olduğu için oda ısınmaya devam eder oda sıcaklığı 23 dereceyi geçer. Kaloriferlerdeki sıcak su yeterince ısı sağlayamadığında oda yavaş yavaş soğur. 23 derecenin altında termostat kombiyi yeniden çalıştırır. Fakat kombi peteklerdeki suyu ısıtana kadar oda sıcaklığı 23 derecenin altına düşmüş olur. Bu şekilde oda sıcaklığı 23 derecenin belirli bir miktar altı ve üstü aralığında gelip gider. Ev kullanımında oda sıcaklığının bu aralıkta salınım yapması önemli bir probleme sebep olmaz fakat hassas işler gerektiren durumlarda bu salınımın mümkün mertebe az olması gerekir. Bu tip durumlarda PID isimli bir algoritma kullanılabilir.

Çizgi izleyen robotta ışık sensörünün ışığının bir kısmının siyah çizgide bir kısmının ise beyaz çizgide olması istenir. Aşağıdaki şekilde gösterildiği gibi bu durumda yansıyan ışık yoğunluğu yaklaşık olarak 60 olmaktadır. Bu değer ortamdaki ışık miktarına, yüzey için kullanılan kâğıda ve kâğıt üzerindeki renklerin niteliğine göre değişiklik gösterebilir.



Resim10.2 Sensörün Bulunması İstenen Bölge

Saat yönünde bir miktar dönme hareketi ile yansıyan ışık yoğunluğu 60'ın üstüne çıkar ve tersinde bir hareket ile 60'ın altına iner. Çizgiyi takip edebilmesi için robotun olabildiğince yansıyan ışık miktarını 60 civarında tutmaya çalışması gerekir. Daha önce söylendiği üzere bu tip bir görev için PID kullanılabilir.

PID algoritmasının **P**roportional (oransal), **I**ntegral (integral) ve **D**erivative (türev) olmak üzere üç bileşeni bulunur. Algoritmanın ismi bileşen isimlerinin kısaltmasından oluşur. Aslında integral ve türev ileri matematiğin konusudur. Fakat burada kolaylık olması açısından bu kavramlar basitçe ele alınacaktır. PID algoritmasının üç bileşeni de hataya bağlıdır. Hata gözlenen (gerçekleşen) ve istenen değer arasındaki farktır.

Hata= Gerçekleşen – İstenen

Algoritmanın bileşenleri hataya farklı şekillerde tepki vererek hatanın istenen seviyede kalıp mümkün mertebe sıfıra yakın olmasına çalışır. Algoritmanın oransal kısmı hatanın şu anki haliyle ilgilenir ve ona oransal bir tepki verir. Aşağıda gösterildiği gibi Hata Kp isimli katsayı ile çarpılarak tepki oluşturulur. Bu katsayılar farklı koşullarda farklı tepkiler oluşturmak için kullanılır. Çizgiyi izlerken kullanılacak tepki ile oda sıcaklığını belirli bir sıcaklıkta tutmak için kullanılan katsayılar farklılaşacaktır.

Kp x Hata

Örneğin ortamdan yansıyan ışık değeri 80 olsun, bu durumda 80-60=20'lik bir hata bulunur. Yani robot siyah şeritten uzaklaşmıştır. Bu hata oranında robotun yönü saat yönünün tersine çevrilir ve hata telafi edilmeye çalışılır.

Gerçek yaşamda bazı durumlarda oransal tepki hatayı telafi etmekte yeterli olsa da birçok durumda yeterli olamaz. Robot zig zaglar çizerek ilerler. Bu da onun 60'ın altında ve üstünde değerler aldığını gösterir. Her bir zig zag hareketinde yeni hatalar oluşmakta ve bu hataların toplamı robotu hedeften saptırmaktadır. Burada integral devreye girer. Hataların toplamı oranında düzeltme yapılması gerekir. İntegral ile geçmişte yapılan hatalar telafi edilmeye çalışılır.

I=Hata Toplamı

Programlama açısından bakılacak olursa aşağıdaki formül bulunacaktır. Bu formül bir döngü içerisinde düşünülmelidir. Her döngü adımında yeni hata integrale eklendiği için toplam hata elde edilir. Bunun için integralin ilk değerinin 0 olarak belirlenmiş olması gerekir. Aksi takdirde toplam hata tam olarak bulunamaz.

I = I + Hata

Oransal ve integralin toplamı ile şu anki hata ve geçmişte yapılan hatalar giderilmeye *çalışılmıştır*. Fakat bazen bu işi yaparken uç durumlar oluşabilir. Bu da verilen tepkinin gereğinden keskin olmasına neden olabilir. Bu durumda gelecekte yapılma ihtimali olan hataların göz önünde bulundurulması gerekir. Gelecekte aşırı bir tepki verilme ihtimali varsa bu dengelenmelidir. Dengeleme görevi türeve düşer. Gelecekteki hata değerini tam olarak belirlemek mümkün değildir. Fakat gelecekteki hata şu anki hata ve bir önceki hata göz önünde bulundurarak tahmin edilebilir. Şu anki hatadan bir önceki hata çıkarılır. Bu değer eğer büyükse son iki hata arasındaki fark büyük olduğundan bir sonraki hatanın da büyük olma ihtimali vardır. Yani aşırı tepki ihtimali doğmuştur. Aşağıda formülü verilen türev ile bu aşırılık telafi edilmeye çalışılır.

T=Güncel Hata – Son Hata

Oluşturulacak tepki bu üçlü bileşen tarafından belirlenir. Fakat üç bileşen tepkiyi oluştururken doğrudan alınmaz. Bunlar birer kat sayı ile çarpılarak tepki oluşturulur. Bu katsayılara sırasıyla Kp (P için), Ki (I için) ve Kd (D için) ismi verilmiştir.

Tepki=Kp x P + Ki x I + Kd x D

Bu görev için gerekli kod aşağıda verilmiştir. Kodda bulunan kat sayılar ve istenen değer aynı ortamda farklı aydınlanma miktarlarına göre bile farklılık gösterir. Hava aydınlıkken çalışan kod hava karardığında tam olarak çalışmayabilir. Katsayılar ve istenen değer öğrenciler ve rehber öğretmen tarafından ayarlanmalıdır.

```
from hub import port
import runloop, color_sensor, motor_pair
async def main():
    Kp=1.75     # Katsayılar farklı ortamlarda
    Ki=0.015    # farklı değerler alabilir.
    Kd=0.3      # Kendi ortamınıza göre düzenleyiniz.
    P=0
```

```
    I=0
    D=0
    son_hata=0
    istenen=60 # Bu değer ortama bağlı olarak değişebilir.
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    while True: # Kendi ortamınıza göre düzenleyiniz.
        gerceklesen=color_sensor.reflection(port.B)
        hata=istenen-gerceklesen # Hata robot hareket yönüne
        P=hata                   # negatif olarak yansıyacaktır.
        I= max(min(I + hata, 100), -100) # Integral birikimini
sınırlıyoruz
        D=hata-son_hata
        Tepki=int(Kp*P+Ki*I+Kd*D) # Tepki tam sayıya çevrildi
        Tepki = max(min(Tepki, 100), -100)# Tepkiyi (-100 ile 100)
sınırlıyoruz
        motor_pair.move(motor_pair.PAIR_1, Tepki, velocity=75)
        son_hata=hata
runloop.run(main())
```

Rehber öğretmen öncelikle bu etkinlikte ne yapılacağını öğrencilere anlatır. Ardından öğrencilere PID konusunu anlatır. Anlattığı konuya göre gerekli programı yazar ve programı yazarken açıklar. Tüm bu süreç bittikten sonra öğrencilerden istenen kodu kendilerinin yazmasını sağlar.

Not

Çalışma alanındaki eğri çizgi için Hafta10_PID_Egri_Mat ismindeki dosyada bulunan sayfalar A4 kâğıdına çıktı alınır. Çıktı alındıktan sonra sayfalar sırası ile birleştirilerek çalışma alanı oluşturulur. Çalışma alanı oluşturulurken iki A4 kâğıdı arasında boşluklar kalabilir. Bu boşluklar kesilerek siyah çizgilerin aralıksız bir şekilde birbirini takip etmesi sağlanır.

Uygula: Dikdörtgen Şeklinde Çizgiyi Takip Eden Robot

Bu etkinlikte PID algoritması kullanarak dikdörtgen şeklinde bir çizgiyi takip eden robot programı yazılacaktır. Robot dörtgenin köşelerinde keskin dönüşler yapmalıdır. Yani geçmişteki ve gelecekteki hatalara nazaran güncel hata ön plandadır. Bu yüzden Kp katsayısı yüksek tutulmalıdır. Kp gereğinden fazla olursa robot çizgi takibini bırakıp çizgiyi geçebilir, gereğinden az olursa robot çizgiyi bulamayabilir veya geç bularak kendi etrafında dönmeye başlayabilir. Bu yüzden Kp katsayısı çok iyi ayarlanmalıdır. Kp'nin ayarlanmasının ardından deneme yanılma yöntemi ile diğer katsayılar da ayarlanabilir. Rehber öğretmen öğrencilere katsayılar hakkındaki bu bilgileri vermeden onların kod yazmalarını sağlar. Ardından sınıfça tartışma yaparak katsayıların nasıl ayarlanabileceğini öğrencilerin anlamalarını sağlar. Örnek katsayılar: Kp=2.55, Ki=0.001 ve Kd=0.016

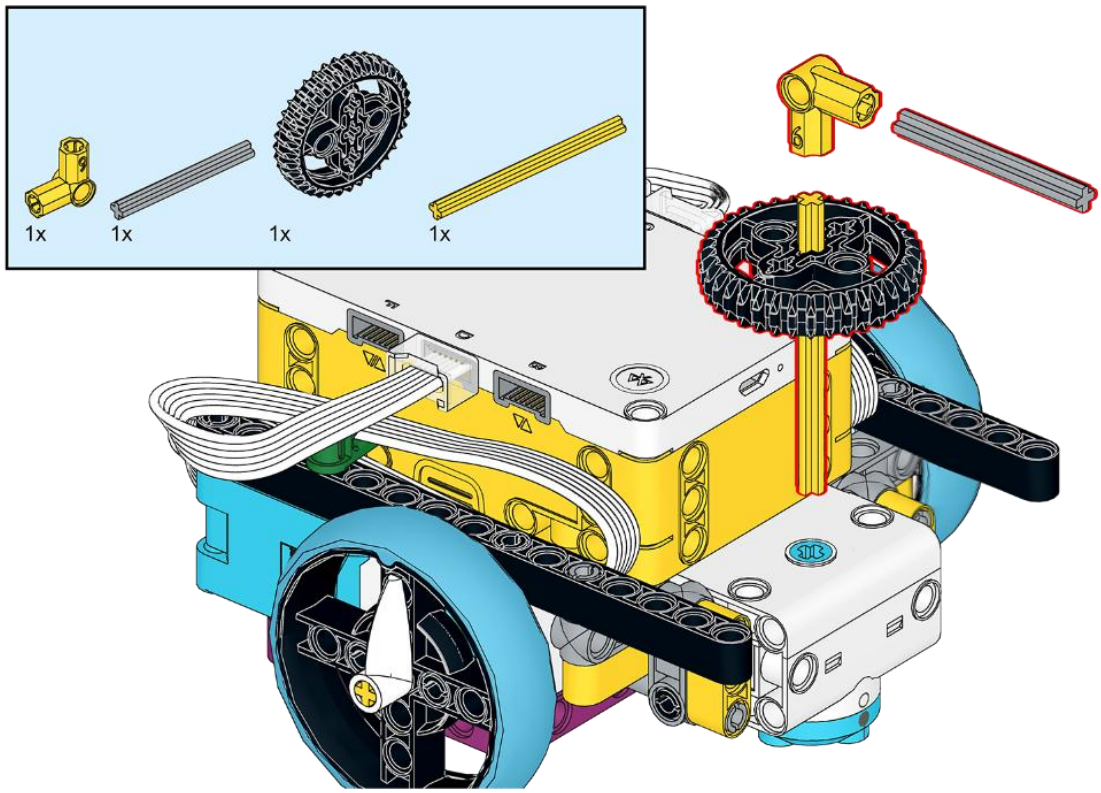
Not

Çalışma alanı için ekte Hafta10_PID_Dortgen_Mat dosyasında bulunan A4 boyutundaki belgenin çıktısı alınır. Bu sayfalar bir bant aracılığı ile birleştirilir. Birleştirme esnasında çizgiler arasında boşluklar bulunabilir. Boşluk bulunan kısımlar makasla kesilerek kesintisiz bir dikdörtgen oluşturulması sağlanır. Ayrıca, program dörtgen çalışma alanı üzerinde denenirken robot köşelerde kâğıtların dışına çıkacak şekilde taşıyorsa, köşelere beyaz A4 kâğıdı konulmalıdır. Aksi halde çalışma masasının rengi programın çalışmasını etkileyebilir. A3 çıktısı almak için uygun yazıcısı olanlar Hafta10_PID_Dortgen_Mat _A3 dosyasının çıktısını alıp doğrudan kullanabilirler.

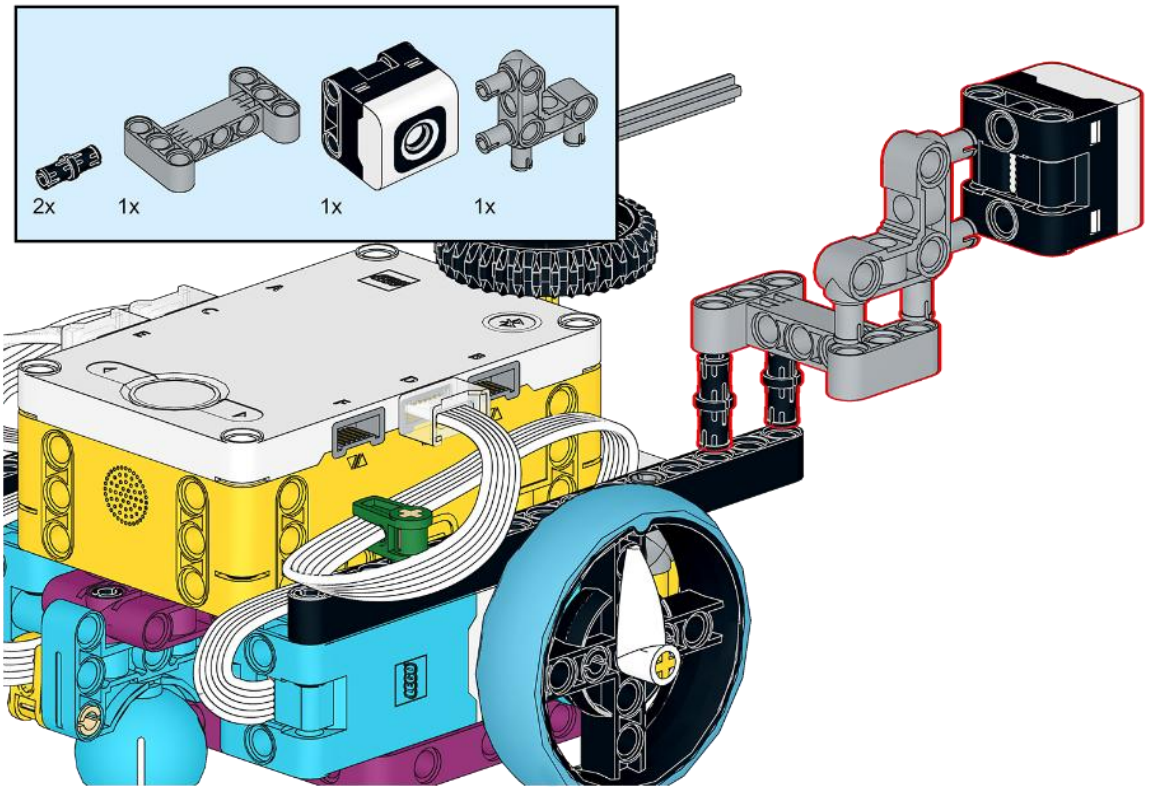
TASARLA

Lego Parçasıyla Robotu Kodluyorum

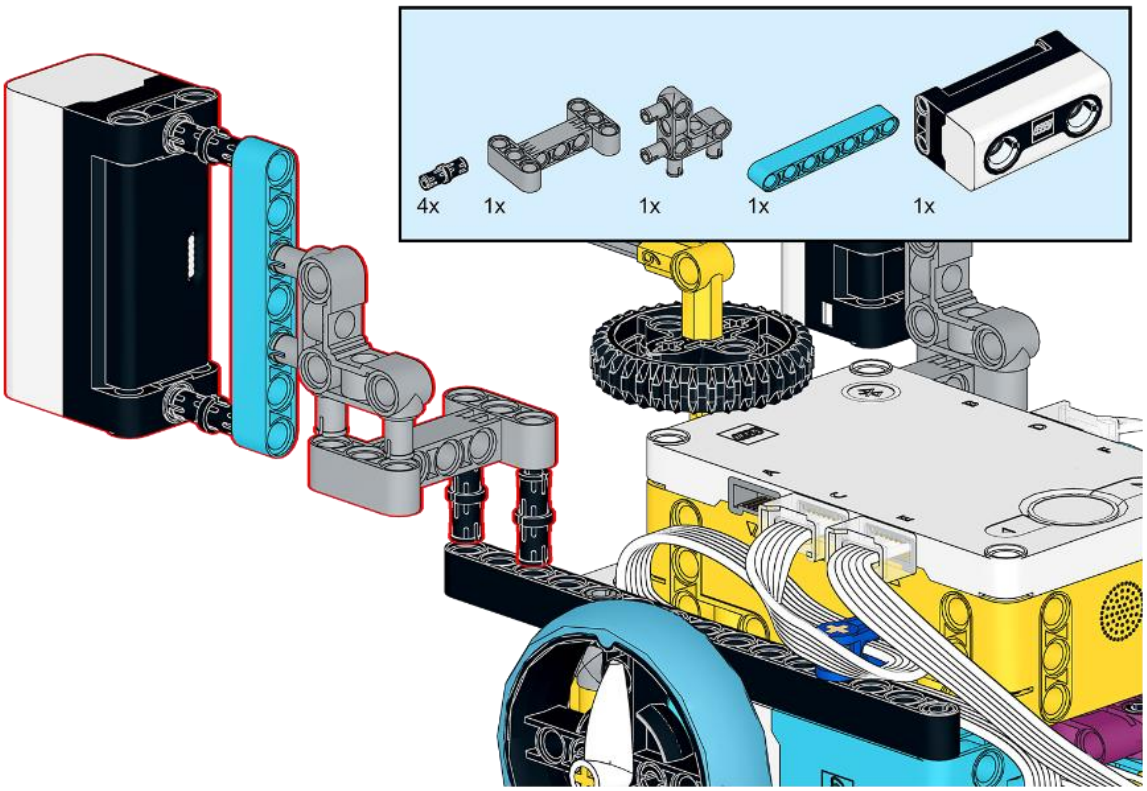
Bu etkinlikte renk sensörü, mesafe sensörü, kuvvet sensörü ve Hub özellikleri birlikte kullanılacaktır. Etkinlik için kullanılacak robotun inşası aşağıda gösterilmiştir.



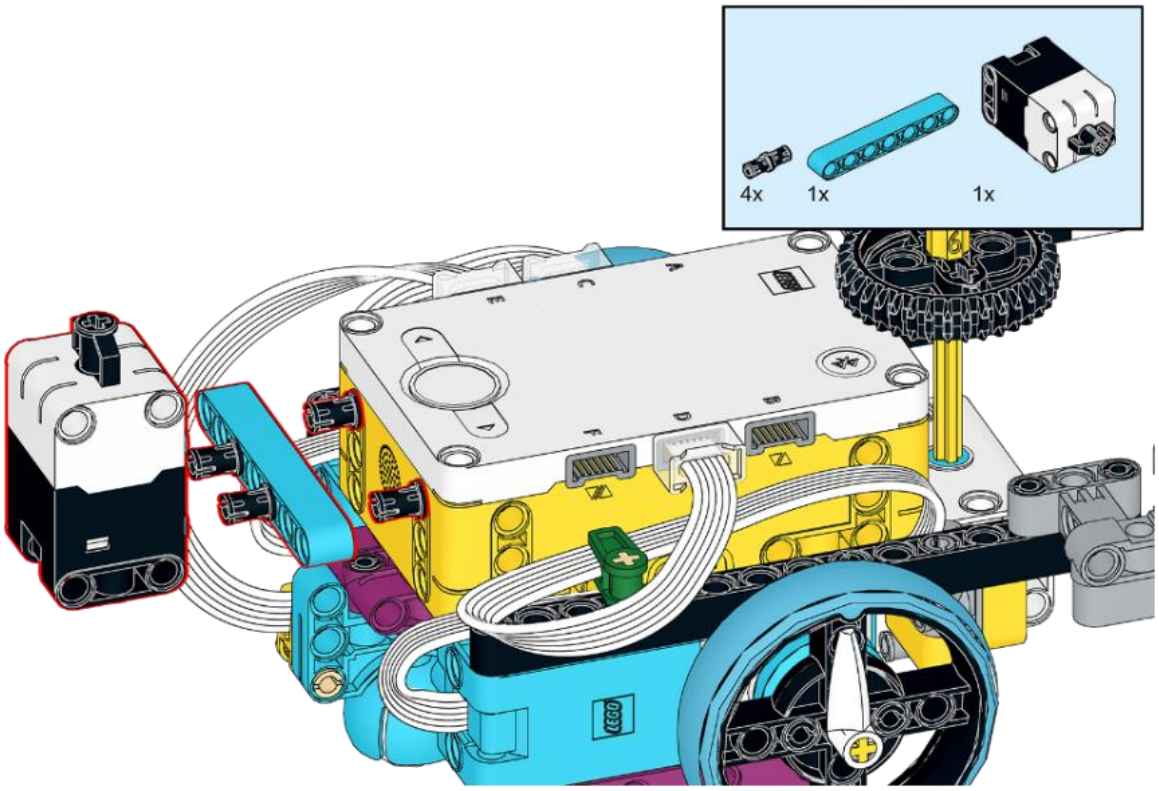
Resim 10.3 Etkinlik İçin Robotun Düzenlenmesi Birinci Adım



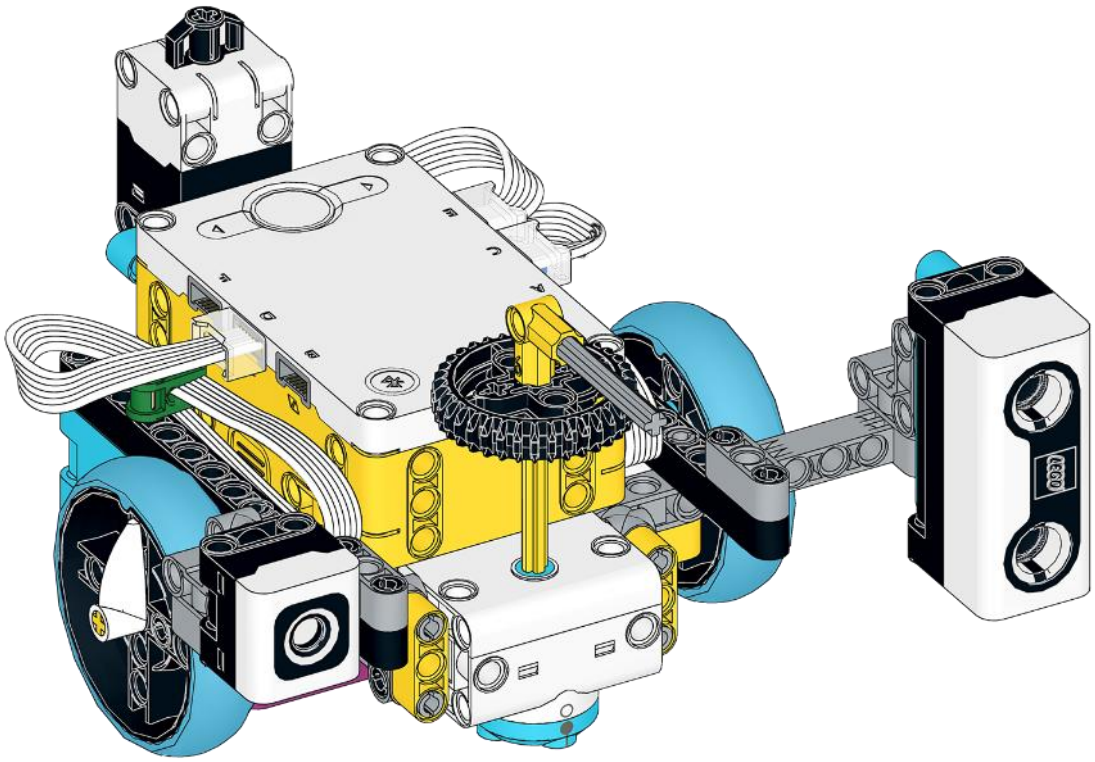
Resim 10.4 Etkinlik İçin Robotun Düzenlenmesi İkinci Adım



Resim 10.5 Etkinlik İçin Robotun Düzenlenmesi Üçüncü Adım



Resim 10.6 Etkinlik İçin Robotun Düzenlenmesi Dördüncü Adım

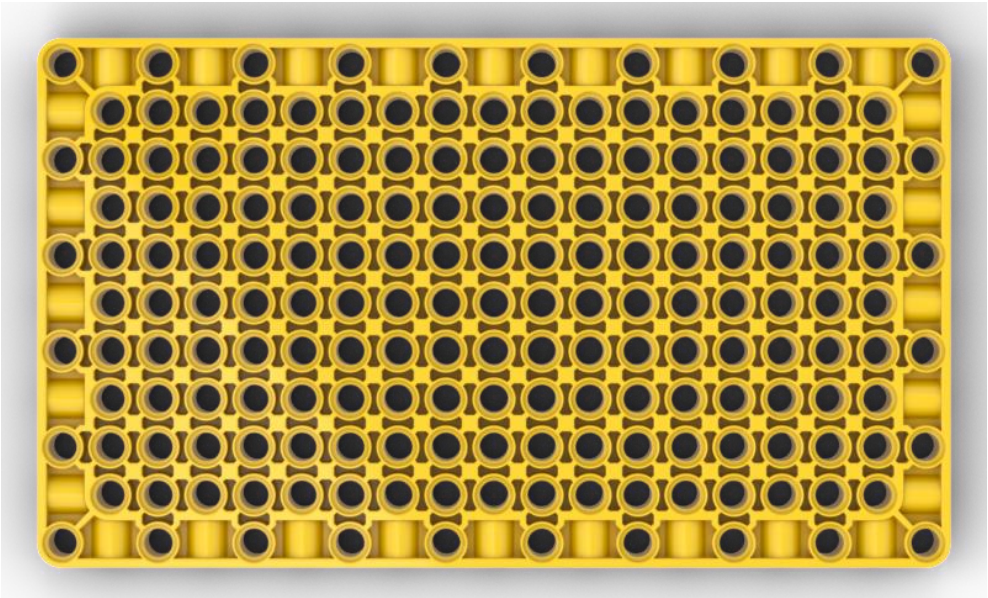


Resim 10.7 Etkinlik İçin Kullanılacak Robot

Robotun hareketi aşağıda örneği verilecek bir Lego parçası ile şu kurallara uygun olacak şekilde kodlanacaktır:

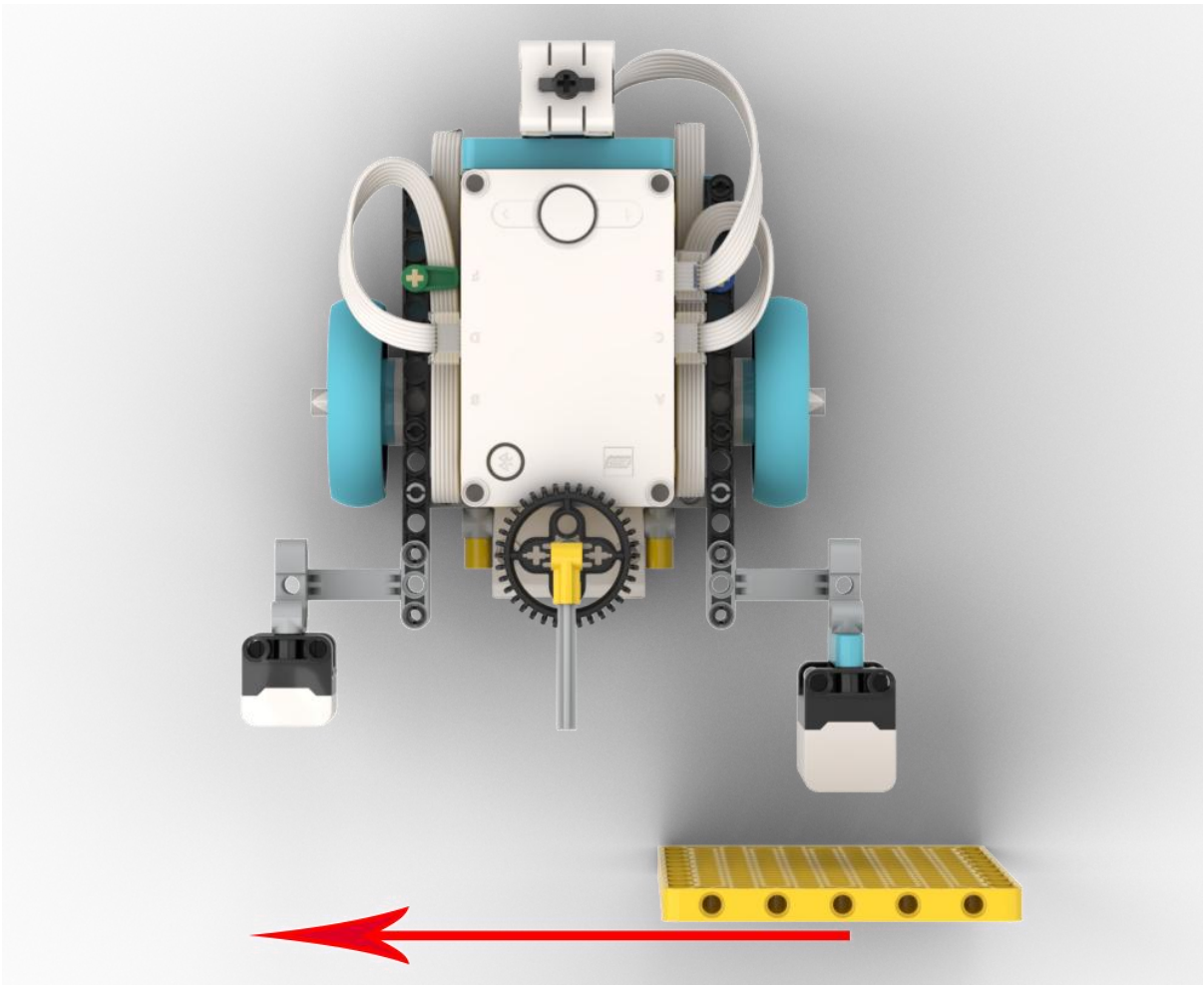
i. Lego parçası önce renk sensörünün ve ardından mesafe sensörünün önünden çok yakın olacak şekilde geçerse bu sağa dön komutu olacaktır.
ii. Lego Parçası önce mesafe sensörünün ve ardından renk sensörünün önünden çok yakın olacak şekilde geçerse bu sola dön komutu olacaktır.
iii. Lego parçası mesafe sensörünün yakınından başlayıp, ondan uzaklaşırsa bu ileri git komutu olacaktır.
iv. Lego parçası mesafe sensörünün uzağından başlayıp yakınına giderse bu geri git komutu olacaktır.
v. İleri-geri ne kadar gidileceğini ve sağa-sola ne kadar dönüleceğini büyük motora bağlı olan değer kolu belirleyecektir. Değer kolu saat yönünün tersine döndükçe artan değerler elde edilmelidir. Değer kolu en sağda iken 0 (sıfır) olmalı. Değer kolu dönüşün ne kadar keskin olacağına karar vermeli. Sıfır olduğunda düz, değerin büyüklüğüne göre daha keskin olmalı. İleri-geri gitmek için değer kolunda belirtilen değer kadar ileri veya geri gidilmelidir.
vi. İstenilen sayıda komut verilebilir.
vii. Kuvvet sensörüne basıldığında kodlama işlemi bitecektir ve robot verilen komutları yerine getirecektir. Örneğin kol değeri 90 iken sola dön, 100 iken sağa dön, 20 iken ileri git, 30 iken geri git komutları sırası ile verildikten sonra kuvvet sensörüne basıldığında robot sırasıyla 10 cm boyunca -90 yönünde, 10 cm boyunca 100 yönünde, 20 cm boyunca ileri, 30 cm boyunca geri gitmelidir.

Bu görev için aşağıdaki Lego parçası kullanılabilir. Parçanın rengi sarı olmak zorunda değildir. Farklı renkteki parçalar da kullanılabilir. Komut verilirken Lego parçası kısa kenarı genişliği ve uzun kenarı ise yüksekliği olacak şekilde sensörlerin karşısında hareket etmelidir.



Resim 10.8 Kullanılan Lego Parçası

Sağa ve sola komutları için Lego parçası aşağıdaki resimde gösterildiği gibi sensörlere çok yakın olmalıdır. Komut vermek için başka Lego parçası da kullanılabilir. Fakat Lego parçasının mümkün mertebe büyük olması komut vermeyi kolaylaştıracaktır.



Resim 10.9 Sola Dön Komutu

Tanımlama: Bu aşamada öğrenciler problemi tanımlamalıdır. Problemi tanımlamak verilenleri, istenenleri, konu ile ilgili var olan bilgileri, ihtiyaçları ve geçmiş problem çözme deneyimlerini ortaya koymak açısından önemlidir. Bu bilgiler problem çözümünde kullanılabilir. Aşağıda örnek tanımlama adımları verilmiştir.

i. Sola dön ve sağa dön komutları iki aşamalı olarak düşünülebilir. Soldaki sensörden (renk) başlanıp sağdaki sensörün (mesafe) terk edilmesi sağa dön ve sağdaki sensörden başlanıp soldaki sensörün terk edilmesi ise sola dön anlamına gelebilir. Görüldüğü gibi komutlarda başlangıç ve terk etme gibi iki adım bulunmaktadır.
ii. Sola veya sağa dönme komutları için Lego parçası sensörlerin çok yakınından geçmelidir. Bu mesafe 6 cm olarak belirlenebilir.
iii. İleri veya geri git komutları mesafe sensörü ile başlar ve mesafe sensöründe biter. Burada ikili bir yapı söz konusu değildir. Fakat genel bir programlama yapısı oluşturmak için bu komutlar da iki adımlı olarak düşünülebilir.
iv. İleri git komutunda mesafe sensörüne yaklaşık 30 cm'den yaklaşmaya başlanmalı ve yaklaşık 15 cm mesafede komut bitirilmelidir.

v. Geri git komutunda mesafe sensöründen uzaklaşmaya yaklaşık 15 cm mesafeden başlanmalı ve 30 cm'de komut bitirilmelidir.
vi. Verilecek komutlar arasına bir miktar süre konulması komutların daha rahat algılanmasına yardımcı olabilir.
vii. Dönme yönü ve ileri-geri gitme miktarı için büyük motordan elde edilecek değerler kullanılabilir.
viii. Kuvvet sensörüne basılması komut verme işinin bittiği anlamına gelecektir. Kuvvet sensörüne basıldığında verilen komutlar yerine getirilmelidir.

Fikir Üretme: Bu aşamada öğrenciler problem çözümüne yönelik daha somut önerilerde bulunmalıdır. Hangi algoritmalar ve programlama yapıları kullanılarak problem çözülebilir, problem çözümü için gerekli parametreler nasıl seçilmelidir ve hangi programlama yöntemleri kullanılabilir gibi sorulara yanıt aranmaktadır. Aşağıda fikir üretme aşaması için bir örnek verilmiştir.

i. Sağa, sola, ileri ve geri komutlarının iki adımlı olarak planlanabilmesi için komutun başladığı ve bittiği evreler için iki farklı değişken kullanılabilir.
ii. Komutlar ve değerler birer listede saklanabilir. Komutlar ve değerler aynı sırada alınarak çalıştırılmaları sıra bazlı gerçekleştirilebilir.
iii. Print komutu kullanılarak büyük motordan alınan değerin konsolda görünmesi sağlanabilir.
iv. Kol döndürülürken her bir derece için bir birimlik artış gerçekleştirilebilir. Değer 100'ü geçmemelidir. Geçerse program vasıtası ile düzeltilmelidir.
v. Soldan veya sağdan komut vermeye başlanması değişkenler yardımıyla belirlenebilir.
vi. Başlangıç mesafesi 6 cm'den büyük ve 15 cm'den küçük olup bitişi 30 cm'den daha büyük olan komutlar ileri git olarak algılatılabilir.
vii. Başlangıç mesafesi 30 cm'den büyük olup bitişi 15 cm ile 6 cm aralığında olan komutlar geri git olarak algılatılabilir.
viii. Her bir komut arasındaki süre için 1.5 saniye belirlenebilir.
ix. Kuvvet sensörü tıklandığında komut verme işleminin sonlandırılması için while döngüsü kullanılabilir.
x. For döngüsü ile komutlar ve değerleri sırayla alınıp uygun kodlar çalıştırılabilir.

Lego parçasıyla robot kodluyorum etkinliği görece zor bir etkinliktir. Öğrenciler tanımlama ve fikir üretme adımlarında çeşitli zorluklara karşılaşıp bu adımları yerine getirmek istemeyebilirler. Bu gibi durumlarda öğrencilere problem çözmeye başlamadan önce problem üzerine düşünmenin faydalı olduğu ve uzman programcıların bu şekilde çalıştığı hatırlatılır. Öğrenciler tam olarak tanımlama ve fikir üretme adımlarını gerçekleştiremiyorsa, en azından doğrudan çözüme başlamak yerine öncelikle problem ve çözümü hakkında düşünmelidir. Öğrenciler her hâlükârda doğrudan probleme geçmek isterse, onlara çözüme başlamaları ve hızlı bir çözüm pratiğinin ardından yeniden genel bir problem çözüm tasarısı ortaya koymaları önerilmelidir.

ÜRET

Lego Parçasıyla Robotu Kodluyorum

Öğrenciler tasarlama adımında ortaya koyduğu temel üzerinden bilgisayar başında kod yazmaya başlarlar. Rehber öğretmen bu adımda öğreten pozisyonunda olmadığı için çözüme yönelik soruları doğrudan yanıtlamaz. Fakat problem görece zor olduğu için çözümün mantığı üzerine yorumlar yaparak öğrencileri yönlendirir. Bu etkinlik öğrencilere büyük bir problemi parçalara ayırarak çözme pratiğini göstermek için iyi bir örnektir. Öğrencilerden problemi bir bütün olarak çözmek yerine öncelikle sola dön komutu üzerine yoğunlaşmaları istenebilir. Bunu başardıktan sonra sırasıyla, sağa dön, ileri git, geri git ve büyük motordan değer oku gibi alt problemler üzerine yoğunlaşabilirler. Her bir alt çözümün doğruluğundan emin olduktan sonra, bu alt çözüm daha önce ortaya konan alt çözüme eklenebilir. Böylece problem çözümü alt çözümler eklenerek devam eder. Bunun yerine öğrenciler her bir alt problemi ayrı ayrı çözdükten sonra bu alt çözümleri birleştirme tasarısını yapabilir ve ardından bütün alt çözümleri bir anda birleştirerek problem çözümünü oluşturabilir. Aşağıda örnek bir kod verilmiştir. Öğrenciler bu koda benzer çözümler üretmek zorunda değildir, çok farklı çözümlere ulaşabilirler.

```
from hub import port, sound
import runloop, color_sensor, distance_sensor, force_sensor, motor_pair,
color, time, motor
async def main():
    miktar_yon=[]
    komutlar=[]
    basladi=False
    bitti=True
    soldan_giris=False
    sagdan_giris=False
    ileri_git=False
    geri_git=False
    mesafe=None
    sayac=1
    await motor.run_to_absolute_position(port.E, 0, 200)
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    while force_sensor.pressed(port.A) == False:
        deger=motor.absolute_position(port.E)
        if deger>100:
            deger=100
        if deger <-100:
            deger = -100
        if basladi == False and bitti == True:
            if sayac==1:
                sound.beep(350)
                sayac=2
            mesafe=distance_sensor.distance(port.F)
            if mesafe != -1 and mesafe < 200:
                if mesafe < 60:
```

```
                sagdan_giris=True
                basladi=True
                bitti=False
            elif mesafe < 150:
                while True:
                    if distance_sensor.distance(port.F) >= 300:
                        break
                ileri_git=True
                basladi=True
                bitti=False
                sound.beep(350)
            elif mesafe < 300:
                while True:
                    if distance_sensor.distance(port.F) <= 150:
                        break
                geri_git=True
                basladi=True
                bitti=False
            sayac=1
        if color_sensor.reflection(port.B) > 5:
            soldan_giris=True
            basladi=True
            bitti=False
            sayac=1
    if basladi==True and bitti==False:
        if soldan_giris==True:
            mesafe=distance_sensor.distance(port.F)
            if mesafe != -1 and mesafe < 60:
                while True:
                    if distance_sensor.distance(port.F) >= 100:
                        break
                komutlar.append("saga_don")
                miktar_yon.append(deger)
                soldan_giris=False
                sound.beep(400)
                await runloop.sleep_ms(1500)
                basladi=False
                bitti=True
                await motor.run_to_absolute_position(port.E, 0, 200)
        if sagdan_giris==True:
            if color_sensor.reflection(port.B) > 5:
                renk=color_sensor.color(port.B)
```

```
                    while True:
                        if renk != color_sensor.color(port.B):
                            break
                    komutlar.append("sola_don")
                    miktar_yon.append(deger)
                    sagdan_giris=False
                    sound.beep(400)
                    await runloop.sleep_ms(1500)
                    basladi=False
                    bitti=True
                    await motor.run_to_absolute_position(port.E, 0, 200)
            if ileri_git==True:
                komutlar.append("ileri_git")
                miktar_yon.append(deger)
                ileri_git=False
                sound.beep(400)
                await runloop.sleep_ms(1500)
                basladi=False
                bitti=True
                await motor.run_to_absolute_position(port.E, 0, 200)
            if geri_git==True:
                komutlar.append("geri_git")
                miktar_yon.append(deger)
                geri_git=False
                sound.beep(400)
                await runloop.sleep_ms(1500)
                basladi=False
                bitti=True
                await motor.run_to_absolute_position(port.E, 0, 200)
    for indeks in range(len(komutlar)):
        if komutlar[indeks]=="ileri_git":
            await motor_pair.move_for_degrees(motor_pair.PAIR_1,
miktar_yon[indeks]*10,0,velocity=400)
        if komutlar[indeks]=="geri_git":
            await motor_pair.move_for_degrees(motor_pair.PAIR_1,
miktar_yon[indeks]*10,0,velocity=-400)
        if komutlar[indeks]=="saga_don":
            await motor_pair.move_for_degrees(motor_pair.PAIR_1,360,
miktar_yon[indeks],velocity=400)
        if komutlar[indeks]=="sola_don":
            await motor_pair.move_for_degrees(motor_pair.PAIR_1,360, -
miktar_yon[indeks],velocity=400)
```

```
runloop.run(main())
```

DEĞERLENDİR

Rehber öğretmen aşağıdaki soruları öğrencilere sorar ve sınıf tartışması içerisinde çözüm önerilerini ortaya koymalarını sağlar.

i. Bugün gerçekleştirilen programlama etkinliklerinden en zorlandığınız hangisidir? Sebebini açıklamaya çalışınız.
ii. PID algoritması nedir? Genel olarak özetleyiniz.
iii. PID günlük yaşamda nerelerde kullanılabilir?
iv. Şimdiye kadar görülen bütün konular kullanılarak bir günlük yaşam problemi için çözüm önerisi getirmeniz beklenmektedir. Probleminizi tanımlayınız ve problem çözümünde kullanılacak robotik kavramlarını açıklayınız.

Proje Hazırlıyorum- Fikir Üretme Aşaması

Öğrenciler geçen hafta boyunca empati ve tanımlama adımlarını gerçekleştirmiştir. Artık öğrencilerin problemleri bellidir ve problemin uygun (verimli, konforlu veya keskin) çözümü için gerekli kriterler oluşturulmuştur. Rehber öğretmen öğrencilerin empati ve tanımlama adımlarındaki ürünlerini değerlendirerek onlara gerekli yönlendirmeleri yapar.

Bu hafta ise öğrenciler problemin çözümü için bir tasarı gerçekleştirecektir. Buna fikir üretme aşaması denir ve öğrenciler bu adımı Tasarım ve Üretim dersinde deneyimlemiştir. Öğrenciler bu hafta problemin çözümüne yönelik fikirlerini ortaya koyacaktır. Bu aşamada başlangıçta olabildiğince farklı fikirlere yer vermek daha verimli olacaktır. Öğrenciler bu farklı fikirler üzerinde düşünerek sonuçta bunlardan bir sentez elde edebilir veya indirgemeye giderek bunlardan birini benimseyebilir. Rehber öğretmen süreçte öğrencileri farklı fikirler üreterek bunların üzerinde düşünmeye yönlendirmelidir. Bunun için grup ve sınıf tartışmaları oluşturabilir. Bu adımda empati ve tanımlama adımında elde edilen bilgilere dayanarak aşağıdaki sorulara cevap aranabilir.

- Robotun ne yapması beklenmektedir?
- Problemin bileşenleri nelerdir?
- Bu bileşenlere yönelik olarak elde ne gibi bilgiler bulunmaktadır?
- Bileşenlerin çözümünde bilinen hangi algoritmalar kullanılabilir?
- Bileşenlerin çözümü için ne gibi algoritmalar üretilebilir?
- Bileşenlerin çözümü için daha verimli algoritmalar oluşturulabilir mi?
- Bileşenler nasıl bir araya getirilebilir?
- Robotun mekanik tasarımı nasıl olmalıdır?
- Robotun üretimi için ne gibi adımlar izlenmelidir?
- Projenin başarıya ulaştığı nasıl test edilebilir?

Bu sorular örnek olarak verilmiştir. Öğrenciler bu adımda kendi sorularını belirleyip onlar üzerine de düşünebilir. Fakat öğrenciler ana fikirden uzaklaşmamalıdır. Rehber öğretmen öğrencileri bu adımda çözüme ve çözümün verimliliğine yönelik yönlendirmelidir?

Sonraki Haftaya Hazırlık:

Öğrenciler bir sonraki haftaya kadar fikir üretme adımını tamamlamış olması gerekmektedir. Öğrenciler gerekli ayarlamaları yaparak laboratuvarları kullanabilir ve rehber öğretmenden destek isteyebilir. Rehber öğretmen öğrencileri yönlendiren pozisyonda olmalıdır. Çözüm konusunda doğrudan müdahalelerde bulunmamalıdır. Bir sonraki haftadan itibaren üretim aşamasına geçilecektir. Öğrencilerin önceki adımlardan eksikliklerinin olmaması gereklidir. Rehber öğretmen bu konuya yönelik olarak öğrencilere gerekli uyarıları yapar.

11. Hafta- Proje Hazırlıyoruz

Haftanın Amacı:

Dersin sonunda, öğrencilerin şimdiye kadar öğrendiklerini kullanarak seçecekleri bir gerçek hayat problemine çözümler üretilmeleri amaçlanmaktadır. Başka bir ifadeyle öğrenciler ders sonunda grup halinde farklı projeler üretecektir.

Haftanın Kazanımları:

- Proje ekibini oluşturur.
- Gerçek hayat problemlerini belirler.
- Yenilikçi fikirler üretir.
- Seçilen probleme tasarlanan robotlar ile çözüm üretir.
- Bir probleme yönelik çözüm önerilerini prototipe dönüştürür.

Öğrencilerin geçen haftalarda grup halinde başladıkları projelerini bu haftada devam etmesi beklenmektedir. Projelerin hazırlık, empati, tanımlama, fikir üretme aşamalarının üzerinden gidilerek *geliştirme ve test etme* sürecine başlanacaktır. Proje geliştirme aşamaları dinamik bir süreçler olmasından dolayı tekrarlanabilir ve fikirler, çözümler yenilenebilir. Bu süreçlerde rehber öğretmenler gözetiminde öğrencilere ders dışı etkinliklerle de destek olunması önerilmektedir.

GELİŞTİRME – TEST ETME

Etkinliğin Amacı: Öğrenciler tanımladıkları problemi çözmek için kendi tasarladıkları bir robotu geliştirir ve test ederler.

Prototip Geliştirme: Grupların belirledikleri robot projesi için yaptıkları fikir üretme çalışması sonucunda üretilmesine karar verdikleri robotu oluşturup programlamaya başlamaları gerekir.

Gruplardan, fikir üretme aşamasında belirledikleri çözümleri etkinlik sonuna kadar hızlıca uygulamaları, robot tasarımı ve programı geliştirmeleri istenir. Çözüm için geliştirilen robotlarda ve programlarda bazı aksaklıkların/problemlerin olması olağandır. Planlanan robotun ilk denemede oluşturulması ve geliştirilen programın sorunsuz çalışması çok karşılaşılan bir durum değildir. Bu tür sorunların ortaya çıkmasının olağan olduğu rehber öğretmen tarafından öğrencilere söylenmeli ve/veya hissettirilmelidir. Ortaya çıkan sorunların çözümüne yönelik bir plan hazırlanıp uygulanmalıdır. Çözüm için gruplara aşağıdakilere benzer bir süreç önerilebilir:

- Problemler maddeler hâlinde listelenmeli (beklenen robot davranışı ile gerçekleşen arasındaki farklar),
- Problemler arasındaki ilişkiler belirlenmeli,
- Öncelik sırası belirlenmeli,
- Öncelik sırasına göre çözümler geliştirilmeli (Çözüm için hipotezler oluşturulmalı),
- Geliştirilen çözümler (hipotezler) denenmeli ve
- Çözümlerin her koşulda istenen şekilde çalışması için gerekli düzenlemeler yapılmalıdır.

Bütün grupların, hazırladıkları prototipleri (tam olarak çalışmasa bile) rehber öğretmene göstermesi sağlanır. Böylece rehber öğretmenin, geliştirilen ürünü test etmesi/görmesi, çeşitli çözüm önerileri getirmesi ve/veya gözden kaçan noktaları ortaya çıkarması sağlanabilir.

Test Etme: Ortaya çıkan son ürünü hedef kitle ile/hedef ortamda değerlendirmek geliştirilen çözümlerin kusursuz bir şekilde çalışması için önemlidir. Hedef kitleye sınıfta ulaşmak her zaman mümkün olmayabilir. Bu nedenle grupların, ortaya çıkardıkları ürünleri diğer öğrencilerin kendi başlarına deneyimlemelerine izin vermesi, gözden kaçan unsurların ortaya çıkarılmasına yardımcı olabilir. Projelerin test aşaması dersin son haftasına kadar devam eder.

12. Hafta- Proje Hazırlıyoruz

Haftanın Amacı:

Bu hafta, öğrenciler hazırladıkları projeleri tamamlayıp, sunum yapmaları beklenmektedir. Geçen hafta başlanan test aşamalarına bu hafta da devam ederek eksikliklerin giderilmesi beklenmektedir.

Haftanın Kazanımları:

- Bir probleme yönelik çözüm önerilerini prototipe dönüştürür.
- Prototipe çalışmasını test eder.
- Projenin aşamalarını sunar.

Öğrencilerin geçen haftalarda grup halinde başladıkları projelerini bu hafta tamamlaması beklenmektedir. Projelerin tamamlanarak sunum yapılması önerilmektedir.

SUNUM

Projenin son aşamasında gruplara projelerini sunmaları konusunda yardımcı olunmalıdır. Gruplar projelerini bir senaryo çerçevesinde tiyatral olarak sunabilir. Projeler değerlendirilirken süreç bazlı değerlendirme tercih edilmelidir. Özellikle dersin son 4 haftası boyunca proje geliştirme aşamaları dikkate alınabilir. Ayrıca proje değerlendirme için aşağıda verilen örnekteki gibi dereceli puanlama anahtarları (rubrikler) de kullanılabilir.

Proje Dereceli Puanlama Anahtarı

Grup İsmi:

		Mükemmel		İyi		Orta		Zayıf		Kötü	
Mekanik Tasarım	**Dayanıklılık:** Robot yapısal bütünlüğü koruyacak sağlamlıkta tasarlanmıştır.	10	9	8	7	6	5	4	3	2	1
	Mekanik Verimlik: Robotun tasarımı basit ve kolaylıkla düzenlenebilir.	10	9	8	7	6	5	4	3	2	1
	Mekanizasyon: Robot amaçlanan görevler için uygun hız, güç ve doğrulukla hareket etmek üzere tasarlanmıştır.	10	9	8	7	6	5	4	3	2	1
Programlama	**Programlama Kalitesi:** Program hedefe uygun olarak tutarlı sonuçlar üretmiştir.	10	9	8	7	6	5	4	3	2	1
	Programlama Verimliliği: Program sistem kaynaklarını en az kullanarak en hızlı sonucu üretmiştir.	10	9	8	7	6	5	4	3	2	1
	Otomasyon/Navigasyon: Program dışardan müdahaleye gereksinim duymadan otonom şekilde görevini yerine getirmiştir.	10	9	8	7	6	5	4	3	2	1
Sunum	**Yenilikçi Sunum:** Yaratıcı sunum teknik ve yöntemleri kullanılmıştır.	10	9	8	7	6	5	4	3	2	1
	Etkin Sunum: Proje hedef kitlenin en hızlı ve kolay anlayacağı şekilde sunulmuştur.	10	9	8	7	6	5	4	3	2	1
	Sunum İçeriği: Proje geliştirme adımları detaylı bir şekilde anlatılmıştır.	10	9	8	7	6	5	4	3	2	1
Yenilikçi Yaklaşım	**Tasarım:** Proje özgün mekanik tasarım özellikleri içermektedir.	10	9	8	7	6	5	4	3	2	1
	Görev Stratejisi: Proje yenilikçi programlama yöntemleri içermektedir.	10	9	8	7	6	5	4	3	2	1
	Yenilikçi: Proje gerçek hayat problem(ler)ine farklı çözüm(ler) getirmektedir.	10	9	8	7	6	5	4	3	2	1
	Toplam:										