DENEYAP

Teknoloji Atölyeleri

YAZILIM TEKNOLOJİLERİ

ORTAOKUL

Dr. Öğr. Üyesi Caner ÖZCAN
Dr. Öğr. Üyesi Rafet DURGUT
Arş. Gör. Dr. Sevil ORHAN ÖZEN
Bilişim Teknolojileri Öğrt. Sercan ÖZEN

TÜBİTAK

TÜBİTAK Deneyap Kitapları 7

YAZILIM TEKNOLOJİLERİ
ORTAOKUL

Dr. Öğr. Üyesi Caner ÖZCAN
Dr. Öğr. Üyesi Rafet DURGUT
Arş. Gör. Dr. Sevil ORHAN ÖZEN
Bilişim Teknolojileri Öğrt. Sercan ÖZEN

TÜBİTAK Deneyap Kitapları DENEYAP TÜRKİYE Projesi kapsamında hazırlanmıştır.

ISBN 978-605-312-442-9
Yayıncı Sertifika No: 47703

Yayım Tarihi: 2023

TÜBİTAK Başkanı: Prof. Dr. Hasan MANDAL
Bilim ve Toplum Başkanı: Ömer KÖKÇAM
Genel Yayın Yönetmeni: Fatma BAŞAR
Editör: İpek PİRPİROĞLU GENCER
Düzelti: Muhammed Said VAPUR
Telif İşleri Sorumlusu: Havva Hilal KAÇAR

TÜBİTAK Bilim ve Toplum Başkanlığı
Tunus Caddesi No: 80 Kavaklıdere 06680 Ankara
Tel: (312) 298 96 50
e-posta: deneyap@tubitak.gov.tr
https://yayinlar.tubitak.gov.tr/deneyap-atolyesi

İçindekiler

Sunuş

Eğitim dünyası günlük hayatta teknolojisiz ortamda yer almayan, okul hayatlarında da öğrenme için teknolojiden faydalanan pek çok çevrimiçi ürün ve teknolojileri kullanmanın yanı sıra, onları üretme becerisini de gösteren çok yönlü bir nesil ile karşı karşıyadır. Teknoloji çağında doğan dijital yerli olarak adlandırdığımız bu nesil hayatın her alanında hızla değişim yaşanan bir ortamda öğrenmeye adapte olmaya çalışmaktadır. Özellikle çocuklarımızın adaptasyon sürecinde hazırlanan öğretim programlarının şekli ve uygulayıcıların kullandıkları öğrenme öğretme yöntemleri kilit noktadır.

Teknolojiyle birlikte değişen ve gelişen ihtiyaç ve pazar koşullarında çocuklarımızı tüketim alışkanlıkları ile baş başa bırakmamak için 81 İlde 100 Deneyap Teknoloji Atölyesi kurulmasına yönelik olarak, T.C. Sanayi ve Teknoloji Bakanlığı, T.C. Gençlik ve Spor Bakanlığı, TÜBİTAK ve Türkiye Teknoloji Takımı Vakfı arasında önemli bir işbirliği tesis edilmiştir. Bu dört önemli kurumun katkılarıyla 2019 yılında Deneyap Teknoloji Atölyeleri hayata geçirilmiştir.

Ülkemizin kalkınması için teknoloji üretme yetkinliği yüksek genç bireyler yetiştirmeyi hedef alan Deneyap Teknoloji Atölyelerinin eğitim modeli kapsamında 36 ay süre ile ücretsiz olarak Tasarım ve Üretim, Robotik ve Kodlama, Elektronik Programlama ve Nesnelerin İnterneti, Yazılım Teknolojileri, İleri Robotik, Nanoteknoloji ve Malzeme Bilimi, Siber Güvenlik, Yapay Zeka, Mobil Uygulama, Enerji Teknolojileri, Havacılık ve Uzay Teknolojileri derslerine ilişkin eğitimler verilmektedir. Verilen eğitimlerden biri de "Yazılım Teknolojileri"dir. Yazılım Teknolojileri, öğrencilere temel programlama mantığı, problem çözümüne yönelik algoritma tasarımı ve tasarlanan algoritmaları C++ programlama dilini kullanarak kodlama becerisi edinmelerini sağlayan bir eğitim içeriği sunmaktadır. Bu içerik doğrultusunda öğrencilerin gerçek hayat problemlerini algoritmaya dönüştürerek akış diyagramları oluşturabilmesi, C++ programlama dilini kullanarak akış diyagramının koda dönüştürülmesi, C++ fonksiyonların kullanımını sağlayabilmesi, nesneye yönelik programlama ile kodlama gerçekleştirebilmesi, dosyalama kavramını kullanarak proje tasarlayabilir hale gelmesi hedeflenmektedir. Eğitim öğrencilerin girişimcilik, yaratıcı düşünme, eleştirel düşünme, karmaşık problemleri çözme, etkili iletişim ve takım çalışması gibi becerileri kazanmalarına yönelik tasarlanmış olup, toplamda 12 hafta sürmektedir. Eğitim sonunda görev temelli bir yarışma düzenlenmektedir.

Okuyuculara sunulan bu kitap Deneyap Teknoloji Atölyeleri kapsamında ortaokul öğrencilerine eğitim verecek eğitmenler için hazırlanan Yazılım Teknolojileri öğretim programını içermektedir. Ancak kitabın tamamı ortaokul seviyesinde Yazılım Teknolojileri eğitimi vermek isteyen tüm kurum, kuruluş ya da eğitmenlerin kullanımına hizmet etmek için hazırlanmıştır.

Sizlere ve ülkemiz öğrencilerine faydalı olması dileğiyle!

Yazılım Teknolojileri Dersi Öğretim Planı Uygulama Kılavuzu

Bu kılavuzda yazılım teknolojileri dersinin öğretimi hakkında aşağıdaki başlıklara yer verilmektedir:

- Yazılım Teknolojileri dersinin hangi öğretim tasarım zemininde oluşturulduğu,
- Belirlenen öğretim tasarımının aşamalarının ne olduğu ve bu öğretim tasarımının aşamalarında eğitmenin süreçte nasıl rol alacağı,
- Eğitmenin yazılım teknolojileri dersinin öğretim süreçlerinde kullanabileceği öğretim yöntem ve teknikleri,
- Eğitmenin yazılım teknolojileri dersinin süreçlerinde kullanabileceği değerlendirme teknikleri,
- Eğitmenin yazılım teknolojilerinde kullanacağı programların tanıtımı,
- Eğitmenin yazılım teknolojilerinde kullanacağı teknolojik araçların tanıtımı.

Yazılım Teknolojileri Dersi Bilgi Paketi

Dersin Amacı

Bu dersin amacı, öğrencilere temel programlama mantığının öğretilmesi, problem çözümüne yönelik algoritmaların tasarlanması, tasarlanan algoritmaların C++ programlama dili kullanılarak kodlanmasıdır. Bu amaç doğrultusunda hedeflerimiz,

- Programlama temellerinin öğretilmesi,
- Verilen probleme yönelik uygun algoritma tasarımlarının geliştirilmesi,
- Akış diyagramından kodlamaya geçiş yapılması,
- C++ programlama dili kullanılarak problem çözümü,
- Nesne yönelimli programlama mantığının geliştirilmesi,
- C++ programlama dili ile proje geliştirilebilmesidir.

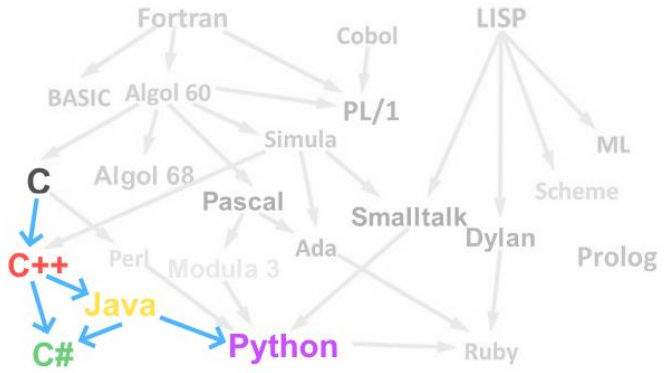
Dersin Çıktıları

Bu dersi alan öğrenciler;

- Gerçek hayat problemlerini algoritmaya dönüştürebilir ve akış diyagramları oluşturabilir.
- C++ programlama dilini kullanarak akış diyagramını koda dönüştürebilir.
- C++ veri tipleri, programlama komutları ve fonksiyonların kullanımını sağlayabilir.
- Nesne yönelimli programlama ile kodlama gerçekleştirebilir.
- İşaretçi ve dosyalama kavramlarını kullanarak proje tasarlayabilir.

Neden C++ Programlama Dili?

C++, 1979 yılında Bjarne Stroustrup tarafından Bell Labs'da geliştirilen nesne yönelimli ve yüksek seviyeli, genel maksatlı programlama dilidir. C++ dili kullanılarak sistem yazılımları, özel yazılımlar, uygulamalar, sürücü yazılımları, kullanıcı taraflı yazılımlar ve gömülü firmware yazılımlar üretilmektedir. C++ dilinin orta seviyeli bir dil olmasından dolayı diğer yüksek seviyeli programlama dillerinden gerekli optimizasyon yapıldığında daha performanslı olduğu söylenebilir. C++ dilini öğrenir, mantığını anlarsak bu dilden etkilenerek oluşmuş programlama dillerini de temel seviyede öğrenmemiz kolay olur. Birçok üniversitede programlamaya giriş dersi olarak C++ eğitimi verilmektedir. Günümüzün en başarılı programcılarının çoğu C veya C ++ ile kod yazmayı öğrenmeye başlamıştır. Resim 1'de verilen programlama dillerine ait aile ağacı grafiğinde özellikle C#, Java ve Python dillerinin atasının C++ olduğunu görebilirsiniz.



Resim 1. *Programlama dillerine ait aile ağacı*

C++ programlama dilinin, farklı birçok uygulamada tercih edilen bir dil olmasını sağlayan iki temel özellik hız ve donanıma yakınlıktır. C++ programlama dili nesnelerin kullanımını sağlayan popüler bir dildir. Programcı için işleri kolaylaştıran yerleşik işlevlerle dolu bir kütüphane sunar. Java ve Python programlamlama dillerinden farklı olarak yorumlayıcı tabanlı değil derleyici tabanlı bir dildir ve bu nedenle bu dillerden nispeten daha hızlıdır. Basit içeriği ile yeni bir programlama dili öğrenmek isteyen programcılara hitap eder. Resim 2'de C++ programlamanın üstün özelliklerini görebilirsiniz.

Resim 2. *C++ programlama dilinin üstün özellikleri*

C++ önemli ve güçlü bir kullanım alanına sahiptir. Bu alanlardan bazılarına örnek verecek olursak;

- Gömülü Sistemler (Robotik Programlama) ve Elektronik Kartlar
- Masaüstü ve Hesaplama Uygulamaları
- Web Tarayıcı Oluşturma, Oyun Programlama, Derleyici Geliştirme
- Yeni Programlama Dili ve Yeni İşletim Sistemi Geliştirme

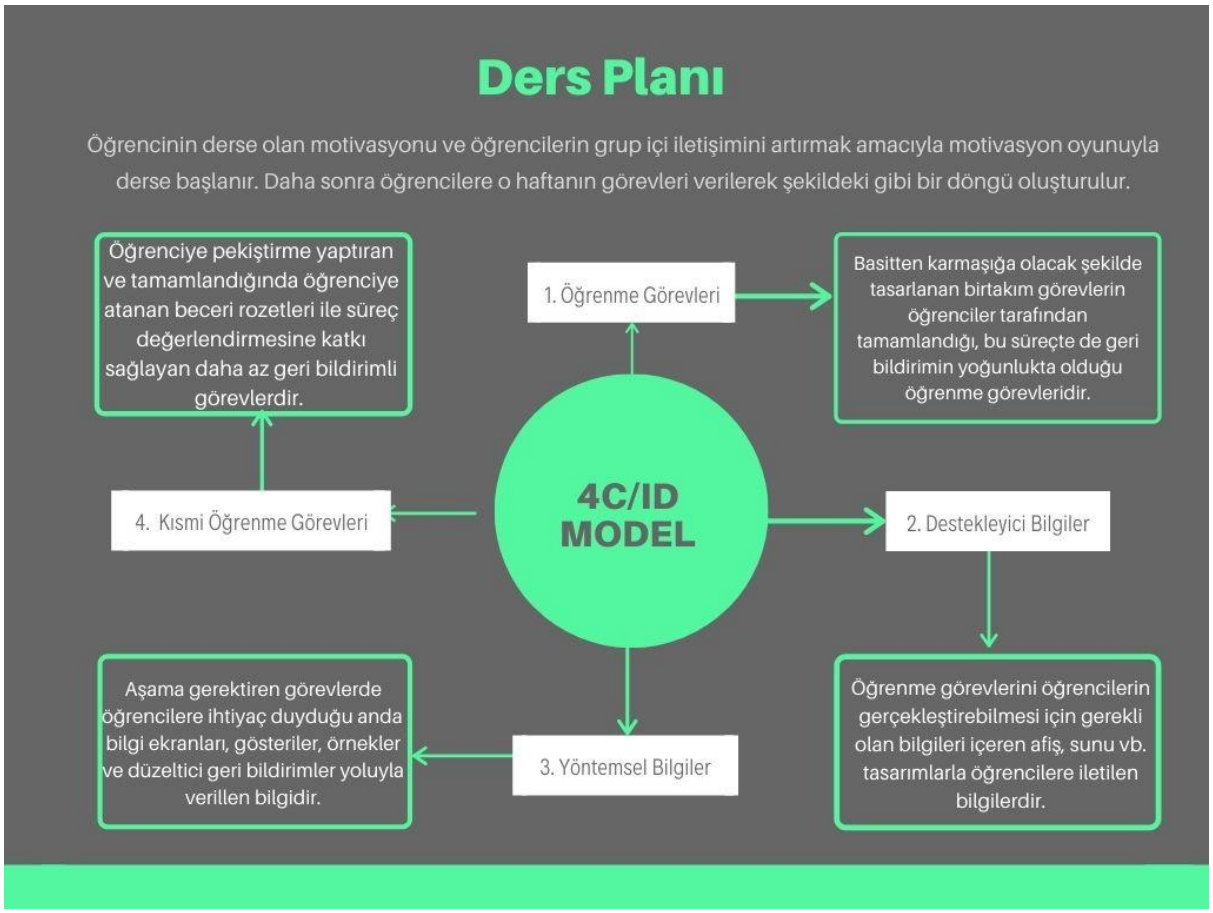
YouTube, Google, Amazon, Twitter, Facebook gibi uygulamaların yapımında C++ programlama dili de kullanılmıştır. Dünya çapında popülerliğini korumaktadır. Yüksek performanslı uygulamalar, oyunlar ve karmaşık araçlar yazmak için ve yazılan uygulamanın direkt olarak donanım ile haberleşmesini için C++ dili kullanmak gereklidir.

Ders Haftalık Planlaması

- Hafta 1: Programlamaya Giriş, Programlama Terimleri, Matematiksel İşlemler, Karşılaştırma İşlemleri, Mantıksal İşlemler, Sayı Sistemleri
- Hafta 2: Algoritma Tasarımı, Operatörler ve Kavramlar, Akış Şemalarına Giriş
- Hafta 3: Örnek Akış Şemaları, Kodlamaya Geçiş, Program Yazmanın Adımları (Hafta İçi Uygulama: C++ Programlama Dilinin Özellikleri, IDE (Derleyici) Kurulumu ve İlk Programlama)
- Hafta 4: C++ Tanımlamalar ve Operatörler, Değişken Tanımlama, Veri Tipleri, Sabitlerin Tanımlanması, Operatörlerin Kullanımı
- Hafta 5: Karar Verme Komutları (if-else, switch), If ifadesi, Switch İfadesi, Mantıksal Operatörler, Operatör Öncelikleri
- Hafta 6: Döngü Komutları (for, while, do-while), İç İçe Döngüler, Rastgele Sayılar
- Hafta 7: Diziler, Karakter Katarları, Örnek Problem Çözümleri
- Hafta 8: Fonksiyonlar, Fonksiyon Sözdizimi, Fonksiyon Çağırma
- Hafta 9: Nesne Yönelimli Programlama, Sınıf Tanımlama, Metod Tanımlama, Nesne Değişkenleri Oluşturma, Yapıcı Yöntemler
- Hafta 10: Nesne Yönelimli Programlama, Nesne Değişkenleri Oluşturma, Yapıcı ve Yıkıcı Metotlar
- Hafta 11: Kütüphane Kullanımı ve Dosyalama İşlemleri
- Hafta 12: Proje Yarışması

Dersin İşleniş Biçimi

Dersin işleniş biçimi Resim 3'te aşağıdaki gibi özetlenmiştir.



Resim 3. *Ders işleniş planı*

Yazılım teknolojileri dersinin öğretim tasarımı Dört Bileşenli Öğretim Tasarımı (4C/ID) modelinden temel alınarak planlanmıştır. Bu tasarıma göre;

Öncelikle her hafta için hazırlanmış olan ders materyal bölümündeki (bu dokümanın ilerleyen bölümlerinde yer almakta) görevler sırasıyla oluşturulan öğrenci gruplarına dağıtılır. Her görev için oluşturulacak öğrenci gruplarının sayısı hazırlanan haftalık ders planında belirtilmiştir.

Eğitmenin verdiği öğrenme görevinin yapılabilmesi için gerekli destekleyici bilgiler (teorik bilgiler) öğrencilere afiş, sunum vb. tasarımlar yoluyla eğitmen rehberinde belirtildiği gibi verilir. Birden fazla aşama gerektiren öğrenme görevlerinde öğrenciler bazı aşamaları geçemediği için görevi tamamlayamayabilir. Bu durumda eğitmen devreye girerek öğrenciye anlık ve hızlı geri bildirimlerle o görevi bitirmesi için işlemsel bilgi (yöntemsel bilgi) vermesi beklenir.

Kısmi öğrenme görevleri ise o haftanın konusunu öğrencinin pratikler yaparak otomatikleşme sağlaması için verilen görevlerdir. Bu aşamada yöntemsel bilgiler gittikçe azaltılır. Eğer öğrenci artık yöntemsel bilgiye gerek duymadan görevleri yerine getirebiliyor ise konuyu özümsediği anlaşılır.

Yazılım Teknolojileri Dersinde Kullanılacak Öğretim Tasarım Modeli

Yazılım teknolojileri dersi kapsamında belirlenen öğrenme hedeflerine erişmek için kullanılacak öğretim tasarımı modeli önem arz etmektedir. Öğretim tasarımı belirlenen hedeflere nasıl ulaşılacağını (hangi öğretim stratejileri ve araçlarının kullanıldığı) ve öğrencinin hedeflere ulaşıp ulaşmadığının kontrolü açısından (değerlendirmenin nasıl yapılabileceği) eğitimcilere yol göstermektedir (Mager, 1984). Öyle ki öğretim tasarımı, eğitim-öğretim ortamlarında uygulanacak her türlü faaliyetin belli bir plana göre uygulanmasıdır (İşman, 2015). Öğretim tasarımının genel amacı, öğrenmeyi daha verimli ve daha kolay hâle getirmektir.

Yazılım teknolojileri dersinin öğretim tasarımı **Dört Bileşenli Öğretim Tasarımı** (4C/ID) modelinden esinlenerek planlanmıştır. Bu modelin seçim nedeni 4C/ID modelinin, karmaşık öğrenme veya mesleki yeterliliklerin öğretilmesi için eğitim programlarının dört bileşen hâlinde tanımlanmasıdır. Burada karmaşık öğrenmeden kasıt, zor anlamında değil; öğrenme bağlamında edinilmiş yeni bilgi, beceri ve tutumların entegrasyonunu ve koordinasyonunu gerektiren bir takım bütüncül görev grubunun başarıyla uygulanmasıdır (Costa ve Miranda, 2019). Bu model, gerçek yaşam durumlarına yönelik öğrenme görevlerinin öğrenmeyi geliştiren unsur olduğunu düşünmektedir (van Merriënboer ve Kester, 2014).

Geleneksel olarak, bu tür derslerin tasarımı “teori” den başlar. Eğitmenler, bu teoriyi öğrencilere “iletmek” için bir dizi ders tasarlar. Buna ek olarak, tüm görevin tek ve küçük yönleriyle pratik yapmak için ödevler tasarlanır. Eğitmenler, bu teoriyi öğrencilere "iletmek" için bir dizi ders ve pekiştirme amaçlı ödevler tasarlar. Örneğin, bu ödevler döngüleri programlama dilinde kullanma veya veri tabanında bilgi oluşturma, okuma, güncelleme ve silme üzerine odaklanabilir. Eğitmenler daha sonra tüm konular ele alınana kadar küçük alıştırmalar ile dersi sürdürür. Bu şekilde geleneksel tasarım modellerinin yaklaşımını takip eder. Teorik bilginin sunulması ile başlayıp belirli uygulama öğelerine bağlanmasını izler. 4C/ID modeli bu yaklaşımı devirir. Profesyonel görevleri belirleyerek başlar, bunları öğrenme görevlerine dönüştürür ve ancak daha sonra hangi “teorinin” öğrencilerin bu öğrenme görevlerini tamamlamalarına yardımcı olması gerektiğini araştırır (Frere Jean vd. 2019).

Yazılım geliştirmek tipik bir karmaşık beceridir. Çünkü programlama dilleri, veri tabanları, geliştirme ortamları vb. hakkında kapsamlı bilgi gerektirir. Ayrıca geliştirilen yazılımı çalıştırmak, temiz ve doğru kodlar yazmak veya bir kullanıcı arayüzü tasarlamak gibi birden fazla beceri gerektirir. Buradan hareketle yazılım teknolojileri dersinde, karmaşık beceriler ve mesleki yeterliliklerin eğitimi üzerine odaklanan Dört Bileşenli Öğretim Tasarımı (4C/ID) modelinden faydalanılabileceğine karar verilmiştir (van Merriënboer, 2016). Bu kapsamda modelin Deneyap Teknoloji Atölyeleri Yazılım Teknolojileri dersinde öğrenme görevleri ile ilgili basamağı bire bir uygulamaya dökülmemiştir. Çünkü bu derste C++ programlama dili verilmektedir. C++ programlama dilinin öğretildiği bu derste, dilin zorluğu ve dikkat gerektirmesi hedef kitle bakımından göz önüne alınmalıdır. Bu nedenle modele ilişkin öğrenme görevleri, belli sayıda parçalı hâlde verilen içeriklerin birleşiminden oluşmaktadır. İçerikte öğrenme görevleri konulara göre kendi içinde karmaşık becerilere hitap eden haftalık etkinlikler şeklindedir. Bu dilin öğretiminde öğrencileri doğrudan karmaşık bütüncül görevlere dahil etmek onların motivasyonunu düşürmeye neden olabilir.

Yazılım Teknolojileri dersi kapsamında 4C/ID modelinde açıklanan karmaşık beceriler, çok sayıda farklı beceriyi aynı anda icra etme olarak düşünülebilir. Öğrencilerin sahip olabileceği

"Biz bunu neden öğreniyoruz?" sorusuna cevap oluşturabilmek ve onların öğrendiklerini bir yazılımcı olarak anlamakta güçlük yaşamalarına engel olmak için modelin basamakları kullanılırken, öğrenme görevleri temel ders konularına ayrılarak haftalık bütünlük içinde verilmeye çalışılmıştır. Örneğin öğrenciler Yazılım Teknolojileri dersinde bir derste problem çözme aşamalarını, diğer derste algoritma, bir başka derste akış şemaları bilgisini öğrenme görevleri şeklinde ayrı ayrı görmektedir. Bu şekilde sunulan öğrenme görevleri birkaç haftalık konu birikiminin ardından küçük bir proje ile sonlanır. Bu noktada bir yazılımcının mesleki hayatında karşılaşabilecekleri durumlara göre, bilgi ve becerilerini birleştirdiği bütüncül uygulamalar söz konusu olacaktır. Örneğin beş hafta boyunca verilen öğrenme görevleri, altıncı hafta tüm öğrenme görevlerini kapsayan karmaşık ancak daha bütüncül ve günlük hayatla ilişkili bir proje görevinde birleşmektedir. Bu anlamda projede kullanılan modelin aşamaları aşağıda açıklanmaktadır.

Dört Bileşenli Öğretim Tasarımı (4C / ID) Bileşenleri ve Modelde Eğitmenin Rolü

4C-ID modeline göre, karmaşık becerileri eğitmek için iyi tasarlanmış ortamlar birbiriyle ilişkili dört bileşenden oluşur:

(1) Öğrenme Görevleri

Öğrenme görevleri gerçek yaşam görevlerine dayanmalı, öğrencinin bu görevleri mantıksal akıl yürütme ve problem çözme becerilerinin bütünleştirmesini gerektirmelidir. Öğrenme görevleri projeler, görevler, vakalar, sorunlar veya diğer türden görevler olabilir. Öğrenme görevleri, öğretim modelinin ve öğrenci öğrenmesinin merkezinde yer alır. Öğrenciler, bir profesyonelin karşılaşabileceği en basit aşamaları içeren görevlerle işe başlar ve daha sonra bir öğrencinin üstesinden gelebilmesi gereken karmaşıklık düzeyindeki görevleri içeren görevlerle bitirir (van Merriënboer, Kirschner ve Kester, 2003). Bu görevler üzerinde çalışırken, eğitmen ve öğretim materyalleri öğrencilerin öğrenme görevlerini yerine getirmelerine yardımcı olmak için gerekli desteği ve rehberliği sağlar (Frere Jean vd. 2019).

Yazılım Teknolojileri dersinde öğrenme görevleri proje ya da problemler şeklinde öğrenciye parça parça iletilmektedir. Öğretim planının bel kemiğini oluşturur ve pratikte karşılaşılan gerçek yaşam durumlarına dayanır. Bu nedenle görevin günlük hayatla ilişkilendirilmesi önemli bir noktadır. Basitten karmaşığa sıralanan görev sınıfları mevcuttur. Bu görevler üzerinde çalışırken, eğitmen ve eğitim materyalleri öğrencilerin görevlerini tamamlamalarına yardımcı olmak için gerekli desteği ve rehberliği sağlar. Haftalık içerikte sunulan öğrenme görevleri gruplama, sıralama, analiz-sentez, tahmin etme gibi tekniklerle öğrencilerin eleştirel düşünmelerini sağlayacak düzeyde kendi içinde karmaşıklaştırılmış parçalı görevlerdir. Bu görev sınıflarının konulara göre gruplaşmasıyla, daha bütüncül ve karmaşık yapıda proje görevlerini oluşturur. Örneğin akış şemaları, karar verme, döngü komutları ve diziler ile ilgili basit ve haftalık öğrenme görevleri proje görevinde gruplaşır. Bu durumda öğrenciler proje görevi öncesi temel yapıyı da edinmiş olarak, tüm bu becerileri kapsayan daha bütüncül ve karmaşık bir görevle karşılaşmış olacaktır. Öğrenciler bu görevler üzerinde çalışırken, eğitmen ve eğitim materyalleri onların görevleri tamamlamasına yardımcı olmak için gerekli desteği ve rehberliği sağlar.

(2) Destekleyici Bilgiler

Destekleyici bilgi, öğrencilere verilen görev veya problemlerin nasıl çözüleceğine yönelik yaklaşımları tanımlar. Destekleyici bilgi genellikle "teori" olarak adlandırılır ve öğrenme görevlerini tamamlamak için gerekli olan zihinsel modeller ve bilişsel stratejiler geliştirmek için bilgi içerir. Destekleyici bilgi, görevin problem çözme, akıl yürütme ve karar verme ile ilgilenen yinelenmeyen yönlerini hedefler. Dersler, çalıştaylar veya çalışma materyalleri şeklinde sunulabilir. Öğrencilerin öğrenme görevlerini yerine, önce ya da çalışırken öğrenim görmeleri için kullanılabilir (Frere Jean vd. 2019). Bu aşamada öğrenciye görevdeki performansına göre bilişsel geri bildirim verilir. Örnek olarak, eğitmen öğrenciye çözümünü bir akranının çözümüyle karşılaştırmasını önerebilir (Costa ve Miranda, 2019). Öğrenme görevinin tamamlanması aşamasında öğrencilere destek olan bilgiler kolay erişilebilir niteliktedir. Bu derste destekleyici bilgi için çeşitli bilgi kartları, tartışma panoları, tekrar ve pekiştirme amacıyla hazırlanan konu ya da kavramları özetleme teknikleri (afiş, kavram haritası, sunu vb.) kullanılmaktadır. Ayrıca eğitmenler öğrencilere görev tamamlama esnasında ihtiyaca dayalı rehberlik etmektedir.

(3) Yöntemsel Bilgi

Öğrenme görevinin rutin kısımlarının nasıl çözüleceğini öğrenciye anlatır ve öğrenciye ihtiyacı olduğu anda anlık olarak sunulur. Öğrenci zaman içinde deneyip kazandıkça yöntemsel bilgi ortadan kalkar. Bu derste eğitmenler öğrencilere görev tamamlama esnasında ihtiyaca dayalı rehberlik etmektedir.

(4) Kısmi Görev Uygulaması

Öğrencilerin görev içinde bir rutin oluşturmak için gerekli becerileri otomatikleşinceye kadar tamamladıkları alıştırma görevleridir. Bu bileşen yalnızca öğrenme görevleri yeterli tekrarı öğrenciye sunmadığı zaman kullanılır. Ancak bu derste kısmi öğrenme görevleri süreçte değerlendirmeyi sağlamak amacıyla da kullanılmıştır. Her kısmi görevin doğru tamamlanmasının ardından öğrenciye görevi tanımlayan bir beceri rozeti verilir. Bu anlamda beceri rozetleri öğrencilere süreçte öğrenilen becerileri pratikleştirme imkânı veren yeni görevler sunmak için kullanılmaktadır.

Kısmi görevlerle bağlantılı bu rozetler ile öğrenciler öğrenme seviyelerine uygun şekilde gruplandırılarak eğitim sonunda projelerde birlikte çalışmaktadır. Bu projeler öğrencilerin edindikleri becerilerin tamamına hitap eden bütüncül görevlerdir. Bu şekilde hem akran öğretimine destek verilmekte hem de 4C/ID öğretim modelinin doğasına uygun bir bütünlük sağlanmaktadır. Ayrıca öğrencilerin kendi öğrenmesini yansıtma işlemi gerçekleştirmesi beklenir.

Eğitmenin Rolü:

4C/ID modelini uyarlandığı Yazılım Teknolojileri dersinde her haftanın içeriği 20-25 dk. arasında değişen öğrenme görevlerinden oluşmaktadır. Bu görevler grup çalışmaları ile bilgi paylaşımı, eğitsel ve motivasyon oyunları ve bilgisayar üzerinde yapılacak öğrenme görevleri şeklindedir. Öğrenme görevleri başında ve aralarında 10 dk. süre ile motivasyon oyunları oynatılmaktadır. Motivasyon oyunlarının her hafta hangi aşamada kaç dakika süre içinde uygulanacağı her haftanın genel özetinde yer alan ders akışı bölümünde belirlenmiştir. Eğitmen motivasyon oyunu saatinde kendi tercih ettiği bir oyunu öğrencileriyle uygulayabilir. Ders

başında ve aralarında oynanacak bu motivasyon oyunlarının öğrencileri monotonluktan çıkarmak, grup etkinlikleri için kendi aralarında kaynaşma sağlamak, dikkatlerini toplamak, motivasyonu artırmak vb. amaçlarla öğrenme görevlerinde öğrenci katılımını destekleyeceği düşünülmüştür.

Her öğrenme görevinin kendine özgü öğrenme materyalleri bulunmaktadır. Bu materyaller, konu anlatımı ve öğrencilerin öğrenme görevlerini tamamlamalarını kolaylaştıracak görsel kartlar, tartışma kartları, çalışma kâğıtları, sunumlar ve oyun uygulamalarını içermektedir. Öğrenme görevlerine ilişkin tüm önemli bilgi ve materyaller eğitmenler için hazırlanan eğitmen rehberi bölümünde toplanmıştır. Eğitmenin hafta başında işlenecek öğrenme görevlerini inceleyerek derse hazırlık yapması ve gerekli materyalleri hazırlayarak öğrenme ortamını düzenlemesi beklenmektedir.

Kitap içeriği hafta hafta bölümlere ayrılmış ve uygulama yönergesi şeklinde hazırlanmıştır. Her haftanın başında ders akışını açıklayan genel özet bölümü yer almaktadır. Bu bölümde:

- Kazanımlar: Haftalık bir dersin 4 saat boyunca öğrencilere aktarılması gereken kazanımlarını içerir.
- Amaç: Haftaya özgü konuların temel amacını içerir.
- Önerilen Ders Akışı: 4 saatlik süre boyunca izlenecek öğrenme adımlarını içerir.

Genel özet bölümünün hemen altında önerilen ders akışı, sırasıyla öğrenme görevinin adına göre açıklanmaktadır. Her görevin adı süre, kazanımlar, eğitim materyalleri, uygulama ve eğitmenin yararlanabileceği konu içeriği olmak üzere alt başlıklardan oluşmaktadır. Eğitmen haftanın konusunu dersten önce eğitmen rehberine uygun şekilde aktarmak için bu adımları çok iyi incelemeli ve hazırlık yapmalıdır.

Model kapsamında öğrenciler öğrenme görevlerinde birtakım uygulamalar gerçekleştirirken eğitmen öğrencilere ihtiyaç duyduğunda ipucu, eleştirel sorular yönelterek geri bildirimlerde bulunmalıdır. Burada eğitmen görev için yeterli süre var ise, görevin yanıtlarını öğrencilere doğrudan vermek yerine onları biraz daha düşünmeye teşvik edecek geri bildirimler iletmelidir. Örneğin bunlar aşağıdaki gibi olabilir:

- *Bu konuya bir de şu açıdan bakmayı deneyin!*
- *Grup arkadaşlarınızla bunu biraz daha tartışmanızı istiyorum. Yeterli süreyi aldığınızda birlikte keşfedebileceğinize eminim.*
- *Cevabının doğru olduğunu söyleyemem ama yaklaştığını söyleyebilirim. Burada .... üzerine biraz daha odaklanmanı istiyorum.*

Eğitmen geri bildirimleri yerine öğrencilerin doğru yanıtları keşfedebilmeleri için akran geri bildirimlerinden de yararlanılabilir. Bunun için derste bilinçli olarak doğru yanıtlara ulaşan öğrencilerin iyi gözlemlenmesi önemlidir. Bu öğrencileri diğerleri ile eşleştirerek birbirlerine görevi açıklamaları istenebilir. Bu açıklamalar yapılırken eğitmen yakından süreci izlemeli ve yanlış öğrenmelerin önüne geçmeli, öğrenme boşluklarını doldurmak için tekrar anlık geri bildirimlerde bulunmalıdır. Bu anlamda eğitmen sınıfta süreci yöneten, izleyen ve öğrencilerle birlikte öğrenmeyi kontrol eden bir göreve sahiptir. Kısaca sürece rehberlik eder. Unutulmamalıdır ki eğitmenin öğrenmeye rehberlik etmesi, bilgiyi doğrudan aktarmasından çok daha zor olacaktır.

Eğitmenin Yazılım Teknolojileri Dersinin Öğretim Süreçlerinde Kullanabileceği Öğretim Metotları ve Teknikleri

Eleştirel Düşünme Teknikleri

Öğrenme görevlerinin öğrenciye sunumunda eleştirel düşünme tekniklerinden yararlanılmaktadır. Bu tekniklerden bazıları örnekleriyle şu şekildedir:

- Sıralama: Algoritma ya da kod satırlarını düzene koyma ya da sıraya dizme.
- Gruplama: Ortak özelliklere sahip nesneleri ya da dizileri gruplama.
- Talimat Verme: İki sayıyı toplama ile ilgili kod yazarı akranına sözde kod yöntemiyle talimat verme.
- Tahmin Etme ve Çıkarımda Bulunma: Ekran çıktısı ya da kodun çözüm getirdiği problemi tahmin etme.
- Analiz ve Sentez: Diziler içinden eleman çıkarma, iki yazıyı birleştirme.

SCAMPER Tekniği

Öğrenme görevlerinin tasarımında kullanılan bir diğer teknik ise SCAMPER yönlendirilmiş beyin fırtınası tekniğidir (Çilci, 2019; İslim, 2011; Yağcı, 2012; Yiğitalp, 2014). Yaratıcı düşünme tekniği (Özyaprak, 2016) olarak da bilinen SCAMPER tekniği ile bir nesne ya da fikri farklı açılardan düşünmeyi sağlayacak sorular yöneltilir. Buna göre her bir harf, temelde farklı soru kalıplarına işaret etmektedir (Özyaprak, 2016). SCAMPER akrostişini oluşturan İngilizce kelimeler ve kullanım örnekleri şu şekildedir:

- S: Substitude (Yer değiştirme): Buradaki dizilerin yerlerine başka ne gelebilir?
- C: Combine (Birleştirme): Hangi kodlar birleştirilirse, problemi çözebiliriz? Ekran çıktısında hangi iki sayıyı bir araya getirip toplanmıştır?
- A: Adapt (Uyarlama): Ortamın sıcaklık değeri 10 derece azalsaydı, kodun ekran çıktısında tahmin edilen hava durumu nasıl olurdu?
- M: Modify, Minify, Magnify (Değiştirme, küçültme, büyütme): Bu kodu daha hızlı çalışır hâle getirmek için nasıl bir değişiklik yapabilirim? Nesne değişkeninin değerlerini daha büyük verirsem, ekran çıktısı nasıl olur?
- P: Put to other uses (Başka amaçlarla kullanma): “Bu değişkeni başka hangi amaçla kullanabilirim?” ya da “bir nesneyi polimorfizm yoluyla başka hangi amaçlarla kullanabilirim?”.
- E: Eliminate (Yok etme, çıkarma): “Bir sınıf tanımlama içerisinden method çıkarsanız, ekran çıktısı nasıl olur?” ya da “diziden eleman çıkarsanız, ne değişir?”
- R: Reverse, Rearrange (Tersine çevirme ya da yeniden düzenleme): Bir dizideki elemanların dizilimini tersine çevirirseniz, ekran çıktısı nasıl değişir?

İşbirlikli Öğrenme

Öğrenciler öğrenme görevlerini tamamlarken küçük gruplar hâlinde çalışmaktadır. Burada bahsi geçen işbirlikli öğrenme sıradan bir grup çalışması değildir. Bunun nedenleri “Her Küçük Grup Çalışması İşbirlikli Öğrenme Değildir” alt başlığı altında ayrıntılı olarak açıklanmaktadır. Grup çalışmalarının işbirlikli öğrenme yapan öğrencilerin hem kendilerini hem de arkadaşlarını kapasitelerinin sonuna kadar geliştirmeye çalışmalarıdır. Bu, tek tek her öğrencinin öğretilenleri tam olarak öğrenmesinden farklı bir durumdur. Grup çalışması sırasında

öğrenciler tek başlarına edinemeyecekleri, ancak başka biriyle etkileşerek kazanacakları öğrenme yaşantılarını deneyimler. Örneğin soru sorma, açıklama yapma, eleştirme, örnek verme gibi.

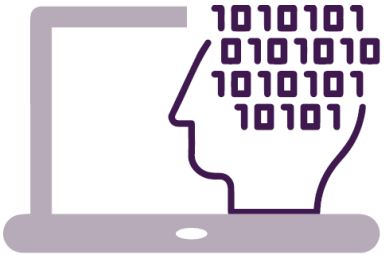
İşbirlikli yöntemin kullanıldığı bu grup çalışmaları bazen istasyon tekniği, bazen de ayrılıp birleşme (jigsaw) tekniği ile tamamlanır. İstasyon tekniği bütün sınıfın her aşamada (her istasyonda) çalışarak bir önceki grubun yaptıklarına katkı sağladığı, bir basamak ileri götürmeyi, yarım kalan işi tamamlamayı öğreten bir yöntemdir. İstasyonlar öğrencilerin eş zamanlı olarak çeşitli öğrenme aktivitelerini gerçekleştirebilecekleri merkezlerdir. Diğer bir işbirlikli teknik olan Jigsaw ile öğrenciler 5-7 kişilik takım oluştururlar. Akademik materyal (ünite) ya da konu gruplardaki öğrenci sayısınca bölümlere (konuya) ayrılır. Her takıma aynı ünite (konu) verilir ve takımlardaki üyelerden ünite parçalarından (konulardan) birini seçmeleri istenir. Her üye kendi konusunu okur. Daha sonra farklı takımlarda aynı konuyu alan üyeler - gruplarından ayrılarak uzmanlık gruplarında bir araya gelirler; konu üzerinde tartışırlar. Sonra kendi takımlarıyla geri birleşerek, takım arkadaşlarını, kendi konularıyla ilgili olarak bilgilendirirler.

Eğitmenin Yazılım Teknolojileri Dersinin Süreçlerinde Kullanabileceği Değerlendirme Teknikleri

Oyun Temelli Düşünme, günlük veya dönemsel yaşam deneyimlerini, yarışma, keşif, senaryolaştırma veya iş birliğine dayalı eylemlere dönüştürmeye dayalı felsefe-düşünme biçimidir. Bu derste de birtakım oyunlaştırma öğelerinden yararlanılmaktadır. Bunlar her ders sonunda öğrenciler tarafından tamamlanan kısmi öğrenme görevlerinden alınacak beceri rozetlerinin kullanımı ile sağlanacaktır. Kısmi öğrenme görevleri, öğrencilerin görev içinde bir rutin oluşturmak için gerekli becerileri otomatikleşinceye kadar tamamladıkları alıştırma görevleridir. Her haftanın son saatinde bu öğrenme görevleri öğrenciye yeterli tekrarı sunmak ve süreç becerilerini değerlendirmek amacıyla süreli olarak kullanılır. Süre kullanılmasının temelinde oyunlaştırma öğeleri ile motivasyonun sağlanması hedeflenmektedir. Ayrıca öğrenciler bu süre içinde kısmi öğrenme görevlerinden kendi seçimiyle istedikleri sayıda görevi tamamlama esnekliğine sahiptir. Diğer bir ifadeyle, öğrenciler haftanın sonunda 5 kısmi görev varsa bunlardan 3'ünü istediği sıralamada yapmak isteyebilir. Eğitmenlerin bu noktada onları tüm kısmi görevleri tamamlama konusunda teşvik etmesi beklenir. Tamamlanan her görevin ardından öğrenci bir beceri rozeti kazanmaktadır. Bu anlamda beceri rozetleri öğrencilere süreçte öğrenilen becerileri pratikleştirme imkânı veren yeni görevler sunmak için kullanılmaktadır. Tüm haftalarda kısmi görevlerin içeriğiyle uyumlu ve göreve tanımlı dört farklı rozet bulunmaktadır. Bunlar:

1. **Analizci Rozet**: Verilen problem için üretilen çözümlerin uygunluğunu kontrol eder ve varsa mantık hataların giderilmesini sağlar.

2. **Kodlayıcı Rozet**: Probleme uygun çözümlerin uygulamaya geçebilmesi için kodlanmasını sağlar.



3. **Tasarlayıcı Rozet**: Verilen probleme uygun çözümün nasıl olabileceği ile ilgili ön hazırlıkları yaparak gerekli algoritma ve akış diyagramlarının hazırlanmasını sağlar.

4. **Denetleyici Rozet**: Verilen problem için üretilen kodlamaların uygunluğunu kontrol eder ve varsa derleyici hatalarının giderilmesini sağlar.

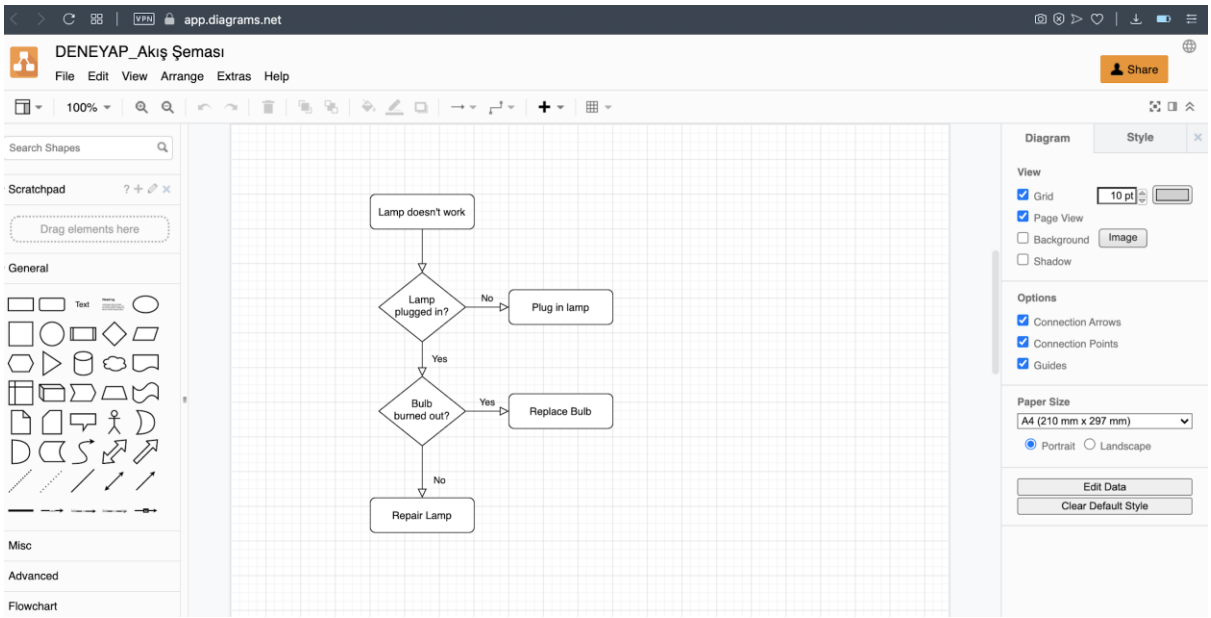
Beceri rozetleri verilen kısmi öğrenme görevleri süreçte değerlendirmeyi sağlamak amacıyla kullanılmıştır. Bu şekilde öğrencilerin etkinliklere katılımını artırmak hedeflenmektedir. Ayrıca öğrenciler tüm hafta boyunca birden fazla analizci ya da kodlayıcı rozeti kazanabilir. Örneğin dönem sonunda edindikleri analizci rozet sayıları öğrencinin portfolyosunda ya da özgeçmişinde yer alacaktır. Bu noktada öğrenciler dönem boyunca edindikleri rozet tür ve sayılarına göre dönem sonunda yer alacakları proje görevlerinde ekip arkadaşlarını bulacaktır. Bu projeler öğrencilerin edindikleri becerilerin tamamına hitap eden bütüncül görevlerdir. Buradaki amaç ise oluşturulacak ekiplerin niteliğini artırmak adına birtakım yeterliliklere gelmiş öğrencilerin bir araya gelmesini sağlamak, ekip üyelerinin birbirlerini beceri anlamında tamamlamalarına imkân vermektir. Bu şekilde hem akran öğretimine destek verilmekte hem de 4C/ID öğretim modelinin doğasına uygun bir bütünlük sağlanmaktadır.

Eğitmenin Yazılım Teknolojilerinde Kullanacağı Programların Tanıtımı

Code::Blocks: Öğrencilerin, Yazılım Teknolojileri dersi kapsamında öğrenecekleri C++ programlama dilindeki programlarını yazmalarını ve derlemelerini kolay bir şekilde gerçekleştirebilmesi için Entegre Geliştirme Ortamı (Integrated Development Environment-

IDE) ile derleyici özelliği olan ve her platform tarafından desteklenen Code::Blocks geliştirme ortamı tercih edilmiştir. Code::Blocks, geliştiriciler için işlevsel araçlarla tamamen yapılandırılabilir ve genişletilebilir açık kaynaklı ve ücretsiz bir IDE çözümüdür. Açık kaynaklı tasarımı sayesinde işlevlerinin büyük kısmı eklentilerle genişletilebilmektedir. Var olan gelişmiş araçlar sayesinde çok başarılı bir hata yakalama çerçevesi sağlamaktadır. Yazılımın derleyici eklentisi, yazılımcıların çok sayıda görevi birleştirmesini kolaylaştıran çalışma alanları sunmaktadır. Yazılım, platformlar arası bir çözümdür ve Mac, Windows ve Linux dahil olmak üzere farklı işletim sistemlerinde çalışır. C++ ile yazılmıştır ve çalışması için kısıtlayıcı kütüphaneler veya çevrilmiş bir dil gerektirmez. Code::Blocks kurulumu ders içeriğinde öğrencilere verilecek materyaller arasında mevcuttur.

Diyagram Çizim Uygulaması: Dersin önemli teknoloji ortamlarından biri ücretsiz çevrimiçi diyagram çizim uygulaması olan draw.io'dur. Bu uygulama Moodle sistemi içinde entegre şekilde kullanılmakta olup, öğrenciler tarafından ayrı bir kullanıcı kaydı ya da kurulum gerektirmemektedir. Öğrenciler uygulamaya farklı bir link üzerinden erişim sağlamaksızın, Moodle üzerinden aktif şekilde kullanabilmektedir. Bu uygulama ile öğrenciler Yazılım Teknolojileri dersindeki projeler için algoritmaların akış şemalarını bu dijital ortamda kolaylıkla çizebilecektir. Resim 4'te Draw.io uygulamasının arayüzü verilmektedir.



Resim 4. *Draw.io uygulamasının arayüzü*

Öğrenci çizimleri Google Drive, Dropbox, Onedrive, Github gibi dijital ortamlarla mevcut cihaz ya da bilgisayarda kaydedilebilir. Bu şekilde eğitmenin öğrenci çizimlerini takip etmesi daha kolay hâle gelecektir. Yazılım Teknolojileri dersinde öğrenci çizimlerinin kayıt ortamları için çoğunlukla Github kullanılmaktadır.

Eğitmenin Yazılım Teknolojilerinde Kullanacağı Diğer Teknolojik Araçların Tanıtımı

Öğrenme Yönetim Sistemi (ÖYS): Derste aktif şekilde kullanılacak olan temel ortamlardan biri öğrencilerin ilgili derse kullanıcı kaydı yapacakları bir Öğrenme Yönetim Sistemidir. Deneyap Teknoloji Atölyeleri tarafından Moodle açık kaynaklı popüler bir öğrenim yönetim sistemi kullanılacaktır. Yazılım Teknolojileri dersleri öğrenme görevleri şeklinde öğrenciler tarafından tamamlanan ya da üretilen ürünlerle ilerlemektedir. Bu nedenle öğrenme görevleri ÖYS ortamında modüler bir yapıda hafta hafta sunulmaktadır.

Yazılım Teknolojileri dersi için Moodle aşağıdaki amaçlara hizmet etmektedir:

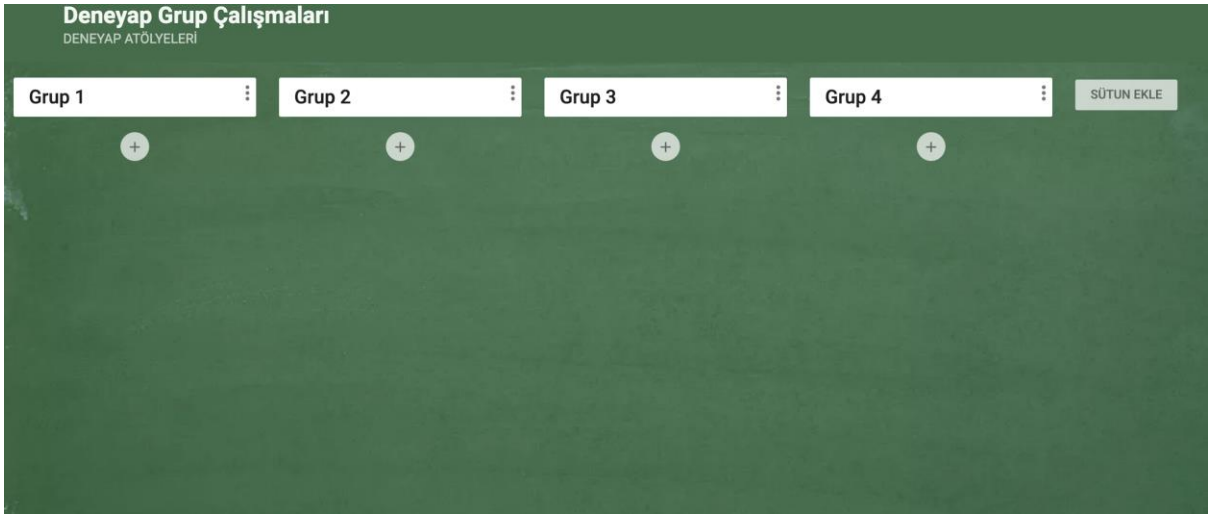
- Dersle ilgili genel duyuruların iletilmesi
- Öğrencilere sunulacak öğrenme materyallerinin paylaşılması
- Öğrenme görevi sırasında kullanılacak diğer teknolojik araçların iletilmesi
- Kısmi öğrenme görevleri gibi bireysel görevlilerin teslimi
- Grup olarak oluşturulacak ürün ya da tamamlanan görevlerin teslimi
- Eğitmenin öğrenci ürünlerini takip etmesi ve geri bildirim sağlaması
- Öğrencilerin kazandığı beceri rozetlerinin iletilmesi, kaydı ve raporlanması
- Öğrenci e-portfolyosunun raporlanması
- Öğrenci-eğitmen, öğrenci-öğrenci ve eğitmen-eğitmen etkileşiminin atölye dışında da sürdürülmesi
- Gerektiğinde BigBlueButton aracılığıyla senkron (canlı) ders ortamının oluşturulması (Oluşturulan canlı ders çalışma odaları destekli olup grup çalışmalarına imkân vermektedir.)
- Eğitmen eğitimleri sürecinde gerekli kaynakların, sunumların ve dokümanların eğitimcilerle paylaşılabilmesi
- Eğitimcilerin fikir alışverişi yapabilmesi için tartışma ortamlarının kurulabilmesi

GitHub: Derste aktif şekilde kullanılacak olan arşivleme profolyo oluşturma ortamlarından biri Github'tır. GitHub açık kaynaklı projeler tarafından tercih edilen ve yazılım geliştirme projeleri için kullanılan web tabanlı popüler bir depolama servisidir. GitHub ile dünya çapında herkes tarafından görüntülenebilen bir projenize, farklı ekip üyeleri ekleyerek takım çalışmaları düzenlenebilir. Ayrıca GitHub üzerinde paylaşılan kodlar ile kişisel gelişim sağlanabilir. Bu anlamda öğrencilerin GitHub ortamını kullanma deneyimini Yazılım Teknolojileri dersinde edinmesi önemli bir beceridir.

Yazılım Teknolojileri dersi için GitHub aşağıdaki amaçlara hizmet etmektedir:

- Her öğrencinin GitHub kullanıcı hesabı alması
- GitHub Branches ile küçük gruplar hâlinde küçük projeler geliştirilmesi
- GitHub Repository ile öğrencilerin kendi depolarını oluşturarak portfolio hazırlanması
- Öğrencilerin takıldıkları noktalarda yardımlaşması için GitHub görev yönetimi (Issues) ile düzenlemeler önerilmesi, yapılacaklar listesinin oluşturulması ve görev atamalarının yapılması
- Tartışma başlıkları açılması

Dijital Pano: Derste aktif şekilde kullanılacak olan temel ortamlardan bir diğeri dijital pano uygulamalarından kullanımı kolay ve popüler olan Padlet'tir. Bu uygulama Türkçe destekli pek çok özelliği ile 3 panoya kadar ücretsiz olan bir ortamdır. Bu uygulama boş bir duvarı içeriklerle doldurma imkânı veren dijital bir panodur. Öğrenciler giriş yapmaksızın padlet ortamında ücretsiz şekilde görsel, video ya da yazı ekleyebileceği bir panoyu birlikte doldurabilmektedir. Ayrıca padlet ortamının raf stili grup çalışmalarını desteklemektedir. Raf stili ile dijital pano üzerinde sütun şeklinde her grubun kendine ait bir alanı bulunur. Gruplar kendine ait sütun altında yer alan + işaretine tıklayarak ürünlerini paylaşabilir. Yazılım Teknolojileri dersinde bu stilden oldukça fazla yararlanılmaktadır. Resim 5'te örnek bir padlet raf stilinin görselini inceleyebilirsiniz.



Resim 5. *Padlet ortamı raf stili örneği*

Yazılım Teknolojileri dersinde padlet ortamı aşağıdaki amaçlara hizmet etmektedir:

- Gruplar ya da öğrenciler için ortak çalışma alanı kurma
- Tartışma ya da beyin fırtınası oluşturma
- Grup ürünlerini paylaşma
- Akranların yaptıklarını inceleme
- Atölye içi ve dışı işbirlikli çalışmayı destekleme
- Oluşan dijital panoyu çıktı hâlinde öğrencilerle paylaşma

Yazılım Teknolojileri dersinde padlet kullanımının avantajları aşağıda verilmektedir.

- Ders aktivitelerinde grup ya da bireysel öğrenci paylaşımlarının kaydını tutma
- Tüm öğrencilere birbirlerinin paylaşımlarını anlık olarak inceleme fırsatı sunma
- Akran öğretimini destekleme
- Ders kaynak ya da materyallerini dijitalleştirme, eğitim maliyetini düşürme
- Öğrencinin multimedya kaynaklarına ders sürecinde erişimini artırma
- Canlı ders yürütürken grup çalışmalarının ürünlerini dijital pano aracılığıyla takip etme

Kaynakça

Costa, J. M., & Miranda, G. L. (2019). Using Alice Software with 4C-ID Model: Effects in Programming Knowledge and Logical Reasoning. *Informatics in Education,* 18(1), 1-15.

Çilci, N., & Aydın, İ. (2019). *Scamper (Yönlendirilmiş Beyin Fırtınası) Tekniğinin 5 ve 6. Sınıf Öğrencilerinin Yaratıcı Yazıları Üzerindeki Etkisi* (Master's thesis). Ordu Üniversitesi, Ordu.

Demirel, Ö.(2005). *Eğitimde Program Geliştirme.* Ankara: Pegem A Yayınları.

Fox, J. (2004). *Rotate, differentiate, and motivate: "how a blend of learning stations and multiple intelligences theory can boost motivation and enhance learning in the middle school classroom* (Unpublished master's thesis). USA, Virginia: College of William & Mary.

Frerejean, J., van Merriënboer, J. J., Kirschner, P. A., Roex, A., Aertgeerts, B., & Marcellis, M. (2019). Designing instruction for complex learning: 4C/ID in higher education. *European Journal of Education, 54*(4), 513-524.

İslim, Ö. F. (2011). Scamper (Yönlendirilmiş Beyin Fırtınası Tekniği). *Uluslararası Bilgisayar ve Öğretim Teknolojileri Sempozyumu,* 22-24.

İşman, A. (2015). *Öğretim Teknolojileri ve Materyal Tasarımı.* Ankara: Pegem Akademi.

Mager, R.F. (1984). Measuring Instructional Results. Belmont, CA: David S. Lake Publishers.

Özyaprak, M. (2016). Yaratıcı Düşünme Eğitimi: SCAMPER Örneği. *Journal of Gifted Education and Creativity, 3 (1), 67-81.*

https://dergipark.org.tr/tr/pub/jgedc/issue/38681/449375

Senemoğlu, N. (2007). *Gelişim öğrenme ve öğretim. (Düzenlenmiş Yeni Baskı).* Ankara: Gönül Yayıncılık.

Van Merriënboer, J. J., & Kester, L. (2014). The four-component instructional design model: Multimedia principles in environments for complex learning. Maastricht University.

Van Merriënboer, J. (2016). *How people learn.* The Wiley handbook of learning technology, 15-34.

Van Merriënboer, J. J., Kirschner, P. A., & Kester, L. (2003). Taking the load off a learner's mind: Instructional design for complex learning. *Educational psychologist, 38*(1), 5-13.

Yağcı, E. (2012). Yönlendirilmiş beyin fırtınası Tekniği: Scamper konusunda veli görüşleri üzerine bir çalışma. Hacettepe Üniversitesi Eğitim Fakültesi Dergisi, 2012(43), 485-494.

Yiğitalp, N. (2014). *Yönlendirilmiş beyin fırtınası (Scamper) tekniğine dayalı eğitimin beş yaş çocuklarının problem çözme becerilerine etkisinin incelenmesi* (Yüksek Lisans Tezi). Hacettepe Üniversitesi, Ankara.

Hafta 1. Programlamaya Giriş

Kazanımlar:

K1. Temel programlama terimlerini açıklar.

K2. Bilgisayarın temel birimlerini açıklar.

K3. Veri saklama birimlerini ayırt eder.

K4. Problem çözme sürecinde aritmetik, mantıksal ve karşılaştırmalı işleçleri kullanır.

K5. Değişkenlere farklı türlerde değer ataması yapar.

K6. İkili ve onlu sayı sistemlerini kullanır.

Haftanın Amacı:

Bu haftanın amacı programlama dünyasına giriş yapılarak, öğrencilerin temel düzeyde matematiksel, karşılaştırma ve mantıksal işleçleri (operatörleri) kullanmalarını sağlamaktır.

Önerilen Ders Akışı:

A. Tanışma: Çember Oyunu (15 dk.)

B. Öğrenme Görevleri

B1. Nasıl Anlatırsın? (20 dk.)

B2. Bilgisayarın Çalışması Neye Benzer? (20 dk.)

- ❑ Ders Arası (5 dk.)
- ❑ Motivasyon Oyunu (10 dk.)

B3. Bilgisayar Verileri Nasıl Saklar? (20 dk.)

B4. Beni Bul ve Değiştir! (20 dk.)

- ❑ Ders Arası (10 dk.)

B5. Farklı Türleri Tanıyalım! (20 dk.)

B6. Sayı Sistemlerini Keşfet! (25 dk.)

- ❑ Ders Arası (5 dk.)
- ❑ Motivasyon Oyunu (10 dk.)

C. Kısmi Öğrenme Görevleri (60 dk.)

A. Tanışma: Çember Oyunu

Süre: 15 dk.

Uygulama: Eğitmen öğrencileri 1 ve 2 olarak gruplar. 1 numaralı öğrenciler dış çember, 2 numaralılar iç çember olarak adlandırılır ve sınıfta iç içe iki çember şeklinde oturmaları sağlanır. Öğrenciler birbirlerine bakacak şekilde oturmaktadır. Eğitmen, öğrencilerden birtakım soruları birbirlerine sorup, bununla ilgili sohbet etmelerini ister. İlk olarak 1 numaralı dış çember öğrencileri, bir dakika süre ile 2 numaralı iç çembere cevap verecektir. Daha sonra 2 numaralılar soru soracaktır. Bir soruya iki taraf da cevap verdikten sonra yeni soruya geçilir. Her yeni soruya geçişte iç çember saat yönünde döndürülerek eşlerin yer değiştirmesi sağlanır. Eşlerin birbirlerine sorabilecekleri örnek sorulardan bazıları aşağıdadır. Bu sorular eğitmen tarafından değiştirilebilir, azaltılabilir ya da artırılabilir.

- Benim için mükemmel bir gün planı oluştursaydınız içinde neler olurdu?
- Bir robot olsaydın, ne işe yarardın?
- On yıl sonra bugün nerede olacağını düşünüyorsun?
- En komik tanıdığın kişi kim ve neden? Seni nasıl güldürüyor?

B. Öğrenme Görevleri

B1. Nasıl Anlatırsın?

Süre: 20 dk.

Kazanımlar: K1. Temel programlama terimlerini açıklar.

Materyaller: Bilgisayar ya da akıllı tahta

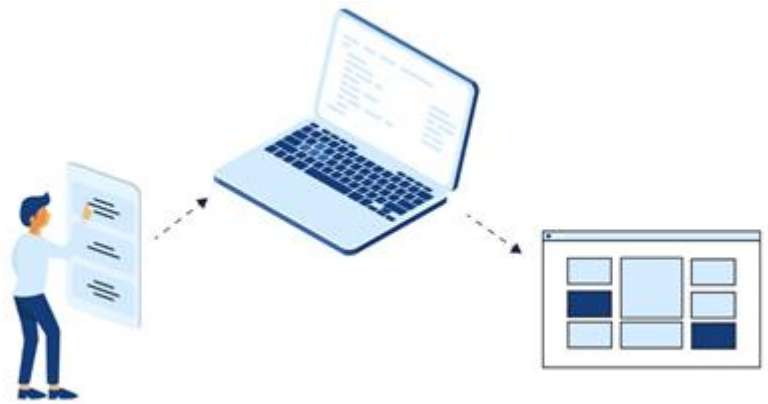
Uygulama: Öğrenciler dörderli gruplara ayrılır. Grupların belirlenmesi için öğrenciler arasındaki benzerliklerden yararlanılır. Örneğin aynı ayda doğanlar, adının baş harfi aynı olanlar vb. özellikler kullanılabilir. Her bir öğrenciye birer parça kâğıt verilir. Her öğrenci kâğıdına onun önemli bir özelliğini ya da o anki ruh hâlini yansıtan bir sıfat ile ismini yazar. Örneğin "Uykulu Özlem" ya da "Çalışkan Ahmet" gibi. Sınıfta oluşan gruplara "programlama" ve "programlama dili" kavramı verilir. Gruptaki her öğrenciden aldıkları kavram ile ilgili akıllarına gelen ilk kelimeyi kâğıtlarına yazmaları istenir. Daha sonra kâğıdı grup içindeki arkadaşları ile değiş tokuş eder ve akıllarına gelen ikinci bir kelime yazarlar. Bu şekilde öğrencinin kendi isminin olduğu kâğıt ona gelene kadar grup içinde yazma devam eder ve her bir gruptaki öğrenci sayısı kadar kavramlarla ilgili özellikler yer alacaktır. Bu işlemden sonra grup içindeki herkes kâğıdında tekrar eden ya da ortak olan kelimeleri grup kâğıdına yazar. Kâğıtların dönme süresi için eğitmen bir uyaran kullanabilir. Değiş ya da çan sesi gibi. Etkinlik bittikten sonra, Programlama ve kavramlarına ilişkin özellikler için tahta ikiye ayrılır. Öğrencilerden grup kâğıtları toplanır ve tahtada bu kelimeler sesli bir şekilde okunarak öğrencilerle tartışılır. Önemli noktalar tahtada yazılarak, programlama ve programlama dili kavramları beyin fırtınası yoluyla özetlenir. Sonuç olarak öğrencilerin aşağıdaki içeriğe ulaşmaları sağlanır.

Konu içeriği:

Programlama	Programlama Dili
Algoritma, Akış diyagramı	Bilgisayar ile konuşma
Yazılım, Kodlama	Bilgisayarla iletişim dili
Talimat verme	Kod dizisi
Bilgisayar kontrolü	Sözdizimi
Bilgisayar görevi	C, C++, Python, Java

Bu bölümdeki amacımız, programlamayla ilgili temel kavramların öğrencilere aktarılmasını sağlamaktır. Programlamanın tanımını, "çeşitli görevleri ya da işlemleri gerçekleştirmek için bilgisayara talimat verme" şeklinde yapabiliriz. Bilgisayarlar güçlü ve hızlı sistemler olmalarına rağmen insanlar gibi akıllı varlıklar olmadıkları için ne yapacaklarını bildiren bazı talimatlara ihtiyaç duyarlar. İşte programlama bu talimatları yazma işlemidir. Bunu yapmak için de bir programlama dili kullanılır.

Tıpkı biz insanların birkaç dili (İngilizce, Almanca, Çince, vb.) anlayabilmesi gibi, bilgisayarlarda da durum böyledir. Bilgisayarlar, programlama dili olarak adlandırılan belirli bir sözdizimi biçimde yazılan talimatları anlar. Görevler ise "iki sayıyı toplama", "sayının karesini alma" gibi basit görevler olabileceği gibi bir dizi çoklu talimat içeren karmaşık görevler de olabilir. Özet olarak, programlama bilgisayarlara belirli bir görevi yerine getirmelerini söylemenin bir yoludur.



Resim 6. *Programlama süreci*

Örneğin İngilizce en popüler ve tanınmış insan dillerinden biridir. İngilizcenin, doğru bir şekilde yazılması için kendi içerisinde uyulması gereken dilbilgisi kuralları seti vardır. Tıpkı insan dilleri gibi, programlama dilleri de sözdizimi adı verilen dilbilgisini takip eder. Programlama dilleri, belirli bir sözdizimi biçiminde yazılır ve bilgisayarın anlayabileceği okunabilir bir formata sahiptir. Hazırlanan programlar daha sonra bilgisayar tarafından yürütülerek istenilen görev ya da işlemlerin gerçekleştirilmesi sağlanır.

Yukarıda belirtildiği üzere bilgisayarlar, programlama dili olarak adlandırılan belirli bir sözdizimi biçiminde yazılan talimatları anlar. Programcının istediği bir görevin bilgisayar

tarafından anlaşılması ve yürütülmesi için ifade etmesini sağlar. Popüler programlama dillerinden bazıları C, C++, Python, Java olarak verilebilir. Bu ders kapsamında C++ programlama dili kullanılarak çalışmalar gerçekleştirilecektir. Bu dile ait bilgiler ilerleyen haftalarda detaylı olarak verilecektir.

B2. Bilgisayarın Çalışması Neye Benzer?

Süre: 20 dk.

Kazanımlar: K2. Bilgisayarın temel birimlerini açıklar.

Materyaller:

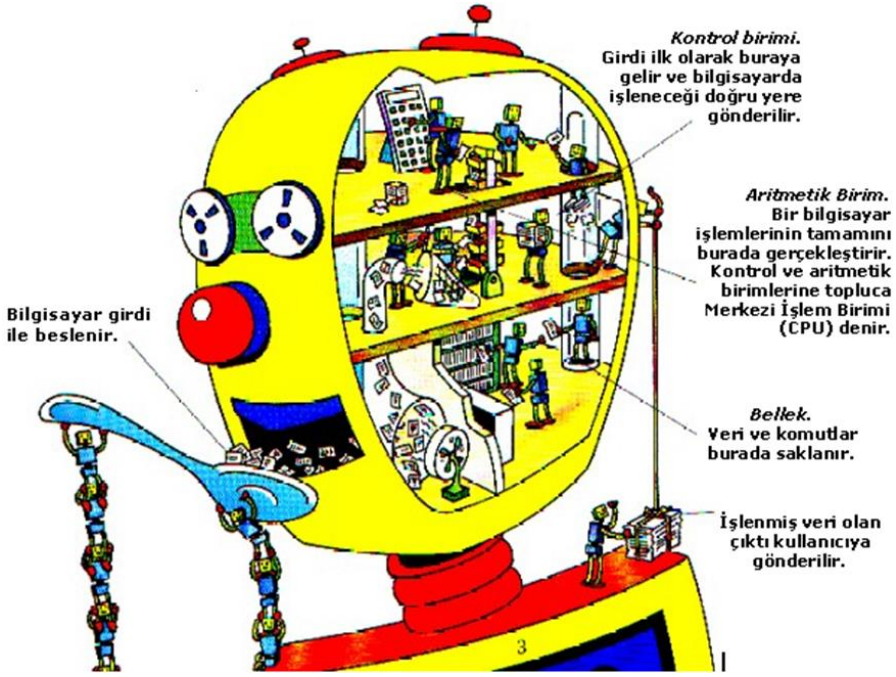
O1. B2. 1. Eşleştirme Kartları

O1.B2.2. Bilgisayarın Temel Mimarisi

Hazırlık: Eşleştirme kartlarından ders öncesi dört adet çıktı alınır. Dört grup için bu kartlar kesikli çizgilerden kesilerek karıştırılır.

Uygulama: Bu etkinlikte bilgisayarın temel bileşenlerini insan organları ile eşleştirme oyunu uygulanacaktır. Öğrencilerle beşerli gruplar hâlinde çalışılır. Eğitmen materyalin çıktısını (4 çıktı olacak şekilde) her grup için ayrı ayrı alır. Çıktıyı her gruba karışık olacak şekilde dağıtmak üzere kesikli çizgilerden keserek eşleştirme kartlarını hazırlar. Eğitmen, öğrencilerden bilgisayarın temel birimlerinin olduğu görsel kartları, insan organlarının bulunduğu görsel kartlar ile ilişkilendirmelerini ister. Bu kartlar gruplara karışık şekilde dağıtılır. Eşleştirilen görsellerin aralarındaki ilişkiyi öğrencilerin kendi arasında tartışmalarını ister. Eğitmen son olarak sınıfta grupların kartlarının kontrolünü yapar ve beyin fırtınası ile konuyu ikinci materyali ekranda yansıtarak aşağıdaki gibi özetler.

Konu içeriği: Bu bölümde öğrencilere temel bilgisayar mimarisinin öğretilmesi amaçlanmaktadır. Bilgisayarlar birkaç bileşenden oluştuğu bilinmektedir. Bu bileşenlerden olan klavye, fare veya monitörü tanımlamak çok kolaydır. Bunların dışında CPU (merkezî işlem birimi) ve bellek gibi daha kompleks bileşenler vardır. CPU, bilgisayarın beyni olarak tüm kararların alındığı, veri işleyen ve yazılım komutlarını gerçekleştiren bölümdür. Genellikle RAM (rastgele erişimli bellek) olarak adlandırılan bellek ise verileri ve talimatları saklamak için kullanılan ve herhangi bir zamanda okunabilen veya değiştirilebilen bir tür veri deposudur.



Resim 7. *Bilgisayarın Temel Birimleri ve İnsan Vücudu*

Kontrol ünitesi, işlemcinin çalışmasını yöneten bir bilgisayarın CPU biriminin bileşenidir. Bilgisayarın bellek, aritmetik ve mantık ile giriş ve çıkış birimlerinin bir programdan alınan talimatlara nasıl yanıt vereceğini bilmesini sağlar. Bir kontrol ünitesi, kontrol sinyallerine dönüştürdüğü ve daha sonra merkezî işlemciye gönderilen giriş bilgilerini alarak çalışır. Bilgisayarın işlemcisi daha sonra bağlı donanıma hangi işlemlerin gerçekleştirileceğini söyler. Bu süreçte verilerin birimler arasında iletilmesi veri akışı olarak tanımlanır. Bilgisayar sisteminin sağlıklı iletişimi için veri akışı çok önemlidir.

Aritmetik/mantık ünitesi, karmaşık bir dijital devredir ve veriler üzerinde aritmetik ve mantık işlemleri gerçekleştirir. İkili sayılar üzerinde hem bit düzeyinde hem de matematiksel işlemler gerçekleştirir ve işlemcide hesaplamaları gerçekleştiren son bileşendir. Kendisine giriş verileri için hangi işlemleri gerçekleştireceğini söyleyen işlenenleri ve kodu kullanır. Bilgiler aritmetik/mantık ünitesi tarafından işlendikten sonra bilgisayarın belleğine gönderilir.

Anakart üzerindeki bileşenlerin birbirlerine bilgi iletmek için kullandıkları yola veri yolu adı verilir. Girdi bir veri yolu üzerinden CPU'ya gelirken, çıktı CPU'dan veri yolu üzerinden çıkar. Veriler, klavye girişi, dosya içeriği, web sunucusu istekleri veya bilgisayar tarafından kullanılan herhangi bir bilgi parçası gibi birçok şeyi ifade edebilir. Talimatlar ise bu iki sayıyı ekleyin, bu verileri buraya taşıyın, daha sonra bu talimata atlayın gibi CPU'ya bir sonraki adımda ne yapacağını belirten özel bir veri türüdür.

B3. Bilgisayar Verileri Nasıl Saklar?

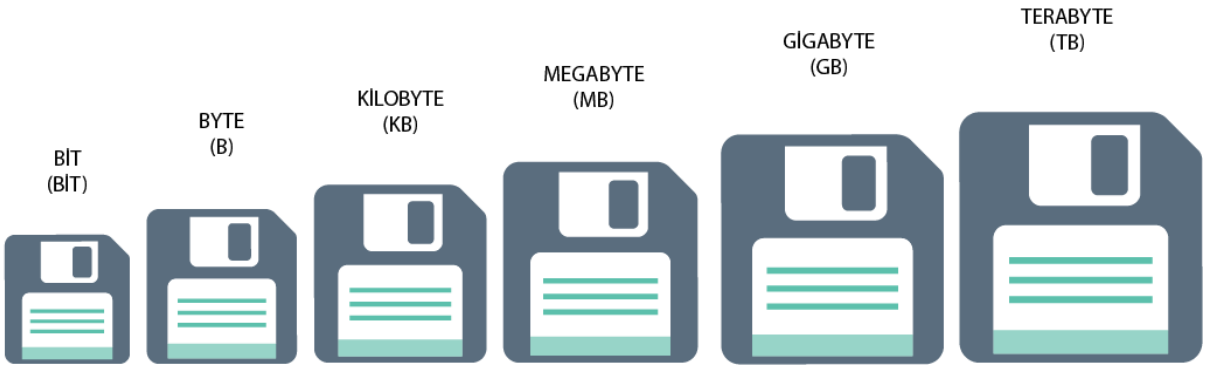
Süre: 20 dk.

Kazanımlar: K3. Veri saklama birimlerini ayırt eder.

Materyaller: O1. B3. 1. Sıralama kartları

Hazırlık: Sıralama Kartları, ön yüz ile arka yüzü iki tarafı eşleşecek şekilde arkalı önlü çıkartılmalıdır.

Uygulama: Eğitmen, öğrencilere bir soru yöneltilir. "Nasıl olur da sayılarla çalışan bilgisayar benim resimlerimi depolayabilir?" Eğitmen bu soruyu öğrencilerine sorarak aralarında tartışma yapmalarını sağlar. Beş dakikalık bir tartışmanın ardından öğrenciler 5'erli kişiler hâlinde 4 gruba ayrılarak onlara eğitmen tarafından sunulan karışık sıralama kartları verilir.

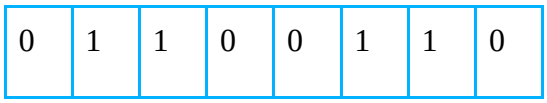


Resim 8. *Sıralama çizgisi*

Veri saklama birimleri ile ilgili kartların üzerine her bir hafıza birimi (bit, byte, mb, gb vb.) için hem görsel hem de açıklama olacak şekilde arkalı önlü kartlar oluşturulur. Öğrenciler 5'erli gruplara ayrılarak, her bir grup üyesinin oluşturulacak kartlardan birini seçmesi istenir. Öğrenciler seçtikleri kartı grup arkadaşlarına göstermeyecektir. Eğitmen iki grup için farklı noktalarda iki sıralama çizgisi oluşturur. Bu çizgi ikiye ayrılan tahta üzerinde ya da zeminde ya da grup masalarında renkli bant yardımıyla oluşturulabilir. Öğrenciler ellerindeki kartların sıralama çizgisinde hangi konuma geleceği konusunda tahmin yürüterek yerleştirecektir. Çizgi üzerine yerleştirilen tüm kartların arka yüzü (görsele ilişkin açıklama bulunan yüz) kapalı konumdadır. Burada amaç grubun elindeki kartların küçükten büyüğe doğru sıralanmış bir çizgi şeklinde olmasıdır. Eğitmen öğrencilerden hazır olduklarında kartları açmalarını ve sıralamayı kontrol etmelerini ister. Eğitmen bu şekilde sıralama çizgisindeki yanlışların nasıl düzeltileceğini öğrencilere keşfettirmeye çalışır. Eğitmen eksik olan noktalara aşağıdaki bilgileri ekleyerek konuyu özetler.

Konu içeriği:

Bilgisayar sistemlerindeki bütün bilgiler ikili sistemde “0” ve “1” ile temsil edilerek saklanır. İkili sistemdeki her bir basamağa bit denir. Bit, nicelik ifade edebilmek için yeterli bir birim olmadığı için en yaygın dijital veri depolama birimi olarak 8 bitlik bayt kullanılır.



8 bit = 1 bayt

Bilgisayar sisteminde her bir karakter 8 bit’ten oluşur. Örneğin: A karakteri bilgisayar sisteminde “00100001” ikili sayısı ile ifade edilir. İşte bu sayının her basamağına 1 bit denir. Depolama bayt cinsinden ifade edildiği için tüm büyük birimlerin kısaltılmış adları kullanılır. Bilgisayar verileri metrik sistemde olduğu gibi bayt (B), kilobayt (KB), megabayt (MB), gigabayt (GB), terabayt (TB), ve petabayt (PB) olarak ifade edilir ve 1 KB 1024 bayttır.

Tablo 1. *Birimler*

Birim	Kısaltma	Kapasite
Bit	b	1 ya da 0
Bayt	B	8 bit
Kilobayt	KB	1024 bayt
Megabayt	MB	1024 kilobayt
Gigabayt	GB	1024 megabayt
Terabayt	TB	1024 gigabayt
Petabayt	PB	1024 terabayt

Tıpkı alfabeyi, konuşulan dilleri ve sayıları anlamak gibi dijital depolama birimlerinde de bilgili olmak önemlidir. Yeni dijital dünya bu yeni dijital depolama sistemlerini öğrenmeyi önemli hâle getirdi. Bu tür bir dil ve ölçümün anlaşılmasının faydaları sürekli artmaktadır. Bu dil sisteminde ne kadar akıcı olursak, dijital dünyayı o kadar hızlı anlarız. Zaman içinde veri hacmi ile daha fazla depolama ihtiyacı arttıkça, kaçınılmaz olarak gerekli yeni kelimeleri de geliştireceğiz.

Örnekler:	
10 MB	= 10240 KB
2048 MB	= 2 GB
3 GB	= 3072 MB
5 MB	= 5120 KB
4096 MB	= 4 GB
2048 PB	= 2147483648 GB
10 TB	= 10485760 MB

B4. Beni Bul ve Değiştir!

Süre: 20 dk.

Kazanımlar: K4. Problem çözme sürecinde aritmetik, mantıksal ve karşılaştırmalı işleçleri kullanır.

Materyaller: O1. B4. 1. Temel işlem tablosu

O1. B4. 2. İşleç bilgi kartları

O1. B4. 3. Örnek olay sunum kartı

Hazırlık: İşleç bilgi kartları, ön yüz ile arka yüzü iki tarafı eşleşecek şekilde arkalı önlü çıkartılmalıdır. 4 grup için 4 adet çıktı alınır ve kesikli çizgiler aracılığıyla kartlar oluşturulur. Diğer iki materyalde (temel işlem tablosu ve örnek olay sunum kartı) tüm gruplara bir tane dağıtılacak şekilde dörder tane çıktı alınır.

Uygulama: Bu öğrenme görevi iki aşamadan oluşmaktadır. İlk olarak öğrenciler beşerli gruplara ayrılır. Eğitmen işleç kavramı hakkında kısa bir giriş yapar.

"Programlama dillerinde kendi başlarına kullanımlarında herhangi bir anlam ifade etmeyen fakat program akışına katkıda bulunan sembol veya sembol topluluklarına işleç denir. Buna örnek olarak toplama ya da çıkarma işleci verilebilir. + sembolü tek başına bir anlam ifade etmezken iki sayı arasında yer aldığında toplama işlemi yapılmasını sağlar."

Bu kısa bilgiden sonra her gruba temel işlem tablosu ve işleç bilgi kartları karışık şekilde verilir. İşleç bilgi kartının ön yüzünde işleç ve örnek kullanımı, arka yüzünde ise açıklaması bulunmaktadır. Grupların ilk görevi işleç bilgi kartlarındaki işlemleri gruplayarak, temel işlem tablosunda doğru yere yazmaktır. İkinci görevde ise, öğrencilere bir örnek olay sunum kartı verilir. İki örnek olay kartı ve kartın arka yüzünde örnek olay görevleri verilmiştir. Onlardan sunum kartının arkasındaki görevleri 5 dk. süresince yapmaları ve grup kâğıtlarına yazmaları istenir. Burada öğrenciler örnek olayı inceleyecek ve görevler sırasında tamamladıkları temel işlem tablosunda yer alan bilgileri kullanacaklardır. Bir örnek olay üzerinde çalışan grup, diğer örnek olayı tamamlamak üzere kartını değiştirir. Görevlerin tamamlanmasının ardından

öğrencilerin grup kâğıtlarında yaptıkları benzer ve farklı işlemler, beyin fırtınası ile sınıfta tartışılır. Sonuç olarak öğrencilerin aşağıdaki bilgilere erişmesi sağlanır.

Konu içeriği:

Aritmetik İşlemler

Matematiksel hesaplamalar bilgisayar programlarında yaygın olarak kullanılmaktadır. Programlama dilinde bir işleç, bir değer veya değişken üzerinde çalışan bir semboldür. İki sayıyı birbirine eklemek (3+5) gibi basit bir hesaplama yapabilen ya da x*x - 2*x + 4 gibi karmaşık bir denklemi çözebilen bir program yazabiliriz. Programlama dilinde bu iki ifade aritmetik ifadeler ve bu ifadelerde kullanılan artı (+), eksi (-) gibi semboller de aritmetik işleçler olarak adlandırılır. Bu ifadelerdeki 3, 5 ve x gibi değerler de işlenen olarak adlandırılır. İfadelerin matematiksel yazılımlarının uygun bir şekilde bilgisayar kodlanmasının gerçekleştirilmesi gerekmektedir.

Tablo 2. *Matematiksel ifadeler ve bilgisayar kodlamaları*

Matematiksel Yazım	Bilgisayar Kodlanması
$x + y - 2xy + 5$	x+y-2*x*y+5
$x - 2y + 3z$	x-2*y+3*z
$x + \frac{x}{3} - 3xy + \frac{1}{y}$	x+x/3-3*x*y+1/y
$\frac{x + y}{2xy}$	(x+y) / (2*x*y)
$x - y + \frac{2x}{y + 4x}$	(x-y)+(2*x) / (y+4*x)

Aşağıdaki tabloda, bilgisayar programlamasında kullanılan önemli aritmetik işleçler listelenmiştir. Tabloda x ve y bir değişken olarak kabul edilmiştir.

***Tablo 3**. Aritmetik işleçler*

İşleç	Açıklama	Kullanım
+	İki işleneni birbirine ekler.	x + y
-	İkinci işleneni birinciden çıkarır.	x - y
*	İki işleneni çarpar.	x * y
/	Birinci işlenenin ikinci ile bölümünden elde edilen tam kısımdır (bölme işlemindeki bölüm).	x / y
%	Birinci işlenenin ikinci ile bölümünden elde edilen kalan kısımdır.	x % y
++	Sayıyı bir arttırma	x++ veya ++x
--	Sayıyı bir azaltma	x-- veya --x

Not: Bölme işlemlerinde elde edilen bölüm kısmı kesirli sayı olarak elde edilebilir. Fakat yukarıdaki tabloda verilen bölme işleminde "/" işleci ile bölme işlemi sonucunda bölümün tam kısmı alınmaktadır.

Karşılaştırma İşlemleri

Bu işleçlerin amacı iki değişken ya da değişken grubunu belirtilen şarta göre karşılaştırmaktır. Bu karşılaştırmalar aynı türdeki değişkenler arasında olmalıdır. Bu işleçleri özellikle ilerleyen haftalarda göreceğimiz koşul yapıları ve döngülerde kullanılır. Yapılan karşılaştırmalar doğruysa "1" yanlışsa "0" sonucunu döndürür.

***Tablo 4.** Karşılaştırma işleçleri*

İşleç	Açıklama	Kullanım
<	Küçüktür	x < y
>	Büyüktür	x > y
<=	Küçük eşittir	x <= y
>=	Büyük eşittir	x >= y
==	Eşittir	x == y
!=	Eşit değildir	x != y

Mantıksal İşlemler

Mantıksal işleçler programlama dillerinde çok önemli bir yere sahiptir ve belirli koşullara göre karar vermemize yardımcı olurlar. İki koşulun sonucunu birleştirmek istediğimizde mantıksal VE ve VEYA istediğimiz sonucu üretmemize yardımcı olur. Bütün şartların sağlanması için koşullar arasına VE, herhangi birinin sağlatılması isteniyorsa koşullar arasına VEYA ve koşulu sağlamayanlar isteniyorsa DEĞİL işleci kullanılmalıdır.

Tablo 5. *Mantıksal işleçler*

İşleç	Açıklama	Kullanım
&&	Mantıksal VE	x && y
||	Mantıksal VEYA	x || y
!	Mantıksal DEĞİL	!x

Mantıksal işleçlerin sonucu şu şekilde belirlenmektedir. Eğer x&&y kullanılıyor ise her iki değişkenin değeri “1” olursa sonuç “1” olur, aksi hâlde sonuç “0” olur. Eğer x||y kullanılıyor ise her iki değişkenin değeri “0” olursa sonuç “0” olur, aksi hâlde sonuç “1” olur. Eğer !x kullanılıyor ise değişkenin değeri “1” ise sonuç “0”, “0” ise sonuç “1” olacaktır. Mantıksal VE için aşağıdaki tabloyu inceleyip çalışma prensibini kontrol edebilirsiniz.

Tablo 6. *Mantıksal VE kullanımı*

İşlenen 1	İşleç	İşlenen 2	Sonuç
0	&&	Sıfırdan farklı	0
Sıfırdan farklı	&&	0	0
0	&&	0	0
Sıfırdan farklı	&&	Sıfırdan farklı	1

Mantıksal VEYA için aşağıdaki tabloyu inceleyip çalışma prensibini kontrol edebilirsiniz.

Tablo 7. *Mantıksal VEYA kullanımı*

İşlenen 1	İşleç	İşlenen 2	Sonuç
0	||	Sıfırdan farklı	1
Sıfırdan farklı	||	0	1
0	||	0	0
Sıfırdan farklı	||	Sıfırdan farklı	1

Mantıksal DEĞİL için aşağıdaki tabloyu inceleyip çalışma prensibini kontrol edebilirsiniz.

Tablo 8. *Mantıksal DEĞİL kullanımı*

İşleç	İşlenen 2	Sonuç
!	Sıfırdan farklı	0
!	0	1

B5. Farklı Atama Türlerini Tanıyalım

Süre: 20 dk.

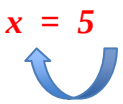
Kazanımlar: K5. Değişkenlere farklı türlerde değer ataması yapar.

Materyaller: O1. B5.1. Yer Değiştirme Görev Kartları

Hazırlık: Materyalin çıktısı arkalı önlü gelecek şekilde 5 adet alınır ve kartlar her grup için ayrı ayrı kesilir.

Uygulama: Eğitmen atama işlemleri için değişken ve değer atama ile ilgili aşağıdaki gibi kısa bir giriş yapar.

Bir değişkene değer atamak için kullanılır. Atama operatörünün sol taraftaki işleneni değişkendir ve atama operatörünün sağ taraftaki işleneni bir değerdir.



x değişkenine 5 değeri atanıyor.

Öğrenciler dörderli beş gruba ayrılır. Eğitmen yukarıdaki örnek üzerinden gruplara farklı tür atama işleçlerini birlikte keşfedeceklerini belirtir ve onlara yer değiştirme görev kartları verir. Eğitmenin yukarıda verdiği değişken atama örneği üzerinden bu görev kartlarını uygulamaları ve aradaki değişimleri tartışmaları istenir. Eğitmen grup çalışmalarının tamamlanmasının ardından her gruptan bir sözcünün görev kartlarından birini diğer gruplara anlatmasını ister. Bu şekilde tüm gruplara bir kartı açıklama görevi verilir. Gruplar açıklama yaparken eğitmen de farklı tür atama işleçlerini aşağıdaki gibi tablo hâlinde tahtada özetler.

Konu İçeriği:

Farklı tür atama operatörleri aşağıda tabloda verilmektedir.

Tablo 9. *Atama operatörleri*

İşleç	Açıklama	Kullanım
=	Sağdaki değeri soldaki değişkene atamak için kullanılır.	x = y z=10
+=	Bu işleç, '+' ve '=' operatörlerinin birleşimidir. Önce soldaki değişkenin geçerli değerini sağdaki değere ekler ve ardından sonucu soldaki değişkene atar.	x+=y (x=x+y)
-=	Bu işleç, '-' ve '=' operatörlerinin birleşimidir. Önce soldaki değişkenin geçerli değerini sağdaki değerden çıkarır ve ardından sonucu soldaki değişkene atar.	x-=y (x=x-y)
*=	Bu işleç '*' ve '=' operatörlerinin birleşimidir. Önce soldaki değişkenin geçerli değerini sağdaki değerle çarpar ve ardından sonucu soldaki değişkene atar.	x*=y (x=x*y)
/=	Bu işleç '/' ve '=' operatörlerinin birleşimidir. Önce soldaki değişkenin geçerli değerini sağdaki değere böler ve ardından sonucu soldaki değişkene atar.	x/=y (x=x/y)
%=	Bu işleç '%' ve '=' operatörlerinin birleşimidir. Önce soldaki değişkenin geçerli değerini sağdaki değere göre modunu alır ve ardından sonucu soldaki değişkene atar.	x%=y (x=x%y)

B6. Sayı Sistemlerini Keşfet!

Süre: 25 dk.

Kazanımlar: K6. İkili ve onlu sayı sistemlerini kullanır.

Materyaller: O1. B6. 1. Sayı sistemleri sunum

O1. B6. 2. Veri dönüştürme örnek olaylar

Hazırlık: Öğrenme yönetim sistemi (ÖYS) üzerinden etkinlik materyallerinden sunum öğrenci erişimine açılır. Örnek olaylar materyali ise beş gruba dağıtmak üzere 5 adet çıktı şeklinde alınır. Öğrenci ürünlerinin paylaşımı için beş gruptan oluşan raf temalı bir padlet ortamı oluşturulur ve padlet linki ÖYS üzerinde öğrencilerle materyal olarak paylaşılır.

Uygulama: Eğitmen öğrencilerden ikişerli gruplar hâlinde interaktif sunum materyalini incelemelerini ister. Öğrencilerden bu materyalde karşılaştıkları basamak, basamak değeri, basamak ağırlığı ve sayı tabanı kavramlarını birbirleriyle tartışmaları beklenir. Devamında eğitmen sayı sistemlerinde basamak, basamak değeri, basamak ağırlığı ve sayı tabanının bilinmesinin önemini aşağıdaki şekilde kısa bir giriş ile vurgular.

Bilgisayarlar sadece sayıları anlayabildiğinden bazı harfleri ve kelimeleri yazdığımızda bunları otomatik olarak sayıya çevirmektedir. Sayıları temsil etme ve bunlarla çalışma tekniğine ***sayı sistemi*** *denir. Sayı sistemlerinde bir doğal sayıyı oluşturan her bir rakam bir* ***basamak*** *olarak adlandırılır. Rakamların bulundukları yerdeki değerine* ***basamak değeri****, her basamağın sahip olacağı üstel ifadeye* ***basamak ağırlığı*** *ve bu doğal sayının tanımlandığı sayı sistemine de* ***sayı tabanı*** *denir.*

Bizler günlük yaşantımızda onlu sayı sistemini kullanırken, bilgisayar sistemleri ikili sayı sistemini kullanır. Onlu sistemde taban 10, ikili sistemde ise taban 2'dir. İkili sayı sisteminin yanı sıra, sekizli ve onaltılı sayı sistemleri de sıklıkla kullanılmaktadır. Onlu sayı sistemlerinde her bir rakam ondalık basamak ya da sadece basamak olarak adlandırılırken, ikili sayı sistemlerinde ikili basamak ya da sadece bit olarak adlandırılır. Sayı sembolleri 0 ile (taban–1) arasında değer almaktadır.

n sayı tabanı ve a, b, c, d, e rakamları n'den küçük olmak üzere,

$A = (abcde)_n$ *sayısının basamakları*

e : n^0 *lar basamağı,*

d : n^1 *ler basamağı,*

c : n^2 *ler basamağı,*

b : n^3 *ler basamağı,*

a : n^4 *ler basamağıdır.*

$A = (abcde)_n = a.n^4 + b.n^3 + c.n^2 + d.n^1 + e.n^0$

şeklinde yazılmasına A sayısının n tabanına göre çözümlenmesi denir.

Bu kavramlardan sonra öğrenciler dörderli gruplar hâlinde çalışacaktır. Eğitmen öğrencilere veri dönüştürme örnek olay kartını dağıtır. Dört kişilik gruplar hâlinde öğrenciler karttaki iki örnek olayla ilgili aşağıdaki görevleri tamamlayacaktır. Eğitmen bunun için aşağıdaki talimatları verir:

Örneklerdeki sayı tabanı, basamak ve basamak değerini bulun.

İlgili örneklere benzer birer örnek de siz yazın. Ancak basamak hesaplama kısmını grup arkadaşlarınıza bırakın.

Öğrenciler tamamladıkları görevleri padlet üzerinden birbiriyle paylaşır. Eğitmen padlet ortamında atılan hatalı bir çözümü sınıfta tartışarak konuyu özetler. Daha sonra öğrenciler grup içinde ellerindeki örnek olay kartını değiştirir ve yukarıdaki görevleri tekrar eder ve padlet ortamında yeniden paylaşırlar. Bu sefer gönderilerin daha az hatalı olması beklenir. Varsa yanlışlar tekrar eğitmen tarafından özetlenir.

Konu İçeriği:

Onlu Sayı Sistemi

Onlu sayı sisteminde 0 ile 9 arasında yalnızca on rakam vardır. Bu sayı sisteminde her sayı 0,1,2,3,4,5,6,7,8 ve 9 ile temsil edilir. Onlu sayı sisteminin tabanı 10'dur, çünkü yalnızca 10 rakam kullanılır. Bu sistemde her bir basamak ağırlığı şu şekilde gösterilebilir:

10^7	10^6	10^5	10^4	10^3	10^2	10^1	10^0

Örnek: Aşağıdaki onlu sayıların basamak analizini inceleyiniz.

$328 = 3 \times 10^2 + 2 \times 10^1 + 8 \times 10^0$

$59 = 5 \times 10^1 + 9 \times 10^0$

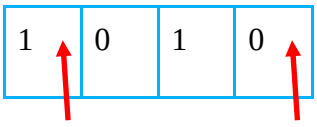
$7401 = 7 \times 10^3 + 4 \times 10^2 + 0 \times 10^1 + 1 \times 10^0$

İkili Sayı Sistemi

İkili sayı sisteminde yalnızca 0 ve 1 olmak üzere iki rakam bulunur. Her sayı, bu sayı sisteminde 0 ve 1 ile gösterilir. İkili sayı sisteminin tabanı 2'dir, çünkü sadece iki rakam kullanılır. Her ikili basamak ayrıca bit olarak ifade edilir. Bu sistemde her bir basamak ağırlığı şu şekilde gösterilebilir:

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Herhangi bir ikili sayıda, en sağdaki basamağa en az anlamlı bit (LSB) ve en soldaki basamağa en anlamlı bit (MSB) denir.



MSB LSB

Verilen bu sayının onlu eşdeğeri, basamak değeri ile her bir basamak ağırlığının çarpımının toplamıdır.

Örnek: Aşağıdaki ikili sayının basamak analizini inceleyiniz.

$(1010)_2 = 1\times2^3 + 0\times2^2 + 1\times2^1 + 0\times2^0$

$= 8 + 0 + 2 + 0$

$= (10)_{10}$

Örnek: Aşağıdaki ikili sayının basamak analizini inceleyiniz.

$(1001101)_2 = 1\times2^6 + 0\times2^5 + 0\times2^4 + 1\times2^3 + 1\times2^2 + 0\times2^1 + 1\times2^0$

$= 64 + 0 + 0 + 8 + 4 + 0 + 1$

$= (77)_{10}$

Onlu Sayı Sisteminden İkili Sayı Sistemine Dönüştürme

İkili sayı sisteminden onlu sayı sistemine ve onlu sayı sisteminden ikili sayı sistemine dönüştürme işlemleri tüm bilgisayar sistemlerinin temelini oluşturduğundan önemli bir kavramdır. Yukarıdaki bölümde ikili sayı sisteminden onlu sayıya dönüştürmenin nasıl yapıldığını örneklerle gösterdik. Onlu sayı sisteminden ikili sayı sistemine dönüştürme için aşağıdaki adımlar kullanılır:

Sayıyı 2 ile bölünüz.

Sonraki yineleme için tamsayı bölümünü alın.

İkili sayının oluşturulması için kalanı alın.

Bölüm 0'a eşit olana kadar adımları tekrarlayın.

Örnek: $(19)_{10}$ sayısını ikili sayı sistemine dönüştürelim.

2 ile Bölme	Bölüm	Kalan	Bit Numarası
19/2	9	1	0
9/2	4	1	1
4/2	2	0	2
2/2	1	0	3
1/2	0	1	4

Sonuç olarak $(19)_{10} = (10011)_2$ olmaktadır.

Örnek: $(145)_{10}$ sayısını ikili sayı sistemine dönüştürelim.

2 ile Bölme	Bölüm	Kalan	Bit Numarası
145/2	72	1	0
72/2	36	0	1
36/2	18	0	2
18/2	9	0	3
9/2	4	1	4
4/2	2	0	5
2/2	1	0	6
1/2	0	1	7

Sonuç olarak $(145)_{10} = (10010001)_2$ olmaktadır.

C. Kısmi Öğrenme Görevleri

Süre: 60 dk.

Materyal: O1. C. 1. Kısmi Öğrenme Görevleri Afişi

Hazırlık: Kısmi öğrenme görevleri afişi öğrencilere ÖYS ortamında süreli ödev olarak açılır.

Uygulama: Öğrenciler afişteki görevleri istedikleri sırada ve sayıda kendi tercihlerine bağlı olarak verilen süre içinde uygulamaya başlar. Her bir görevi tamamlayan öğrencilere, göreve ilişkin beceri rozeti verilir. Öğrenciler görev seçimlerini ve tamamladıkları görevleri eğitmene bildirmelidir. Eğitmenler ihtiyaç duydukları zaman öğrencilere görevi tamamlama aşamasında anlık geri bildirimlerde bulunur. Öğrenciler, ÖYS üzerinden süreli ödev olarak açılan kısmi öğrenme görevlerinden kendi tercihlerine göre istedikleri sayıda görev tamamlayıp eğitmene iletir. Süre bitiminde eğitmen görevleri ve yanıtlarını ÖYS ya da GitHub üzerinden öğrencilere gönderir.

Kısmi Öğrenme Görevleri Yanıtlar: Görevler, öğrenciye verilecek beceri rozetleri ile isimlendirilmiştir. Her bir rozetin altında ilgili görevin yanıtları verilmektedir. Eğitmen bu yanıtları uygulama süresinin sonunda GitHub üzerinden öğrencilerle paylaşır.

Tasarlayıcı 1: Tasarlayıcı rozeti 3 farklı görevden oluşmaktadır. Bu görevlerden birinin tamamlanması Tasarlayıcı rozetinin kazanılması için yeterlidir. Üç görevi de yapan öğrenciler üç kere Tasarlayıcı rozetini almış olur.

1. Kişiye özel bir oyuncak tasarlamalısın. Düğmeye bastığımızda şarkı söyleyen, adımızı hatırlayan, saate göre günaydın/iyi akşamlar/iyi geceler dileyen bir oyuncak için giriş-çıkış, CPU, Bellek görevleri neler olmalıdır? Çizerek anlatır mısın?

Cevap: Bu soru tamamen öğrencinin hayal yeteneğine bağlı olup basitçe; Düğme: Giriş, Hoparlör: Çıkış, Adın hatırlanması: Bellek, Saatin hesaplanması: CPU olarak düşünülebilir.

2. Aşağıdaki ifadeyi bilgisayar diline çeviriniz.

3x-2xy+4y

Cevap: 3*x - 2*x*y+4*y

3. Arda öğretmen sınav sorularının puan değerini ikili tabana göre aşağıdaki gibi belirledi. Sınavdan en yüksek kaç puan alabilirsin? Cevap anahtarına göre doğruların 1001011 şeklinde ise sınavdan kaç alırsın?

Soru	1	2	3	4	5	6	7
Puan	64	32	16	8	4	2	1

Cevap: Sınavdan en yüksek 127 alabilirsin.

Sınavdn alacağın not: 64+8+2+1 = 75

Analizci: Ayşe'nin parka gitmesi için; Ödevlerini bitirmiş olması (A); Hava sıcaklığının 20-30 derece arasında olması (B); Karnının tok olması (C) gerekiyor. Buna göre Ayşe'nin parka gitmesini mantıksal operatörler ile yazın.

Cevap:

A = Ödevini bitirdi mi? (0 ise bitirmedi 1 ise bitirdi.)

B = 20<Hava sıcaklığı<30 (Doğru/Yanlış)

C = Karnı tok mu? (0 ise aç, 1 ise tok.)

(A==1 && B ==1 && C==1) Kısaca: (A&&B&&C)

Kodlayıcı: Ali bilgisayara 1'den 10'a kadar olan sayıları aşağıdaki koşullara uygun olarak yazdırmak istiyor.

Eğer sayı <= 5 ise sayı = sayı * 2

Eğer sayı > 5 ise sayı = sayı % 2

Buna göre bilgisayar ekranında ne yazacaktır?

Cevap:

2

4

6

8

10

0

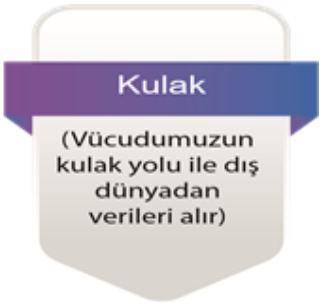
1

0

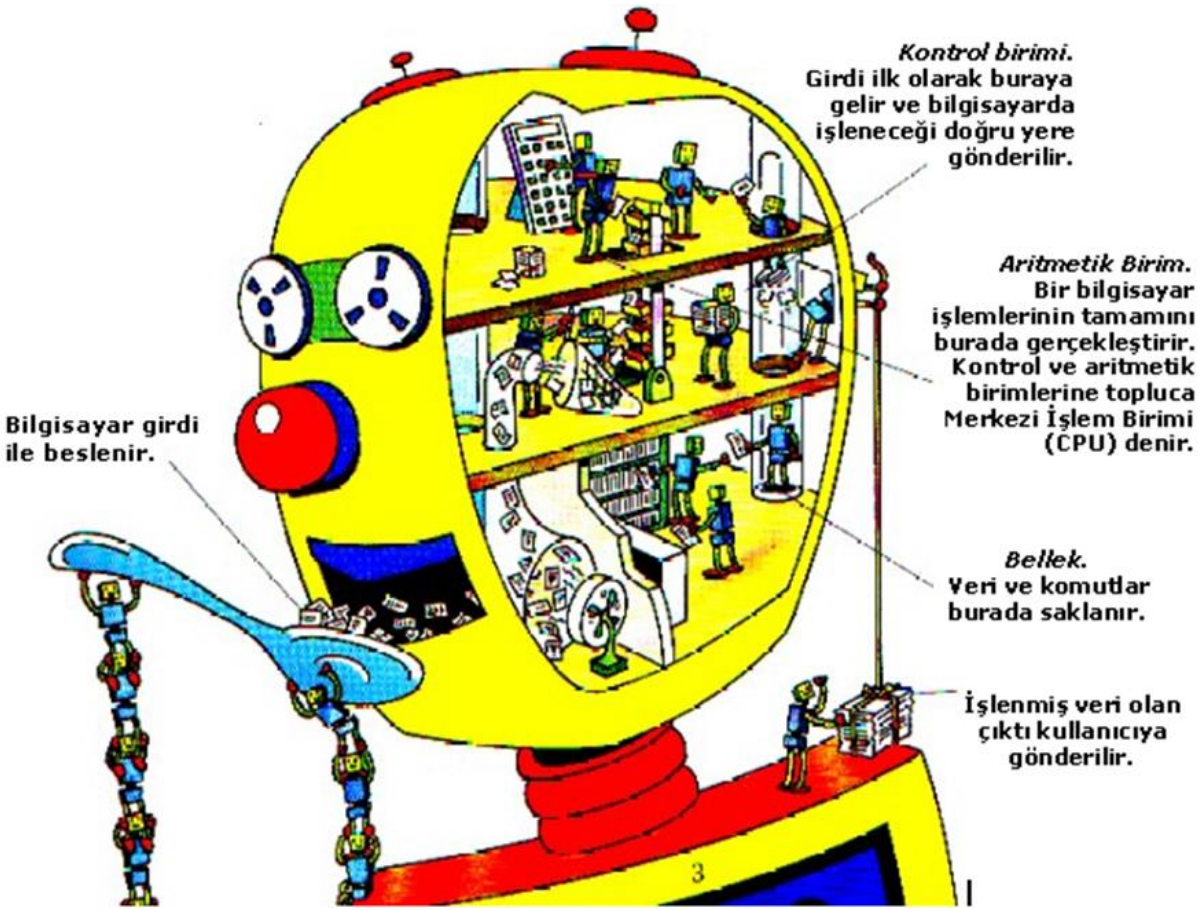
1

Hafta 1. Ders Materyalleri

O1. B2. 1. Eşleştirme Kartları

O1. B2. 2. Bilgisayarın Temel Mimarisi



O1. B3. 1. Sıralama Kartları (Ön Yüz)

BİT
(BİT)

1 BİT

BİT
(BİT)

2 BİT

BİT
(BİT)

5 BİT

BİT
(BİT)

6 BİT

BYTE
(B)

1 BYTE

KİLOBAYT
(KB)

1 KB

MEGABYTE
(MB)

1 MB

GİGABYTE
(GB)

1 GB

TERABYTE
(TB)

1 TB

BİT
(BİT)

8 BİT

O1. B3. 1. Sıralama Kartları (Arka Yüz)

Bilgisayar sistemlerindeki bütün bilgiler ikilik sistemde “0” ve “1” ile temsil edilerek saklanır. İkilik sistemdeki her bir basamağa bit denir.	Bilgisayarda depolanan bilgilerin “01” , “10”, “00”, “11” şeklinde bir araya gelmesiyle oluşur.	Bilgisayarda depolanan bilgilerin “01010” , “10001”, “00000”, “11111” şeklinde beş tane rakamın bir araya gelmesiyle oluşur.
Bilgisayarda depolanan bilgilerin “010101” , “100010”, “000000”, “111111” şeklinde altı tane rakamın bir araya gelmesiyle oluşur.	Bilgisayarda depolanan bilgilerin “01010101” , “10001001”, “00000000”, “11111111” şeklinde sekiz tane rakamın bir araya gelmesiyle oluşur.	1024 tane byte’ın bir araya gelmesiyle oluşan hafıza birimidir.
1024 tane Kilobyte’ın bir araya gelmesiyle oluşan hafıza birimidir.	1024 tane Megabyte’ın bir araya gelmesiyle oluşan hafıza birimidir.	1024 tane Gigabyte’ın bir araya gelmesiyle oluşan hafıza birimidir.
	Bilgisayarda depolanan bilgilerin “01010101” , “10001001”, “00000000”, “11111111” şeklinde sekiz tane rakamın bir araya gelmesiyle oluşur.	

O1. B4. 1. Temel İşlem Tablosu

Aritmetik İşlemler	Karşılaştırmalı İşlemler	Mantıksal İşlemler
Programlama dilinde iki ifade arasında toplama, çıkarma, bölme, yüzdesini alma gibi matematiksel işlemler için kullanılan sembollerdir.	*Programlama dilinde aynı türdeki değişkenler arasında belli bir koşula göre karşılaştırma yaptıran sembollerdir. Koşul doğru ise 1, yanlış ise 0 olarak program sonuçlanır.*	*Programlama dilinde iki ya da daha fazla koşulun birlikte değerlendirilmesinde kullanılan sembollerdir.*

O1. B4. 2. İşleç bilgi kartları

Not: İşleç bilgi kartları önlü arkalı olarak tasarlanmış olup, bu şekilde çıktı alınmalıdır. Ön yüz işleç hakkında açıklama verirken arka yüz kullanım örneğini oluşturur.

Ön Yüz	Arka Yüz
İki işleneni toplar. İşleç: a **+** d	Arda'nın yaşı (a=5) ve Duru'nun yaşını (d=4) toplayalım. Çıktı: 5 + 4 = 19
İkinci işleneni birinciden çıkarır. İşleç: a **-** d	Arda'nın yaşından (a=5) ve Duru'nun yaşını (d=4) çıkaralım. Çıktı: 5 - 4 = 1
İki işleneni çarpar. İşleç: a ***** d	Arda'nın yaşı (a=5) ve Duru'nun yaşını (d=4) çarpalım. Çıktı: 5 * 4 = 20
İkinci işlenen birinciye böler. İşleç: a **/** d	Arda'nın yaşını (a=5) ve Duru'nunkine (d=4) bölelim. Çıktı: 5 / 4 = 1
Bölme işleminden kalanı verir. İşleç: a % d	Arda'nın yaşını (a=5) ve Duru'nun yaşına (d=4) bölüp, kalanı yazalım. Çıktı: 5 % 4 = 1

Sayıyı bir arttırma İşleç: a++	Arda'nın yaşını (a=5) bir arttıralım. Çıktı: 5++ = 6
Sayıyı bir azaltma İşleç: a--	Duru'nun yaşını (d=4) bir azaltalım. Çıktı: 4-- = 3
Büyüktür. İşleç: a > d	Arda'nın yaşı (a=5), Duru'nun (d=4) yaşından büyüktür. Çıktı: 1 (Doğru)
Küçüktür. İşleç: <	Arda'nın yaşı (a=5), Duru'nun (d=4) yaşından küçüktür. a < d Çıktı: 0 (Yanlış)
Küçük eşittir İşleç: <=	Arda'nın yaşı (a=5), Duru'nun (d=4) yaşından küçük eşittir. a <= d Çıktı: 0 (Yanlış)
Büyük eşittir İşleç: >=	Arda'nın yaşı (a=5), Duru'nun (d=4) yaşından büyük eşittir. a >= d Çıktı: 1 (Doğru)

Eşittir İşleç: ==	Arda'nın (a=5) ve Duru'nun (d=4) yaşı eşittir. a == d Çıktı: 0 (Yanlış)
Eşit değildir. İşleç: !=	Arda'nın (a=5) ve Duru'nun (d=4) yaşı eşit değildir. a != d Çıktı: 1 (Doğru)
VE İşleç: &&	Arda'nın yaşı a=5 ve Duru'nun yaşı d=4'tür. ((a==5) && (d>5)) Çıktı: 1 && 0 = 0 (Yanlış) ((a!=5) && (d>5)) Çıktı: 0 && 0 = 0 (Yanlış) ((a==5) && (d<5)) Çıktı: 1 && 1 = 1 (Doğru)
VEYA İşleç: ||	Arda'nın yaşı a=5 ve Duru'nun yaşı d=4'tür. ((a==5) || (d>5)) Çıktı: 1 || 0 = 1 (Doğru) ((a!=5) || (d>5))

	Çıktı: 0 || 0 = 0 (Yanlış) ((a==5) || (d<5)) Çıktı: 1 || 1 = 1 (Doğru)
DEĞİL İşleç: !	Arda'nın yaşı a=5 ve Duru'nun yaşı d=4'tür. ! (a==4) Çıktı: 1 (Doğru) ! (d!=5) Çıktı: 0 (Yanlış)

O1. B4. 3. Örnek Olay Sunum Kartı

Bir okulda yaşı 15'in üzerinde olan ve 9. sınıfta okuyan öğrenciler için gezi planı yapılıyor. Okul müdürü sizden bu öğrencilerin isimlerini istedi. Bu durumda bu kriterlere uygun öğrencilerin ismini listelemek için aşağıdaki gibi bir ifade oluşturmalıyız. Bu örnekte iki koşul bulunmaktadır. Bu iki koşulun da doğru (1) olması gerekmektedir. Bunun için, Eğer Yaş > 15 **&&** Sınıf == 9 ise, öğrenci ismi yaz 1. Koşul 2. Koşul	GÖREV Örnek olaydaki koşul ifadelerini temel işlem tablosundaki semboller ile değiştirmeyi deneyin. Önceki sonuçlar ile yeni sonuçları karşılaştırın.

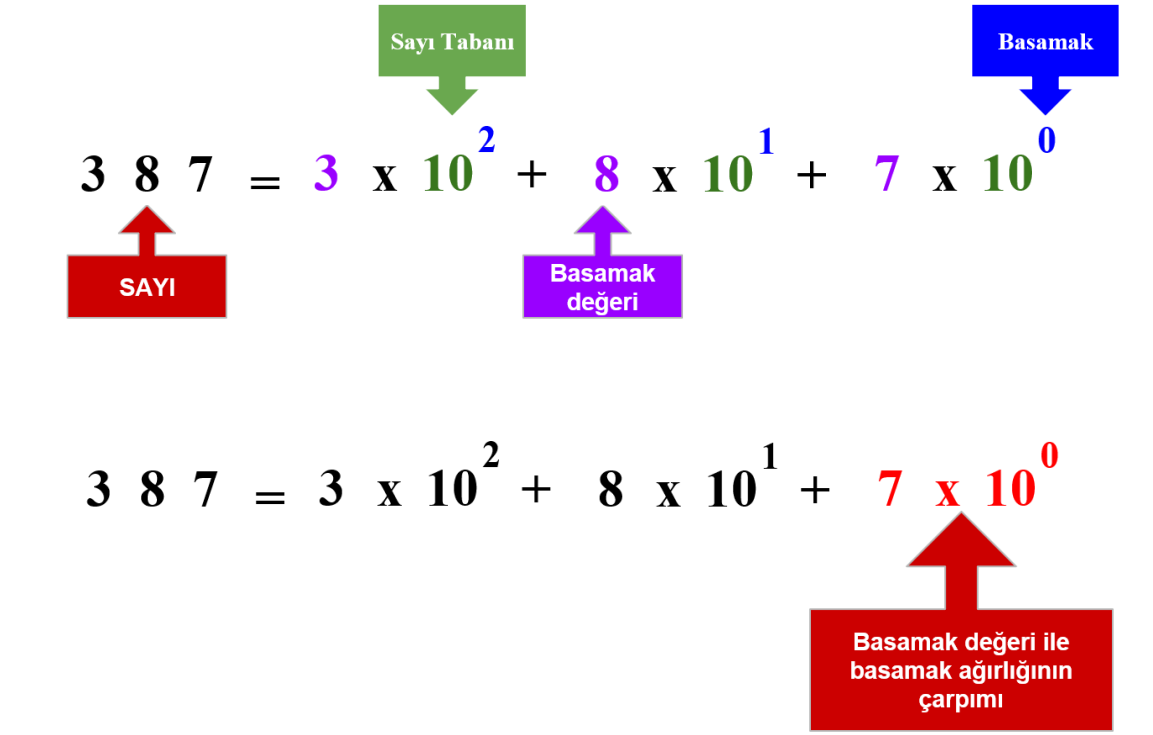
O1. B5. 1. Yer Değiştirme Görev Kartları

*Görev kartları arkalı önlü olacak şekilde basılmalıdır. Görev kartlarındaki her satır bir görevi oluşturmaktadır.

Görevler	İpuçları
Görev Örnekteki = atama işlecinin ardından += işlecini uygulayın ve sonucu tahmin etmeye çalışın.	**İpucu** += atama işleci '+' ve '=' operatörlerinin birleşimidir. Önce soldaki değişkenin geçerli değerini sağdaki değere ekler ve ardından
Görev Örnekteki = atama işlecinin ardından, -= işlecini uygulayın ve sonucu tahmin etmeye çalışın.	**İpucu** '-' ve '=' operatörlerinin birleşimidir. Önce soldaki değişkenin geçerli değerini sağdaki değerden çıkarır ve ardından sonucu soldaki
Görev Örnekteki = atama işlecinin ardından, *= işlecini uygulayın ve sonucu tahmin etmeye çalışın.	**İpucu** '*' ve '=' operatörlerinin birleşimidir. Önce soldaki değişkenin geçerli değerini sağdaki değerle çarpar ve ardından sonucu soldaki

Görev Örnekteki = atama işlecinin ardından /= işlecini uygulayın ve sonucu tahmin etmeye çalışın.	**İpucu** '/' ve '=' operatörlerinin birleşimidir. Önce soldaki değişkenin geçerli değerini sağdaki değere böler ve ardından sonucu soldaki
Görev Örnekteki = atama işlecinin ardından %= işlecini uygulayın ve sonucu tahmin etmeye çalışın.	**İpucu** '%' ve '=' operatörlerinin birleşimidir. Önce soldaki değişkenin geçerli değerini sağdaki değere göre modunu alır ve ardından sonucu soldaki

O1.B6.1.Sayı Sistemleri Sunum



Resim 9. *Basamak işlemi sorusu*

O1.B6.2. Veri Dönüştürme Örnek Olaylar

İkilik sayı sisteminden, onluk sisteme dönüşümünü inceleyiniz.

Bunun için sayının basamak değerleri, basamak pozisyon (konum) değeri ile çarpılıp toplanır. $(1010)_2$ sayısını onluk sisteme çevirelim.

$$(1010)_2 = 1\times2^3 + 0\times2^2 + 1\times2^1 + 0\times2^0$$
$$= 8 + 0 + 2 + 0$$
$$= (10)_{10}$$

Onluk sayı sisteminden, ikilik sisteme dönüşümünü inceleyiniz.

$(19)_{10}$ sayısını, ikili sayı sistemine dönüştürelim.

Bunun için aşağıdaki adımlar kullanılır:

1. Sayıyı 2 ile bölünüz.
2. Sonraki yineleme için tamsayı bölümünü alın.
3. İkili bassamak için kalanı alın.
4. Bölüm 0'a eşit olana kadar adımları tekrarlayın.

2 ile Bölme	Bölüm	Kalan	Bit Numarası
19/2	9	**1**	0
9/2	4	**1**	1
4/2	2	**0**	2
2/2	1	**0**	3
1/2	0	**1**	4

Sonuç olarak $(19)_{10} = (11001)_2$ olmaktadır.

O1. C. 1. Kısmi Öğrenme Görevleri Afişi

Kısmi öğrenme görevleri afişine ulaşmak için buraya tıklayınız.

Hafta 2. Algoritma Tasarımı

Kazanımlar

K1. Verilen algoritmanın temel özelliklerini analiz eder.

K2. Bir problemin çözümüne uygun farklı algoritmalar tasarlar.

K3. Bir problemin çözümüne uygun algoritmanın akış şemasını oluşturur.

K4. Bir problemin çözümüne uygun algoritmanın akış şemasını bilgisayar ortamında çizer.

Amaç

Haftanın amacı bir problemi algoritma ve akış diyagramları kullanarak çözmek ve problemin çözümünü bilgisayar kullanarak aktarmaktır.

Önerilen Ders Akışı

A. Giriş: Çizgi ve Nokta Oyunu (10 dk.)

B. Öğrenme Görevleri

B1. Algoritmayı Tanıyorum! (25 dk.)

B2. Algoritmaları Eşleştirelim (25 dk.)

- ❏ Ders Arası (5 dk.)
- ❏ Motivasyon Oyunu (10 dk.)

B3. Algoritma Yazıyorum (25 dk.)

B4. Çıkartma Algoritması (25 dk.)

- ❏ Ders Arası (5 dk.)

B5. Zamanla Yarışalım (40 dk.)

- ❏ Ders Arası (10 dk.)

C. Kısmi Öğrenme Görevleri (60 dk.)

A. Giriş: Çizgi ve Nokta Oyunu

Süre: 10 dk.

Uygulama: Eğitmen derse girişte öğrenme sırasında grupları belirlemek için Çizgi-Nokta oyununu uygular. Bu oyunda öğrencilerden isimlerinin baş harfini alfabetik sıraya dizerek çizgi şeklinde durmaları istenir. Öğrenciler çizgiyi oluştururken hiç konuşmadan beden ya da işaret dili kullanacaktır. Bunun için eğitmen "Birbirinizin ismini konuşmadan sessizce tahmin edip çizgiyi oluşturalım" diyerek, kendisi de çizgiye katılır. Çizgi tamamlandığında herkes ismini söyler ve kontrol yapılır. Çizgiden sonra eğitmen "Şimdi de lütfen sizinle aynı mevsimi seven bir arkadaşınızı bulup ikili gruplar oluşturun. Yine bu işlemi de sessizce yapın. İşaret kullanabilirsiniz. Bu şekilde arkadaşınızla bir nokta oluşturmuş olacaksınız." diyerek nokta olmalarını ister. Oyun nokta aşamasında bitirilir. Nokta olanlar dersin devamında grup çalışmalarında ikili grupları oluşturacaktır.

Eğitmene Öneriler: Eğitmen çizgi ve nokta oluşturma talimatları değiştirebilir. Örnek talimatlar: Doğdukları aylara göre çizgi olmak ve aynı göz rengine sahip olan biriyle nokta olmak vb. Süreye bağlı olarak sadece bir kez ya da daha fazla çizgi ve nokta aşamaları gerçekleştirebilir. Oyun nokta oldukları aşamada bitirilmelidir.

B. Öğrenme Görevleri

B1. Algoritmayı Tanıyorum!

Süre: 25 dk.

Kazanımlar: K1. Verilen algoritmanın temel özelliklerini analiz eder.

Materyaller: O2. B1. 1. Doğru algoritma hangisi?

Hazırlık: Ders materyali ÖYS üzerinden öğrenci erişimine açılır ya da çıktı alınır.

Uygulama: Eğitmen algoritmanın ne olduğu ile ilgili konuya dikkat çeker ve aşağıdaki gibi kısa bir giriş yapar.

> *Algoritma, bir problemi ya da sorunu çözmek için izlenecek mantıksal kural ve adımların listesidir. Diğer bir ifadeyle, bir problemi çözmek için takip edilecek sonlu sayıda adımdan oluşan bir çözüm yoludur. Mevcut var olan bilgilerden istenilenlere erişmemizi sağlar. Bir algoritmayı anlamanın en iyi yolu onu bir tarif (ilaç, yemek vb.) olarak düşünmektir. Günlük hayattaki yaşantı şeklimizde düzenli olarak birtakım işlemlerin sıra ile yapılması şeklindedir. Yani bir işi yapabilmek için bir takım alt işleri peş peşe gerçekleştiririz.*

Daha sonra öğrencilerden çizgi-nokta oyunundaki noktaların ikili grup hâlinde birleşmelerini ve ÖYS'de haftanın materyallerinden "Doğru Algoritma Hangisi" başlıklı olanı açmalarını ister. Daha sonra eğitmen öğrencilere materyal üzerinden aşağıdaki görevleri yapmalarını ister.

1. *Verilen algoritmaları arasındaki benzerlik ve farklılıkları düşünün. En iyi yazılan kek algoritmasını bulun.*

2. *Verilen algoritma üzerinden doğru algoritma yazma özellikleri hakkında grup arkadaşınızla tartışın.*

Bu görevleri yaparken öğrencilerden algoritmanın özellikleri üzerine düşünmeleri beklenir. Eğitmen görev sonunda sınıfta öğrencilere beyin fırtınası yaptırarak, onların algoritmanın özelliklerini aşağıdaki gibi keşfetmelerine imkân verir.

- *Bir başlangıç noktası olmalı*
- *Basit olmalı*
- *Algoritma içindeki ifadeler herkes tarafından aynı şekilde anlaşılabilir olmalı*
- *Yorum gerektirmemeli ve belirsiz ifadelere sahip olmamalı*
- *Her adımda tek bir iş yapılmalı*
- *Adımların gerçekleşme sırası doğru olmalı*
- *Mümkün olan en az adım ile en kısa sürede gerçekleşmeli*
- *Sonsuz döngüye girmemeli*
- *Bir bitiş noktası olmalı*

Eğitmene Öneriler: Görev sonunda eğitmen öğrencilere en iyi yazılmış algoritmanın diğerinden farklı ve benzer olan yanları üzerine sorular sorabilir. Bunlar algoritmanın özellikleri için ipuçları verecektir.

B2. Algoritmaları Eşleştirelim

Süre: 25 dk.

Kazanımlar: K1. Verilen algoritmanın temel özelliklerini analiz eder.

Materyaller: O2. B2. 1. Algoritmaları Eşleştirelim Çalışma Kâğıdı

Hazırlık: Nokta hâlinde oturan öğrencilerden yanlarındaki bir noktayla birleşmeleri ve dörtlü gruplar hâlinde oturmaları istenir. Çalışma kâğıdının her gruba ikişer tane dağıtmak üzere 8 adet çıktısı alınır.

Uygulama: Eğitmen dörtlü gruplar hâlinde oturan öğrencilerden "Algoritmaları Eşleştirelim" materyalini açmalarını ister. Materyalde iki farklı problemin metin, sözde kod ve akış diyagramı şeklinde hazırlanmış algoritması vardır. Öğrenciler problemi ve yazım şekillerini eşleştirerek tahmin etmeleri istenir. Bu etkinlik ile öğrencilerden bir problemin algoritmasının üç farklı şekilde oluşturulabildiğini keşfetmeleri sağlanır.

Eğitmene Öneriler: Bu etkinlikte öğrencilerin edinmesi gereken içerik için aşağıdaki bilgiler incelenebilir.

a) Metin olarak yazım: Problemin çözüm adımları, düz metin olarak açık cümlelerle ifade edilir. Algoritmadaki her bir satıra numara verilir. "Başla" ile başlayıp "Bitir" ile bitirilir.

b) Sözde kod (pseudocode) yazım: Problemin çözüm adımları komut benzeri anlaşılır metinlerle ifade edilir. Sözde kod konuşma dilinde ve programlama mantığı altında, "eğer", "iken" gibi koşul kelimeleri ve '>', '=', '<' gibi ifadeler ile birlikte yazılır. Algoritma yazma ile kod yazma arasında kalan bir kısımdır ama kodlamaya daha yakındır. İyi yazılmış sözde koddan bir programlama diline kolaylıkla geçiş yapabilirsiniz.

c) Akış diyagramlarının çizilmesi: Problemin çözüm adımları, görsel olarak simge ya da sembollerle gösterilir. Akış şemalarının algoritmadan farkı, algoritma adımlarının semboller şeklinde kutulara yazılması ve adımlar arasındaki ilişki ve yönün oklar ile gösterilmesidir.

B3. Algoritma Yazıyorum?

Süre: 25 dk.

Kazanımlar: K2. Bir problemin çözümüne uygun farklı algoritmalar tasarlar.

K3. Bir problemin çözümüne uygun algoritmanın akış şemasını oluşturur.

Materyaller: O2. B3. 1. Nereye Gidelim Haritası

O2. B3. 2. Görev Kartları

O2. B3. 3. Akış Diyagramları Bilgi Afişi

Hazırlık: Materyallerden "Nereye Gidelim Haritası" ve "Bilgi Afişi" ÖYS'de öğrencilerin kullanımına açılır. Görev kartları ise iki adet çıktı alınarak kesilir ve tüm kartlar bir torba içine karışık olarak atılır. Öğrencilerin grup görevlerini paylaşması beş grup sütunundan oluşan raf temelli padlet linki öğrencilere ÖYS üzerinden paylaşılır.

Uygulama: Öğrenciler noktalar hâlinde ikişerli gruplara geri döner. Gruplar ÖYS'den ders materyali olan haritayı açar. Eğitmen her grubun torba içinden bir görev kartı seçmesini ister. Görev kartlarında haritada bir yerden diğerine ulaşmak için izlenmesi gereken talimatların algoritmasını sözde kod ile yazması istenir. Torbada bir görev iki kere tekrarlanmaktadır. Bu nedenle iki grup aynı problem üzerine çalışır. İkili gruplar sözde kodları oluşturduktan sonra, aynı problem üzerine gruplar bir araya gelip yeni grupları oluştururlar. Birleşen yeni gruplar artık dörder kişiden oluşmaktadır. Yeni grupların oluşturdukları sözde kodları akış diyagramlarının kullanarak akış şemasına dönüştürmeleri istenir. Eğitmen bu aşamada öğrencilerden "Akış diyagramları bilgi afişini" kullanmalarını ister ve afişe ÖYS'den açabileceklerini belirtir. Öğrencilerden etkinlik sonunda grup ürünlerinin ekran görüntülerini padlet ortamında paylaşmaları istenir. Paylaşımlar üzerinden eğitmen konuyu özetler.

Eğitmene Öneriler: Eğitmen bu etkinlikte akış diyagramlarına giriş yapmaktadır. Draw.io programını kullanarak öğrencilerin akış diyagramlarını bilgisayar ortamında oluşturabileceklerini söyleyebilir. Etkinlik süresine bağlı olarak kısaca program öğrencilere tanıtılabilir. Programın kullanım kılavuzu ders öncesi öğrencilere ÖYS üzerinden gönderilebilir.

B4. Çıkartma Algoritması

Süre: 25 dk.

Kazanımlar: K2. Bir problemin çözümüne uygun farklı algoritmalar tasarlar.

K3. Bir problemin çözümüne uygun algoritmanın akış şemasını oluşturur.

K4. Bir problemin çözümüne uygun algoritmanın akış şemasını bilgisayar ortamında çizer.

Materyaller: O2. B4. 1. Çıkartma Algoritması Çalışma Kâğıdı

O2. B4. 2. Draw.io Program Kullanım Kılavuzu

O2. B3. 3. Akış Diyagramları Bilgi Afişi

Hazırlık: Eğitmen nokta oluşturan öğrencilerle ikişerli gruplar oluşturur. Çalışma kâğıdının 10 adet çıktısı alınır. Akış Diyagramları Bilgi Afişi ve program kullanım kılavuzu ÖYS'de öğrencilerin erişimine açılır. Öğrencilerin Draw.io programını açmaları istenir. Öğrenciler ders öncesi program kullanım kılavuzunu incelemiş olmalıdır.

Uygulama: Eğitmen nokta oluşturan öğrencilerle ikişerli gruplar oluşturur. Eğitmen her gruba sözde kodlar hâlinde ancak karışık sırada yazılmış, iki sayı arasındaki çıkartma algoritmasının çıktısı verilir. Gruplar sırasıyla aşağıdaki görevleri tamamlamalıdır.

1. Sözde kod hâlinde verilen çıkartma algoritmasını doğru şekilde sıralayın. Bu aşamada ÖYS'de erişimi açılan "Akış Diyagramları Bilgi Afişini" inceleyin. Nokta hâlindeki bir diğer grup ile sıralamanızı karşılaştırın.
2. Sıralanan algoritmanın akış şemasında kullanılacak diyagramları belirleyin ve "Draw.io" programını kullanarak algoritmayı çizin. Bu aşamada Draw.io program kullanım kılavuzundan yararlanın.

Eğitmene Öneriler: Draw.io programında akış şemaları çizim özelliklerinden öğrencilere kısaca bahsedilebilir. Program kılavuzunu ders öncesinde inceleyenler, incelemeyenler ile gruplanabilir.

B5. Zamanla Yarışalım!

Süre: 40 dk.

Kazanımlar: K2. Bir problemin çözümüne uygun farklı algoritmalar tasarlar.

K3. Bir problemin çözümüne uygun algoritmanın akış şemasını oluşturur.

Materyaller: O2. B5. 1. Grup görevleri

Hazırlık: Eğitmen beşerli dört grup oluşturur. Her grup kendisine bir isim vermelidir. Grup görevlerinin bir adet çıktısı alınır ve kesilerek kartlara dönüştürülür.

Uygulama: Zamanla yarışılan bu etkinlikte öğrenciler gruplar hâlinde çalışmaktadır. Atölyenin birbirinden uzak farklı dört köşesine grup isimleri yazılır. Her grup bir torba içinden görevlerini seçer. Torba içinde grup sayısı kadar görev bulunmaktadır. Eğitmen her gruba 2 dk. tanımlar. 2 dk. sonra, eğitmen grupların görev numarasını ve görevin yanıtını grup isminin altına yapıştırmasını ister. Eğitmen yanıt kâğıdının yapıştırılması için 30 sn verir. Tüm gruplar yanıtlarını yapıştırdıktan sonra, saat yönünün tersine gruplar yer değiştirir. Yeni ekip, eski ekibin görevini 2 dk. içinde tamamlamaya çalışır ve yanıtlarını grup isminin altına yapıştırır. Bu şekilde bir grup tüm görevlere dönene kadar işlem tekrar edecektir. Grup görevlerinin tamamlanması toplamda 10 dk.'da bitirilmelidir. Son olarak eğitmen görev numarasına göre grupların yanıtlarını karşılaştırır. En çok görevi doğru tamamlayan gruba ödül verilir.

Eğitmene Öneriler: Grupların yanıtları öğrenciler tarafından padlet ortamında paylaşılabilir. Bu şekilde öğrenciler nerede hatalı olduklarını ve nasıl düzeltmeleri gerektiğini ders sonunda birbirlerinden öğrenebilir. Bunun için padlet etkinlik paylaşım linki öğrencilere ÖYS üzerinden paylaşılmalıdır.

C. Kısmi Öğrenme Görevleri

Süre: 60 dk.

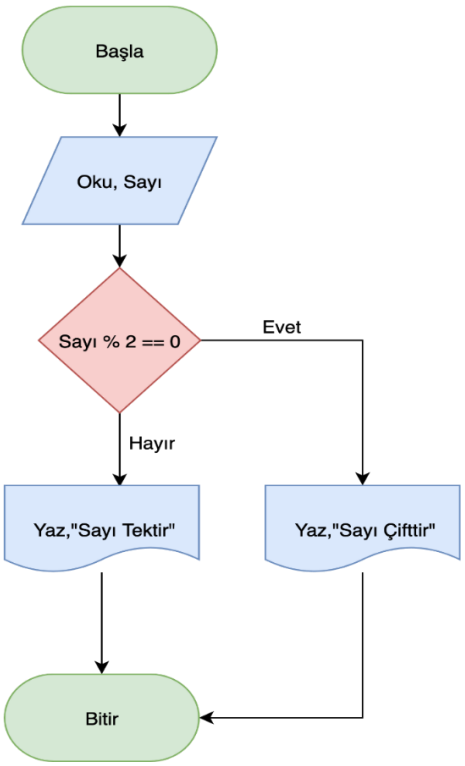
Materyal: O2. C. 1. Kısmi Öğrenme Görevleri Afişi

Hazırlık: Kısmi öğrenme görevleri afişi ÖYS ortamında öğrencilere süreli ödev olarak açılır.

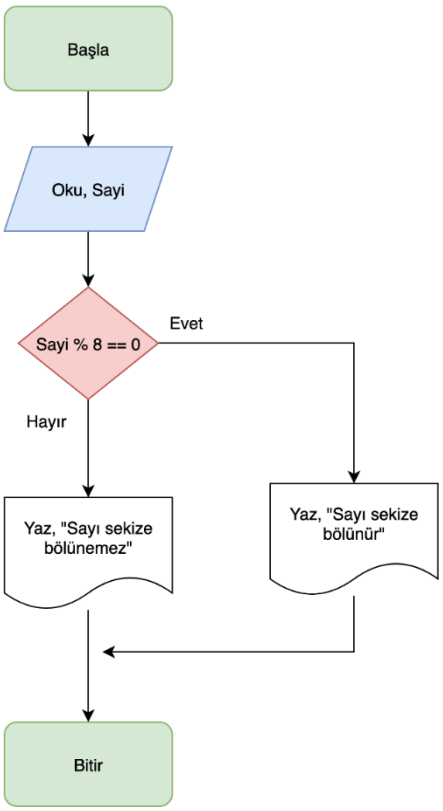
Uygulama: Öğrenciler afişteki görevleri istedikleri sırada ve sayıda kendi tercihlerine bağlı olarak verilen süre içinde uygulamaya başlar. Her bir görevi tamamlayan öğrencilere, göreve ilişkin beceri rozeti verilir. Öğrenciler görev seçimlerini ve tamamladıkları görevleri eğitmene bildirmelidir. Eğitmenler ihtiyaç duydukları zaman öğrencilere görevi tamamlama aşamasında anlık geri bildirimlerde bulunur. Öğrenciler, ÖYS üzerinden süreli ödev olarak açılan kısmi öğrenme görevlerinden kendi tercihlerine göre istedikleri sayıda görev tamamlayıp eğitmene iletir. Süre bitiminde eğitmen görevleri ve yanıtlarını ÖYS ya da GitHub üzerinden öğrencilere gönderir.

Kısmi Öğrenme Görevleri Yanıtlar: Görevler, öğrenciye verilecek beceri rozetleri ile isimlendirilmiştir. Her bir rozetin altında ilgili görevin yanıtları verilmektedir. Eğitmen bu yanıtları uygulama süresinin sonunda GitHub üzerinden öğrencilerle paylaşır.

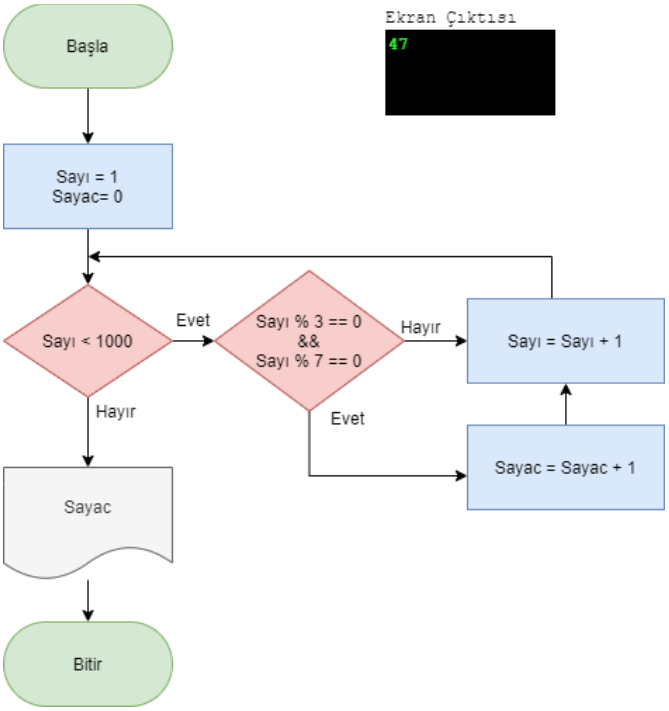
Tasarlayıcı 1: Aşağıdaki algoritma klavyeden girilen sayının çift olduğunu belirleyen programın akış şeması verilmiştir. Bu algoritma girilen sayının 8 ile bölünüp bölünmediğini gösterseydi, şema nasıl değişirdi?



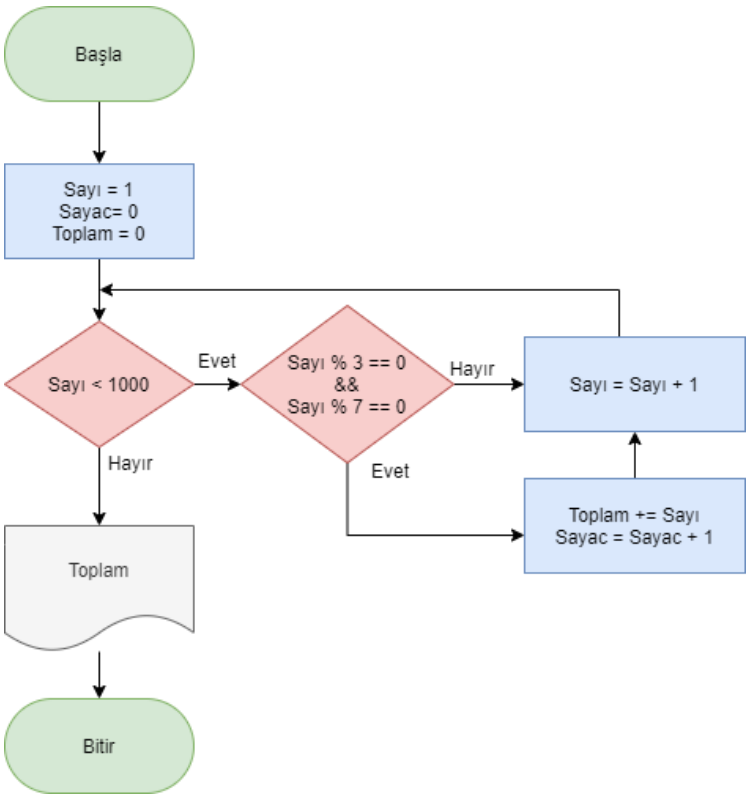
Yanıt:



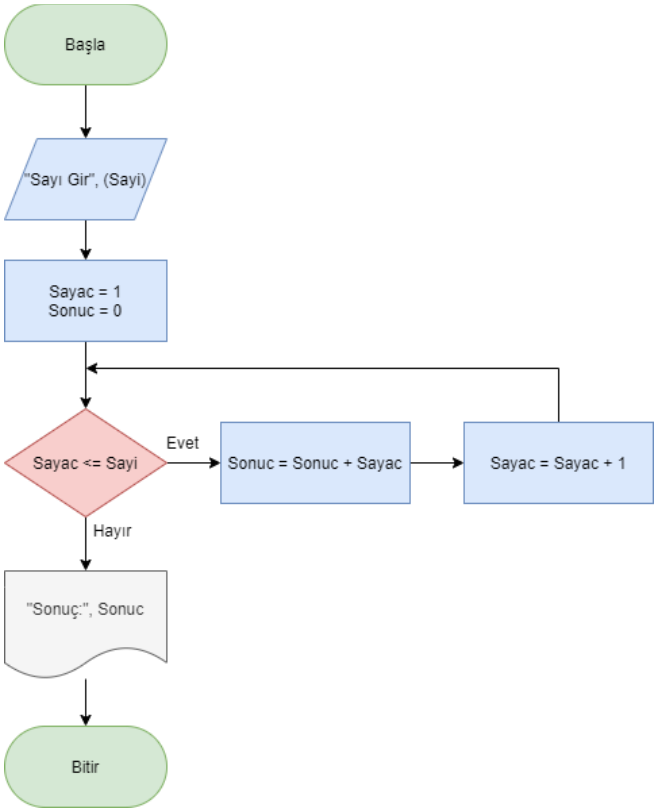
Tasarlayıcı 2: Aşağıdaki algoritma üç ve yedi ile tam bölünen 1000'den küçük sayıların toplamını yazdırıyor olsaydı, şema nasıl değişirdi?



Yanıt:

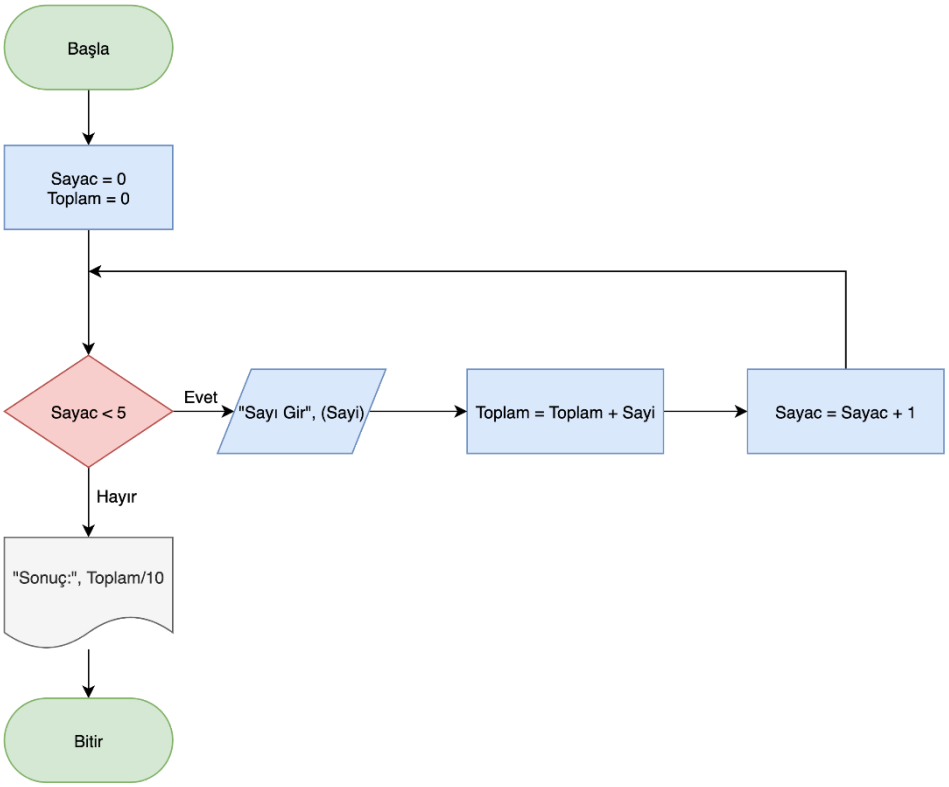


Analizci: Akış şeması verilen algoritmanın temel problemini tahmin edin. Sayı=5 için çıktı nedir?



Yanıt: 1'den Klavyeden girilen sayıya kadar olan tüm sayıları toplar. Sayı 5 için, Sonuc = 1 + 2 + 3 + 4 + 5 = 15 olur.

Denetleyici: Akış şeması verilen algoritmanın ekran çıktısını (4,6,8,4,2) için tahmin edin.



Yanıt: 4+6+8+4+2 = 24

Çıktı = 24/10 = 2.4 olur.

Hafta 2. Ders Materyalleri

O2. B1. 1. Doğru algoritma hangisi?

1. Başla
2. Sebzeleri hazırla
3. Tencereye suyu koy
4. Ocağı aç
5. Suyun kaynamasını bekle
6. Sebzeleri tencereye koy
7. Baharatları ekle
8. 10 dakika bekle
9. Ocağı kapat
10. Bitir

A

1. Başla
2. Sebzeleri hazırla
3. Tencereye suyu koy
4. Ocağı aç ve suyun kaynaması nı bekle
5. Sebzeleri tencereye koy
6. Baharatları ekle
7. 10 dakika bekle ve ocağı kapat
8. Bitir

B

1. Sebzeleri hazırla
2. Tencereye suyu koy
3. Ocağı aç
4. Suyun kaynamasını bekle
5. Sebzeleri tencereye koy
6. Baharatları ekle
7. 10 dakika bekle
8. Ocağı kapat

C

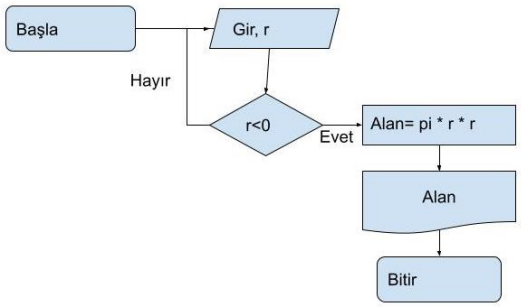
1. Sebzeleri hazırla
2. Tencereye suyu koy
3. Ocağı aç
4. Suyun kaynamasını bekle
5. Sebzeleri tencereye koy
6. Baharatları ekle
7. Dördüncü adıma geri dön
8. 10 dakika bekle
9. Ocağı kapat

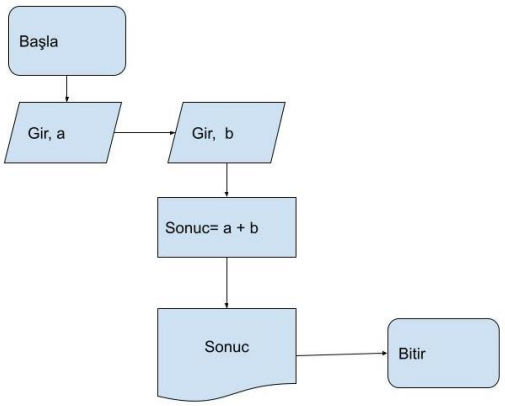
D

O2. B2. 1. Algoritmaları Eşleştirelim Çalışma Kâğıdı

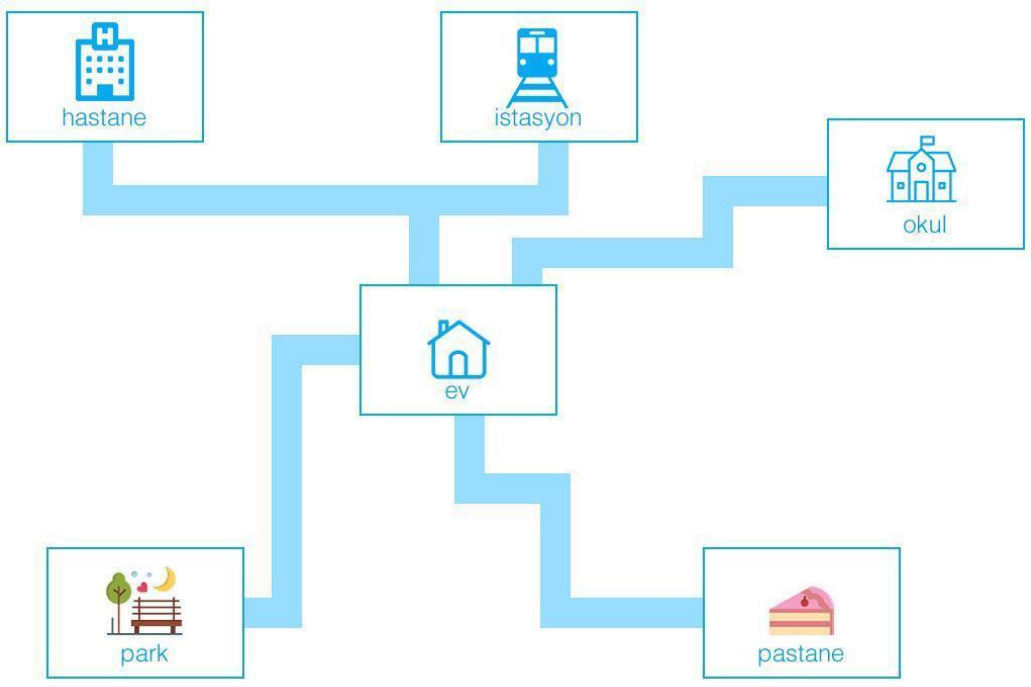
Aşağıda metin şeklinde yazımı, sözde kod ve akış diyagramı olmak üzere bir probleme üç farklı şekilde algoritma yazılmıştır. Hangi algoritma hangi probleme aittir, lütfen eşleştiriniz. Eşleştirme için aşağıdaki kısaltmalardan uygun olanı kutucuk altına yazınız.

Problemler	**Algoritma İfade Şekli**
P1. Çemberin alanını hesaplama	MY: Metinsel Yazım
P2. İki sayıyı toplama	SK: Sözde Kod
	AD: Akış Diyagramı

<table>
<tr>
<td>1. Başla
2. Birinci sayıyı gir (a)
3. İkinci sayıyı gir (b)
4. İşlemi yap.
(sonuc = a+b)
5. Ekrana yaz (sonuc)
6. Bitir.</td>
<td>1. Başla.
2. Gir, r
3. Eğer r < 0 ikinci adıma git.
4. $alan = \pi \times r \times r$
5. Yaz, Sonuc
6. Bitir.</td>
</tr>
<tr>
<td>Problem:
Algoritma İfadesi:</td>
<td>Problem:
Algoritma İfadesi:</td>
</tr>
<tr>
<td> 
</td>
<td>1. Başla.
2. Yarıçap gir (r).
3. Eğer yarıçap sıfırdan küçükse ikinci adıma git.
4. Alanı hesapla ($alan = \pi \times r^2$)
5. Sonucu ekrana yazdır.
6. Bitir.</td>
</tr>
<tr>
<td>Problem:
Algoritma İfadesi:</td>
<td>Problem:
Algoritma İfadesi:</td>
</tr>
</table>

1. Başla 2. Gir, a 3. Gir, b 4. Sonuc = a+b 5. Yaz, Sonuc 6. Bitir.	
Problem: *Algoritma İfadesi:*	*Problem:* *Algoritma İfadesi:*

O2. B3. 1. Nereye Gidelim Haritası



Resim 10. *Nereye gidelim haritası*

O2. B3. 2. Görev Kartları

Görev

Evinden çıkıp okula gitmek isteyen Arda'nın, izlemesi gereken yolu tarif eden bir algoritma oluştur.

Görev

Evinden çıkıp hastaneye gitmek isteyen Karaca'nın, izlemesi gereken yolu tarif eden bir algoritma oluştur.

Görev

Evinden çıkıp parka gitmek isteyen Defne'nin izlemesi gereken yolu tarif eden bir algoritma oluştur.

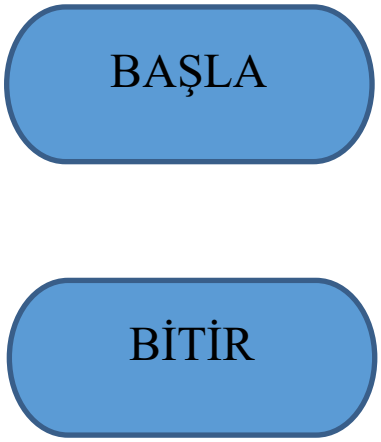
Görev

Evinden çıkıp pastaneye gitmek isteyen Güney'in, izlemesi gereken yolu tarif eden bir algoritma oluştur.

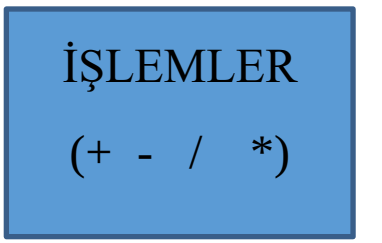
Görev

Hastaneden istasyona gitmek isteyen Karaca'nın izlemesi gereken yolu tarif eden bir algoritma oluştur.

O2. B3. 3. Akış Diyagramları Bilgi Afişi



Başla ve Bitir adımları yandaki şekildeki gibi gösterilir. Algoritmamızı meydana getirecek olan diğer bütün adımlar bu iki şekil arasına eklenir.

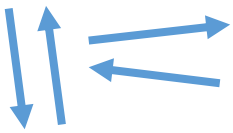


Toplama, çıkarma, aritmetik işlemlerin yapılması gerektiği durumlarda dikdörtgen kullanılır. Örneğin; üçgenin alanını hesaplama

Kullanıcı girişlerini şekilsel olarak göstermek için paralelkenar kullanılır. Örneğin; Kullanıcıdan istenen ad, soyad, yaş vb.)

Yukarıda kullanıcıdan alınan bilgileri ekrana yazdırılması gerektiğinde yandaki şekil kullanılır.

Algoritma akışında işlemler tek yönlü iken karar yapılarında iki farklı ok çıkabilir ve bazı



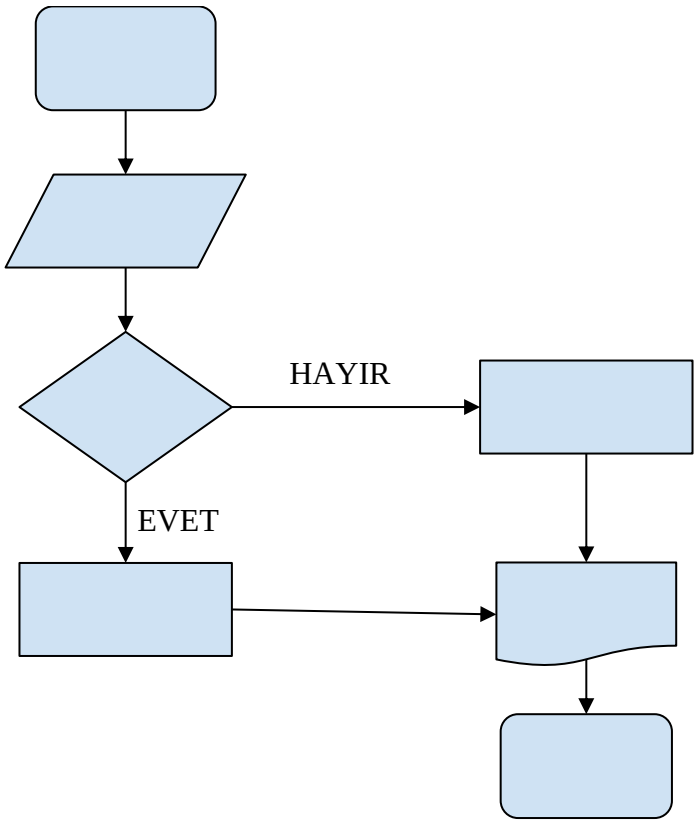
Yandaki oklar algoritmanın akış yönünü bize belirtir.

O2. B4. 1. Çıkartma Algoritması Çalışma Kâğıdı

Görev: Aşağıdaki problemin çözümüne yönelik sözde kodu verilen çıkartma algoritmasını doğru şekilde sıralayın ve akış diyagramındaki boşluklara yazın.

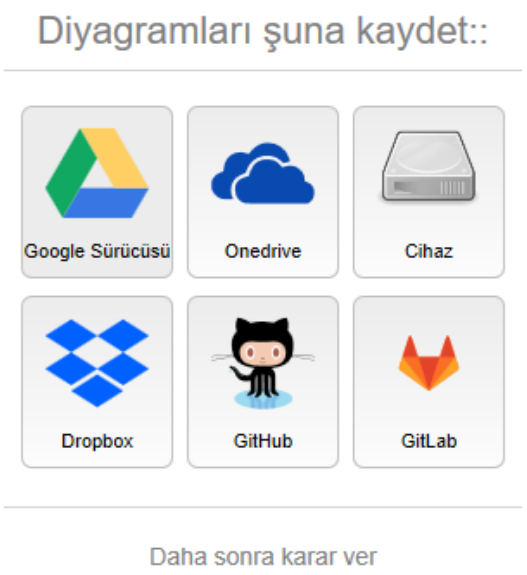
Problem: Klavyeden girilen iki sayı arasında küçük olanı büyük olandan çıkartma işlemini yazınız.

Başla	Değilse, Sonuc=Sayi2 - Sayi1	Gir, Sayi1, Sayi2	Eğer Sayi1 > Sayi2
Yaz, Sonuc	Sonuc=Sayi1 - Sayi2	Bitir	



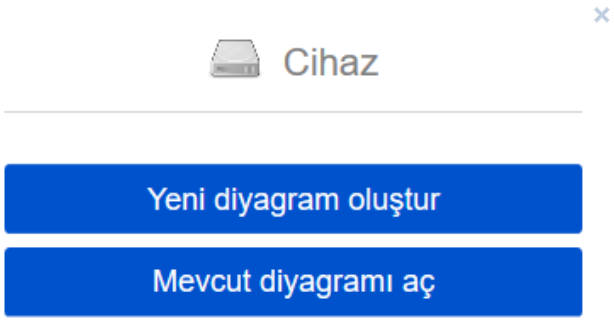
O2. B4. 2. Draw.io program kullanım kılavuzu

Akış diyagramını bilgisayar ortamında çizmek için draw.io isimli websitesi kullanacağız. Ücretsiz olan bu uygulama sayesinde çizdiğimiz şekilleri kaydedebilir, istediğimiz kişilerle paylaşabiliriz. Draw.io isimli uygulamaya ilk giriş yaptığımızda aşağıdaki görüntü karşımıza gelecektir. Çizdiğimiz diyagramı nereye kaydetmek istediğimizi belirtiyoruz. Cihaz seçeneğini tıklayarak, şimdilik kendi bilgisayarımızda kaydedelim.



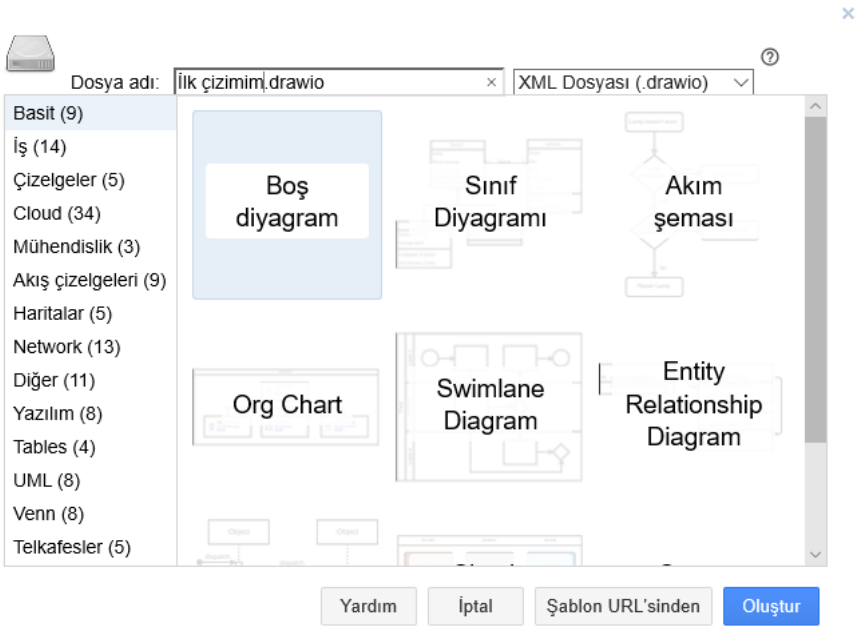
Resim 11. *Draw.io giriş ekranı*

Ardından, yeni diyagram oluştur butonuna tıklayarak boş bir sayfa oluşturalım.



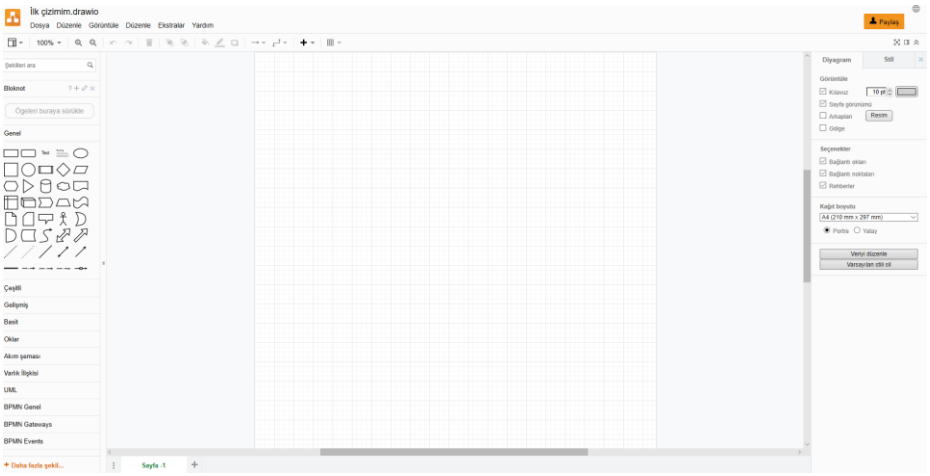
Resim 12. *Draw.io yeni diyagram oluşturma ekranı*

Draw.io ile basitten profesyonel seviyeye kadar birçok seviyede çizim yapılabilir. Biz şimdilik, boş diyagram seçeneğini seçerek bir isim verebiliriz.



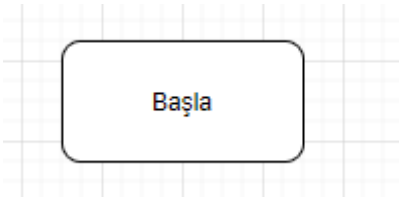
Resim 13. *Draw.io yeni diyagram tipi seçme ekranı*

Karşımıza aşağıdaki çizim ekranı gelecektir.



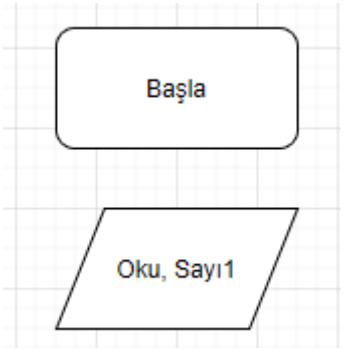
Resim 14. *Draw.io çizim ekranı*

Bu aşamada **Genel** isimli araç kutusunda öğrendiğimiz bloklar mevcuttur. Eklemek istediğimiz bloğu çizim alanına sürekle bırak yapıyoruz. Ardından blok üzerine çift tıklayarak içerisine yazı ekliyoruz.



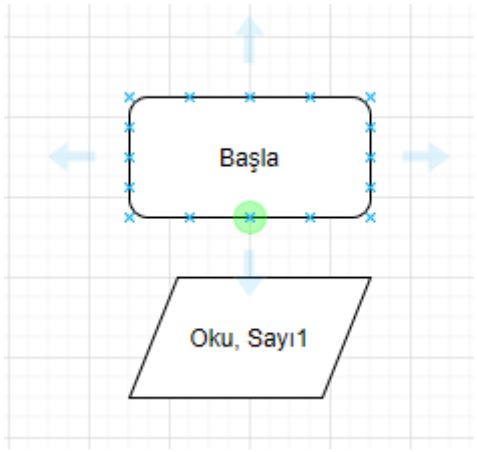
Resim 15. *Genel bloğu*

Ardından bir tane de paralel kenar ekleyelim.



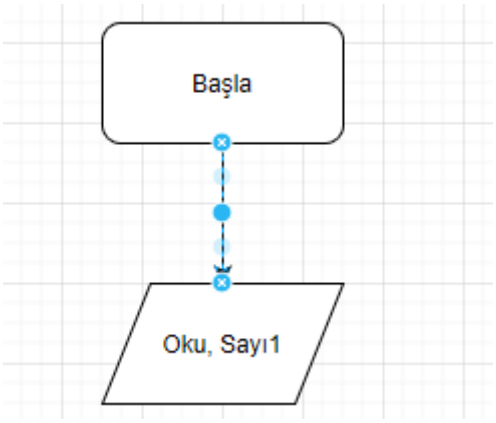
Resim 16. *Paralel kenar bloğu*

İki bloğu birbirine bağlamak için ilk bloğun üzerine fareyi getiriyoruz. Fare üzerine geldiğinde, çizgi çizebileceğimiz noktaları görüyoruz.



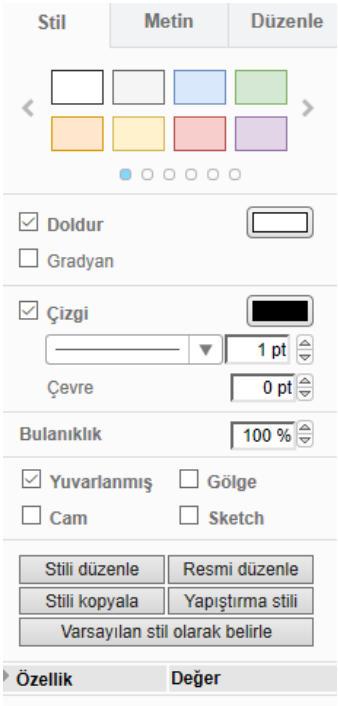
Resim 17. *Blokları bağlama öncesi*

Ardından yine fare yardımıyla, hangi sonraki blok ile bağlantıyı tamamlıyoruz.



Resim 18. *Blokları bağlama sonrası*

Eğer bloğun renklerini değiştirmek istersek sağ paneli kullanabiliriz.

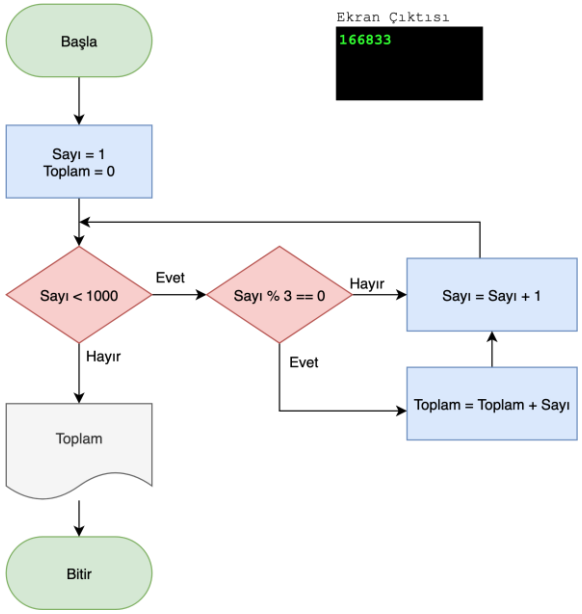


Resim 19. *Blok özellikleri ekranı*

O2. B5. 1. Grup görevleri

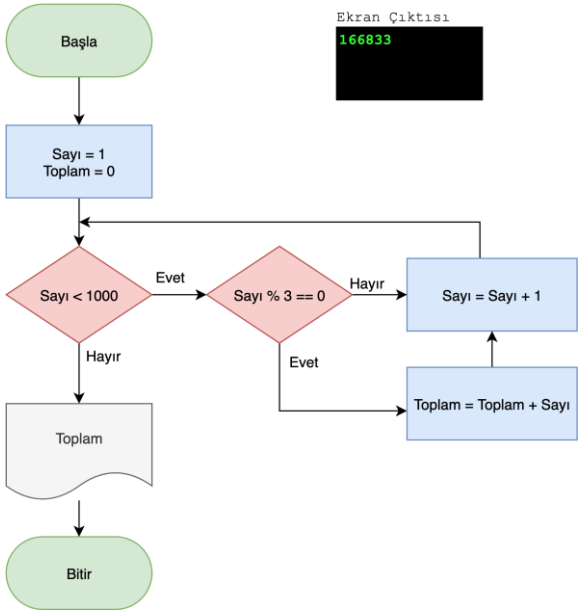
Görev

Yanda verilen akış şemasında Sayı=Sayı + 1 basamağı çıkarılırsa, algoritmanın temel problemi nasıl değişir?



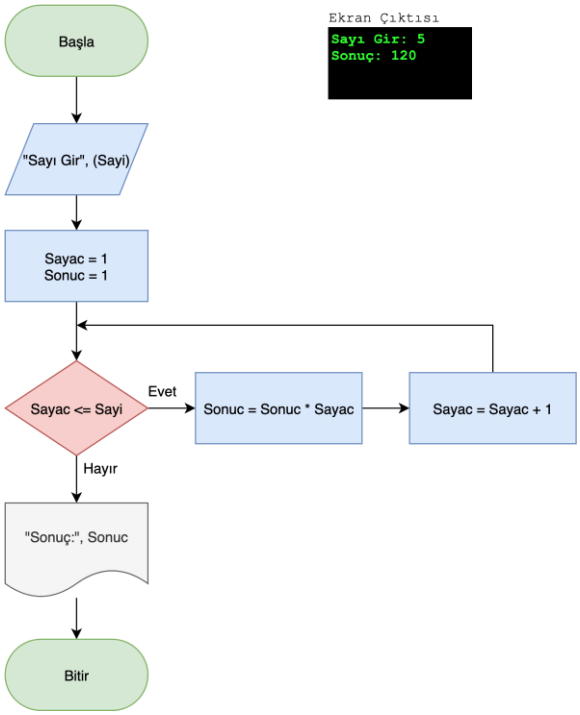
Görev

Yanda verilen akış şemasında Sayı %3 == 0 olan basamak Sayı %9 == 0 ile değiştirilirse, algoritmanın temel problemi nasıl değişir?



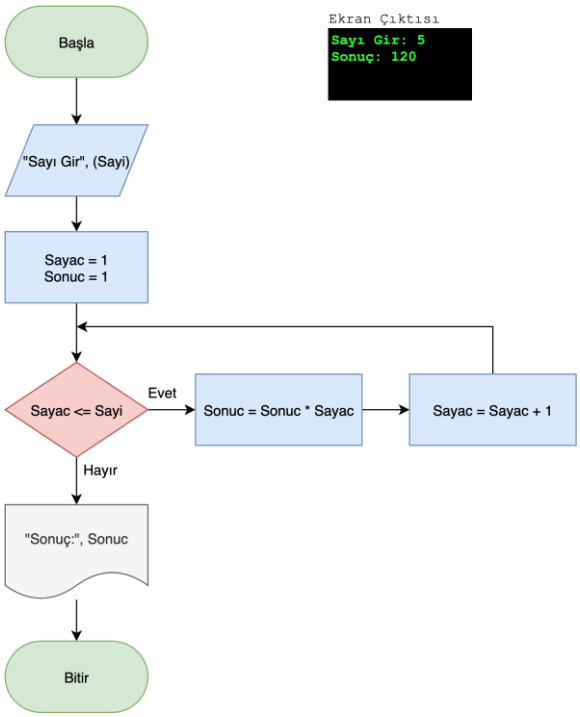
Görev

Yanda verilen akış şemasına ait programda Sayac <= Sayi basamağı çıkarılırsa program çıktısı nasıl değişir?



Görev

Yanda verilen akış şemasına ait programda girilen sayı 4 olsaydı, algoritmanın ekran çıktısı nasıl değişirdi?



O2. C. 1. Kısmi Öğrenme Görevleri Afişi

Kısmi öğrenme görevleri afişine ulaşmak için buraya tıklayınız.

Hafta 3. Algoritmada Değişkenler ve Değerleri

Kazanımlar

K1. Verilen algoritmada temel kavramları analiz eder.

K2. İhtiyaç duyulan algoritmayı tasarlar.

K3. Bir problemin çözümünde değişkenleri kullanır.

K4. Verilen algoritmanın akış diyagramını sözde koda dönüştürür.

K5. Verilen algoritmanın akış diyagramındaki değişkenleri ayırt eder.

Amaç

Haftanın amacı bir problemin çözümü için tasarlanan akış diyagramlarında kullanılan değişkenler ve değişkenlerin değerlerini keşfetmektir.

Önerilen Ders Akışı

A. Giriş: Sona Kalan Kazanır! (10 dk.)

B. Öğrenme Görevleri

B1. Algoritma Terimlerini Keşfet! (25 dk.)

B2. Otomatik Park Etme (25 dk.)

- ❏ Ders Arası (5 dk.)
- ❏ Motivasyon Oyunu (10 dk.)

B3. Değişkenler Değerlidir! (25 dk.)

B4. Sözde Kod Yazalım (25 dk.)

- ❏ Ders Arası (10 dk.)

B5. Algoritmayı Test Edelim (30 dk.)

- ❏ Ders Arası (5 dk.)
- ❏ Motivasyon Oyunu (10 dk.)

C. Kısmi Öğrenme Görevleri (60 dk.)

A. Giriş: Sona Kalan Kazanır!

Süre: 10 dk.

Uygulama: Eğitmen tüm öğrencileri oldukları yerde ayağa kalkmalarını ister. Basit sorular soracağını ve bu sorular için cevabı "evet, o benim" diyenlerin yerlerine oturmaları gerektiğini, en son ayakta kalanın ise oyunu kazanacağını belirtir. Ayakta kalan son öğrenciye sürpriz olarak küçük bir hediye verilir. Bu buz kırıcı teknik, basitten karmaşığa sorular sorularak uygulanabilir. Örneğin: Kimin kız kardeşi var? / Kim Galatasaraylı? / Kim eylül ayında doğdu? / Kimin gözleri siyah? / Kimin kolunda saat var?

Eğitmene Öneriler: Süreye bağlı olarak sorular çeşitlendirilip arttırılabilir.

B. Öğrenme Görevleri

B1. Algoritma Terimlerini Keşfet!

Süre: 25 dk.

Kazanımlar: K1. Verilen algoritmada temel kavramları analiz eder.

Materyaller: O3. B1. 1. Çiftleri Toplama Algoritması

O3. B1. 2. Algoritma Terimleri Görev Kartları

Hazırlık: Birinci materyalden de 10 adet çıktı alınır. İki öğrenciye bir tane gelecek şekilde sınıfta dağıtılır. Görev kartları ise beş adet çıktı alınarak kesilir ve her gruba karışık şekilde dağıtılır.

Uygulama: Öğrenciler dörderli gruplar hâlinde çalışmaktadır. Bu etkinlikte öğrencilerin tanımlayıcı, değişken, sabit, sayaç, döngü, atama/aktarma ve ardışık toplama/çıkarma terimlerini öğrenmeleri beklenir. Eğitmen öğrencilerden çiftleri toplama algoritmasının akış şemasındaki boşlukları doldurmalarını ister. Doğru algoritmaya eriştikten sonra eğitmen görev kartlarından ilkini Tanımlayıcı görev kartını sınıfta uygular. Diğer görev kartlarını ise gruplara ikişer tane olacak şekilde rastgele dağıtır ve aşağıdaki talimatı verir. Etkinlik sonunda eğitmen aşağıdaki konu içeriğini yapılan görevler üzerinden özetler.

Çiftleri toplama algoritması üzerinde "algoritma terimleri görev kartlarını" uygulayın.

Konu İçeriği:

1) Tanımlayıcı: Algoritmadaki değişkenleri, sabitleri, kayıt alanlarını ve özel bilgi tiplerini adlandırmak programcı tarafından gerçekleştirilen işlemlerdir. Tanımlayıcıların yerini tuttukları ifadelere çağrışım yapacak şekilde adlandırılmaları algoritmanın anlaşılması açısından önemlidir. Tanımlayıcı isimlerinde İngiliz alfabesindeki "A-Z" ve "a-z" arası harfler, "0-9" arası rakamlar, sembollerden alt çizgi "_" kullanılabilir. Bununla birlikte tanımlayıcı ismi, rakamla başlayamaz veya sadece rakamlardan oluşamaz.

2) Değişken: Algoritmanın her çalıştırılmasında, girdiğimiz değerleri alan veya programın çalışmasıyla bazı değerlerin atandığı bellek alanlarıdır. Değişken

isimlendirilmeleri, yukarıda sayılan tanımlayıcı kurallarına uygun biçimde yapılmalıdır. Örneğin bir ismin aktarıldığı değişken "ad" olarak, bir isim ve soy ismin aktarıldığı değişken "adSoyad", ev telefon numarasını aktarıldığı değişken "evTel", ev adresinin aktarıldığı değişken "evAdres" ve iş adresinin aktarıldığı değişken "isAdres" olarak tanımlanabilir. Burada özellikle İngiliz alfabesini kullandığımıza dikkat edelim.

3) Sabit: Uygulamanın çalıştığı süre boyunca, içeriği sabit olan değer ve ifadelerin saklanması için kullanılır. Algoritma boyunca kolay takip edilebilmesi için kullanılmaktadır. Değişkenler gibi isimlendirme kurallarına uygun olarak oluşturulmalıdırlar.

4) Atama/Aktarma: Bir değişkene değer atamak için gerçekleştirilen işlemdir. Algoritma adımlarında sağdaki değeri soldaki değişkene atamak için kullanılan bir işlemdir.

5) Sayaç: Algoritma tasarımlarında bazı işlemlerin belirli sayıda yaptırılması ve bu süreçte oluşan değerlerin sayılması gerekebilir. Bu amaçla kullanılan sayma işlemlerine sayaç denir.

6) Döngü: Algoritma tasarımlarında bir veya birden fazla işlem satırını, bir koşula bağlı olarak, belirli sayıda veya bir koşul sağlandığı sürece tekrarlayarak çalıştıran kalıplardır. Örneğin 1 ile 100 arasındaki çift sayıların toplamını hesaplayan programda T=2+4+6 ….+100 hesabını teker teker yapmak yerine 1 ile 100 arasında ikişer artan bir döngü kullanmak ve döngü değişkenini toplam hesabında kullanmak daha uygun olacaktır.

7) Ardışık Toplama/Çarpma: Algoritmalarda var olan değere yeni bir değerin eklenmesi ya da var olan değerin yeni bir değerlerle çarpılarak oluşan bu yeni değerin kullanılması işlemidir.

B2. Otomatik Park Etme

Süre: 25 dk.

Kazanımlar: K2. İhtiyaç duyulan algoritmayı tasarlar.

Materyaller: O3. B2. 1. Otomatik Park Etme Algoritması

Hazırlık: Eğitmen iki nokta oluşturan öğrencileri birleştirerek dörderli gruplar oluşturur.

Materyalin beş adet çıktısı alınır.

Uygulama: Öğrenciler dörderli gruplar hâlinde çalışır. Eğitmen öğrencilerden "Otomatik park etme algoritması" üzerinde aşağıdaki problemleri incelemelerini ve algoritmayı tekrar düşünmelerini ister.

1. Otoparkta boş yer yok ise, algoritma nasıl çalışacaktır? Grup içinde tartışın ve grup kâğıdına yazın.
2. Algoritmanın daha iyi çalışması için basamaklarda nasıl bir değişiklik yapılmalıdır? Değiştirilen algoritma önerinizi grup kâğıdına yazınız.

Eğitmen birinci görevin grup içinde tartışılmasının ardından, grupların saat yönünde yer değiştirmelerini ister. Yeni grup, bir önceki grubun grup kâğıdında tamamlanan görevi inceleyip kontrolünü gerçekleştirir. Ardından önceki grubun kâğıdından ikinci göreve devam edilir. İki görevin tamamlanmasının ardından sınıfta beyin fırtınası yapılarak algoritmalarda "Sonsuz döngü" problemi aşağıdaki gibi özetlenir.

> *Akıllı araç ilk sokağa geldiğinde yer olmadığı (sensörler yardımıyla) algılayacak ve sonraki sokağa kadar ilerleyecektir. İkinci sokakta P2 ve P3 alanlarının boş olması durumunda P2 alanına aracını park edecektir. Eğer otoparkta boş yer yok ise, 2. ve 3. adımlarda **sonsuz döngü** dediğimiz istenmeyen durumla karşılaşılacaktır. Bu durumu da göz önüne alarak, otoparkın sonuna gelindiğinde akışın sonlanmasını istiyorsak aşağıdaki şekilde yazabiliriz.*
>
> ***Algoritmayı Sonsuz Döngüden Kurtarma:***
>
> 1. *Başla*
> 2. *Sonraki sokağa kadar ilerle.*
> 3. *Eğer otoparkın sonuna geldiysen 7. adıma git.*
> 4. *Eğer sokakta yer yoksa 2. adıma git.*
> 5. *Sokağa gir.*
> 6. *İlk uygun yere park et.*
> 7. *Bitir.*

Eğitmene Öneriler: Eğitmen öğrencilere algoritmadaki problemi keşfetmeleri için ipuçları kullanabilir. Örneğin; "Basamakları tek tek çalıştırmayı dene"; "İkinci basamağın tekrar çalıştırmayı düşünebilirsin"; "son basamağın nasıl çalıştığına dikkat et" vb. gibi ifadelerle problemin kaynağını doğrudan öğrenciye aktarmaktan kaçınılmalıdır.

B3. Değişkenler Değerlidir!

Süre: 25 dk.

Kazanımlar: K3. Bir problemin çözümünde değişkenleri kullanır.

Materyaller: O3. B3. 1. Örnek Kod Çalıştırma Tablosu

Hazırlık: Materyal ÖYS üzerinden materyal olarak öğrencilere yayınlanır.

Uygulama: Eğitmen öğrencilere aşağıdaki örnek olayı verir. Öğrenciler ikili gruplar hâlinde çalışmaktadır.

Örnek Olay: Arkadaşınız aklından iki sayı tutmuştur ve sizden bu iki sayıdan büyük olanı bulmanızı istiyor. Bunun için bir algoritma yazmak istiyorsunuz.

Eğitmen ilgili algoritmanın sözde kodunu, değişken, değişken değerleri ve ekran çıktısını içeren aşağıdaki talimatları verir.

1. Örnek olaya ilişkin algoritma için değişkenleri ve değerlerini ayırt edin.
2. Bu değişkenleri kullanarak akış şemasını çizin ve sözde kodları oluşturup sıralayın.

Önceki iki görev iki kişilik gruplarda yapıldı. Şimdi iki grup birleşerek dörderli yeni gruplar oluşturun ve elinizdeki kodları birlikte kontrol edip, kâğıt üzerinde adım adım çalıştırmayı deneyin.

Etkinlik sonunda eğitmen öğrencilerden Örnek Kod Çalıştırma Tablosu'nu ÖYS üzerinden açıp incelemelerini ister. Eğitmen değişkenler ve değerleri üzerine konuyu özetler. Bunu yaparken eğitmen her bir adımın açıklaması için aşağıdaki tabloyu kullanılır.

Adım	Açıklama
1) Başla	Programın Başlangıcı
2) Oku, (Sayi1,Sayi2)	Kullanıcıdan değerlerin alınması. Burada biz klavyeden alındığını düşünüyoruz. Eğer bir mobil cihaz üzerinde ya da web sitesi üzerinde çalışacaksa farklı teknikler kullanılabilir.
3) Eğer Sayi1>=Sayi2	**Sayi1**, **Sayi2**'den büyük eşit olup olmadığı kontrol ediliyor. Eğer bu koşul doğru ise bir alt satıra devam edeceğiz. Değilse aksi takdirde yazan satıra gideceğiz. Eğer aksi takdirde satırı yok ise bir alt adımdan devam edeceğiz.
3.1) Yaz, Sayi1	Ekrana **Sayi1**'in değerini yaz.
4) Aksi takdirde	**Sayi1, Sayi2**'den büyük değilse
4.1) Yaz, Sayi2	Ekrana **Sayi2**'nin değerini yaz
5) Bitir.	Programın Sonu

Eğitmene Öneriler: Eğitmen bu etkinlikte değişken ve değişken değerlerine dikkat çekmelidir. Bunun için aşağıdaki bilgileri de ekleyebilir.

Değişken zamanla değeri değişen veri saklayıcıdır. Örneğin Yaş bilgisi değişkendir. Yaşınızın kaç olduğu ise değerdir. Bilgisayar ortamında değişkenler hafızadan yer ayırmak için kullanılır. Daha sonra içerisine değerler yazarak kullanılır. İçerisindeki değerleri program çalıştığı süre boyunca unutmazlar ve siz değiştirmediğiniz sürece değişmezler. Algoritma ya da program yazarken genellikle daha kısa ve anlamlı isimler veririz. Değişken ve değerlere günlük yaşamdan örnekler verirsek;

Değişken	***Değişken İsmi***	***Değer***
Adınız	*Ad*	*“Ahmet”, “Mehmet”, “Ali”,”Ayşe”*
Yaşınız	*Yas*	*“12”,”13”,”25”*
Boyunuz	*Boy*	*“150 cm””,”140 cm”*
Telefon numaranız	*Tel*	*“05555555555”*
Sınıfınız	*Sinif*	*“4”,”5”,”6”,”7”*

B4. Sözde Kod Yazalım!

Süre: 25 dk.

Kazanımlar: K4. Verilen algoritmanın akış diyagramını sözde koda dönüştürür.

K5. Verilen algoritmanın akış diyagramındaki değişkenleri ayırt eder.

Materyaller: O3. B4. 1. Akış Şemasını Doldurma

O3. B4. 2. Sözde Kod Çalıştırma Tablosu

Hazırlık: Öğrencilerin ikişerli gruplar hâlinde oturmaları sağlanır. Ders materyali ÖYS üzerinden öğrencilere sunulur.

Uygulama: Eğitmen gruplara bir örnek olay verir ve “Akış Şemasını Doldurma” kâğıdını dağıtır.

> *Örnek Olay: Bir okuldaki 10 öğrencinin sınav notunun ortalamasını hesaplamak istiyorsunuz. Bunun için akış şemasındaki boş yerleri doldurarak algoritmayı tasarlayınız.*

Öğrenciler akış şemasını doldurduktan sonra gruplar saat yönünde birbirleriyle yer değiştirir. Yeni gelen grup, önceki grubun akış şemasındaki sözde kodları kontrol ederek gerektiğinde düzenleme yapar. Daha sonra gruplardan sözde kod çalıştırma tablosu oluşturmaları istenir. Bunu yaparken algoritmanın her bir adımını basamak basamak çalıştıracaklardır. Oluşturulacak tablo aşağıdaki gibidir:

Tablo 10. *Sözde kod çalıştma tablosu*

Adım	Değişken değerleri	Ekran Çıktısı
Başla		
Sayac=0, Toplam=0	Sayac=1 Toplam=0	
Eğer 0 < 10	Doğru, Evet	
Oku, Sayi	Sayi=1	
Toplam = 0 + 1	Toplam = 1	

Sayac = 0 + 1	Sayac =1	
Eğer 1 < 10	Doğru, Evet	
Oku, Sayi	Sayi=2	
Toplam = 1 + 2	Toplam = 3	
Sayac = 1 + 1	Sayac =2	
Eğer 2< 10	Doğru, Evet	
Oku, Sayi	Sayi=3	
Toplam = 3 + 3	Toplam = 6	
Sayac = 2 + 1	Sayac =3	
Eğer 3 < 10	Doğru, Evet	
Oku, Sayi	Sayi=4	
Toplam = 6 + 4	Toplam = 10	
Sayac = 0 + 1	Sayac =4	
…	**Girilen Sayılar: 1,2,3,4,5,6,7,8,9,10**	
Eğer 9 < 100	Doğru, Evet	
Oku, Sayi	10	
Toplam = 45 + 10	Toplam = 55	
Sayac = 9 + 1	Sayac =10	
Eğer 10 < 10	Yanlış, Hayır	
Yaz, 55 / 10		5.5
Bitir	Sayac = 10 Toplam=55	

Tablonun ardından öğrencilerin ikinci kez yer değiştirmeleri istenir. Gruplar şimdi de kendi oluşturdukları tablo ile önceki grubun tablosunu karşılaştıracaklardır. Etkinlik sonunda eğitmen hangi basamakta hangi değişkenleri ve değişken değerlerinin programda nasıl çıktı oluşturduğuna dikkat çeker.

Eğitmene Öneriler: Bu etkinlikte eğitmen örnek olayın sözde kodunu aşağıdaki açıklamaları yaparak özetleyebilir. Özetleme sırasında kod içindeki değişkenlere ve değişken değerlerine vurgu yapılır.

Tablo 11. *Sözde kod çalıştma tablosu*

Adım	Açıklama
1) Başla	Programın Başlangıcı
2) Sayac=0, Toplam=0	Burada bize iki adet değişken gerekli. Kaç adet sayı girildiğini **Sayac** isimli değişkende tutuyoruz. İstediğimiz sayıların toplamını ise **Toplam** isimli değişkende tutuyoruz.
3) Sayac<10 Olduğu Sürece	Buradaki koşulumuz 10 değerine ulaşıp ulaşmadığımız. **Sayac** değişkenimiz 10'a ulaşana kadar devam edeceğiz.
3.1) Oku, Sayi	Klavyeden sayi al. Sayi isimli değişkene at.
3.2) Toplam=Toplam+Sayi	Toplam değişkenine bu sayiyi ekle.
3.3) Sayac= Sayac+1	Elimizdeki Sayac değerini bir arttırıyoruz.
4) Yaz, Toplam/10	Ekrana Toplam/10 değerini yazdırıyoruz.
5) Bitir.	Programın sonu

B5. Algoritmayı Test Edelim?

Süre: 30 dk.

Kazanımlar: K4. Verilen algoritmanın akış diyagramını sözde koda dönüştürür.

K5. Verilen algoritmanın akış diyagramındaki değişkenleri ayırt eder.

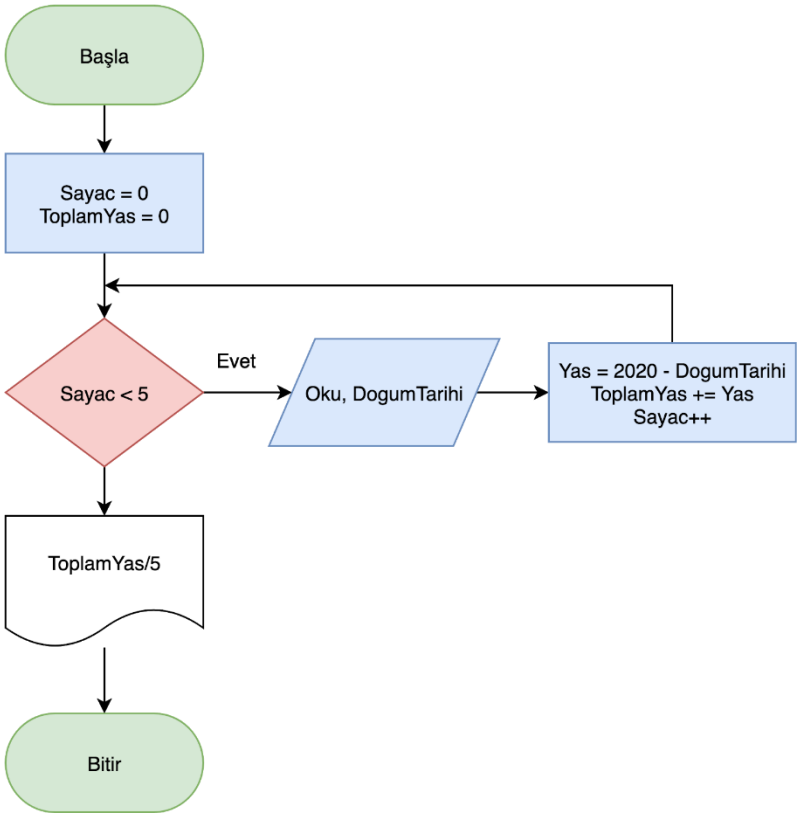
Materyaller: O3. B4. 2. Sözde Kod Çalıştırma Tablosu

Hazırlık: Draw.io programı öğrenci bilgisayarlarında kurulu olmalıdır.

Uygulama: Eğitmen öğrencilere aşağıdaki görevi verir. Öğrenciler ikişerli gruplar hâlinde çalışır.

> *Görev 1: Sevdiğiniz 5 kişinin doğum tarihlerini girerek yaşlarının ortalamasını bulan bilgisayar programı için akış diyagramı ve sözde kodunu Draw.io programını kullanarak bilgisayarda çiziniz.*

Öğrencilerin çizmesi gereken akış şeması aşağıdaki gibi olmalıdır:



Eğitmen ikinci göreve geçmeden önce iki grubu birleştirerek dörderli yeni gruplar oluşturur ve aşağıdaki talimatı verir.

> *Görev 2: Çizdiğiniz akış şemasındaki sözde kodları karşılaştırın ve adım adım çalıştırıp, test etmek için Sözde Kod Çalıştırma tablosu oluşturun.*

Öğrenciler bir önceki öğrenme görevi olan B4 etkinliğindekine benzer bir sözde kod çalıştırma tablosu hazırlayacaktır. Eğitmen bu materyalin aynısını kopyalayarak tekrar ÖYS üzerinden öğrencilerin erişimine açabilir. Hazırlanacak sözde kod çalıştırma tablosu aşağıdaki gibi olmalıdır:

Tablo 12. *Sözde kod çalıştma tablosu*

Adım	Değişken değerleri	Çıktı
1) Başla	Programın Başlangıcı	
2) Sayac=0, ToplamYas=0	Sayac=0 ToplamYas=0	
3) Eğer 0<5	Doğru	
3.1) Oku, DogumTarihi	DogumTarihi=2001	
3.2) Yas = 2020 - 2001	Yas = 19	
3.3) ToplamYas += Yas	ToplamYas = 19	

Adım	Değişken değerleri	Çıktı
3.4) Sayac ++	Sayac=1	
4) Eğer 1 <5	Doğru	
4.1) Oku, DogumTarihi	Doğum Tarihi=2003	
4.2) Yas = 2020 - 2003	Yas = 17	
4.3) ToplamYas += Yas	ToplamYas = 36	
4.4) Sayac ++	Sayac=2	
5) Eğer 2<5	Doğru	
5.1) Oku, DogumTarihi	DogumTarihi=2005	
5.2) Yas = 2020 - 2005	Yas = 15	
5.3) ToplamYas += Yas	ToplamYas = 51	
5.4) Sayac ++	Sayac=3	
6) Eğer 3<5	Doğru	
6.1) Oku, DogumTarihi	DogumTarihi=2007	
6.2) Yas = 2020 - 2007	Yas = 13	
6.3) ToplamYas += Yas	ToplamYas = 64	
6.4) Sayac ++	Sayac=4	
7) Eğer 4<5	Doğru	
7.1) Oku, DogumTarihi	DogumTarihi=2004	
7.2) Yas = 2020 - 2004	Yas = 16	
7.3) ToplamYas += Yas	ToplamYas = 80	
7.4) Sayac ++	Sayac=5	
8) Eğer 5<5	Yanlış	
9) Yaz, ToplamYas/5	80/5	16
10) Bitir.	Programın sonu	

Öğrencilerin oluşturduğu tablolar tahtaya asılır ve eğitmen tarafından tabloda sıralanan adımlar aşağıdaki gibi açıklamalarla özetlenir.

Tablo 13. *Sözde kod çalıştma tablosu*

Adım	Açıklama
1) Başla	Programın Başlangıcı
2) Sayac=0, ToplamYas=0	Burada bize iki adet değişken gerekli. Kaç kişinin bilgisi girildiğini **Sayac** isimli değişkende tutuyoruz. İstediğimiz sayıların toplamını ise **ToplamYas** isimli değişkende tutuyoruz.
3) Eğer Sayac<5	Buradaki koşulumuz 5 değerine ulaşıp ulaşmadığımız. **Sayac** değişkenimiz 5'e ulaşana kadar devam edeceğiz.
3.1) Oku, DogumTarihi	Klavyeden kişinin doğum yılını al.
3.2) Yas = 2020 - DogumTarihi	Yaşını hesapla
3.3) ToplamYas += Yas	Kişinin yaşını toplam yaş değerine ekliyoruz
3.4) Sayac ++	Elimizdeki Sayac değerini bir arttırıyoruz.
4) Yaz, ToplamYas/5	Ekrana ortalama yaş değerini (Toplam/5) değerini yazdırıyoruz.
5) Bitir.	Programın sonu

Eğitmene Öneriler: Eğitmen açıklama sırasında değişkenler ve değerlerine vurgu yapmalıdır. Öğrencilerin grup tablolarını online oluşturup, Öğrenme Yönetim Sisteminde paylaşmaları istenebilir.

C. Kısmi Öğrenme Görevleri

Süre: 60 dk.

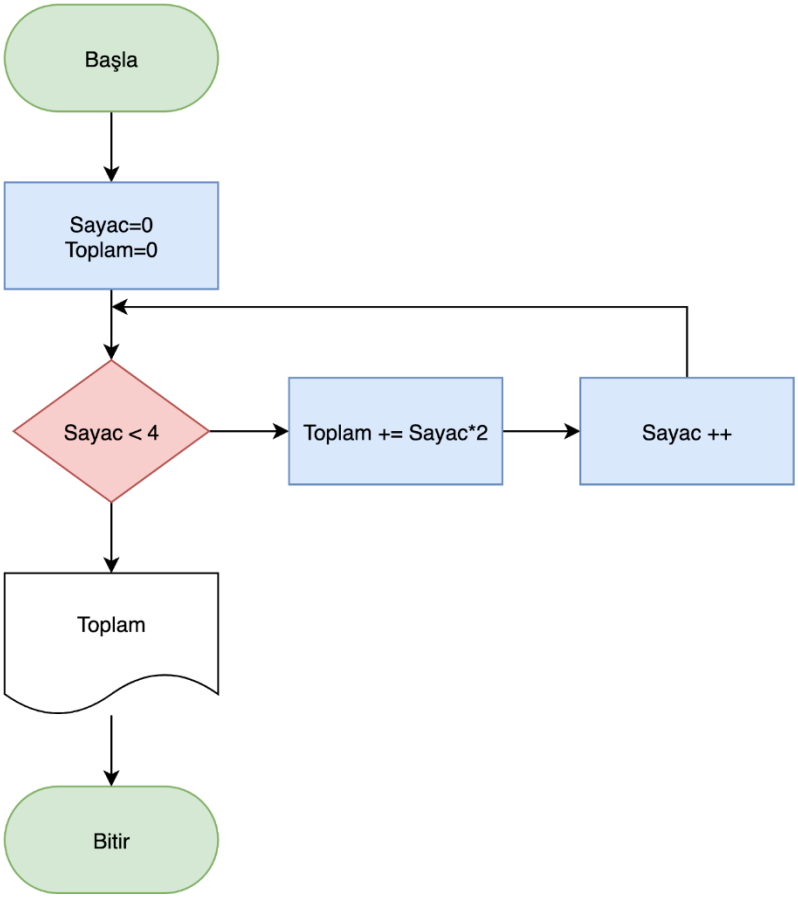
Materyal: O3. C. 1. Kısmi Öğrenme Görevleri Afişi

Hazırlık: Kısmi öğrenme görevleri afişi öğrencilere ÖYS ortamında süreli ödev olarak açılır.

Uygulama: Öğrenciler afişteki görevleri istedikleri sırada ve sayıda kendi tercihlerine bağlı olarak verilen süre içinde uygulamaya başlar. Her bir görevi tamamlayan öğrencilere, göreve ilişkin beceri rozeti verilir. Öğrenciler görev seçimlerini ve tamamladıkları görevleri eğitmene bildirmelidir. Eğitmenler ihtiyaç duydukları zaman öğrencilere görevi tamamlama aşamasında anlık geri bildirimlerde bulunur. Öğrenciler, ÖYS üzerinden süreli ödev olarak açılan kısmi öğrenme görevlerinden kendi tercihlerine göre istedikleri sayıda görev tamamlayıp eğitmene iletir. Süre bitiminde eğitmen görevleri ve yanıtlarını ÖYS ya da GitHub üzerinden öğrencilere gönderir.

Kısmi Öğrenme Görevleri Yanıtlar: Görevler, öğrenciye verilecek beceri rozetleri ile isimlendirilmiştir. Her bir rozetin altında ilgili görevin yanıtları verilmektedir. Eğitmen bu yanıtları uygulama süresinin sonunda GitHub üzerinden öğrencilerle paylaşır.

Kodlayıcı: Akış diyagramını veren arkadaşın bu diyagramın sözde kod hâlini öğretmenine sunmak istiyor. Arkadaşına yardım etmek için sözde kodları oluşturur musun?



Cevap:

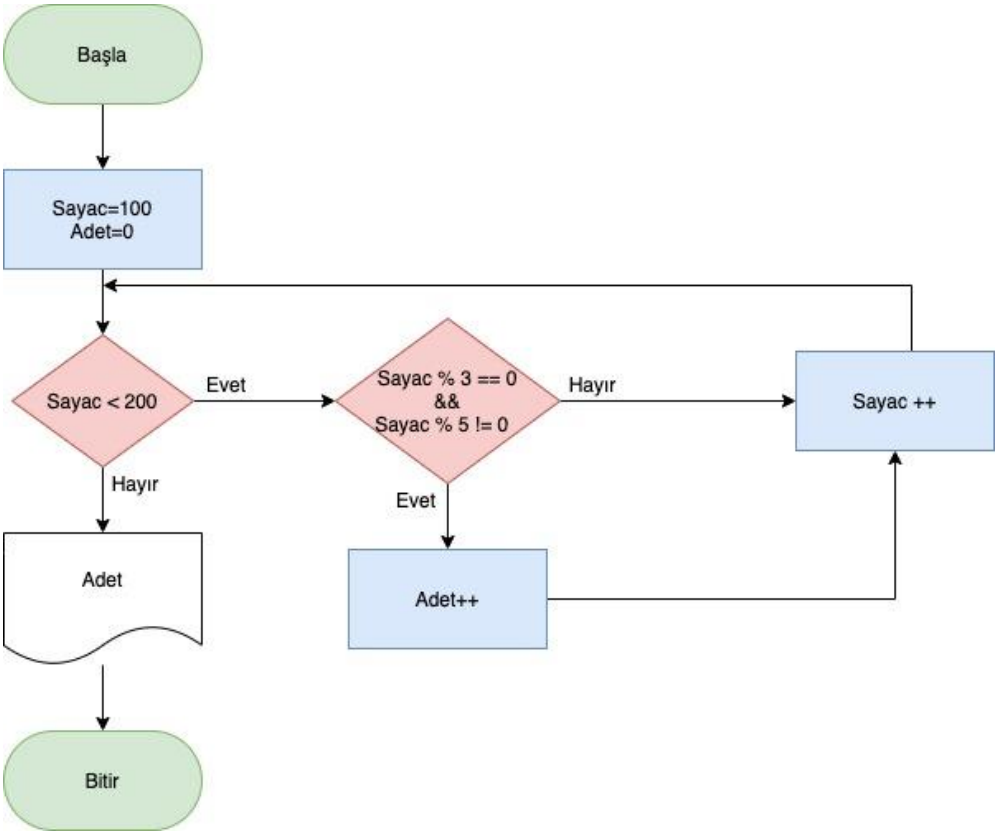
- Başla
- Sayac=0, Toplam=0
- Sayac < 4 Olduğu Sürece
    a) Toplam = Toplam + Sayac * 2
    b) Sayac ++
- Yaz, Toplam
- Bitir.

Analizci: Kodlayıcı rozetine ait görevde yer alan akış diyagramını kâğıt üzerinde adım adım çalıştıran tabloyu hazırlayıp, her bir değişkenin ekran çıktısını tahmin edebilir misin?

Tablo 14. *Sözde kod çalıştma tablosu*

Değişken	Değer	Ekran Çıktısı
Sayac=0, Toplam=0	Sayac=0, Toplam=0	
Eğer Sayac < 4	Doğru	
Toplam = Toplam + Sayac * 2	Toplam = 0 + 0*2 = 0	
Sayac++	Sayac= 1	
Eğer Sayac < 4	Doğru	
Toplam = Toplam + Sayac * 2	Toplam = 0 + 1*2 =2	
Sayac++	Sayac= 2	
Eğer Sayac < 4	Doğru	
Toplam = Toplam + Sayac * 2	Toplam = 2 + 2*2 =6	
Sayac++	Sayac= 3	
Eğer Sayac < 4	Doğru	
Toplam = Toplam + Sayac * 2	Toplam = 6 + 3*2 =12	
Sayac++	Sayac= 4	
Eğer Sayac < 4	Yanlış	
Yaz, Toplam		12
Bitir		

Tasarlayıcı: İki arkadaş 100 ile 200 arasında 3'e bölünen ancak 5 ile bölünmeyen kaç sayı olduğunu merak edip bir yarışa girmiştir. Biri diğerini geçmek için daha hızlı bir yöntem olan bir algoritma oluşturacaktır. Buna göre hazırlayacağı algoritmanın sözde kod ve akış diyagramı nasıl olmalıdır? Onun yerine bunları sen oluşturabilir misin?


Başla
Sayac=100
Adet=0
Sayac < 200
Evet
Sayac % 3 == 0
&&
Sayac % 5 != 0
Hayır
Sayac ++
Hayır
Evet
Adet
Adet++
Bitir

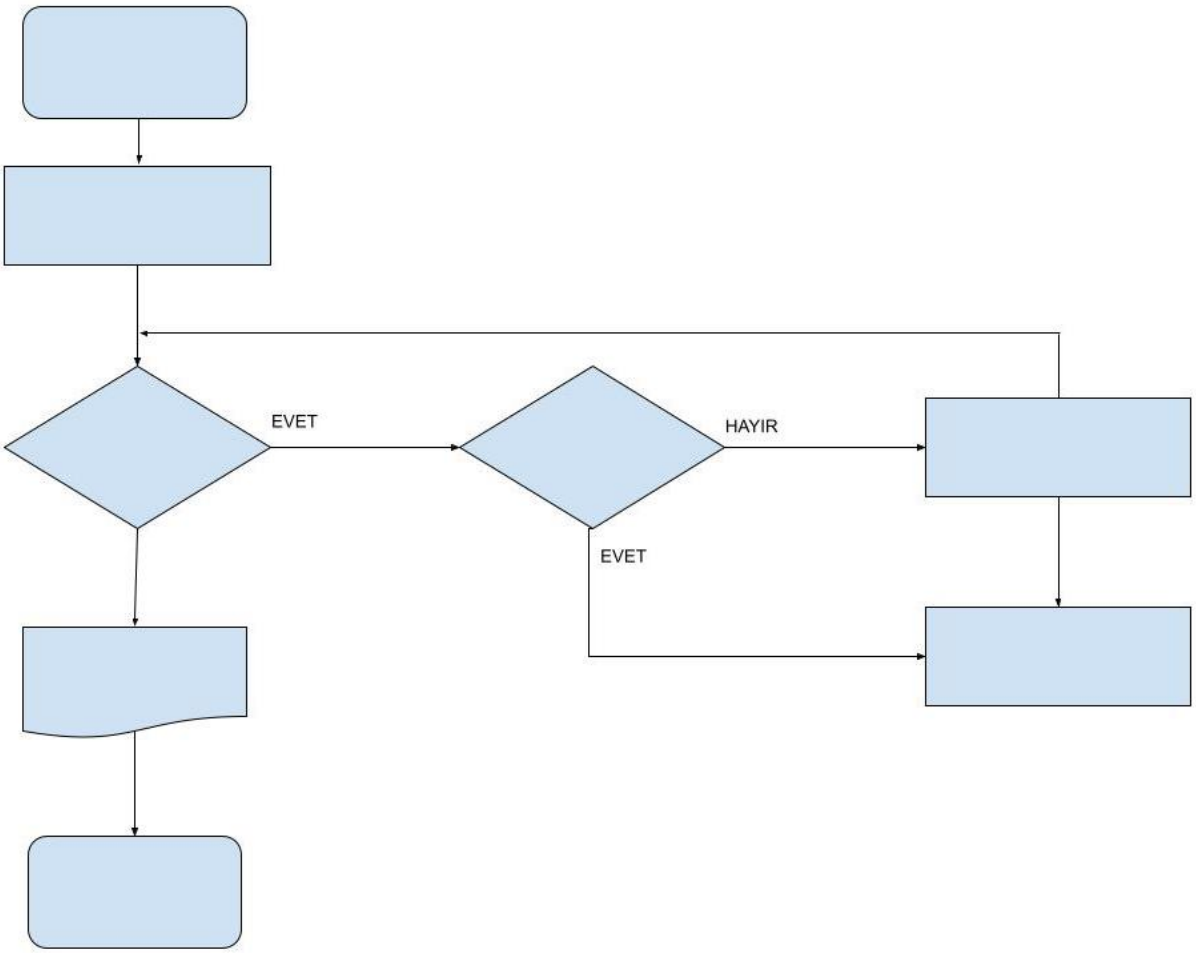
Hafta 3. Ders Materyalleri

O3. B1. 1. Çiftleri Toplama Algoritması

Görev: Aşağıdaki problemin çözümüne yönelik sözde kodu verilen algoritmayı doğru şekilde sıralayın ve akış diyagramındaki boşluklara yazın.

Problem: 1 ile 100 arasındaki çift sayıların toplamını yazdırma

Başla	Sayi = 1, Toplam = 0	Sayi % 2 ==0	Sayi < 100
Yaz, Toplam	Toplam = Toplam + Sayi	Bitir	Sayi = Sayi + 1



O3. B1. 2. Algoritma Terimleri Görev Kartları

Tanımlayıcı

Algoritmadaki değişkenleri, sabitleri, kayıt alanlarını ve özel bilgi tiplerini adlandırmak için programcı tarafından oluşturulan kelimelerdir. Tanımlayıcıların yerini tuttukları ifadelere çağrışım yapacak şekilde adlandırılmaları algoritmanın anlaşılması açısından önemlidir. Tanımlayıcı isimlerinde İngiliz alfabesindeki "A-Z" ve "a-z" arası harfler, "0-9" arası rakamlar, sembollerden alt çizgi "_" kullanılabilir. Bununla birlikte tanımlayıcı ismi, rakamla başlayamaz veya sadece rakamlardan oluşamaz.

Görev

Çiftleri toplama algoritmasındaki tanımlayıcı kurallar doğru mudur?

Değişken

Algoritmanın her çalıştırılmasında farklı değerler alan bellek alanlarıdır. Değişken isimlendirilmeleri, tanımlayıcı kurallarına uygun biçimde yapılmalıdır. Örneğin bir ismin aktarıldığı değişken "ad" olarak, bir isim ve soy ismin aktarıldığı değişken "adsoyad", ev adresinin aktarıldığı değişken "evadres" ve iş adresinin aktarıldığı değişken "isadres" olarak tanımlanabilir. Burada özellikle İngiliz alfabesi kullanılır.

Görev

Çiftleri toplama algoritmasındaki değişkenleri bulup, isimlerini değiştirmeyi deneyip çıktıyı test edin.

Sabit Algoritmadaki değeri değişmeyen ifadelerdir. Algoritma boyunca kolay takip edilebilmesi için kullanılmaktadır. Değişkenler gibi isimlendirme kurallarına uygun olarak oluşturulmalıdırlar.	**Görev** Çiftleri toplama algoritmasındaki sabitleri bulun.
Atama/Aktarma Bir değişkene değer atamak için gerçekleştirilen işlemdir. Algoritma adımlarında sağdaki değeri, soldaki değişkene atamak için kullanılan bir işlemdir.	**Görev** Çiftleri toplama algoritmasındaki değişken atama ifadelerini bulunuz ve bu ifadeyi farklı türde kullanmayı deneyin. Aradaki değişimi tartışın.
Sayaç Algoritma tasarımlarında bazı işlemlerin belirli sayıda yaptırılması ve bu süreçte oluşan değerlerin sayılması gerekebilir. Bu amaçla kullanılan sayma işlemlerine sayaç denir.	**Görev** Çiftleri toplama algoritmasına yeni bir sayaç eklemek için problemi değiştirin.

Döngü

Döngüler, tekrar etmesi gereken işlem bloklarını verilen sayıda gerçekleştiren işlem akışlarını sağlamaktadır. Örneğin 1 ile 100 arasındaki çift sayıların toplamını hesaplayan programda T=2+4+6 ….+100 hesabını teker teker yapmak yerine, 2'den 100'e kadar ikişer artan bir döngü kullanmak ve döngü değişkenini toplam hesabında kullanmak daha uygun olacaktır.

Görev

Çiftleri toplama algoritmasındaki döngüyü bulup, tartışın.

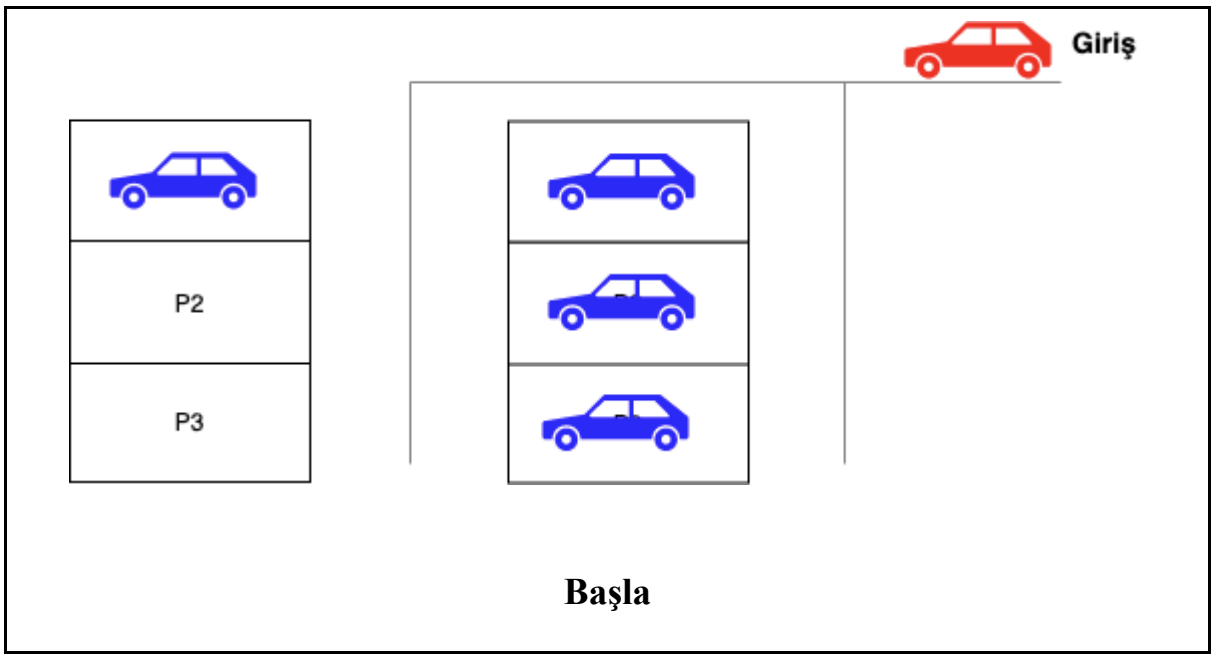
Ardışık Toplama/Çarpma

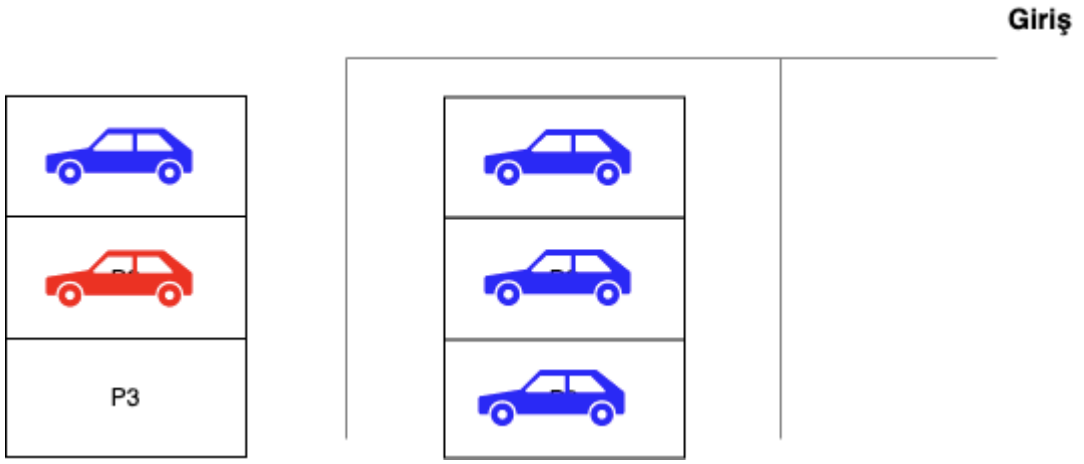
Algoritmalarda aynı değerin üzerine yeni değerlerin eklenmesi ya da aynı değerin yeni değerlerle çarpılarak kullanılmasını sağlayan işlemlerdir.

Görev

Çiftleri toplama algoritmasındaki ardışık toplama ifadesi yerine, ardışık çarpma yapıldığında oluşan yeni problemi tartışın.

O3. B2. 1. Otomatik Park Etme Algoritması





Bitir

Şekilde görülen kırmızı arabanın otomatik olarak uygun alana park etme algoritması:

1. Başla.
2. Sonraki sokağa kadar ilerle.
3. Eğer sokakta yer yoksa 2. adıma git.
4. Sokağa gir.
5. İlk uygun yere park et.
6. Bitir.

O3. B3. 1. Örnek Kod Çalıştırma Tablosu

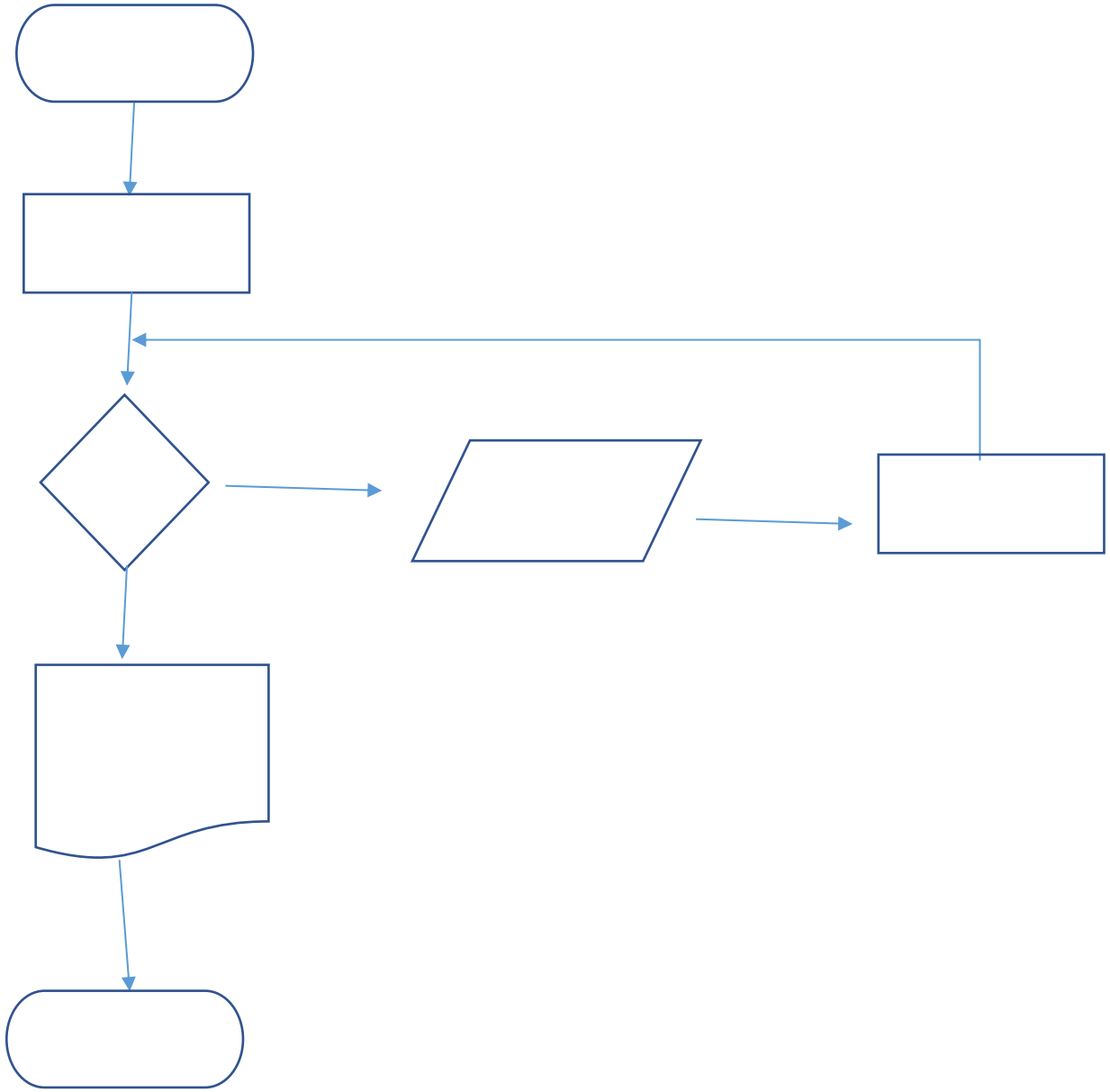
Örnek Olay: Arkadaşınız aklından iki sayı tutmuştur ve sizden bu iki sayıdan büyük olanı bulmanızı istiyor. Bunun için bir algoritma yazmak istiyorsunuz.

Tablo 15. *Örnek kod çalıştma tablosu*

Adım	Değişken değerleri	Ekran Çıktısı
1) Başla.		
2) Oku, (Sayi1,Sayi2)	**Sayi1**=19, **Sayi2**=15	
3) Eğer Sayi1>Sayi2	Doğru, Evet	
3.1) Yaz, Sayi1		19
4) Aksi takdirde		
4.1) Yaz, Sayi2		
5) Bitir.		

Klavyeden Sayi1 olarak 19, Sayi2 olarak 15 girdiğimizi düşünelim. 3.Adımda 19>15 koşulu kontrol ediliyor. Bu koşul doğru olduğu için, 3.1. Adıma gidiyoruz. Orada da ekrana 19 değerini yazdırıyoruz. 4 ve 4.1 adımlar hiç çalıştırılmadan 5. Adıma giderek programımızı sonlandırıyoruz.

O3. B4. 1. Akış Şemasını Doldurma



O3. B4. 2. Sözde Kod Çalıştırma Tablosu

Algoritmanın sözde kodlarını çalıştırmak için aşağıdaki tabloyu kullanabilirsiniz. Algoritmanın adım sayısına göre satırları artırabilirsiniz.

Tablo 16. *Sözde kod çalıştma tablosu*

Adım	Değişken değerleri	Ekran Çıktısı

O3. C. 1. Kısmi Öğrenme Görevleri Afişi

Kısmi öğrenme görevleri afişine ulaşmak için buraya tıklayınız.

Hafta 4. C++ Dilinde Değişken ve Veri Tipleri

Kazanımlar

K1. Değişken isimlendirirken uyulması gereken kuralları uygular.

K2. Bir algoritmadaki sabitleri ayırt eder.

K3. Veri tipine uygun veri tanımlar.

K4. C++ temel giriş/çıkış akışlarını analiz eder.

K5. Operatörleri bir algoritma içinde kullanarak sonucunu yordar.

Amaç

Haftanın amacı, C++ değişken tanımlama adımlarının veri tipleri ile birlikte öğrenilerek sabit tanımlamanın nasıl gerçekleştirildiğini öğrenmektir. Ayrıca C++ temel giriş/çıkış akışlarının nasıl kullanıldığını, C++ işleçlerin örnekler üzerinde nasıl çalıştığını görmelerini sağlamaktadır.

Önerilen Ders Akışı

A. Giriş: İnmeyen Çubuk (15 dk.)

B. Öğrenme Görevleri

B1. Kuralları Belirle! (30 dk.)

B2. Sabitleri Ayıralım (15 dk.)

- ❏ Ders Arası (5 dk.)
- ❏ Motivasyon Oyunu (10 dk.)

B3. Uzman Sensin (40 dk.)

B4. Çıktıları Karşılaştır (15 dk.)

- ❏ Ders Arası (5 dk.)

B5. Operatörlerle Yüzleş! (30 dk.)

- ❏ Ders Arası (5 dk.)
- ❏ Motivasyon Oyunu (10 dk.)

C. Kısmi Öğrenme Görevleri (60 dk.)

A. Giriş: İnmeyen Çubuk

Süre: 15 dk.

Materyaller: 2 adet ince ve ortalama 1–1,5 metre uzunluğunda çubuk (Oyuncu sayısına göre uzunluk değişebilir)

Uygulama: Oyuncular 2 gruba ayrılır ve birbirleriyle karşı karşıya iki sıra hâlinde hizalanırlar. Oyunculardan işaret parmaklarını kaldırmaları ve kollarını ileri doğru uzatmaları istenir. Çubuk yatay şekilde tutulur ve herkesin sadece işaret parmağıyla tutacağı şekilde yerleştirilir. Herkesin işaret parmağı çubuğa her zaman değmesi gerektiği ve diğer parmakların değmesinin yasak olduğu söylenir. Amaç; çubuğun tüm grupla birlikte, yere paralel olacak şekilde yere kadar indirilmesidir. Hangi grup çubuğu önce yere indirirse, o grup oyunu kazanır (Kaynak**:** https://app.ogrenmetasarimlari.com).

B. Öğrenme Görevleri

B1. Kuralları Belirle!

Süre: 30 dk.

Kazanımlar: K1. Değişken isimlendirirken uyulması gereken kuralları uygular.

Materyaller: O4. B1. 1. Değişkenler Çalışma Yaprağı

O4. B1. 2. Hatalı Değişkenler

O4. B1. 3. Değişken İsmi Olamazlar

Hazırlık: Eğitmen "Hatalı Değişkenler" materyalinin çıktısını alıp, tek tek keserek materyali beş gruba ayırır. Her grupta dörder tane kart bulunmalıdır. Diğerleri ise öğrenme yönetim sistemi üzerinden açılır.

Uygulama: Eğitmen önceki etkinliklerde değişken kavramı hakkında öğrenilenleri Değişkenler Çalışma Yaprağı üzerinden hatırlatır. Eğitmen sunumun ardından bu etkinlikte değişkenlere isim verme üzerinde duracaklarını ifade eder. Bunun için öğrenciler dörderli gruplara ayrılır. Eğitmen tek tek kesilip beş gruba ayrılmış Örnek Değişkenler' materyalini öğrencilere dağıtır. Daha sonra gruplara

> *''Elinizde C++ dilinde değişken isimlerine ilişkin örnekler var ve grup görev kartı var. Görev kartında hatalı ya da doğru yazılan değişken isminden bir kurala erişmeniz bekleniyor. Bunun için elinizdeki değişken isimleri arasındaki benzerlikleri keşfetmeye çalışın. Bu benzerlikler değişken ismi yazma kuralını bize verecektir. Sizden beklenen bu benzerlikten yararlanıp değişkene isim verme kuralını keşfetmektir. Her grup yalnızca bir kurala erişmelidir.'*

talimatı verilir. Her grup bir tane değişken belirleme kuralına erişmelidir. Bu kurala erişmek için eğitmen her öğrencinin kartında yazan değişken isimlerini grup üyelerininki ile karşılaştırmalarını ister. Bu şekilde kartlar arası ortak noktaların değişken belirleme kuralına işaret edeceği ipucunu verir. Gruplar beş dakika kartlar üzerinde çalışırlar. Eğitmen sırayla grupların belirledikleri kuralı sesli şekilde okutur. Eğitmen tüm kurallar tamamlanınca değişken ismi olmayan yasaklı kelimelerin olduğu konusunda bir hatırlatma yapar. Bu

kelimeler "Değişken İsmi Olamazlar" materyali ile Öğrenme Yönetim Sistemi üzerinden öğrencilerle paylaşılır. Bununla birlikte eğitmen aşağıdaki açıklamayı kullanarak öğrencilere uyarıda bulunur.

> *"Artık değişkenlere isim verme kuralları belli olduğuna göre, herkes kendisine bir kâğıt çıkarsın. Bu kuralların tümünü uygulayan bir veriyi tutacak değişken ismi belirleyin ve grup olarak verdiğiniz isimleri inceleyin."*

Eğitmen etkinlik sonunda değişken ismi olmayan yasaklı kelimelerin olduğu konusunda bir hatırlatma yapar. Bu kelimeler 'Değişken İsmi Olamazlar' materyali ile Öğrenme Yönetim Sistemi üzerinden öğrencilerle paylaşılır. Bununla birlikte eğitmen aşağıdaki açıklamayı kullanarak öğrencilere uyarıda bulunur.

> *"Anahtar kelimeleri yanlışlıkla değişken adı olarak kullanmayın. Anahtar kelimeleri bu şekilde kullanmak öngörülemeyen sonuçlara neden olabilir ve birçok sıkıntıya sebep olabilir. Örneğin, "short" bir sayıyı temsil etmek için ayrılmış bir kelimedir ve bu kelimeyi bir değişken adı olarak kullanmaya çalışırsanız, program oluşturulduğunda derleyici size bir hata verecektir."*

Eğitmene Öneriler: Sınıfta erişilmesi gereken temel kurallar aşağıdaki gibi olmalıdır.

Sınıfta erişilmesi gereken temel kurallar aşağıdaki gibi olmalıdır.

1. İsim alt çizgi ile başlayabilir ancak sayı ile başlayamaz. Diğer her karakter bir harf, alt çizgi veya sayı olabilir.
2. İsim bir harf ile başlayabilir ancak sayı ile başlayamaz. Diğer her karakter bir harf, alt çizgi veya sayı olabilir.
3. Değişken adları C++' da büyük/küçük harfe duyarlıdır; bu nedenle "numara", "Numara" ve "NUMARA" gibi değişkenler üç ayrı değişken olarak ele alınır.
4. İsim içerisinde Türkçe karakter kullanılmaz.
5. İsim yazarken boşluk bırakılmaz.

'Değişken İsmi Olamazlar' materyali Öğrenme Yönetim Sistemi üzerinden öğrencilerle paylaşılırsa, ihtiyaç anında öğrencilerin açıp bakma imkânı olur. Eğitmen ayrıca öğrencilere aşağıdaki ipucunu vererek önerilerde bulunabilir.

İPUCU:

C++ programlamada, değişkenlerin adları için küçük harfler kullanmak standart bir uygulamadır. Ayrıca programlarınızın okunabilirliğini artırmak için, ogrenci_yas, sinav_notu vb. gibi daha uzun ve daha açıklayıcı adlar seçebilirsiniz.

B2. Sabitleri Ayıralım!

Süre: 15 dk.

Kazanımlar: K2. Bir algoritmadaki sabitleri ayırt eder.

Materyaller: O4. B2. 1. Sabitler için Örnek Kod

Hazırlık: Her grup için materyalin bir çıktısı alınır. Öğrenciler dörderli gruplar hâlinde oturmaktadır.

Uygulama: Eğitmen değişken ismi tanımlamalarının ardından sabitler için aşağıdaki kısa bilgi ile konuya giriş yapar.

> *"Bir programın yürütülmesi sırasında içeriği hiç değişmeyecek olan veriler, bir değişken yerine sabit bir tanımlama ile saklanmalıdır."*

Eğitmen öğrencilere örnek koddaki sabitleri tahmin etmelerini ister. Bunu yaparken aşağıdaki üç temel kurala dikkat etmeleri gerektiğini belirtir:

1. "const" anahtar sözcüğünü kullanarak tanımlanır.
2. Sabit adları, değişken adlarından ayırt etmek için büyük harfle yazılır.
3. Sabitler her zaman tanımlama sırasında değerlerini almalıdır.

Eğitmen öğrencilerin sabitleri ayırması için gruplara 3 dk. süre tanır. Süre sonunda tahminleri alır ve konuyu aşağıdaki bilgilerden yararlanarak özetler:

> *Bir programın yürütülmesi sırasında içeriği hiç değişmeyecek olan veriler, bir değişken yerine sabit bir tanımlama ile saklanmalıdır. Bu, derleyicinin kodu hatalar açısından kontrol etmesini sağlar. Yazdığımız program, sabitte depolanan değeri değiştirmeye çalışırsa, derleyici bir hata bildirir ve derleme başarısız olur.*
>
> *Aşağıda verilen bilgileri sabit olarak değerlendirebilirsiniz.*
>
> - *Pi sayısı*
> - *Evinizdeki oda sayısı*
> - *Eviniz ile okulunuz arasındaki uzaklık*
> - *Kimlik numaranız*
> - *Öğrenci numaranız*
> - *Doğum tarihiniz*

```
const sabitTipi SABITADI = deger;
```

```
const double PI = 3.14159265;
```

Sabitleri kullanmak çok fazla avantaj sağlar. Örneğin matematikteki Pi sayısını her kullanmamız gerektiğinde, program boyunca 3.14 gibi bir sayı yazdığımızı varsayalım. Ardından uygulamanızın daha yüksek bir hassasiyet gerektirdiğini fark edersek; her 3.14

sayısını, 3.14159265 gibi daha yüksek hassasiyet değeri ile değiştirmemiz gerekir. Bu değişikliği uygun bir şekilde hatasız olarak gerçekleştirmek de zaman alacaktır. Bunun yerine, Pi sayısını 3.14 değeri ile sabit olarak tanımlasaydık ve daha sonra hassasiyeti artırmamız gerektiğinde sadece en baştaki tanımlamadaki değeri değiştirmemiz yeterli olacaktır.

B3. Uzman Sensin

Süre: 40 dk.

Kazanımlar: K3. Veri tipine uygun veri tanımlar.

Materyaller: O4. B3. 1. Veri Tipleri Çalışma Kâğıtları

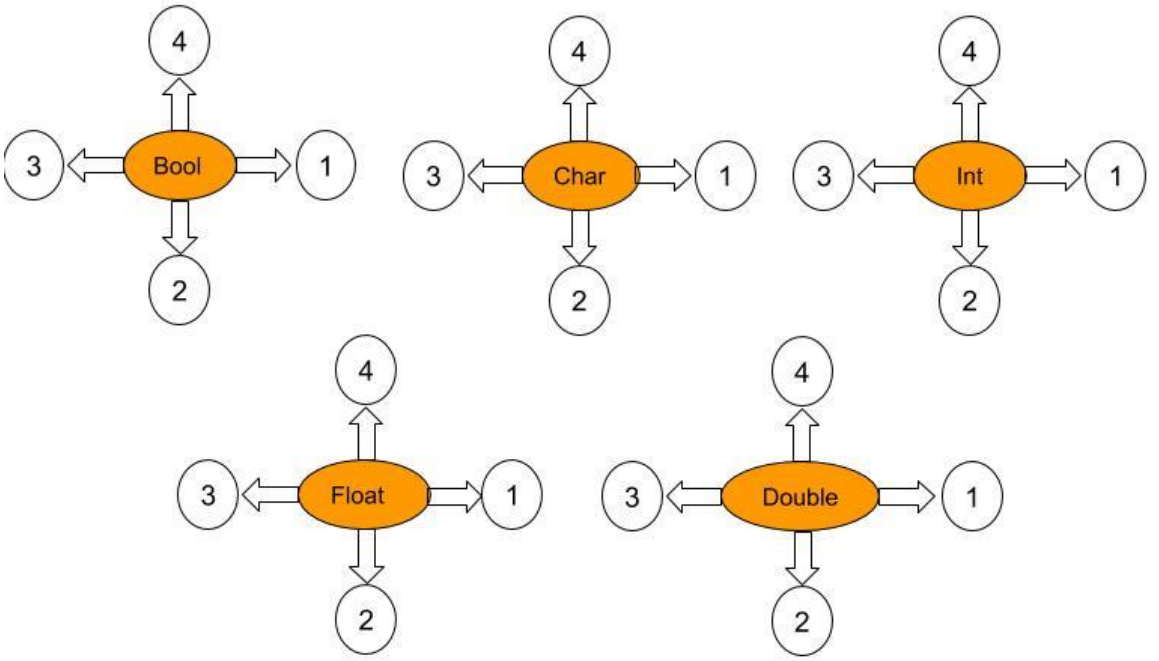
O4. B3. 2. Değişken Atama Kodları

Hazırlık: Eğitmen veri tipleri çalışma kâğıtlarını ve değişken atama kodlarının birer çıktısını alır, bunları keserek beş gruba ayırır.

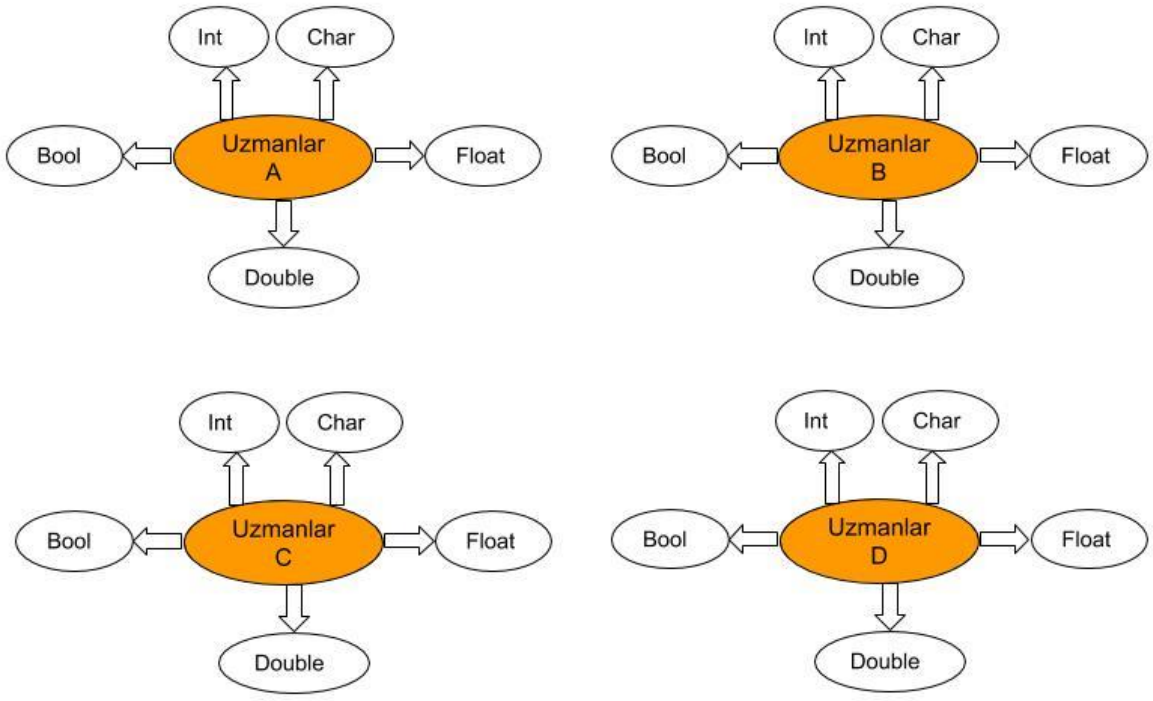
Uygulama: Öğrenciler dörderli beş gruba ayrılır. Eğitmen gruplara beş parçaya ayırdığı materyalleri dağıtır. Bu materyal ile her bir grup bir veri tipini temsil etmektedir. Ayrıca eğitmen gruplardan grup üyelerinin her birinin veri tipi adi_no şeklinde kodlanmalarını ister. Örneğin Bool ekip üyeleri, Bool_1, Bool_2… şeklinde dört kişiden oluşmaktadır. Eğitmen öğrencilerden temsil ettikleri veri tipinin özellikleri hakkında bilgi kartlarını incelemeleri ve örnekleri grup içinde beş dakika içinde tartışmalarını ister. Bu tartışmada her bir grup üyesinin temsil ettiği veri tipini başkasına anlatacak seviyede uzman olmaları istenir. Beş dakika ekipler kendi içlerinde çalıştıktan sonra, aynı numara ile kodlanmış öğrenciler bir araya gelir. Örneğin 1 numaralı grup üyeleri birleşerek beş kişiden oluşan uzmanlar grubu elde edilir.

Uzman gruplarının sayısı, beşer kişiden dört gruptur. Bu uzmanlar grubunda artık tüm veri tiplerinden birer kişi bulunmaktadır. Eğitmen bu yeni gruplardaki her bir bireyin temsil ettiği ve uzmanı olduğu veri tipini diğerine açıklamalarını ister. Ayrıca uzman öğrenciler temsil ettiği veri tipine ilişkin değişken atama kodlarını diğer grup üyelerine uygulatarak açıklayacaktır. Ancak kodları çalıştırmadan önce program çıktısında ne göreceklerini tahmin etmelerini ister. Örneğin bir numaralı uzman grubundaki Bool temsilcisi, diğer uzmanlara Bool değişken tanımlama kodlarını bilgisayar ortamında uygulayarak anlatır. Bu şekilde tüm uzmanlar birbirlerine değişken tanımlama kodlarını anlatarak ortak bir veri dosyası hazırlayacaktır. Artık bu kod dosyasında tüm veri tiplerine ilişkin değişken tanımlama programları yer alır. Eğitmen grup çalışmaları sırasında öğrencileri takip eder, desteğe ihtiyaç duyulduğunda geri bildirimlerde bulunur. Gruplar 10 dk. kadar birbirlerine açıklamalarda bulunur.

Eğitmene Öneriler: Eğitmen grupların dağılıp birleşmeleri için çan ya da zil sesi kullanabilir. Sınıf düzeni aşağıdaki gibi olmalıdır. Uzman gruplarına dağılmadan önce öğrencilerin grup içinde birlikte anlamlandıramadıkları noktaları eğitmenlere danışmaları istenebilir.



Oturma Düzeni 1



Oturma Düzeni 2

Resim 20. *Oturma düzenleri*

Ayrıca eğitmen uzman gruplarının her birinin birer öğretmen olarak rol aldıkları konusunda öğrencilere hatırlatıcı ve teşvik edici geri bildirimlerde bulunmalıdır. Örneğin "Bu işi çok iyi başardın; Konuyu diğer arkadaşlarına senin öğretiyor olman, benim açıklama yapmamdan çok daha etkili olacaktır" vb. gibi.

B4. Çıktıları Karşılaştır...

Süre: 15 dk.

Kazanımlar: K4. C++ temel giriş/çıkış akışlarını analiz eder.

Materyaller: O4. B4. 1. Çıktıları Karşılaştır Dosyası

Hazırlık: Öğrenciler ikili gruplar hâlinde oturmaktadır. Öğrenme yönetim sisteminden materyal öğrenci erişimine açılır.

Uygulama: Eğitmen öğrencilerin örnek iki kodun dosyasını öğrenme yönetim sistemi indirip kendi bilgisayarlarında açmalarını ister. Öğrencilerden cin ve cout arasındaki çıktı farklarını keşfetmeleri istenir. İlk olarak eğitmen öğrencilerden her iki kodu da bilgisayarlarında çalıştırıp karşılaştırmalarını ister. Daha sonra cin ve cout komutlarının görevlerini aralarında tartışmaları istenir. Sonunda eğitmen beyin fırtınası aracılığıyla öğrencilere sorular eşliğinde konuyu özetleyerek açıklar.

Eğitmene Öneriler: Eğitmen konuyu özetlerken aşağıdaki bilgileri kullanabilir.

C++ programlama, giriş ve çıkış işlemlerini gerçekleştirmek için birçok farklı kütüphane sunmaktadır. C++ 'da giriş ve çıkış, bayt serisi şeklinde veya daha yaygın olarak akış olarak bilinir. Giriş akışında, bayt akış yönü aygıttan (klavye, disk sürücü, ağ bağlantısı, vb.) ana belleğe doğrudur. Çıkış akışında ise bayt akış yönü tersine ana bellekten aygıta (ekran, yazıcı, disk sürücü, vb.) doğrudur. C++'da bulunan iki anahtar kelime cin ve cout çıktıları yazdırmak ve girdi almak için kullanılmaktadır. Bu iki ifade girdi ve çıktı almak için en temel yöntemlerdir. C++'da cin ve cout kullanmak için programda ***iostream*** *başlık dosyasını eklemek gerekmektedir. Aşağıdaki örnekleri inceleyiniz.*

B5. Operatörlerle Yüzleş!

Süre: 30 dk.

Kazanımlar: K5. Operatörleri bir algoritma içinde kullanarak sonucunu yordar.

Materyaller: O4. B5. 1. Operatör Çalışma Kâğıtları

O4. B5. 2. Destekleyici Bilgiler

Hazırlık: Her bilgisayarda Code::Blocks kayıtlı ve aktif çalışır durumda olmalıdır. Çalışma kâğıtlarından 10 adet çıktı alınır ve ikişerli olacak şekilde öğrencilere dağıtılır. Destekleyici bilgiler ÖYS üzerinden materyal olarak erişime açılır.

Uygulama: Eğitmen dört kişilik gruplar oluşturur, ancak öğrencilerin bilgisayar başında ikişerli oturmalarını ister. İkişerli oturan her öğrenciye Operatör Çalışma Kâğıtları dağıtılır. Öğrenciler bu çalışma kâğıtlarında verilen kodların ekran çıktısını kâğıt üzerinde tahmin etmeye çalışacaktır. Eğitmen bu çalışma kâğıtlarını doldururken ÖYS üzerinden erişime açılan Destekleyici Bilgiler materyalinden faydalanabileceklerini belirtir. Dörtlü grup hâlinde çalışma kâğıdındaki kodları tartışmalarına 5 dk. kadar izin verilir. Daha sonra grup üyeleri çalışma yaprağındaki kodları paylaşarak tahmin ettikleri çıktıyı kontrol etmek için mevcut kodları

Code::Blocks üzerinde yazıp kontrolünü yaparlar. Hatalı tahminler dörtlü grup içinde tekrar tartışılır.

Eğitmene Öneriler: Öğrenciler destekleyici bilgileri açıp incelemeden eğitmene doğrudan soru yöneltebilir. Bu durumlarda onların materyal üzerinde çalışmalarını teşvik eden <*Bu konudaki soruna destekleyici bilgiler cevap verecektir. Lütfen materyali sistemden açıp inceleyin*> şeklinde geri bildirimler verilmelidir.

C. Kısmi Öğrenme Görevleri

Süre: 60 dk.

Materyal: O4. C. 1. Kısmi Öğrenme Görevleri Afişi

Hazırlık: Kısmi öğrenme görevleri afişi öğrencilere ÖYS ortamında süreli ödev olarak açılır.

Uygulama: Öğrenciler afişteki görevleri istedikleri sırada ve sayıda kendi tercihlerine bağlı olarak verilen süre içinde uygulamaya başlar. Her bir görevi tamamlayan öğrencilere, göreve ilişkin beceri rozeti verilir. Öğrenciler görev seçimlerini ve tamamladıkları görevleri eğitmene bildirmelidir. Eğitmenler ihtiyaç duydukları zaman öğrencilere görevi tamamlama aşamasında anlık geri bildirimlerde bulunur. Öğrenciler, ÖYS üzerinden süreli ödev olarak açılan kısmi öğrenme görevlerinden kendi tercihlerine göre istedikleri sayıda görev tamamlayıp eğitmene iletir. Süre bitiminde eğitmen görevleri ve yanıtlarını ÖYS ya da GitHub üzerinden öğrencilere gönderir.

Kısmi Öğrenme Görevleri Yanıtlar: Görevler, öğrenciye verilecek beceri rozetleri ile isimlendirilmiştir. Her bir rozetin altında ilgili görevin yanıtları verilmektedir. Eğitmen bu yanıtları uygulama süresinin sonunda GitHub üzerinden öğrencilerle paylaşır.

Kodlayıcı: Kodlayıcı rozetine ait iki görev bulunmaktadır. Aşağıdaki görevlerden birinin yapılması kodlayıcı rozetinin kazanılması için yeterlidir.

1. Kullanıcıdan alacağınız yarıçap bilgisi ve tanımlayacağınız sabit pi = 3.14 değerini kullanarak dairenin alanı ve çevresini hesaplayıp ekrana yazdıran programı kodlayınız.

```
#include <iostream>

using namespace std;

int main()

{
```

```
    int yari_cap;
    const float PI_DEGERI = 3.14;
    float alan, cevre;
    cout << "Dairenin yaricap uzunlugunu giriniz: ";
    cin >> yari_cap;
    alan = PI_DEGERI * yari_cap * yari_cap;
    cout << "Dairenin alani: " << alan;
    cevre = 2 * PI_DEGERI * yari_cap;
    cout << "\nDairenin cevresi: " << cevre;
    return(0);
}
```

2. 2 kişinin yaşları toplamını hesaplayıp, ekrana yazdıran programın kodunu yazınız.

```
#include <iostream>
using namespace std;
int main()
{
    int yas1, yas2, yas3;
    cout << "1. kisinin yasini giriniz:";
    cin >> yas1;
    cout << "2. kisinin yasini giriniz:";
    cin >> yas2;
    cout << "\nYaslarinizin toplami: " << yas1+yas2;
    return 0;
}
```

Denetleyici: Verilen değişken tanımlamalarından ve ilk değer atamalarından hangileri uygun olmayan değişken tanımı ve değer atamasıdır?

int __xyz5;	int _xyz5;	bool xyz = -1
~~short xyz = 34452;~~	~~int xyz=5.2;~~	int xyz = '*';
int xyz = 34452;	char xyz = '\192';	~~float xyz = 12345.12345;~~

Tasarlayıcı: Verilen C++ kodunun çıktısı tahmin edin.

Kod

```
#include <iostream>
using namespace std;
int main()
{
   const int x;
   x = 10;
   cout << x<<;
}
```

Çıktı

Verilen kodda sabit olarak tanımlanan x değişkenine tanımlama satırının dışında değer atanmaya çalışılmaktadır. Daha önce belirtildiği gibi sabitler tanımlanırken ilk değerlerini alır ve daha sonra içerikleri değiştirilemez.

Analizci: 2 kişinin yaşları toplamını hesaplayan, program kodu verilmiştir. Kod üzerinde eksik olan satırları tamamlayınız.

Eksik Kodlar

```
#include <iostream>
using namespace std;
int main()
{
    int yas1, yas2;
    cout << "1. kisinin yasini giriniz:";
    cin >> yas2;
    return 0;
}
```

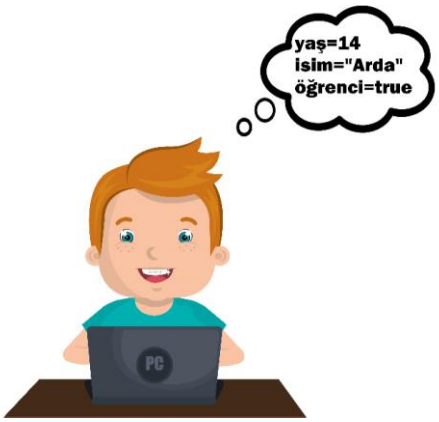
Tamamlanmış Kodlar

```
#include <iostream>
using namespace std;
int main()
{
    int yas1, yas2;
    cout << "1. kisinin yasini giriniz:";
    cin >> yas1;
    cout << "2. kisinin yasini giriniz:";
    cin >> yas2;
    cout << "\nYaslarinizin toplami: " << yas1+yas2;
    return 0;
}
```

Hafta 4. Ders Materyalleri

O4. B1. 1. Değişkenler Çalışma Yaprağı

C++ programlarında veri değerlerinin bilgisayarın belleğinde saklanabildiği saklayıcı yapılara değişken adı verilir. Saklanan değere değişkenin adı kullanılarak ulaşılabilir. Değişken dediğimiz kavram bir öğrencinin yaşı, ismi veya öğrencilik durumu olabilir.



Resim 21. *Değişkenleri düşünme*

Değişken kavramını daha iyi anlamak için şu örnekleri kullanabiliriz.

- *Bir bardak ve bir şişe süt olduğunu varsayalım. Bardağa biraz süt koyalım ve bardakta ne kadar süt olduğuna bakalım. Bardağa biraz daha süt koyalım ve tekrar bardakta ne kadar süt olduğuna bakalım. Sütün miktarı bardak dolana kadar az az değişerek herhangi bir değer olabilir. İşte bu nedenle süt miktarına "değişken" diyoruz. Yani değişebilen bir miktar söz konusudur.*

Resim 22. *Değişken kavramını anlama*

- *Kilomuz ve yaşımız bir değişkendir.*

Resim 23. *Kilo ve yaş değişkenleri*

- *Kumbaradaki para miktarımız bir değişkendir.*

Resim 24. *Kumbara değişkeni*

- *Günlük yürüme miktarımız bir değişkendir.*

Resim 25. *Günlük yürüme değişkeni*

- *Genel olarak farklı kişilerin adları ve soyadları değişkendir (tabii ki aynı olması da mümkündür).*

Resim 26. *Kimlik değişkeni*

Bir değişkenin bir programda kullanılabilmesi için önce tanımlanması gerekir. Bir değişken tanımladığınızda, tür belirtilir ve uygun miktarda bellek ayrılır. Bu bellek alanı, değişkenin adı referans alınarak ele alınmaktadır.

veri-tipi degisken_adi;

Sayılar, karakterler ve hatta tam kayıtlar gibi veriler, bir program tarafından işlenmelerini sağlamak için değişkenlerde saklanır. Değişkenler, özellikle bir sınıfa aitlerse, nesne olarak da adlandırılır. Aynı veri türünden birden fazla değişken adı oluşturulmak istenirse bunlar virgülle ayrılacak şekilde aşağıdaki gibi yazılabilir:

veri-tipi deg_adi1, deg_adi2, deg_adi3;

O4. B1. 2. Hatalı Değişkenler

*Her satır bir grubu temsil etmektedir. Satırdaki her sütun ise bir grup üyesinin kartıdır. Ortadaki sütun gruba ait görev kartını içerir. Lütfen aşağıdaki kartları kesikli noktalardan kesip öğrencilerinize dağıtın. Ancak Kural satırlarını gruplara etkinlik sonunda verin.

_sayi1	_birinci_sayi	Bu grupta sadece bir değişken ismi hatalı yazılmıştır. Kural hatalı olanda gizlidir.	_1sayi	1sayi
Kural 1: Değişken ismi alt çizgi ile başlayabilir. Ancak numara ile değil.				
ikincisayi	sayi_ikinci	Bu grupta sadece bir değişken ismi hatalı yazılmıştır. Kural hatalı olanda gizlidir.	sayi_2	2sayi
Kural 2: Değişken ismi bir harf ile başlayabilir. Ancak numara ile değil.				

ücuncüsayi	üçüncüsayı	Bu grupta sadece bir değişken ismi doğru yazılmıştır. Kural doğru olanda gizlidir.	uçuncusayi	ucuncusayı
Kural 3: Değişken isminde Türkçe karakter kullanılmaz.				
SAYIdort	sayiDORT	Bu gruptaki tüm değişkenler doğru yazılmış ve her biri farklı bir değişkendir. Kural doğru olanda gizlidir.	SayiDort	Sayidort
Kural 4: Değişken adları C++' da büyük/küçük harfe duyarlıdır; bu nedenle “numara”, “Numara” ve “NUMARA” gibi değişkenler üç ayrı değişken olarak ele alınır.				

O4. B1. 3. Değişken İsmi Olamazlar!

C++ ANAHTAR KELİMELER:					
and	auto	bool	break	case	catch
char	class	concept	const	continue	default
delete	do	double	else	enum	extern
false	float	for	friend	goto	if
int	long	namespace	new	not	operator
or	private	protected	public	register	return
short	signed	sizeof	static	struct	switch
template	this	throw	true	try	typedef
union	unsigned	virtual	void	volatile	while

UYARI:

Yukarıda verilen anahtar kelimeleri yanlışlıkla değişken adı olarak kullanmayın. Anahtar kelimeleri bu şekilde kullanmak öngörülemeyen sonuçlara neden olabilir ve birçok sıkıntıya sebep olabilir. Örneğin, "short" bir sayıyı temsil etmek için ayrılmış bir kelimedir ve bu kelimeyi bir değişken adı olarak kullanmaya çalışırsanız, program oluşturulduğunda derleyici size bir hata verecektir.

O4. B2. 1. Sabitler için Örnek Kod

Aşağıda yer alan sabitleri bulup yuvarlak içine alınız. Sabitleri ayırmak için aşağıdaki üç kurala dikkat ediniz.

1. "const" anahtar sözcüğünü kullanarak tanımlarız.
2. Sabit adları, değişken adlarından ayırt etmek için büyük harfle yazılır.

Sabitler her zaman tanımlama sırasında değerlerini almalıdır.

```
#include <iostream>
using namespace std;
int main()
{
    const int DOGUM_YILI = 2005;
    int boy = 175, kilo=72;

    cout << "Dogum yilim: " << DOGUM_YILI << endl;
    cout << "Boyum: " << boy << endl;
    cout << "Kilom: " << kilo << endl;

    return(0);
}
```

O4. B3. 1. Veri Tipleri Çalışma Kâğıtları

Veri Tipi	Açıklama	Boyut	Örnek
bool	Bool veri tipi en basit veri tipidir ve bir bayt uzunluğundadır. Doğru (1) ve yanlış (0) olmak üzere iki değerden birini saklayabilir. Programınızda yalnızca iki olasılığınız olduğunda bool türünü kullanabilirsiniz. Doğru (true-1) veya yanlış (false-0) değerlerini alabilen veri tipidir.	1 Bayt	Işığın açık olup olmadığı (0: kapalı, 1: açık)
char	Bir karakter (char) veri türü, standart ASCII tablosunda gösterilen herhangi bir harf veya simge olabilecek 256 farklı karakterden herhangi birini içerebilir. Dolayısıyla karakter kodu, her karakterle ilişkilendirilmiş bir tam sayıdır. Örneğin A harfini kodu 65 ile temsil edilir. Bir karakter değişkeni iki tek tırnakla ifade edilir.	1 Bayt	İsminin baş harfi: ‘A’, ‘d’

	Örneğin, ‘C’, ‘5’, ‘*’ ve ‘T’ karakter ifadeleridir. “char” anahtar sözcüğünü kullanarak aşağıdaki gibi bir karakter türü bildirebilirsiniz: Bir karakter tutabilen tek bir bayttır.		

int	Int (tam sayı) veri türü temel olarak tam sayıları temsil eder (içerisinde kesirli parça yoktur). Tam sayılarla işlem yapmak için üç farklı tam sayı türü vardır. Bu türler değer aralıkları ile ayırt edilir.
	
	❖ **int**
	❖ **short int** (veya **short**)
	❖ **long int** (veya **long**).
	
	Programlamada en sık kullanılan veri türüdür. Tam sayı türleri virgüllü sayıları veya kesirleri saklayamaz (örneğin, 5.1 veya 1/4). 16 bit bilgisayarlar için int veri tipi short veri tipine eşdeğer olurken, 32 bit bilgisayarlar için int veri tipi long veri tipine eşdeğer olur.
	
	`int sayi = 5;`
	`cout << sayi;`
	
	Short veri tipi, int veri tipinin yarısı boyutunda, yani 2 bayttır. -32,768 ile 32,767 arasında değer alabilir. Nispeten küçük değerleriniz olduğunda, bu tür daha kullanışlıdır. Bellek açısından baktığımız zaman int türünün sadece yarısını kapladığı için iki kat daha verimli olmaktadır.
		4 Bayt	Yaş (17), Sınıf (3)

	`short sayi = 5;` `cout << sayi;` `short int sayi = 5;` `cout << sayi;`
	
	Tam sayı veri tipleri arasındaki diğer veri tipimiz de long veri tipidir. Int veri tipi gibi 4 bayt uzunluktadır. Dolayısıyla -2,147,483,648 ile 2,147,483,647 arasında değer alabilir. Aşağıdaki şekillerde tanımlayabiliriz:
	
	`long sayi = 597;` `cout << sayi;` `long int sayi = 5` `cout << sayi;`		

float	Gerçek hayattaki veriler her zaman tam sayı olmak zorunda değil. Örneğin, kişinin boyu, marketteki bir ürünün fiyatı ya da bir sayının karekökü ondalık yani virgüllü bir sayı olabilir. Ondalık sayıları ya da kesirleri saklamak için float ve double veri türlerine ihtiyacımız vardır. Tam sayıların aksine, ondalık sayılar önceden belirlenmiş bir hassasiyette saklanmalıdır. Tiplerin değer aralığı ve hassasiyeti ayrılan bellek miktarına göre tespit edilir.	4 Bayt	Boy (1,75), Kilo (60,5)
double	Double veri tipi 8 bayt uzunluğundadır. Görüldüğü üzere son derece yüksek bir hassasiyete sahiptir. Virgüllü sayıların kullanımında bilimsel gösterim kullanılabilir. Aşağıda bazı sayıların bilimsel gösterimde nasıl yazıldığını görebilirsiniz: 2.27e+5 -3.15e+3 342.2e-3 44.78e-2 Yukarıdaki örnekteki 2.27e+5 değeri, $2{,}27 \times 10^5$ değeri ile aynıdır.	8 Bayt	Pi Sayısı (3.1415926535)

O4. B3. 2. Değişken Atama Kodları

Bool değişken tanımlama.

```
#include <iostream>
using namespace std;
int main()
{
    bool deger = true;
    cout << "Deger : " << deger;
    return 0;
}
```

Kodun Çıktısı:

```
Deger: 1
```

Char değişken tanımlama.

```
#include <iostream>
using namespace std;
int main() {
     char harf = 'a';
     cout << "Yazilan harf: " << harf;
     return 0;
}
```

Kodun Çıktısı:

```
Yazilan harf: a
```

Int değişken tanımlama.

```
#include <iostream>
using namespace std;
int main()
{
    int yas = 15;
    int kilo = 52;
    int boy = 167;
    cout << "Benim yasim: " << yas;
    cout << ", kilom: " << kilo;
    cout << " ve boyum: " << boy;
    return 0;
}
```

Kodun Çıktısı:

```
Benim yasim: 15, kilom: 52 ve boyum: 167
```

Float değişken tanımlama.

```
#include <iostream>
using namespace std;
int main()
{
    float sayi1 = 309.572;
    float sayi2 = -78.271;
    cout << "Sayi 1: " << sayi1 << "\nSayi 2: " << sayi2;
    return 0;
}
```

Kodun Çıktısı:

```
Sayı 1: 309.572
Sayı 2: -78.271
```

Double değişken tanımlama.

```
#include <iostream>
using namespace std;
int main()
{
    double sayi1 = 2.2e-308;
    double sayi2 = -2.3e-308;
    cout << "Sayi 1: " << sayi1 << "\nSayi 2: " << sayi2;
    return 0;
}
```

Kodun Çıktısı:

```
Sayı 1: 2.2e-308
Sayı 2: -2.3e-308
```

O4. B4. 1. Çıktıları Karşılaştır

```cpp
#include <iostream>
using namespace std;
int main()
{
    cout << "DENEYAP projesi ile gelecegimizi sekillendiriyoruz!";
    return 0;
}
```

```cpp
#include <iostream>
using namespace std;
int main()
{
    int yas;
    cout << "Yasinizi giriniz:";
    cin >> yas;
    cout << "\nYasiniz: " << yas;
    return 0;
}
```

O4. B5.1. Operatör Çalışma Yaprağı

```cpp
#include <iostream>
using namespace std;
int main()
{
    int a = 23, b = 5;
    cout << "Ekleme: " << (a + b) << endl;
    cout << "Cikarma: " << (a - b) << endl;
    cout << "Carpma: " << (a * b) << endl;
    cout << "Bolme: " << (a / b) << endl;
    cout << "Mod alma: " << (a % b) << endl;
    cout << "Arttirma: " << a++ << endl;
    cout << "Arttirma: " << ++a << endl;
    cout << "Azaltma: " << b-- << endl;
    cout << "Azaltma: " << --b << endl;
return 0;
}
```

Çıktı:

```
#include <iostream>
using namespace std;
int main()
{
    int x = 5, y = 4;
    cout << (x < y) << endl;
    cout << (x > y) << endl;
    cout << ((x-1) <= y) << endl;
    cout << (x >= (y+2)) << endl;
    cout << (x == y) << endl;
    cout << (x != y) << endl;
    return 0;
}
```

Çıktı:

```
#include <iostream>
using namespace std;
int main()
{
    int x = 5, y = 0;
    cout << ((x <= y) || (y>0)) << endl;
    cout << ((x > 4) && (y==0)) << endl;
    cout << (x && !y) << endl;
    cout << (!(x-2) || (y+2 > 2)) << endl;
    return 0;
}
```

Çıktı:

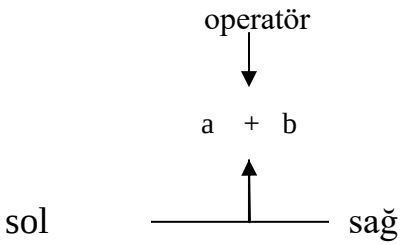
```
#include <iostream>
using namespace std;
int main()
{
    float x, y, z;
    x = 2.4;
    y = x;
    z = x + 1.3 + (y * 2.0);

    cout << "x: " << x << endl;
    cout << "y: " << y << endl;
    cout << "z: " << z << endl << endl;

    x = y = 3.4;
    cout << "x: " << x << endl;
    cout << "y: " << y << endl << endl;

    x+= 5.1;
    y+= y * 2.0;
    cout << "x: " << x << endl;
    cout << "y: " << y << endl << endl;

    x/= 2.0;
    y*= 3.0;
    cout << "x: " << x << endl;
    cout << "y: " << y << endl << endl;

    return 0;
}
```

Çıktı:

O4. B5. 2. Destekleyici Bilgiler

Aritmetik Operatörler

Matematiksel hesaplamalar bilgisayar programlarında yaygın olarak kullanılmaktadır. Programlama dilinde bir operatör, bir değer veya değişken üzerinde çalışan bir semboldür. Aşağıda görüldüğü üzere a değişkeni sol işlenen, b değişkeni sağ işlenen ve + sembolü de operatörü oluşturmaktadır.



Aşağıdaki tabloda, bilgisayar programlamasında kullanılan önemli aritmetik operatörler listelenmiştir.

Tablo 17. *Önemli aritmetik operatörler*

Operatör	Açıklama	Kullanım
+	İki işleneni birbirine ekler.	x + y
-	İkinci işleneni birinciden çıkarır.	x - y
*	İki işleneni çarpar.	x * y
/	İkinci işlenenle birinciyi böler.	x / y
%	Tamsayı bölümünün kalanını verir.	x % y
++	Sayıyı bir arttırma	x++ veya ++x
--	Sayıyı bir azaltma	x-- veya --x

Tekli operatör olan ++ veya -- işleçleri değişken değerlerini arttırmak için kullanılır. “x++” ifadesinde x değişkeni var olan değeri ile işleme girer ve işlem tamamlandıktan sonra x değişkeninin değeri bir arttırılır. “++x” ifadesinde ise önce değişkenin değeri bir arttırılır ve daha sonra işlem yapılır. Aynı şekilde -- operatörü içinde benzer kullanım geçerlidir. Aşağıdaki örneği inceleyiniz.

```
a = 5
b = ++a          // a değeri 6 olur, b değeri 6 olur
```

```
c = 2
d = c++;          // c değeri 3 olur, d değeri 2 olur

a = 5
b = --a           // a değeri 4 olur, b değeri 4 olur

c = 2
d = c--;          // c değeri 1 olur, d değeri 2 olur
```

UYARI:
Bir ön-ek operatörünün değişken değerini hemen değiştirdiğini, bir son-ek operatörünün değişken değerini daha sonra değiştirdiğini unutmayalım.

Karşılaştırma Operatörleri

Bu işleçlerin amacı iki değişken veya değişken grubunu belirtilen şarta göre karşılaştırmaktır. Bu karşılaştırmalar aynı türdeki değişkenler arasında olmalıdır. Bu işleçleri özellikle ilerleyen haftalarda göreceğimiz koşul yapıları ve döngülerde kullanılır. Yapılan karşılaştırmalar doğruysa "1" yanlışsa "0" sonucunu döndürür.

Tablo 18. *Karşılaştırma operatörleri*

İşleç	Açıklama	Kullanım
<	Küçüktür	x < y
>	Büyüktür	x > y
<=	Küçük eşittir	x <= y
>=	Büyük eşittir	x >= y
==	Eşittir	x == y
!=	Eşit değildir	x != y

UYARI:
Programlamada en çok yapılan hatalardan biri iki ifadeyi karşılaştırmak için atama operatörünün (=) kullanılmasıdır. Böyle bir atama yaptığınızda soldaki değer bir

değişkense derleyici bir hata mesajı oluşturmaz. Bu hata, programlamaya yeni başlayanların en çok dikkat etmesi gereken bir hatadır.

Atama Operatörleri

Atama işleçleri, bir değişkene değer atamak için kullanılır. Atama operatörünün sol taraftaki işleneni değişkendir ve atama operatörünün sağ taraftaki işleneni bir değerdir. Farklı tür atama operatörleri aşağıda tabloda verilmektedir.

Tablo 19. *Atama operatörleri*

İşleç	Açıklama	Kullanım
=	Sağdaki değeri soldaki değişkene atamak için kullanılır.	x = y z=10
+=	Bu işleç, '+' ve '=' operatörlerinin birleşimidir. Önce soldaki değişkenin geçerli değerini sağdaki değere ekler ve ardından sonucu soldaki değişkene atar.	x+=y (x=x+y)
-=	Bu işleç, '-' ve '=' operatörlerinin birleşimidir. Önce soldaki değişkenin geçerli değerini sağdaki değerden çıkarır ve ardından sonucu soldaki değişkene atar.	x-=y (x=x-y)
*=	Bu işleç "*" ve "=" operatörlerinin birleşimidir. Önce soldaki değişkenin geçerli değerini sağdaki değerle çarpar ve ardından sonucu soldaki değişkene atar.	x*=y (x=x*y)
/=	Bu işleç "/" ve "=" operatörlerinin birleşimidir. Önce soldaki değişkenin geçerli değerini sağdaki değere tam böler ve ardından sonucu soldaki değişkene atar.	x/=y (x=x/y)
%=	Bu işleç "%" ve "=" operatörlerinin birleşimidir. Önce soldaki değişkenin geçerli değerini sağdaki değere göre bölümünden kalanı alır ve ardından sonucu soldaki değişkene atar.	x%=y (x=x%y)

Mantıksal Operatörler

Mantıksal operatörler programlama dillerinde çok önemli bir yere sahiptir ve belirli koşullara göre karar vermemize yardımcı olurlar. İki koşulun sonucunu birleştirmek istediğimizde mantıksal VE ve VEYA istediğimiz sonucu üretmemize yardımcı olur. Bütün şartların sağlanması için koşullar arasına VE, herhangi birinin sağlatılması isteniyorsa koşullar arasına VEYA ve koşulu sağlamayanlar isteniyorsa DEĞİL işleci kullanılmalıdır.

Tablo 20. *Mantıksal operatörler*

İşleç	Açıklama	Kullanım
&&	Mantıksal VE	x && y
||	Mantıksal VEYA	x || y
!	Mantıksal DEĞİL	!x

Mantıksal işleçlerin sonucu şu şekilde belirlenmektedir. Eğer x&&y kullanılıyor ise her iki değişkenin değeri "1" olursa sonuç "1" olur, aksi hâlde sonuç "0" olur. Eğer x||y kullanılıyor ise her iki değişkenin değeri "0" olursa sonuç "0" olur, aksi hâlde sonuç "1" olur. Eğer !x kullanılıyor ise değişkenin değeri "1" ise sonuç "0", "0" ise sonuç "1" olacaktır. A ve B değişkenleri için oluşturulan aşağıdaki tabloyu inceleyebilirsiniz.

Tablo 21. *Mantıksal operatörlerin kullanımı*

A	B	A && B	A || B	! A
True	True	True	True	False
True	False	False	True	False
False	True	False	True	True
False	False	False	False	True

UYARI:
x veya x+1 gibi sayısal bir değer, değeri 0 ise "false", 0 dışında bir değer ise "true" olarak yorumlanır.

Operatör Önceliği

C++ programlarınızı oluştururken aritmetik operatörlerin önceliğine dikkat etmeniz gerekmektedir. Operatör önceliği değerlendirme sırasını, yani operatörlerin ve işlenenlerin nasıl gruplandığını belirler. Aşağıdaki tablodaki öncelik sırasını inceleyiniz.

Tablo 22. *Operatör öncelikleri*

Öncelik	Operatör	İşlem Yönü
1	()	Soldan sağa
2	~ ++ - -	Sağdan sola
3	* / %	Soldan sağa
4	+ -	Soldan sağa
5	<< >>	Soldan sağa

6	< <= > >=	Soldan sağa
7	== !=	Soldan sağa
8	&	Soldan sağa
9	^	Soldan sağa
10	|	Soldan sağa
11	&&	Soldan sağa
12	||	Soldan sağa
13	= += -= *= /= %=	Sağdan sola

O4. C. 1. Kısmi Öğrenme Görevleri Afişi

Kısmi öğrenme görevleri afişine ulaşmak için buraya tıklayınız.

Hafta 5. Karar Mantık Yapıları

Kazanımlar

K1. C++ programlama dilinde tekli karar yapısını kullanarak programı test eder.

K2. C++ programlama dilinde çoklu karar yapısını kullanarak programı test eder.

K3. C++ programlama dilinde iç içe karar yapısını kullanarak programı test eder.

Amaç

Bu haftanın amacı öğrencilerin programlamada karar yapılarını kullanarak programın akışını kontrol edebilmelerini sağlamaktır.

Önerilen Ders Akışı

A. Giriş: Ders Öncesi Enerji (10 dk.)

B. Öğrenme Görevleri

B1. Karar Mantık Yapılarını Tanıyalım! (40 dk.)

- ❏ Ders Arası (10 dk.)

B2. Görevleri Kodlayalım (60 dk.)

- ❏ Ders Arası (10 dk.)

B3. İç İçe Koşula Farklı Bir Bakış (40 dk.)

- ❏ Ders Arası (10 dk.)

C. Kısmi Öğrenme Görevleri (60 dk.)

A. Giriş: Ders Öncesi Enerji

Süre: 10 dk.

Uygulama: Sınıfta bulunan her öğrenci kendine bir oyun arkadaşı seçer. Herkes seçtiği oyun arkadaşı ile taş, kâğıt ve makas oyununu karşılıklı oynar, oyunu kaybeden kazandığı kişinin arkasına geçerek onun takipçisi olur. Kazananlar sürekli bir diğer kazananla karşılaşarak aynı şekilde taş, kâğıt ve makas oyununu tekrar eder, kaybeden her kişi kazananın arkasına geçmektedir ve kazananın takipçisi olmaktadır. En uzun kuyruğa ulaşan bir başka ifadeyle hiç kaybetmeyen kişi oyunu kazanır.

Eğitmene Öneriler: Oyundaki amaç kazananı belirlemekten çok grup dinamiğini canlı tutmaktır. Oyunu kaybeden öğrenci, kazananın arkasına geçmektedir ve kaybettiği arkadaşı bir sonraki diğer kazananla yarıştığında arkasında bulunduğu için aslında farkında olmadan kaybettiği arkadaşına destek vermektedir.

B. Öğrenme Görevleri

B1. Karar ve Mantık Yapılarını Tanıyalım!

Süre: 40 dk.

Kazanımlar: K1: C++ programlama dilinde tekli karar yapısını kullanarak programı test eder.

K2: C++ programlama dilinde çoklu karar yapısını kullanarak programı test eder.

K3: C++ programlama dilinde iç içe karar yapısını kullanarak programı test eder.

Materyaller: O5. B1. 1. Karar Yapılarını Tanıyalım Görev Kâğıtları

Hazırlık: Eğitmen bu konu için hazırlanan 5 sayfalık görev kâğıtlarının çıktısını alarak derse gelir.

Uygulama: Sınıf her bir grupta öğrenci sayısı eşit olacak şekilde 5 gruba ayrılır. Daha sonra her bir grubun ortak karar verecek şekilde 1 ile 100 arasında sayı belirlemeleri istenir. Eğitmen bu bölüm için hazırlanan görev kâğıtlarının çıktısını alarak her bir gruba birer karar yapısı içeren görev verir. Gruplardan daha önceden belirledikleri sayı ile birlikte elindeki karar yapılarının bulunduğu görev kâğıtlarını karşılaştırılıp, görev kâğıtlarındaki algoritmanın hangi çıktıyı vereceğine yönelik cevap istenir. Eğitmen bulunan çıktı sonuçlarının doğru veya yanlışlığı hakkında öğrencilere geri bildirim verir. Her bir gruptaki öğrencilerin doğru sonucu bulması hedeflenir. Daha sonra her bir görev kâğıdının her bir gruba dolaşması sağlanarak öğrencilerin çeşitli karar yapı örneklerinin görünmesi sağlanır. Görev kâğıtları dolaşırken her değişimde eğitmen karar yapılarına uygun sayılar belirleyerek ve her değişimde bu sayıları değiştirerek grupların o sayıya yönelik programın hangi çıktıyı vereceği öğrenciler tarafından tahmin edilir.

NOT: Her bir görev kâğıdının her gruba ulaşması sağlanır. Öğrenciler etkinliği bitirdikten sonra her bir görev kâğıdının günlük yaşamda hangi problemi çözmek için yazıldığını öğrencilerin tahmin etmesi beklenir ve bununla ilgili sınıfta eğitmen eşliğinde tartışma gerçekleştirilir.

Eğitmene Öneriler: Yukarıdaki etkinliğin amacı; öğrencinin program akışının bir noktasında verilen karara göre (seçilen sayıya göre) yapılacak işlemlerin ve ekran çıktısının nasıl değişebileceğini öğrencilere hissettirmektir.

Yukarıdaki etkinlik bittikten sonra eğitmenin karar yapılarının C++ dilinde nasıl kodlandığına yönelik aşağıdaki gibi kısa bir bilgilendirme yaparak özetleme yapması beklenir.

Bilgisayar programlamanın da olmazsa olmazı karar ifadeleridir. Belirli şartlara (sıcaklık değeri, klavyede bir tuşa basılıp basılmadığı, bir değer girilip girilmediği) göre yapılması gereken işlemleri karar ifadeleri ile gerçekleştiririz. C++ programlama dilinde bu amaç için kullanılan "if","if-else" ve "switch" ifadeleridir.

Eğer koşula göre yapılacak bir veya daha çok işlem varsa ilgili blok '{' ve '}' arasına yazılır.

```
if(koşul)
{
        koşul doğru ise yapılacak işlemler
}
```

Bazı durumlarda koşul gerçekleşmediğinde de işlem yapmak isteriz. Koşulun gerçekleşmemesi durumunu işleme almak için "aksi takdirde" anlamına gelen "else" komutu yazılır.

```
if(koşul)
{
        koşul doğru ise yapılacak işlemler
}
else
{
        koşul yanlış ise yapılacak işlemler
}
```

Bazı durumlarda koşullarımız birden fazla olabilir. Bu durumlarda her bir koşul için "else if" kullanmaktayız.

```
if(koşul1)
        koşul1 için yapılacak işlem
else if(koşul2)
        koşul2 için yapılacak işlem
else if(koşul3)
        koşul3 için yapılacak işlem
else
        diğer tüm durumlar için yapılacak işlem
```

B2. Görevleri Kodlayalım

Süre: 60 dk.

Kazanımlar: K1: C++ programlama dilinde tekli karar yapısını kullanarak programı test eder.

K2: C++ programlama dilinde çoklu karar yapısını kullanarak programı test eder.

K3: C++ programlama dilinde iç içe karar yapısını kullanarak programı test eder.

Materyaller: O5. B1.1 Karar Yapılarını Tanıyalım Görev Kartları

Hazırlık: Eğitmen B1 öğrenme görevindeki materyali bu etkinlikte de kullanmaya devam eder.

Uygulama: Öğrencilerin materyalde belirtilen ve bir ders önceki yaptıkları "Karar Yapılarını Tanıyalım" etkinliği içerisinden seçtiği gerçek yaşamla ilgili iki karmaşık görevi kendi başlarına kodlaması beklenir. Öğrenciler kodlama yaparken eğitmenlerden biri sınıfta dolaşarak öğrencilerin ihtiyaç duyduğu anda yöntemsel bilgiyi verecektir. Diğer eğitmen ise kodları bilgisayarda yazarak öğrencilerin seçtiği görevlerdeki bazı aşamaları ihtiyaç hâlinde öğrencilerin görmesine olanak sağlar.

Eğitmene Öneriler: Her bir bilgisayara iki öğrenci oturduğu düşünülürse, iki öğrencinin birer görev yapması sağlanarak zamanın etkili bir şekilde kullanımı sağlanabilir. İmkan dahilinde eğer her öğrenci için bir bilgisayar varsa her öğrencinin en az iki görevi kodlaması istenir. Kodlanamayan diğer görevler öğrencilere ödev olarak verilebilir.

Uygulama esnasında eğitmenin görevi, öğrencilerin ihtiyaç duyduğu zamanlarda yöntemsel bilgi vermesidir. Bu yöntemsel bilgi; kodun tamamını öğrencilere göstermek yerine, öğrencilerin zorluk yaşadığı aşamalarda ihtiyaç duydukları ipuçlarıdır. İpuçları öğrencileri doğru cevabı direkt söylemez ancak üzerinde düşünmesine teşvik eder.

B3. İç İçe Koşula Farklı Bir Bakış

Süre: 40 dk.

Kazanımlar: K3: C++ programlama dilinde iç içe karar yapısını kullanarak programı test eder.

Materyaller: O5. B3. 1. SWITCH Yapısı Kullanımı

Hazırlık: Eğitmen dersin bu bölümü için hazırlanan materyalleri ÖYS üzerinden paylaşır.

Uygulama: Bu bölüm için hazırlanan materyal sunum üzerinden paylaşılarak if yapısı ile switch yapısı konusunda kod yazımında ne gibi farklılıkların olduğuna yönelik sınıf içi tartışma ortamı oluşturularak öğrencilerin bu farklılıkları bulması istenir. Daha sonra eğitmen tarafından aşağıdaki gibi konu öğrencilerin de katılımı ile özetlenir.

Birden çok koşul olduğu durumlarda "switch" bloğunu kullanabiliriz. Her koşul durumunu "case" komutları ile belirterek daha kolay okunabilir bir kod parçası oluşturabiliriz. Söz dizimi şu şekildedir;

```
switch(değiskenAdi)
{
        case durum 1:
                Yapılacak işlemler
        break;
        case durum 2:
                Yapılacak işlemler
        break;
        …
        default:
                Yapılacak işlemler
        break;
}
```

break: Bu komut, koşul bloğundan çıkmamızı sağlar.

default: Verilen koşulların gerçekleşmemesi hâlinde çalışır.

Eğitmene Öneriler: Eğitmenler tarafından konu özetlendikten sonra öğrencilerin switch komutunu kullanarak, aynı örneği bilgisayarda kodlaması istenir. Kodlama esnasında eğitmenlerin öğrencilere işlemsel bilgiyi ihtiyaç duydukları anda vermesi beklenir.

C. Kısmi Öğrenme Görevleri

Süre: 60 dk.

Materyal: O5. C. Kısmi Öğrenme Görevleri Afişi

Hazırlık: Kısmi öğrenme görevleri afişi öğrencilere ÖYS ortamında süreli ödev olarak açılır.

Uygulama: Öğrenciler afişteki görevleri istedikleri sırada ve sayıda kendi tercihlerine bağlı olarak verilen süre içinde uygulamaya başlar. Her bir görevi tamamlayan öğrencilere, göreve ilişkin beceri rozeti verilir. Öğrenciler görev seçimlerini ve tamamladıkları görevleri eğitmene bildirmelidir. Eğitmenler ihtiyaç duydukları zaman öğrencilere görevi tamamlama aşamasında anlık geri bildirimlerde bulunur. Öğrenciler, ÖYS üzerinden süreli ödev olarak açılan kısmi öğrenme görevlerinden kendi tercihlerine göre istedikleri sayıda görev tamamlayıp eğitmene iletir. Süre bitiminde eğitmen görevleri ve yanıtlarını ÖYS ya da GitHub üzerinden öğrencilere gönderir.

Kısmi Öğrenme Görevleri Yanıtlar: Görevler, öğrenciye verilecek beceri rozetleri ile isimlendirilmiştir. Her bir rozetin altında ilgili görevin yanıtları verilmektedir. Eğitmen bu yanıtları uygulama süresinin sonunda GitHub üzerinden öğrencilerle paylaşır.

Analizci:

```
#include <iostream>
using namespace std;
int main()
{
    int sayi = 13;
    if(sayi % 2 == 0)
        cout << sayi /2;
    else
        cout << (sayi-1) /2;
}
```

Yukarıdaki kodu bilgisayarına yazan bir programcı sizce ne programlamak istemiştir?

Cevap: Verilen sayının yarısını bulmaya çalışmaktadır. Sayı tek olsa dahi orta noktası bulunmaktadır.

Denetleyici:

```
#include <iostream>
using namespace std;
int main()
{
    int sayi = 13;
    if(sayi % 2 == 0)
        cout << sayi /2;
    else
        cout << (sayi-1) /2;
}
```

Yukarıdaki kodu bilgisayarına yazan bir programcı nasıl bir ekran çıktısı ile karşılaşacaktır. Çıktıyı tahmin ettikten sonra, lütfen kodları bilgisayarınıza yazarak tahmininizi test ediniz.

Cevap: 6

Kodlayıcı: Ali adlı bir öğrenci istediği programı yazmaya başlamadan önce yandaki akış şemasını oluşturmuştur. Fakat akış şemasını koda dönüştürme konusunda yardıma ihtiyacı vardır. Bunun için Ali'ye yardım ediniz.

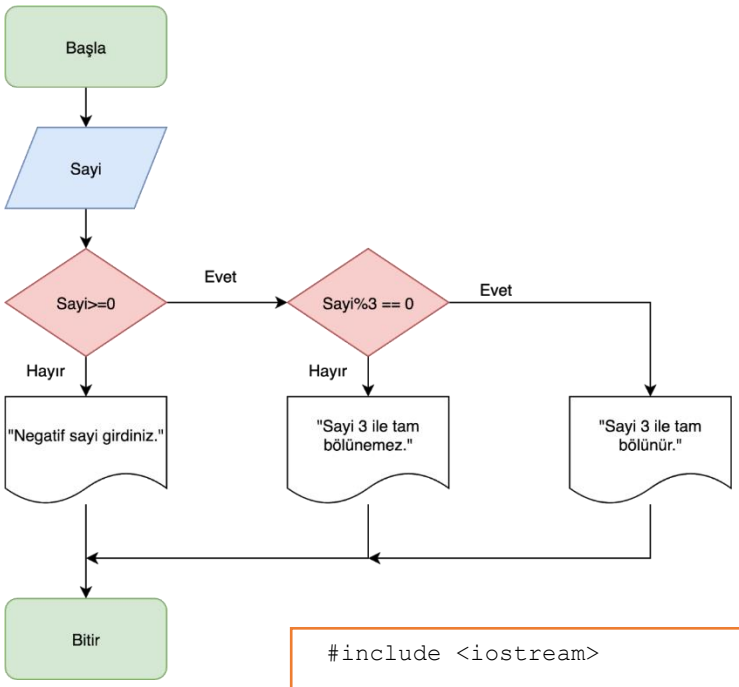
Cevap:

```
#include <iostream>
using namespace std;
int main()
{
    int sayi;
    cin >> sayi;
    if(sayi>10)
        sayi = sayi -10;
    else
        sayi = sayi +10;
    cout << sayi;
}
```

Hafta 5. Ders Materyalleri

O5. B1.1 Karar Yapılarını Tanıyalım Görev Kâğıtları (Toplam 5 görev)

1. Görev



1. **Grup Ekran Çıktısı:**

   ……...

2. **Grup Ekran Çıktısı:**

   ……...

3. **Grup Ekran Çıktısı:**

   ……...

4. **Grup Ekran Çıktısı:**

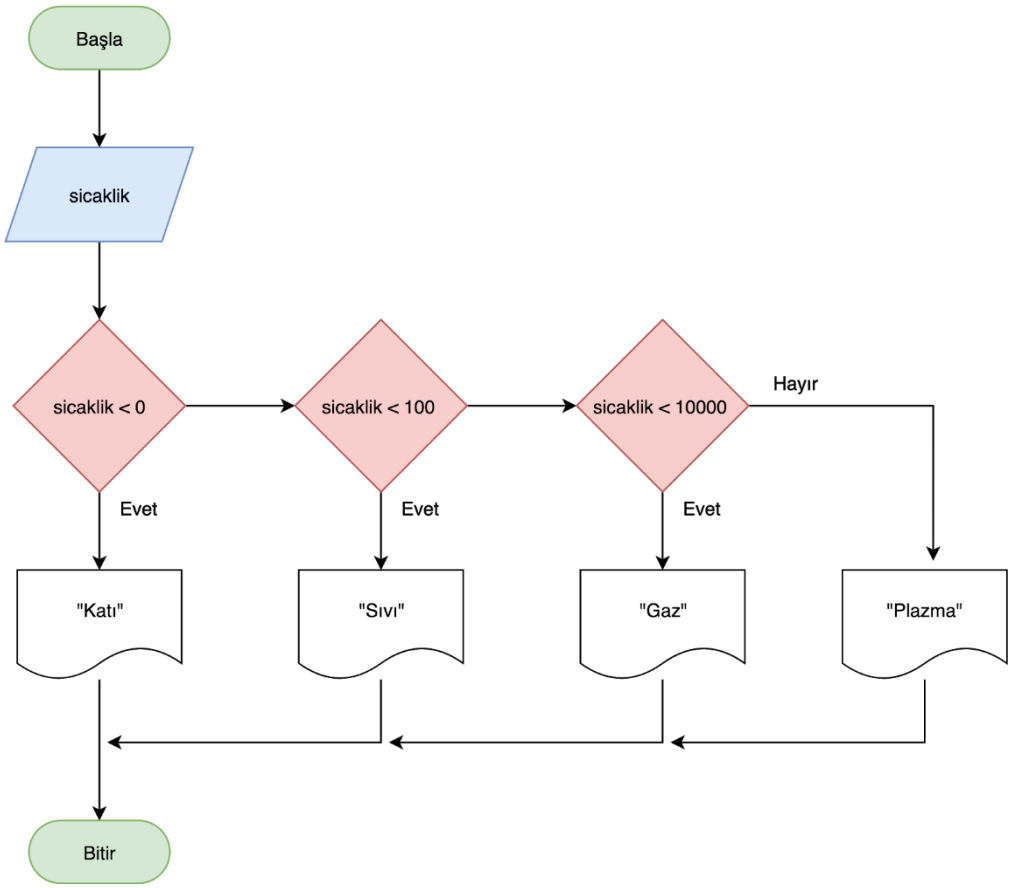
   ……...

5. **Grup Ekran Çıktısı:**

   ……...

```
#include <iostream>
using namespace std;
int main()
{
      int sayi;
      cin >> sayi;

      if(sayi >= 0)
      {
          if(sayi%3 == 0)
              cout << "Sayi 3 ile tam bolunur.";
          else
                  cout << "Sayi 3 ile tam bolunemez.";
      }
      else
      {
          cout << "Negatif sayi girdiniz.";
      }
      return 0;
 }
```

2. Görev



1. **Grup Ekran Çıktısı:**

   ……...

2. **Grup Ekran Çıktısı:**

   ……...

3. **Grup Ekran Çıktısı:**

   ……...

4. **Grup Ekran Çıktısı:**

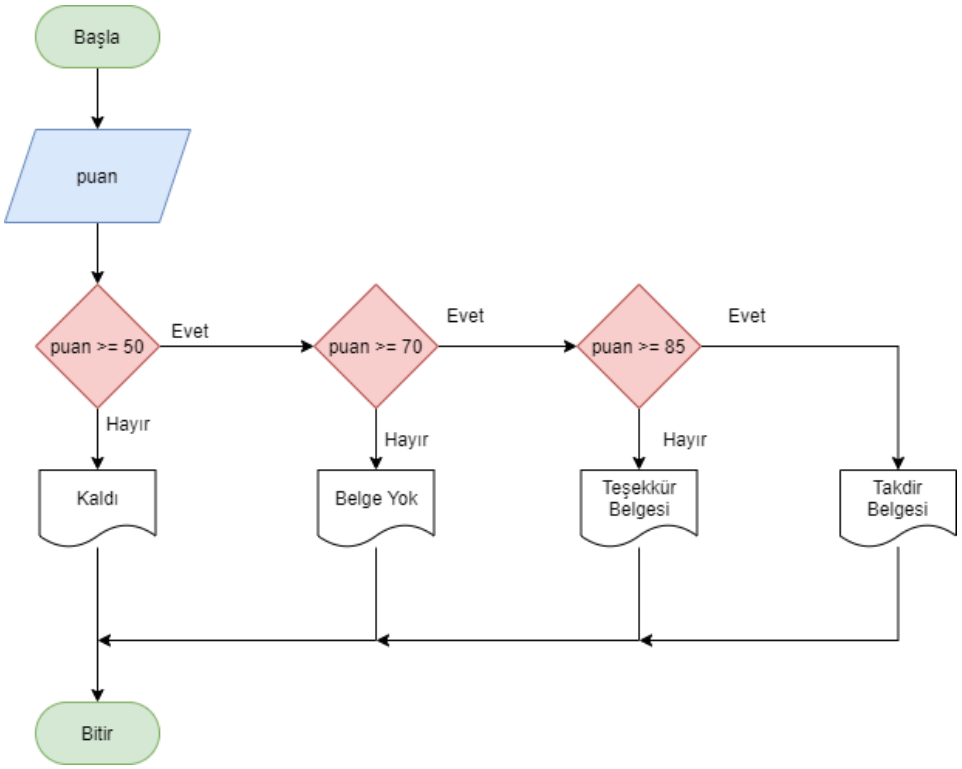
   ……...

5. **Grup Ekran Çıktısı:**

   ……...

```cpp
#include <iostream>
using namespace std;

int main()
{
    int sicaklik;
    cin >> sicaklik;

    if(sicaklik < 0)
        cout << "Kati";
    else if(sicaklik < 100)
        cout << "Sivi";
    else if(sicaklik < 10000)
        cout << "Gaz";
    else
        cout << "Plazma";
    return 0;
}
```

3. Görev



1. **Grup Ekran Çıktısı:**

   ……...

2. **Grup Ekran Çıktısı:**

   ……...

3. **Grup Ekran Çıktısı:**

   ……...

4. **Grup Ekran Çıktısı:**

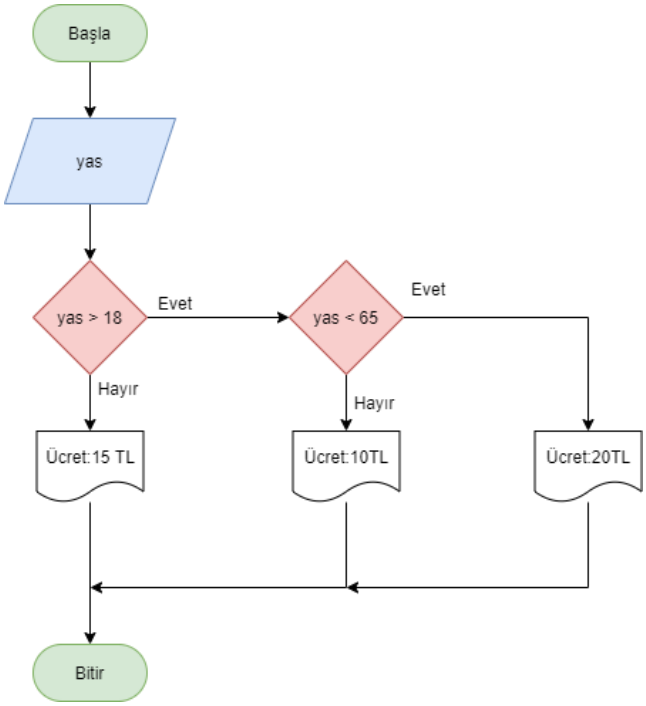
   ……...

5. **Grup Ekran Çıktısı:**

   ……...

```
#include <iostream>
using namespace std;
int main()
{
    int puan;
    cout << "notu giriniz:";
    cin >> puan;
    if(puan<50)
        cout << "kaldi";
    else if(puan<70)
        cout << "belge yok";
    else if(puan<85)
        cout << "tesekkür belgesi";
    else if(puan<=100)
        cout << "takdir belgesi";
    else
        cout <<"Hatali giris";
    return 0;
}
```

4. Görev



1. Grup Ekran Çıktısı:

……

2. Grup Ekran Çıktısı:

……

3. Grup Ekran Çıktısı:

……

4. Grup Ekran Çıktısı:

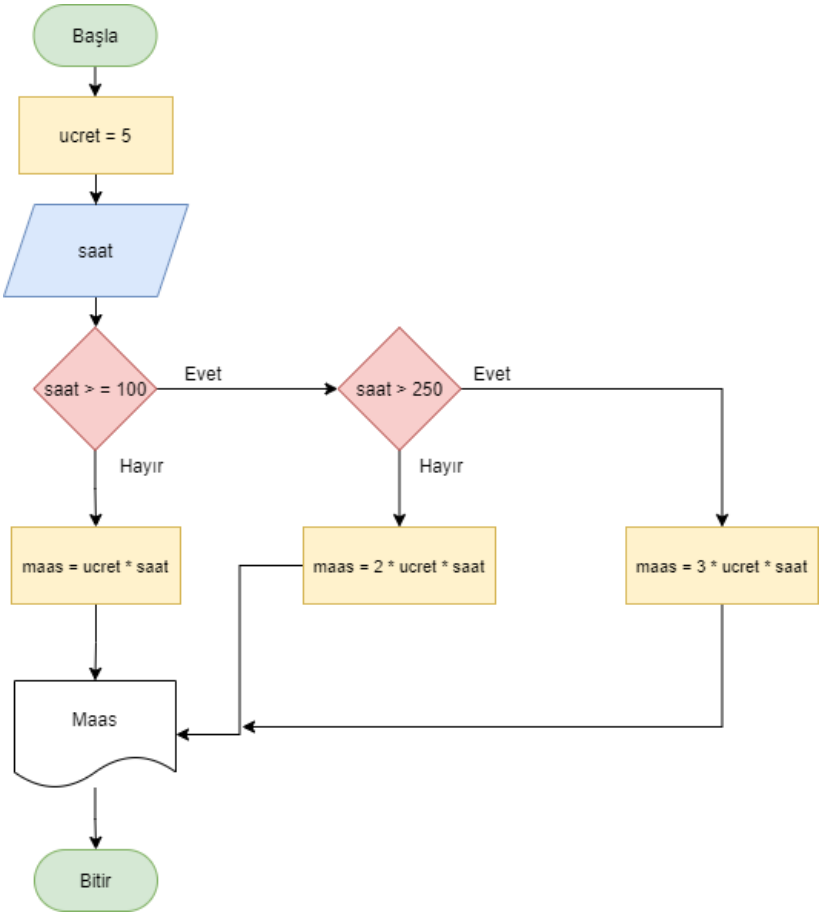
……

5. Grup Ekran Çıktısı:

……

```
#include <iostream>
using namespace std;
int main()
{
    int yas;
    cout << "yasinizi giriniz:";
    cin >> yas;

    if(yas<18)
        cout << "Ucret: 15 TL ";
    else if(yas<65)
        cout << "Ucret: 20 TL ";
    else
        cout << "Ucret: 10 TL";
    return 0;
}
```

5. Görev



1. Grup Ekran Çıktısı:

……

2. Grup Ekran Çıktısı:

……

3. Grup Ekran Çıktısı:

……

4. Grup Ekran Çıktısı:

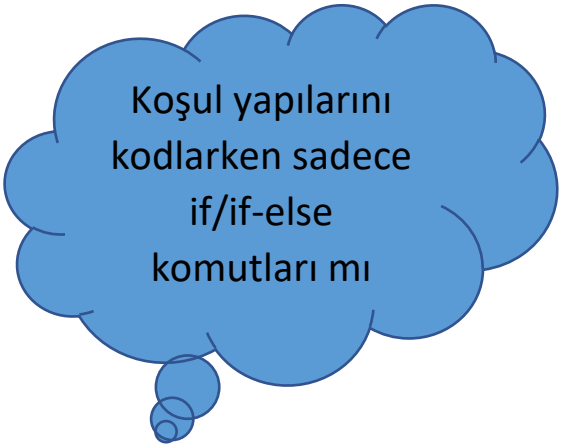
……

5. Grup Ekran Çıktısı:

……

```
#include <iostream>
using namespace std;
int main()
{
    int saat,maas;
    cout << "kac saat calisti:";
    cin >> saat;
    if(saat<100)
        maas = saat * 5;
    else if(saat<250)
        maas = saat * 5 * 2;
    else
        maas = saat * 5 * 3;
   cout << "Maasiniz: "<< maas;
    return 0;
}
```

O5. B3. 1. SWITCH Yapısı Kullanımı



Cevap: Birden çok koşul olduğu durumlarda "switch" bloğunu kullanabiliriz. Gelin if ile switch kullanımını hesap makinesi yapımı için gereken kodlamayı yaparak kıyaslayalım.

If/ If Else Kullanımı

```
#include <iostream>
using namespace std;
int main()
{
    char islem;
    cin >> islem;
    if(islem == '+')
        cout << "Toplama islemi";
    else if(islem == '-')
        cout << "Cikarma islemi";
    else if(islem == '*')
        cout << "Carpma islemi";
    else if(islem == '/')
        cout << "Bolme islemi";
    else
        cout << "Hatali giris.";
}
```

Switch/Case Kullanımı

```
#include <iostream>
using namespace std;
int main()
{
    char islem;
    cin >> islem;
    switch(islem){
    case '+':
        cout << "Toplama islemi";
        break;
    case '-':
        cout << "Cikarma islemi";
        break;
    case '*':
        cout << "Carpma islemi";
        break;
    case '/':
        cout << "Bolme islemi";
        break;
    default:
        cout << "Hatali giris.";
    }
}
```

O5. C. 1. Kısmi Öğrenme Görevleri Afişi

Kısmi öğrenme görevleri afişine ulaşmak için buraya tıklayınız.

Hafta 6. Döngü Yapıları

Kazanımlar

K1. Problem çözme süreçlerinde döngü yapılarını kullanarak algoritma tasarlar.

K2. Problem çözümünde döngü yapısını kullanır.

Amaç

Bu haftanın amacı öğrencilerin programlamada döngü yapılarını kullanarak programın akışını kontrol edebilmelerini sağlamaktır.

Önerilen Ders Akışı

A. Giriş: Sona Kalan Kazanır! (10 dk.)

B. Öğrenme Görevleri

B1. Döngüleri Tanıyalım (40 dk.)

- ❏ Ders Arası (10 dk.)

B2. Döngülerin Ne İçin Kullanıldığını Keşfediyorum (50 dk.)

- ❏ Ders Arası (10 dk.)

B3. Döngü Görevlerini Kodlayalım! (60 dk.)

- ❏ Ders Arası (10 dk.)

C. Kısmi Öğrenme Görevleri (50 dk.)

A. Giriş: Sona Kalan Kazanır!

Süre: 10 dk.

Uygulama: Bütün öğrencilerin oldukları yerde ayağa kalkmalarını ister. Basit sorular soracağını ve Eğitmen bu sorular için cevabı "evet, o benim" diyenlerin yerlerine oturmaları gerektiğini, en son ayakta kalanın ise oyunu kazanacağını belirtir. Ayakta kalan son öğrenciye sürpriz olarak küçük bir hediye verilir. Bu buz kırıcı teknik, basitten karmaşığa sorular sorularak uygulanabilir. Örneğin: Kimin erkek kardeşi var? / Kim kova burcu? / Kim ekim ayında doğdu? / Kimin gözleri kahverengi? / Kimin kolunda saat var?

Eğitmene Öneriler: Süreye bağlı olarak sorular çeşitlendirilip arttırılabilir.

B. Öğrenme Görevleri

B1. Döngüleri Tanıyalım

Süre: 40 dk.

Kazanımlar: K1. Problem çözme süreçlerinde döngü yapılarını kullanarak algoritma tasarlar.

Materyaller: O6. B1.1. Döngüleri Tanıyalım 1

O6. B1.2. Döngüleri Tanıyalım 2

Hazırlık: Eğitmen iki materyali de dört gruba dağıtmak için birer adet çıktısını alır.

Uygulama: Öğrenciler beşerli dört grup olarak çalışmaktadır. Döngülerin mantığını kavratmak için hazırlanan iki görev kâğıdının çıktısı her gruba bir adet olacak şekilde dağıtılır. Birinci görev arının tüm çiçeklerdeki nektarı almak için "ilerle" ve "nektarı al" bloklarının kaç kez kullanılması gerektiğidir. Öğrenciler blok kodları dizdikten sonra ikinci görev kâğıtları dağıtılır ve yine verilen blok kodlar kullanılarak arıların çiçeklerden nektarı alması için gerekli kodu yazması beklenir. Etkinliklerin cevabı aşağıda verilmiştir.

Resim 27. *Çözüm 1*

Resim 28. *Çözüm 2*

Eğitmene Öneriler: Öğrencilerin Görev 1'deki kodları ile Görev 2'deki kodları karşılaştırması istenir ve öğrencilere şu soru sorulur? Görev 1'de 1000 tane çiçek olsaydı ve arıya tüm nektarı aldıracak olsaydık 1000 kez mi ilerler ve nektarı al kodunu kullanırsınız? Öğrencilerin bu soru üzerinde tartışması sağlanarak döngünün neden kullanıldığı hakkında fikir sahibi olması sağlanır. Daha sonra eğitmen aşağıdaki bilgiler doğrultusunda döngüler konusunu özetler.

Döngüler, belirli sayıda bir görevi gerçekleştirmek için ya da programda belli bir durdurma kriteri sağlanana kadar aynı işlemlerin tekrarlanmasına olanak verir. Bir işi 10 kez istenildiği sayıda tekrar ettirmek için ya da istenilen şart sağlanması için döngüler kullanılır.

Döngü olmadan olmaz mı?

Bu bir kaba 12 bardak süt dökmek için 12 tane ayrı bardak kullanmaya benzemektedir. Yapılabilir ama mantıklı değildir.

B2. Döngülerin Neden Kullanıldığını Keşfetme

Süre: 50 dk.

Kazanımlar: K2. Problem çözümünde döngü yapısını kullanır.

Materyaller: O6. B2.1 Döngü Çeşitleri Afişi

O6. B2.2 Döngüleri Neden Yazıyorum

Hazırlık: Eğitmen O6. B2. 1 kodlu materyali ÖYS üzerinden erişime açar. Diğer O6.B2.2. kodlu materyalden ise 10 adet çıktı alır.

Uygulama: Her iki öğrenci için "O6. B2. 2" kodlu materyalin çıktısı alınarak öğrencilerle paylaşılır. Öğrencilerden verilen kodun hangi problemi çözmek için veya hangi amaca yönelik olduğunu bulup yazmaları istenir. Eğitmen destekleyici bilgi olarak döngüler için hazırlanan

afişi (O6. B2. 1) öğrencilerle paylaşarak döngüler hakkında aşağıdaki gibi bir özetleme yapar.

Döngüleri C++ programlama dilinde "for", "while" ve "do-while" komutları ile gerçekleştirebiliriz. Bir döngüyü istediğimiz komut ile yapabiliriz. Üç farklı komut olmasının nedeni ise bazı işlemleri bazı döngüler ile yapmak daha kolay olmaktadır. Döngüler için oluşturulan afiş öğrencilerle paylaşılır.

Eğitmene Öneriler: Yukarıdaki kodlamalar ve problem bulma bittikten sonra döngüler aşağıdaki gibi özetlenir.

Bir döngüde olması gerekenler şunlardır;

1) Kontrol değişkeni: Döngü başlangıç ve bitişi hangi değişken üzerinden kontrol edilecek.
2) Değişken başlangıç değeri
3) Bitiş koşulu
4) Değişken güncellemesi: Her tekrar sonrasında kontrol değişkeni nasıl etkilenecek?
5) Döngü gövdesi: Döngü içerisinde gerçekleştirilecek işlemler.

B3. Döngü Görevlerini Kodlayalım

Süre: 60 dk.

Kazanımlar: K2. Problem çözümünde döngü yapısını kullanır.

Materyaller: O6.B3.1. Döngü Görevlerini Bilgisayarlarımızda Kodlayalım!

Hazırlık: Eğitmen dersin bu bölümü için hazırlanan materyalleri ÖYS üzerinden paylaşır. Code Blocks uygulaması öğrencilerin kodlama yapabilmesi için hazır duruma getirilir.

Uygulama: Öğrenciler ikili gruplar hâlinde bilgisayar başındadır. Eğitmenler döngü görevi etkinliğindeki uygulamalarından iki tanesini öğrencilerin seçmesini sağlayarak, öğrencilerin bu derste kodlama yapmasını ister. Diğer görevler ise öğrencilere kodlanması için ev görevi olarak verilebilir. Öğrenciler kodlama esnasında daha önceki derste hazırlanan afişi destekleyici bilgi olarak kullanılırken, öğrencinin ihtiyaç duyduğu anda gerekli işlemsel bilgiyi eğitmenlerden biri sınıfı dolaşarak, biri de kodlamayı öğrencilerin de görebileceği şekilde bilgisayarda kodlayarak sağlayabilir.

Eğitmene Öneriler: Eğitmen görevleri öğrencilere dağıtmadan önce C++ programlama dilinde nasıl rastgele sayı üretildiği hakkında aşağıdaki gibi çok kısa bir bilgi verir.

C++ programlama dilinde rastgele sayı üretmek için rand() hazır fonksiyonu kullanılır. 0 ila 32767 arasında rastgele sayı üretir. Üst sınırı sınırlandırmak için mod (%) operatörü kullanılır. 0-100 arasında rastgele sayı üretmek istersek rand()%100 şeklinde kullanırız. Alt sınırı arttırmak/azaltmak istersek toplama işlemini kullanırız. Örneğin 10 ile 100 arasında rastgele sayı üretmek için 10 + (rand() % 90) şeklinde kullanırız.

Döngü görevleri ve cevapları aşağıdaki gibidir.

Görev 1: Merve adlı öğrenci 1'den 20'ye kadar olan tek sayıları bulan bir program yazıp ekranda göstermek istemektedir. Merve bunun için sizce nasıl bir kod yazmalı?

CEVAP 1:

For

```
int i;
 for(i=1;i<20;i++)
     if(i%2==1)
         cout << i << endl;
```

While

```
int i=1;
 while(i<20)
 {
     if(i%2==1)
         cout << i << endl;
     i++;
 }
```

Do-While

```
int i=1;
do
{
     if(i%2==1)
         cout << i << endl;
     i++;
}while(i<20);
```

Görev 2: Ali kardeşi Buğra'nın 1'den 30'a kadar 3'erli olarak sayı saymasını istemektedir. Buradan hareketle kardeşinin doğru sayıp saymadığını kontrol etmesi için bilgisayarda 1'den 30'a kadar küçükten büyüğe olacak şekilde 3'e bölünebilecek sayıları ekranda göstermek istiyor. Bunun için nasıl bir kod yazmalı?

Cevap 2:

```
#include <iostream>
using namespace std;
int main()
{
	for(int i=0; i < 30; i++)
		if(i%3==0)
			cout << i <<endl;
	return 0;
}
```

Görev 3: Rafet öğretmen, sınıfında bulunan 10 öğrencinin matematik dersinde aldığı notları klavyeden teker teker girerek sınıfın matematik dersi not ortalamasını bulan bir program yazmak istiyor. Bunun için nasıl bir kod yazmalıdır?

Cevap 3:

```
#include <iostream>
using namespace std;
int main()
{
	int toplam = 0;
	for(int i=0; i < 10; i++)
	{
		int puan;
		cout << i+1 <<". ogrenci puani:";
		cin >> puan;
		toplam += puan;
	}
	cout << "ortalama : " << toplam /10 ;
}
```

Görev 4: Defne öğretmen aldığı 10 tane kitabı sınıfındaki öğrencilere çekiliş yoluyla dağıtmak istemektedir. Sınıftaki öğrencilerin okul numaraları 50 ile 100 arasındadır. Defne öğretmen bunun için 50 ile 100 arasında 10 tane rastgele sayı üreten bir program yazarak çıkan numaraya sahip öğrencilere kitabı vermeyi planlamaktadır. Bunun için nasıl bir kod yazmalıdır?

```
#include <iostream>
#include <cstdlib>
using namespace std;
int main()
```

```
{
        srand(time(0));
        for(int i=0; i<10;i++)
        cout << 50 + rand() % 50 <<endl;
}
```

Görev 5: Ahmet okul kütüphanesindeki raflara herkesin kolayca kitapları bulabilmesi için sayı etiketleri yapıştırmak istiyor. Kütüphanede 30 tane raf olduğu düşünülürse Ahmet'in 1'den 30'a kadar sayıları sıralayıp ekranda göstermesi gerekmektedir. Buradan hareketle Ahmet'in nasıl bir kod yazması gereklidir, bilgisayarımızda kodlayalım.

CEVAP 5:

```
int sayi = 1;
while(sayi<30)
{
cout << sayi <<endl;
sayi ++;
}
```

C. Kısmi Öğrenme Görevleri

Süre: 50 dk.

Materyal: O6. C.1. Kısmi Öğrenme Görevleri Afişi

Hazırlık: Kısmi öğrenme görevleri afişi öğrencilere ÖYS ortamında süreli ödev olarak açılır.

Uygulama: Öğrenciler afişteki görevleri istedikleri sırada ve sayıda kendi tercihlerine bağlı olarak verilen süre içinde uygulamaya başlar. Her bir görevi tamamlayan öğrencilere, göreve ilişkin beceri rozeti verilir. Öğrenciler görev seçimlerini ve tamamladıkları görevleri eğitmene bildirmelidir. Eğitmenler ihtiyaç duydukları zaman öğrencilere görevi tamamlama aşamasında anlık geri bildirimlerde bulunur. Öğrenciler, ÖYS üzerinden süreli ödev olarak açılan kısmi öğrenme görevlerinden kendi tercihlerine göre istedikleri sayıda görev tamamlayıp eğitmene iletir. Süre bitiminde eğitmen görevleri ve yanıtlarını ÖYS ya da GitHub üzerinden öğrencilere gönderir.

Kısmi Öğrenme Görevleri Yanıtlar: Görevler, öğrenciye verilecek beceri rozetleri ile isimlendirilmiştir. Her bir rozetin altında ilgili görevin yanıtları verilmektedir. Eğitmen bu yanıtları uygulama süresinin sonunda GitHub üzerinden öğrencilerle paylaşır.

1) **Analizci**: Aşağıdaki kod ekrana kaç kez merhaba yazar. (Cevap **0**)

```
for(int i=0;i>10;i++)
```

```
    {
        cout << "merhaba" << endl;
    }
```

2) **Kodlayıcı**: Ahmet bilgisayara rastgele 2 sayı ürettirip, üretilen sayıdaki büyük olanı ekrana yazdırmak istiyor. Ahmet'in nasıl bir kod yazması gerekmektedir?

```
#include <iostream>
#include <cstdlib>
#include <ctime>

using namespace std;

int main()
{
    int sayi1, sayi2;
    srand(time(0));


    sayi1 = rand();
    sayi2 = rand();

    if(sayi1>sayi2)
        cout << sayi1;
    else
        cout << sayi2;
}
```

3) **Kodlayıcı**: Bir sınıftaki öğrencilerin numarası 5 ile 25 arasında değişmektedir. Sınıfa giren matematik öğretmeni Sercan, 3 ile tam bölünebilen öğrenci numaralarını bulduran bir program yazmak istediğini belirtmiştir. Siz matematik öğretmeninize kodları nasıl yazarak yardımcı olurdunuz?

```
for(int i=5;i<25;i++)
        if (i%3 == 0)
            cout << i <<endl;
```

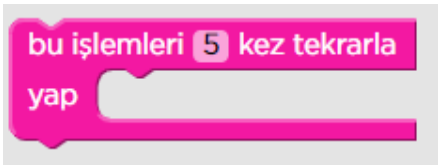
Hafta 6. Ders Materyalleri

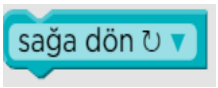
O6. B1.1. Döngüleri Tanıyalım Görev 1

** https://studio.code.org/s/express-2020/lessons/22/levels/1?section_id=2984849*

Yukarıdaki arının tüm çiçeklerdeki nektarı almasını isteseydiniz **blok** kodlardan hangisini kaç kere kullanırdınız?

O6. B1. 2. Döngüleri Tanıyalım Görev 2





** https://studio.code.org/s/express-2020/lessons/22/levels/1?section_id=2984849*

Yukarıdaki arının tüm çiçeklerdeki nektarı almasını isteseydiniz **blok** kodlardan hangisini kaç kere kullanırdınız?

O6. B2.1 Döngü Çeşitleri Afişi

Resim 29. *Döngü çeşitleri afişi*

O6. B2.2 Döngüleri Neden Yazıyorum

Kod

```
for(int sayi=0;sayi<10;sayi++)
```

```
for(int sayi=10;sayi>=0;sayi--)
```

```
for(int sayi=10;sayi>=0;sayi-=2)
```

```
for(int i=15;i>=0;i-=3)
```

```
#include <iostream>
using namespace std;
int main()
{
    for(int i=0;i<10;i++)
        cout << "DENEYAP" << endl;
    return 0;
}
```

```
#include <iostream>
using namespace std;
int main()
{
    for(int i=1; i<10; i++)
    {
        cout << i <<endl;
    }
    return 0;
}
```

```
int sayi = 1;
    while(sayi<100)
    {
        cout << sayi <<endl;
        sayi ++;
    }
```

Problem

Sol kısımda bulunan kodlar hangi temel problem ya da amaç için yazılmıştır? Bu sütunda belirtiniz.

```
int i;
    for(i=1;i<20;i++)
        if(i%2==1)
            cout << i << endl;
```

```
 int i=1;
    while(i<20)
    {
        if(i%2==1)
            cout << i << endl;
        i++;
    }
```

```
int i=1;
    do
    {
        if(i%2==1)
            cout << i << endl;
        i++;
    }while(i<20);
```

O6. B3. 1 Döngü Görevlerini Bilgisayarlarımızda Kodlayalım!

1

Merve adlı öğrenci 1'den 20'ye kadar olan tek sayıları bulan bir program yazıp ekranda göstermek istemektedir. Merve bunun için sizce nasıl bir kod yazmalı?

2

Ali kardeşi Buğra'nın 1'den 30'a kadar 3'erli olarak sayı saymasını istemektedir. Buradan hareketle kardeşinin doğru sayıp saymadığını kontrol etmesi için bilgisayarda 1'den 30'a kadar küçükten büyüğe olacak şekilde 3'e bölünebilecek sayıları ekranda göstermek istiyor. Bunun için nasıl bir kod yazmalı?

3

Caner öğretmen sınıfında bulunan 5 öğrencinin matematik dersinde aldığı notları klavyeden girerek bulunmasını isteyen bir program yapmak istiyor. Bunun için nasıl bir kod yazmalı.

4

Defne öğretmen aldığı 10 tane kitabı sınıfındaki öğrencilere çekiliş yoluyla dağıtmak istemektedir. Sınıftaki öğrencilerin okul numaraları 50 ile 100 arasındadır. Defne öğretmen bunun için 50 ile 100 arasında 10 tane rastgele sayı üreten bir program yazarak çıkan numaraya sahip öğrencilere kitabı vermeyi planlamaktadır. Bunun için nasıl bir kod yazmalıdır.

5

Ahmet okul kütüphanesindeki raflara herkesin kolayca kitapları bulabilmesi için sayı etiketleri yapıştırmak istiyor. Kütüphanede 30 tane raf olduğu düşünülürse
Ahmet'in 1 den 30'a kadar sayıları sıralayıp ekranda göstermesi gerekmektedir. Buradan hareketle Ahmet'in nasıl bir kod yazması gereklidir, bilgisayarımızda kodlayalım.

O6. C. 1. Kısmi Öğrenme Görevleri Afişi

Kısmi öğrenme görevleri afişine ulaşmak için buraya tıklayınız.

Hafta 7. Diziler ve Katarlar

Kazanımlar

K1. C++ programlamada dizi kavramını açıklar.

K2. C++ programlamada tek boyutlu ve çok boyutlu dizileri kullanır.

K3. C++ programlamada dizilere farklı türde değerler atar.

K4. Dizileri döngü içinde kullanır.

K5. Karşılaşılan problemlere dizileri kullanarak çözüm üretir.

K6. Diziler ve katarlar arasındaki farkı ayırt eder.

Amaç

Haftanın amacı, tek boyutlu ve çok boyutlu dizi tanımlama ve diziler üzerinde farklı işlemlerin gerçekleştirilmesi ile örnek çözümlerin öğretilmesi amaçlanmaktadır. Katarların tanımlanması ve kullanımını uygulayarak dizilerden farkını öğrenir.

Önerilen Ders Akışı

A. Giriş: Ekip İşi (15 dk.)

B. Öğrenme Görevleri

B1. Dizileri Tanıyalım! (20 dk.)

B2. Dizilere Değer Verelim! (25 dk.)

- ❏ Ders Arası (5 dk.)
- ❏ Motivasyon Oyunu (10 dk.)

B3. Döngülerle Diziler (20 dk.)

B4. Dizilerle Kodlayalım (25 dk.)

- ❏ Ders Arası (5 dk.)

B5. Kodlama Ekibi (25 dk.)

B6. Farkı Bul (25 dk.)

- ❏ Ders Arası (5 dk.)
- ❏ Motivasyon Oyunu (10 dk.)

C. Kısmi Öğrenme Görevleri (50 dk.)

A. Giriş: Ekip İşi

Süre: 15 dk.

Uygulama: Oyuncular isimlerinin alfabetik sırasına göre sıraya dizilir ve sıradan devam edecek şekilde beşerli gruplar oluşturur. Çeşitli görevlerin yazılı olduğu liste oyuncuların rahatça görebileceği bir yere asılır ya da görev listesi tahtaya yazılır. Oyunculardan 10 dakika içinde listedeki tüm görevlerin bitmiş olması istenir. Saati rahat takip etmeleri için kronometre kullanılabilir. Oyunculara, görev dağılımlarını nasıl yapacakları gibi konularda kendilerinin karar vermeleri istenir. Süre bitiminde, listedeki görevlerin nasıl tamamlandığına bakılır. Gruba ve süreye bağlı olarak görev sayısı ve görevler değişebilir.

Örnek liste:

*Grubun burç haritasını çıkarın.

*Grubun uzmanlık alanlarını sıralayın.

*Grubun ortalama yaşını bulun.

*Herkesin dahil olduğu yaratıcı bir fotoğraf çekilin.

*Kolaylaştırıcılara bir iyilik yapın.

(Kaynak: https://app.ogrenmetasarimlari.com)

B. Öğrenme Görevleri

B1. Dizileri Tanıyalım!

Süre: 20 dk.

Kazanımlar: K1. C++ programlamada dizi kavramını açıklar.

K2. C++ programlamada tek boyutlu ve çok boyutlu dizileri kullanır.

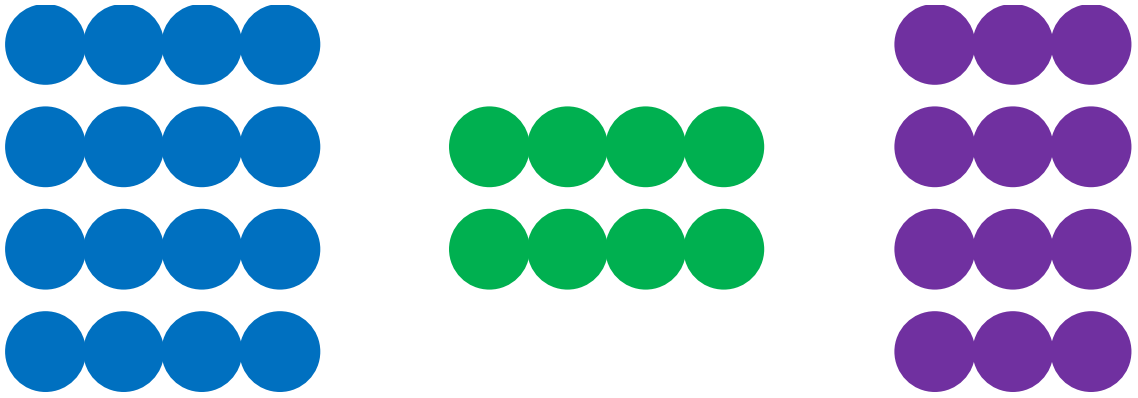
Materyaller: O7. B1. 1. Destekleyici Bilgi: Dizileri Tanıyalım!

Mavi, yeşil ve mor renkli oyun hamurları

Hazırlık: O7. B1. 1. materyali ÖYS üzerinden erişime açılır. Eğitmen ayrıca farklı renklerde oyun hamurları getirir.

Uygulama: Eğitmen öğrencileri ikili gruplar hâlinde çalıştırır. Sınıfta iç içe geçmiş iki çember oluşturulur. İç çemberdekiler tek boyutlu dizi, dış çemberdekiler ise çok boyutlu dizi temsilcileridir. Materyaldeki destekleyici bilgiler ikiye ayrılarak iç çemberdekilere tek boyutlu dizi, dıştakilere ise çok boyutlu dizi materyali dağıtılır. 1 dk. kadar temsilcilerin destekleyici bilgileri incelemeleri istenir. Daha sonra 1 dk. süre içinde iç çemberdekiler, dıştakilere tek boyutlu diziler hakkında anladıklarını aktarır. Eğitmen süre bitişini hatırlatmak için bir çan ya da zil kullanabilir. Daha sonra dış çemberdekilere sıra geçer ve onlar da 1 dk. süre içinde iç çemberdekilere çok boyutlu dizileri açıklamaya çalışır.

Öğrencilerin birbirlerine açıklama yapmalarının ardından eğitmen dış çemberin saat yönünde birer öğrenci yana kayacak şekilde dönmesini ister. Bu şekilde öğrenci çiftleri değiştirilir. Her çifte farklı renklerde ve çeşitli büyüklüklerde oyun hamurları dağıtılır. Öğrenci çiftlerinden bu hamurlar ile aşağıdaki gibi iki boyutlu diziler oluşturmaları istenir.



Resim 30. *Farklı dizi boyutları*

Eğitmen öğrencilerden oluşturdukları dizilerin boyutlarını küçük ve büyük şeklinde değiştirerek, bunları boyut bilgileri ve indis numaraları ile etiketlemelerini ister. Etkinlik sonunda eğitmen grup çalışmalarını takip eder ve özellikle hatalı diziler üzerinden örnekler alarak konuyu özetler.

Eğitmene Öneriler: Etkinlik materyali olarak oyun hamuru yerine farklı renk ve büyüklükte minik ponponlar, farklı şekillerde kesilmiş renkli kâğıtlar ya da madenî paralar ile de yürütülebilir.

B2. Dizilere Değer Verelim!

Süre: 25 dk.

Kazanımlar: K2. C++ programlamada tek boyutlu ve çok boyutlu dizileri kullanır.

K3. C++ programlamada dizilere farklı türde değerler atar.

Materyaller: O7. B2. 1. Destekleyici Bilgiler: Dizilere Değer Verelim!

O7. B2. 2. Görev Kartları: Dizilere Değer Verelim!

Hazırlık: Öğrenciler B1 etkinliğindeki oturma düzeninde iç içe çember şeklinde oturmaya devam eder. Ancak bu sefer iç çemberdekiler çok boyutlu dizi, dış çemberdekiler ise tek boyutlu dizi temsilcisi olarak yer değiştirmektedir. O7. B2. 2. materyali B1 etkinliğindeki gibi ikiye ayrılır. O7. B2. 1. Destekleyici bilgiler ise ÖYS üzerinden erişime açılır.

Uygulama: Eğitmen süreci B1 etkinliğindeki gibi yönetir. İç çemberdekilere çok boyutlu dizilere değer atama materyali verilirken, dış çemberdekilere tek boyutlu dizilere değer atama

materyali verilir. Çiftler birbirlerine birer dakika arayla materyaldekileri açıklamaya çalışır. İlk olarak bir dakika kadar temsilcilerin destekleyici bilgileri incelemeleri istenir. Daha sonra 1 dk. süre içinde iç çemberdekiler, dıştakilere tek boyutlu diziler hakkında anladıklarını aktarır. Eğitmen süre bitişini hatırlatmak için bir çan ya da zil kullanabilir. Daha sonra dış çemberdekilere sıra geçer ve onlar da 1 dk. süre içinde iç çemberdekilere çok boyutlu dizileri açıklamaya çalışır.

Öğrencilerin birbirlerine açıklama yapmalarının ardından eğitmen dış çemberin saat yönünde birer öğrenci yana kayacak şekilde dönmesini ister. Birleşen yeni çiftlere görev kartları verilir. Bu sefer görev kartları verilirken, iç çembere tek boyutlu, dış çembere çok boyutlu dizi görev kartları verilir. Ancak öğrencilerden bu kartlarda yer alan görevleri tamamlarken birbirlerine yardım etmeleri gerektiği belirtilir. Öğrenciler görevleri tamamladıktan sonra kartların üzerine grup cevaplarını ve isimlerini yazarak, eğitmene teslim eder.

Eğitmene Öneriler: Eğitmen aralıklı olarak görevlerin yapılma aşamasında öğrencileri takiptedir. Gerektiğinde yanlış öğrenme ya da açıklamaları engellemek ya da öğrenci sorularını yanıtlamak için anlık geri bildirimlerde bulunur.

B3. Döngülerle Diziler

Süre: 20 dk.

Kazanımlar: K4. Dizileri döngü içinde kullanır.

Materyaller: O7. B3. 1. Destekleyici Bilgi: Döngülerle Diziler

O7. B3. 2. Görev Kartı: Döngülerle Diziler

Hazırlık: Öğrenciler B2 etkinliğindeki oturma düzenindedir. Materyaller ÖYS sisteminde öğrenci erişimine açılır.

Uygulama: Öğrenciler B2 etkinliğinde oturdukları gibi çiftler hâlinde B3 etkinliğine devam etmektedir. İç çemberdekiler çok boyutlu diziyi, dıştakiler ise tek boyutlu diziyi temsil eder. Bu sefer öğrenciler çemberlerin karşısındaki arkadaşı ile değil de yanında oturan arkadaşıyla eşleşir. Tüm öğrencilerden Öğrenme Yönetim sisteminde erişime açılan birinci materyali açmaları istenir. İlk olarak öğrencilerin 2 dakika kadar birinci materyali incelemeleri istenir. Eğitmen burada aşağıdaki açıklamayı yapar.

> *Dizilere ilk değer atamanın diğer bir yolu da döngüleri kullanmaktır. Bunu gerçekleştirmek için, önce diziyi normalde yaptığımız gibi tanımlar ve daha sonra oluşturacağımız döngü içerisinde istediğimiz değerleri atarız. Bunun için ilk materyaldeki örneği inceleyiniz.*

Daha sonra dış çemberde yan yana oturan çiftlere Görev 1, iç çemberde yan yana oturan çiftlere ise Görev 2 üzerinde çalışmaları söylenir. Çiftlerin 5 dk. görev üzerinde çalışmaları beklenir. Eğitmenler anlık geri bildirimler ve görev üzerinde çalışmaya motive etmek için öğrencileri takip etmelidir. Görevler üzerinde çalışma süresi bittikten sonra, öğrencilere "Şimdi

karşınızdaki arkadaşınızla eşleşin ve çalıştığınız görevler hakkında birbirinizle tartışın. Kodunuz üzerinde hatalı olan noktalar varsa bunu birlikte düzeltmeye çalışın." denir. Burada öğrencilerin çiftler hâlinde yaptıklarını artık karşılarındaki arkadaşlarıyla paylaşmaları istenir. Bunun için 5 dk. süre verilir. Süre sonunda eğitmen tahtaya görevlerin doğru yanıtlarını yazar. Eğitmen çiftlerin çalışmalarını gezerek takip eder.

Eğitmene Öneriler: Öğrencilerin yaptıkları görevlerin doğru yanıtları aşağıdaki gibidir:

Görev 1 Yanıtı

```
#include <iostream>
using namespace std;
int main()
{
    int yaslar[5] = {15, 14, 17, 12, 16};
    int i;
    for(i=0; i<5; i++){
        cout << i+1 <<". arkadasimin
yasi: " << yaslar[i] << endl;
    }
    return 0;
}
```

Görev 2 Yanıtı

```
#include <iostream>
using namespace std;
int main(){
   int d_yili[2][3] = {{2005, 2004,
2003},
                       {2008, 2006,
2002}};
   int k = 1;
   for(int i=0; i<2;i++){
      for(int j=0; j<3; j++,k++){
        cout << k << ". arkadasimin dogum
yili: " << d_yili[i][j] << endl;
      }
   }
   return 0;
}
```

B4. Dizilerle Kodlayalım!

Süre: 25 dk.

Kazanımlar: K5. Karşılaşılan problemlere dizileri kullanarak çözüm üretir.

Materyaller: O7. B4. 1. Görev Kodu

Hazırlık: Öğrenciler B3 etkinliğindeki oturma düzeninde iç içe çember şeklinde oturmaya devam eder ve çiftler hâlinde çalışır. Materyaller kesikli çizgiden kesilerek bir torba ya da fanus içerisine atılır. Bir görevden iki tane çıktı alınarak, beş görev toplamda 10 göreve çıkarılır.

Uygulama: Eğitmen, çiftlere torba içinden bir görev kodu seçmelerini ister ve aşağıdaki talimatı verir.

> *Kod içindeki değişkenleri ayırt edin. Bu değişkenlerin nasıl bir veri tutabileceğini düşünün ve onlara daha anlamlı isimler vererek kodu değiştirin. Değiştirdiğiniz kodu bilgisayarda çalıştırıp çıktısını inceleyin. Bu şekilde görev kodunun günlük hayatta hangi probleme çözüm üretebileceği konusunda bir öneri getirin.*

Öğrenciler ikili gruplar hâlinde çıkan görev üzerinde 5 dk. kadar çalışır. Torbada toplam 10 görev kodu olmasına rağmen aynı görev kodundan iki tane bulunmaktadır. Bu nedenle beş dk. sonra aynı kod üzerinde çalışan çiftlerin bir araya gelmeleri istenir. Bu şekilde dörtlü grup olan öğrenciler çalıştıkları görev üzerinde yaptıklarını birbirlerine aktarır. Eğitmen dörtlü grupların 3-5 dk. kadar kendi aralarında tartışmalarına izin vermelidir. Süre sonunda grupların yaptıklarını incelemek için onları dinler, sorularını cevaplar ve yanlış öğrenmelerin önüne geçer. Eğitmen öğrencilerden tüm grupların üzerinde çalıştıkları görevleri ve çözümlerini Öğrenme Yönetim Sisteminde paylaşmalarını ister. Öğrencilerden üzerinde çalıştıkları görev dışında kalan diğer dört görevi evde tamamlamaları istenebilir.

Eğitmene Öneriler: Verilen görevlerin doğru yanıtları aşağıdadır:

```cpp
#include <iostream>

using namespace std;

int main(){

    int dizi1[5], dizi2[5], dizi3[5];

    int i;

    for(i=0; i<5;i++){

        cout << "1. Dizinin " << (i+1) << ". elemanini giriniz: ";

        cin >> dizi1[i];

    }

    cout << endl;

    for(i=0; i<5;i++){

        cout << "2. Dizinin " << (i+1) << ". elemanini giriniz: ";

        cin >> dizi2[i];

    }

    cout << endl;

    for(i=0; i<5;i++){

        dizi3[i] = dizi1[i] + dizi2[i];

        cout << "3. Dizinin " << (i+1) << ". elemani: " << dizi3[i] << endl;

    }

}
```

Görev Kodu 1

Yandaki program kodu iki dizinin karşılıklı olarak elemanlarını toplayıp, yeni diziye kaydetmeyi amaçlamaktadır.

```cpp
#include <iostream>
using namespace std;
int main()
{
    int sayilar[] = {19, 11, 21, 13, 15};
    int i, x, y, n = 5;
    cout << "Dizi: ";
    for(i=0; i < n; i++)
        cout << sayilar[i] << " ";
    x = sayilar[0];
    y = 0;
    for(i=1; i < n; i++){
        if(x < sayilar[i]){
            x = sayilar[i];
            y = i;
        }
    }
    cout << x << " ve " << y;
}
```

Görev Kodu 2

Yandaki program dizideki en büyük elemanı ve bu elemanın indis numarasını yazdırmayı amaçlamaktadır.

```cpp
#include <iostream>
using namespace std;
int main()
{
    int dizi[] = {12, 67, 78, 45, 78, 78, 32, 16, 16, 57};
    int i, j, n = 10;
    cout << "Dizi: ";
    for(i = 0; i < n; i++)
        cout << dizi[i] << " ";
    cout  << "\nElemanlar: ";
    for(i = 0; i < n-1; i++)
        if(dizi[i] == dizi[i+1])
            cout <<  dizi[i] << " ";
}
```

Görev Kodu 3

Yandaki program dizide ardışık olarak tekrar eden elemanları bulmayı amaçlamaktadır.

```cpp
#include <iostream>
using namespace std;
int main()
{
    int dizi[100];
    int i, boyut, tek=0, cift=0;
    cout << "Eleman sayisini girin : ";
    cin >> boyut;
    cout<<"\nDizi elemanlarini girin :\n";
    for(i=0; i<boyut; i++){
        cout << "Elemani girin dizi[" << i << "] : ";
        cin >> dizi[i];
    }
    for(i=0; i<boyut; i++){
        if(dizi[i]%2==0)
            cift++;
        else
            tek++;
    }
    cout << "\nCift eleman sayisi: " << cift;
    cout << "\nTek eleman sayisi: " << tek;
    return 0;
}
```

Görev Kodu 4

Yandaki program girilen dizide tek ve çift sayı adedini ekrana yazdırmayı amaçlamaktadır.

```cpp
#include <iostream>
using namespace std;
int main()
{
    int sayilar[100];
    int i, n, bol5=0;
    cout << "Eleman sayisini girin : ";
    cin >> n;
    cout<<"\nDizi elemanlarini girin :\n";
    for(i=0; i<n; i++){
```

Görev Kodu 5

Girilen dizide 5'e bölünebilen sayı adedini ekrana yazdırmayı amaçlamaktadır.

```
        cout << "Elemani girin dizi[" << i << "] : ";
        cin >> sayilar[i];
    }
    for(i=0; i<n; i++){
        if(sayilar[i]%5==0)
            bol5++;
    }
    cout << "\n5 ile bolunebilen eleman sayisi: " << bol5;
    return 0;
}
```

B5. Kodlama Ekibi!

Süre: 25 dk.

Kazanımlar: K5. Karşılaşılan problemlere dizileri kullanarak çözüm üretir.

Materyaller: O7. B5. 1. Problem

O7. B5. 2. Görev Kartı

Hazırlık: Problem öğrenme yönetim sisteminden materyal öğrenci erişimine açılır. İkinci materyalden ise 10'ar tane çıktı alınır. Bu materyal kesikli çizgilerden kesilerek kart hâlinde hazırlanır.

Uygulama: Eğitmen bu etkinlikte öğrencilerin kodlama ekibi oluşturarak, bir problemi birlikte kodlamalarını bekler. Bunun için dört kişiden oluşan grupları kendi içlerinde çiftlere ayırır ve iş bölümü yaptırır. Eğitmen ilk etapta öğrencileri iç içe geçmiş çember şeklinde oturtur. İç çemberde yer alanlar çiftler hâlinde eşleşmiştir. Dış çemberdekiler de aynı şekildedir. İç çemberdekilere iç çember görevi, dıştaki çiftlere ise dış çember görevi verilir. Eğitmen öğrencilere aşağıdaki açıklamayı yapar.

> *İç çember ve dış çember görevleri birbirini tamamlayan kod yazma görevidir. İki görev birleştirilince bir problemin kodlarını oluşturmaktadır. Probleme öğrenme yönetim sisteminden erişebilirsiniz. Görevleri yaparken dikkat edeceğiniz önemli bir nokta ise şudur: Problem tek olduğu için değişken isimlerinin ortak olmasına dikkat edin.*

5 dk. kadar öğrenciler görevler üzerinde çalışır. Daha sonra eğitmen çiftlerin karşılarında oturan dış çember çiftiyle eşleşmelerini ister. Bu şekilde gruplar artık dört kişiye ulaşmıştır. Dört kişilik ekiplere kodlama ekibi görev kartı verilir. Artık bu grup, temel problemi çözecek kodlama ekibini oluşturur. Kodlama ekipleri yazdıkları kod satırlarını birleştirerek bir araya getirir ve kodu bilgisayar ortamında çalıştırır. Eğitmen sonuca ulaşamayan grupların, tamamlayan gruplardan destek almalarını sağlar.

Eğitmene Öneriler: İç çember ve dış çember grupları, sınıfın düzenine göre öğrenci mevcudunun ikiye bölünmesi ile grup 1 ve grup 2 şeklinde de oluşturulabilir. Problemin kodlarını aşağıda bulabilirsiniz. Dört kişilik grup toplandığında, turuncu kod satırlarına iç çemberdekilerin, sarı kod satırlarına dış çemberdekilerin erişmesi beklenir. Yeşil kod satırları ve diğer tamamlamalar ise kodlama ekibinin görevidir.

```
1.  #include<iostream>
2.  using namespace std;
3.  int main()
4.  {
5.      int i, j, satir=3, sutun=3, matris1[3][3], matris2[3][3], sonuc[3][3];
6.      for(i = 0; i < satir; i++) {
7.          for(j = 0; j < sutun; j++) {
8.              cout << "1. Matrisin [" <<i<< "," <<j<<"]. elemanini giriniz: ";
9.              cin >> matris1[i][j];
10.         }
11.     }
12.     cout << endl;
13.     for(i = 0; i < satir; i++) {
14.         for(j = 0; j < sutun; j++) {
15.             cout << "2. Matrisin [" <<i<< "," <<j<<"]. elemanini giriniz: ";
16.             cin >> matris2[i][j];
17.         }
18.     }
19.     cout<<"\nSonuc Matrisi: " << endl;
20.     for(i = 0; i < satir; i++) {
21.         for(j = 0; j < sutun; j++) {
22.             sonuc[i][j] = matris1[i][j] + matris2[i][j];
23.             cout << sonuc[i][j] << " ";
24.         }
25.         cout << endl;
26.     }
27. }
```

B6. Farklı Bul!

Süre: 25 dk.

Kazanımlar: K6. Diziler ve katarlar arasındaki farkı ayırt eder.

Materyaller: O7. B6. 1. Farkı Bul!

Hazırlık: Öğrenciler ikili gruplar hâlinde oturmaktadır. Öğrenme yönetim sisteminden materyal öğrenci erişimine açılır.

Uygulama: Öğrenciler materyali sistem üzerinden açar. Gruplara materyaldeki kodun basamaklarını adım adım çalıştırarak incelemeleri ve kod çıktısını kontrol etmeleri istenir. Eğitmen materyal öncesi şu açıklamayı yapar.

> *Koda bakacak olursanız klavyeden girilen karakterlerden sırasıyla "Arda" ismini karakter dizisi kullanarak, "Duru" ismini ise "katar" kullanarak ekrana yazdırıyoruz. Buradaki dizi ve katar arasındaki farkı kodun basamaklarını tek tek inceleyerek tahmin etmeye çalışın.*

Eğitmen çiftlere 5 dk. kadar tartışma süresi tanır. Her çiftten tahminlerini isimlerinin yazılı olduğu kâğıda yazmaları istenir. Tahminler yazıldıktan sonra, öğrenciler kâğıtlarını tahtaya yapıştırır. Beyin fırtınası eşliğinde tüm tahminler eğitmen tarafından okunur ve aradaki fark açıklanır.

Eğitmene Öneriler: Eğitmen öğrenci tahminlerini sınıfta tahtaya yapıştırmak yerine, padlet.com kullanarak dijital tartışma panosu da oluşturabilir. Eğitmen aradaki farkı açıklarken aşağıdaki içerikten yararlanabilir.

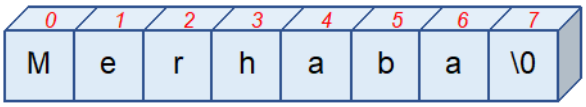
> *Programlamada metin türünde verilerimizi saklamak için kullanılan özel karakter dizileridir. Katarlar, null ('\0') karakter ile sonlandırılmış tek boyutlu karakter dizileri olarak tanımlanabilir. Aşağıda verilen örnekte, "Merhaba" sözcüğünden oluşan bir katar oluşturulmaktadır. Normalde verilen kelime 7 harften oluşsa da sondaki null karakteri tutmak için de bir karakterlik alan gerektiği için bellekte toplamda 8 karakterlik alan ayrılması gerekmektedir.*

```
char kelime[] = {'M', 'e', 'r', 'h', 'a', 'b', 'a', '\0'};
```

Dizilerdeki ilk değer atama yöntemlerini hatırlarsanız aşağıdaki gibi bir ilk değer ataması yapabiliriz. Dizi değişkeni, çift tırnak işareti içine alınmış bir karakter dizisi içerir.

```
char kelime[] = "Merhaba";
```

Aslında yukarıdaki tanımlamada gördüğünüz üzere null karakteri bir katar sabitinin sonuna yerleştirmezsiniz. C++ derleyicisi diziyi oluşturduğunda '\0' değerini dizinin sonuna otomatik olarak ekler. Yukarıdaki tanımlama sonucu bellekte şu şekilde bir yerleşim söz konusu olacaktır.



C. Kısmi Öğrenme Görevleri

Süre: 50 dk.

Materyal: O7. C. 1. Kısmi Öğrenme Görevleri Afişi

Hazırlık: Kısmi öğrenme görevleri afişi öğrencilere ÖYS ortamında süreli ödev olarak açılır.

Uygulama: Öğrenciler afişteki görevleri istedikleri sırada ve sayıda kendi tercihlerine bağlı olarak verilen süre içinde uygulamaya başlar. Her bir görevi tamamlayan öğrencilere, göreve ilişkin beceri rozeti verilir. Öğrenciler görev seçimlerini ve tamamladıkları görevleri eğitmene bildirmelidir. Eğitmenler ihtiyaç duydukları zaman öğrencilere görevi tamamlama aşamasında anlık geri bildirimlerde bulunur. Öğrenciler, ÖYS üzerinden süreli ödev olarak açılan kısmi öğrenme görevlerinden kendi tercihlerine göre istedikleri sayıda görev tamamlayıp eğitmene iletir. Süre bitiminde eğitmen görevleri ve yanıtlarını ÖYS ya da GitHub üzerinden öğrencilere gönderir.

Kısmi Öğrenme Görevleri Yanıtlar: Görevler, öğrenciye verilecek beceri rozetleri ile isimlendirilmiştir. Her bir rozetin altında ilgili görevin yanıtları verilmektedir. Eğitmen bu yanıtları uygulama süresinin sonunda GitHub üzerinden öğrencilerle paylaşır.

Tasarlayıcı: Fadime, 10 arkadaşının Facebook beğeni sayılarının toplamını ve beğeni ortalamalarını merak etmektedir. Bu amaçla bir program yazmak ister, ancak yardıma ihtiyacı vardır. Fadime için bu programı sen tasarlar mısın?

```
#include <iostream>
using namespace std;
int main()
{
    int sayilar[] = {17, 13, 12, 9, 6, 11, 3, 14, 2, 19};
    int toplam = 0, n = 10;
```

```
    float ort;
    cout << "Dizi: " << endl;
    for (int i=0; i < n; i++) {
        cout << sayilar [i] << " ";
        toplam += sayilar[i];
    }
    cout << "\nDizinin toplami: " << toplam << endl;
    ort = (float)toplam / n;
    cout << "Dizinin ortalamasi: " << ort << endl;
    return 0;
}
```

Analizci: Arkadaşın (bilgisayar) aklından 1-9 arasında rastgele bir sayı tutar. Sen de tutulan bu sayıyı 3 tahminde bulmaya çalışan bir program yazıyorsun. Kural gereği tutulan sayıyı 3 tahminde bulamazsan oyun sona erer. Eğer 3 tahminden birinde sayıyı bulursan program tutulan sayıyı kaçıncı tahmin hakkında bulduğunu ekrana yazdırır.

```
#include <iostream>
#include <ctime>
using namespace std;
int main()
{
    int sayi;
    int tahmin = -1;
    int tahmin_sayisi = 0;
    int tahmin_limiti = 3;
    bool outOfGuesses = false;
    srand(time(NULL));

    sayi = rand() % 9 + 1;
    cout << sayi;
    while(tahmin != sayi && tahmin_sayisi < tahmin_limiti){
        cout << "Tahmininizi girin: ";
        cin >> tahmin;
        tahmin_sayisi++;
    }
```

```
    if(tahmin == sayi){
        cout << "Tebrikler, " << tahmin_sayisi <<". denemede kazandiniz!" << endl;
    } else {
        cout << "Uzgunum, 3 hakkinizda bilemediniz!" << endl;
    }
    return 0;
}
```

Kodlayıcı: Aşağıdaki matris yapısında voleybol oyuncularının numaraları verilmektedir. Koç, oyuncuları iki maç öncesi aşağıdaki gibi aynı sırada görmek istiyor. İki maçtada sıralamanın aynı olduğundan emin olmak için bir kod tasarlamayı düşünüyor, ancak yardıma ihtiyacı var. Koç için bu programı sen tasarlar mısın? Program içerisinde tanımlama bölümünde oyuncuların ilk dizilimi aşağıdaki matristeki gibi olmalıdır.

1. Maç Oyuncu Sırası:

1	3	5	7	9
11	13	15	17	19
2	4	6	8	10
12	14	16	18	20

2. Maç Oyuncu Sırası:

1	3	5	7	9
11	13	15	17	19
2	4	6	8	10
12	14	16	18	20

```
#include <iostream>
using namespace std;
int main()
{
    int matris1[4][5] = { {1, 3, 5, 7, 9},
                          {11, 13, 15, 17, 19},
```

```
                    {2, 4, 6, 8, 10},
                    {12, 14, 16, 18, 20}};

    int matris2[4][5] = { {1, 3, 5, 7, 9},
                    {11, 13, 15, 17, 19},
                    {2, 4, 6, 8, 10},
                    {12, 14, 16, 18, 20}};
    bool durum = true;
    int i, j;
    for (i = 0; i < 4; i++)
        for (j = 0; j < 5; j++)
            if (matris1[i][j] != matris2[i][j])
                durum = false;
    if (durum)
        cout << "Sıralama aynı";
    else
        cout << "Sıralama farklı";
    return 0;
}
```

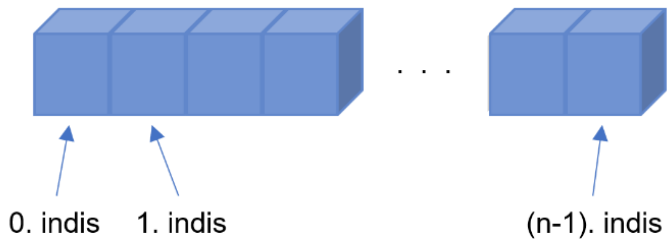
Hafta 7. Ders Materyalleri

O7. B1. 1. Destekleyici Bilgi: Dizileri Tanıyalım

Diziler: Dizi, tek bir veri parçasını depolayabilen klasik bir değişkenin aksine, birden çok veri öğesini depolayabilen bir veri yapısıdır. Diziler tek boyutlu ve çok boyutlu olmak üzere ikiye ayrılır.

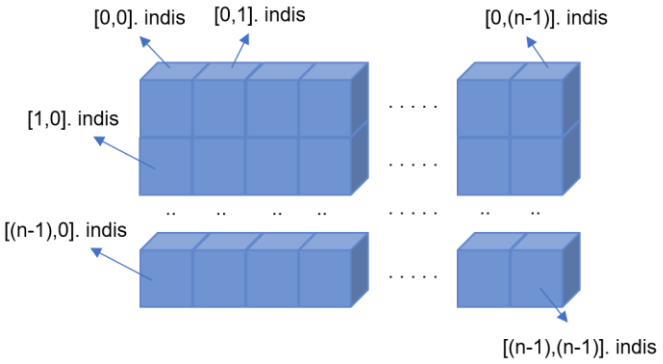
İndis: Bir dizi, bir veri kümesi tutar. Kümenin her üyesine bir eleman denir. İndis, dizinin hangi elemanına eriştiğinizi gösteren bir sayıdır.

Tek Boyutlu Diziler: Tek bir veri türü içeren ve birden fazla değişkeni bir arada tutmaya yarayan veri yapısıdır.



UYARI: Dizilerin indislerinin numaralandırması sıfırdan başlar. Bu nedenle n elemandan oluşan tek boyutlu bir dizideki son elemanın indisi n değil (n-1) olur.

Çok Boyutlu Diziler: Dizilerin elemanları da bir dizi tutabilir. Bu dizileri ifade etmek için dizilerin dizisi ya da dizilerden oluşan diziler ifadesi kullanılır. Aşağıda iki boyutlu dizi örneği verilmektedir.



UYARI: Ayrıca çok boyutlu bir dizide saklanabilecek toplam eleman sayısı, tüm boyutların çarpımı ile hesaplanabilir. 3*2, 6 elemanlı dizi anlamına gelir.

5 elemanlı tek boyutlu dizi

İndisler: **0** **1** **2** **3** **4**

2*4'lük 8 elemanlı iki boyutlu dizi

İndisler: **0, 0** **0, 1** **0, 2** **0, 3**

1, 0 **1, 1** **1, 2** **1, 3**

3 elemanlı tek boyutlu hatalı dizi: Dizi aynı veri türünde olmalıdır. 	2*3'lik iki boyutlu ancak hatalı dizi: Dizi aynı veri türünde olmalıdır. 

O7. B2. 1. Destekleyici Bilgi: Dizilere Değer Verelim!

Tek Boyutlu Diziler: Tek bir veri türü içeren dizilerdir.	***Çok Boyutlu Diziler:*** Bir veya daha fazla dizi içeren dizilerdir.

veri_tipi dizi_adi[eleman_sayisi];
 	
Beş adet öğrenciye ait öğrenci numarasını saklamak için oluşturulan "numaralar" dizisine değer atamak için:
 	
`int numaralar[5];`
 	
Uyarı! Numaralar dizisi bir boyutludur ve 5 elamana sahiptir. Dizi içerisindeki herhangi bir elemana ulaşmak için dizi adı ve ardından eleman indis numarasını içeren köşeli parantezler kullanılır. Yukarıda tanımlanan numaralar dizisinin beşinci elemanına erişmek için şu şekilde bir ifade yazılır:
 	
`numaralar[4]`
	veri_tipi dizi_adi[boyut1][boyut2]…[boyutN];
	
	Dört adet öğrencinin bir dersten aldığı iki yazılı notunu saklamak için oluşturulan "notlar" dizisine değer atamak için:
	
	`int notlar[4][2] = {{90, 70},`
	`{50, 80},`
	`{85, 86},`
	`{50, 70}};`
	
	Uyarı! Notlar dizisi 4*2 = 8 elemana sahiptir. Dizi içerisindeki herhangi bir elemana ulaşmak için dizi adı ve ardından eleman indis numarasını içeren köşeli parantezler kullanılır.
	
	sayilar [2][2] dizisi için eleman indis numarası;
	sayilar[0][0] : birinci satır ilk eleman
	sayilar[0][1] : birinci satır ikinci eleman
	sayilar[1][0] : ikinci satır birinci eleman

	sayilar[1][1] : ikinci satır ikinci eleman
veri_tipi dizi_adi[eleman_sayisi] = {değer1, değer2, değer3, …}; Beş adet öğrenci adlarının ilk harflerini saklamak için oluşturulan "isim" dizisine değer atamak için: **char** isim [5] = {'H', 'K', 'A', 'R', 'S'}; **char** isim [] = {'H', 'K', 'A', 'R', 'S'}; **Uyarı!** Dizinin ilk değerlerini bu şekilde atarsanız dizi eleman sayısı alanını boş bırakabilirsiniz. Derleyici, bu şekilde dizinin elemanları verildiği zaman dizi boyutunun ne olacağına kendisi karar verecektir.	veri_tipi dizi_adi[boyut1][boyut2]…[boyutN] = {değer1, değer2,..değerN}; Beş adet öğrenci ad ve soyadlarının ilk harflerini saklamak için oluşturulan "harfler" dizisine değer atamak için: **char** harfler[2] [5] = {{'H', 'K', 'A', 'R', 'S'}, {'M', 'N', 'P', 'S', 'D'} }; **Uyarı!** Dizi 2*5 = 10 elemana sahiptir ve başlangıç değeri ataması gerçekleştirilmiştir. Değer atamada bu yöntem satır ve sütunları gösterdiği için daha çok tercih edilir.
int numara[5] = {90, 70, 50}; **int** notlar[5] = {90, 70, 50, 0, 0}; **Uyarı!** Dizi ilk değer atamasında eleman sayısı 5'tir. Ancak diziye sadece üç değer girilmiş, yani eleman sayısı kadar değer girilmemiştir. Bu durumda 5 elemanlı dizinin 4. ve 5. elemanlarının değeri 0 olacaktır. Yukarıdaki iki tanımlamada aynıdır.	**int** sayilar [2][3][5]; Yukarıda tanımlanan *sayilar* dizisi 2 * 3 * 5 = 30 elemana sahiptir. **Uyarı!** Çok boyutlu bir dizide istediğiniz sayıda boyuta sahip olabilirsiniz. Ancak, çok fazla boyut tanımlamanız bilgisayarın belleğini hızlı bir şekilde doldurabilir. Çok boyutlu dizilerin en basit formu iki boyutlu dizilerdir. İki boyutlu bir dizi dizi elemanlarının da bir dizi olması hâlidir.

O7. B2. 2. Görev Kartları: Dizilere Değer Verelim!

Tek Boyutlu Diziler	*Çok Boyutlu Diziler*
Sıra Sizde!	**Sıra Sizde!**

`int notlar[5] = {90, 70, 50, 80, 85};`
 	
Örnekteki notlar dizisine benzer bir başka dizide siz oluşturun ve diziye değer atayın. Yazdığınız dizinin eleman sayısını birlikte tartışın.
	`int notlar[2][5]={{90, 70, 50, 80, 85},`
	`{86, 50, 70, 90, 95} };`
	
	Örnekteki notlar dizisine benzer bir başka dizide siz oluşturun ve diziye değer atayın. Yazdığınız dizinin eleman sayısını birlikte tartışın.

Sıra Sizde!
 	
`char harfler[] = {'H', 'K', 'A', 'R', 'S'};`
 	
Örnekteki harfler dizisinde 2. indis'ten sonra gelen dizi elemanı hangisidir?
	Sıra Sizde!
	
	`char harfler[2] [5] = {{'H', 'K', 'A', 'R', 'S'},`
	`{'M', 'N', 'P', 'S', 'D'} };`
	
	Örnekteki harfler dizisinde (1,2) numaralı indis'ten (2.satır 3. sütun olmakta) önce gelen dizi elemanı hangisidir?

O7. B3. 1. Destekleyici Bilgi: Döngülerle Diziler

Problem: Ayşe öğretmen sınıfındaki beş öğrencisinin Yabancı Dil sınav notlarını bir program kullanarak listelemek istemektedir. Bunun için öğrencilerinden biri aşağıdaki kodları tasarlamaktadır.

```
#include <iostream>
using namespace std;
int main(){
    int notlar[5];
    int i;
    for(i=0; i<5; i++){
        cout << i+1 << ". ogrenci notunu giriniz: ";
        cin >> notlar[i];
    }
    return 0;
}
```

Kodun Çıktısı:

1. ogrenci notunu giriniz: *75*
2. ogrenci notunu giriniz: *65*
3. ogrenci notunu giriniz: *90*
4. ogrenci notunu giriniz: *87*
5. ogrenci notunu giriniz: *81*

O7. B3. 2. Görev Kartı: Döngülerle Diziler

Problem: 5 arkadaşın yaş bilgilerini değişkende saklama ve ekrana yazdırma

```
#include <iostream>
using namespace std;
int main(){
    int yas1 = 15;
    int yas2 = 14;
    int yas3 = 17;
    int yas4 = 12;
    int yas5 = 16;
    cout << "1. arkadasimin yasi: " << yas1 << endl;
    cout << "2. arkadasimin yasi: " << yas2 << endl;
    cout << "3. arkadasimin yasi: " << yas3 << endl;
    cout << "4. arkadasimin yasi: " << yas4 << endl;
    cout << "5. arkadasimin yasi: " << yas5 << endl;
    return 0;
}
```

Kodun Çıktısı:

1. arkadasimin yasi: 15
2. arkadasimin yasi: 14
3. arkadasimin yasi: 17
4. arkadasimin yasi: 12
5. arkadasimin yasi: 16

Görev dış çember: Yukarıdaki problemin çözümünü, kod çıktısı aynı olacak şekilde tek boyutlu dizi kullanarak yeniden programlayınız.

Görev iç çember: Yukarıdaki problemin çözümünü, kod çıktısı aynı olacak şekilde çok boyutlu dizi kullanarak yeniden programlayınız.

O7. B4. 1. Görev Kodu

```
#include <iostream>
using namespace std;
int main(){
    int dizi1[5], dizi2[5], dizi3[5];
    int i;
    for(i=0; i<5;i++){
        cout << "1. Dizinin " << (i+1) << ". elemanini giriniz: ";
        cin >> dizi1[i];
    }
    cout << endl;
    for(i=0; i<5;i++){
        cout << "2. Dizinin " << (i+1) << ". elemanini giriniz: ";
        cin >> dizi2[i];
    }
    cout << endl;
    for(i=0; i<5;i++){
        dizi3[i] = dizi1[i] + dizi2[i];
        cout << "3. Dizinin " << (i+1) << ". elemani: " << dizi3[i] << endl;
    }
    return 0;
}
```

Görev Kodu 1

```cpp
#include <iostream>
using namespace std;
int main()
{
    int sayilar[] = {19, 11, 21, 13, 15};
    int i, maks, yer, n = 5;
    cout << "Dizi: ";
    for(i=0; i < n; i++)
        cout << sayilar[i] << " ";
    maks = sayilar[0];
    yer = 0;
    for(i=1; i < n; i++){
        if(maks < sayilar[i]){
            maks = sayilar[i];
            yer = i;
        }
    }
    cout << "\nDizinin en buyuk elemani " << yer <<". indisteki " << maks;
  return 0;
}
```

Görev Kodu 2

```cpp
#include <iostream>
using namespace std;
int main()
{
    int dizi[] = {12, 67, 78, 45, 78, 78, 32, 16, 16, 57};
    int i, j, n = 10;
    cout << "Dizi: ";
    for(i = 0; i < n; i++)
        cout << dizi[i] << " ";

    cout  << "\nTekrar eden elemanlar: ";
    for(i = 0; i < n; i++)
        if(dizi[i] == dizi[i+1])
            cout <<  dizi[i] << " ";
    return 0;
}
```

Görev Kodu 3

```cpp
#include <iostream>
using namespace std;
int main()
{
    int dizi[100];
    int i, boyut, tek=0, cift=0;
    cout << "Dizi boyutunu girin : ";
    cin >> boyut;
    cout<<"\nDizi elemanlarini girin :\n";
    for(i=0; i<boyut; i++){
        cout << "Elemani girin dizi[" << i << "] : ";
        cin >> dizi[i];
    }
    for(i=0; i<boyut; i++){
        if(dizi[i]%2==0)
            cift++;
        else
            tek++;
    }
    cout << "\nCift eleman sayisi: " << cift;
    cout << "\nTek eleman sayisi: " << tek;
    return 0;
}
```

Görev Kodu 4

```cpp
#include <iostream>
using namespace std;
int main()
{
    int sayilar[100];
    int i, n, bol5=0;
    cout << "Dizi boyutunu girin : ";
    cin >> n;
    cout<<"\nDizi elemanlarini girin :\n";
    for(i=0; i<n; i++){
        cout << "Elemani girin dizi[" << i << "] : ";
        cin >> sayilar[i];
    }
    for(i=0; i<n; i++){
        if(sayilar[i]%5==0)
            bol5++;
    }
    cout << "\n5 ile bolunebilen eleman sayisi: " << bol5;
    return 0;
}
```

Görev Kodu 5

O7. B5. 1. Problem

Haftalık oyun oynayan üç arkadaşın oyun sonunda aldıkları puanların ilk iki haftası aşağıda verilmiştir. Üçüncü hafta ise ilk iki haftanın puan toplamından oluşmalıdır. Buna göre bu üç arkadaşın üçüncü hafta puanlarını ekrana yazdıran programı tasarlayınız.

Hafta 1:

Oyuncu	Puan		
Ayşe	1	0	1
Burcu	0	0	0
Can	0	2	0

Hafta 2:

Oyuncu	Puan		
Ayşe	1	1	1
Burcu	2	0	0
Can	0	2	0

Hafta 3: ??

Oyuncu	Puan		
Ayşe			
Burcu			
Can			

O7. B5. 2. Görev Kartı

İç Çember Görev Kartı

Problemde yer alan ilk hafta skorlarını sırayla ekrana yazdıran kod satırlarını tamamlayınız.

Dış Çember Görev Kartı

Problemde yer alan ikinci hafta skorlarını ekrana yazdıran kod satırlarını yazınız.

Kodlama Ekibi Görev Kartı

İlk hafta ve ikinci hafta skorlarını ekrana yazdıran kod satırlarını birleştirin. Bu iki skorların toplamını ekrana yazdıran yeni kod satırlarını oluşturun. Şimdi elinizde temel problemi çözecek kod satırı parçaları vardır. Bunları bir araya getirerek temel problemin çıktısını oluşturacak programı çalıştırın.

O7. B6. 1. Farkı Bul!

```
#include<iostream>
using namespace std;
int main()
{
    char dizi[4];
    char katar[5];
    int i;
    cout << "Ilk ismin karakterlerini giriniz: " << endl;
    for(i=0; i < 4; i++) {
        cin >> dizi[i];
    }
    cout << "Ikinci ismin karakterlerini giriniz: " << endl;
    for(i=0; i < 4; i++) {
        cin >> katar[i];
    }
    katar[4] = '\0';
    cout << "Ilk isim: ";
    for(i=0; i < 4; i++) {
        cout << dizi[i];
    }
    cout << "\nIkinci isim: ";
    cout << katar;

    return 0;
}
```

O7. C. 1. Kısmi Öğrenme Görevleri Afişi

Kısmi öğrenme görevleri afişine ulaşmak için buraya tıklayınız.

Hafta 8. Fonksiyonlar

Kazanımlar

K1. Verilen problemin çözümü için fonksiyon tanımlar.

K2. Verilen problemin çözümü için fonksiyon içeren programlar geliştirir.

K3. Verilen problemin çözümünde fonksiyon çağırma kodunu geliştirir.

Amaç

Bu haftanın amacı öğrencilerin programlamada fonksiyon oluşturmalarını, fonksiyon kullanabilmelerini sağlamaktır.

Önerilen Ders Akışı

A. Giriş: Taş, Kâğıt, Makas (10 dk.)

B. Öğrenme Görevleri

B1. Fonksiyonları Tanıyalım (30 dk.)

- ❑ Ders Arası (5 dk.)
- ❑ Motivasyon Oyunu (10 dk.)

B2. Fonksiyonların Nasıl Kullanıldığını Keşfediyorum (40 dk.)

- ❑ Ders Arası (10 dk.)

B3. Fonksiyonları Kullanarak Kodlama Yapalım (60 dk.)

- ❑ Ders Arası (10 dk.)

C. Kısmi Öğrenme Görevleri (60 dk.)

A. Giriş: Taş, Kâğıt, Makas

Süre: 10 dk.

Uygulama: Sınıfta bulunan her öğrenci kendine bir oyun arkadaşı seçer. Herkes seçtiği oyun arkadaşı ile taş, kâğıt ve makas oyununu karşılıklı oynar, oyunu kaybeden kazandığı kişinin arkasına geçerek onun takipçisi olur. Kazananlar sürekli bir diğer kazananla karşılaşarak aynı şekilde taş, kâğıt ve makas oyununu tekrar eder, kaybeden her kişi kazananın arkasına geçmektedir ve kazananın takipçisi olmaktadır. En uzun kuyruğa ulaşan bir başka ifadeyle hiç kaybetmeyen kişi oyunu kazanır.

B. Öğrenme Görevleri

B1. Fonksiyonları Tanıyalım

Süre: 30 dk.

Kazanımlar: K1. Verilen problemin çözümü için fonksiyon tanımlar.

Materyaller: O8. B1.1. Fonksiyonların Özelliklerini Keşfediyorum!

Hazırlık: Eğitmen dersin bu bölümü için hazırlanan materyalin beş tane olacak şekilde çıktısını alır, her bir kutucuğu keserek derse hazırlar.

Uygulama: Eğitmen öncelikle her bir grup dörderli olacak şekilde sınıfı beş gruba böler. Her bir görev kâğıdı kutucuklarından kesilerek her bir gruba aynı materyal 5 parça olacak şekilde dağıtılır. Daha sonra eğitmen sınıftaki gruplara görevleri şu şekilde verir:

Şimdi her grubun elinde fonksiyonların ne işe yaradıklarını ve özelliklerini anlatan; günlük yaşamla ile ilişkisi kurularak keşfedebileceğiniz örnekler var. Şimdi her grubun her bir örnekten yola çıkarak fonksiyonların ne gibi özellikleri olabileceğine yönelik grupça bir şeyler yazmasını istiyorum. Daha sonra gruplardan gelen fonksiyonlarla ilgili her bir özelliği beyin fırtınası yoluyla tahta da özetleyeceğiz. Görevleriniz başladı ve süreniz 10 dk.

Eğitmen etkinlik sonunda fonksiyonlarla ilgili özellikleri öğrencilerle birlikte aşağıdaki gibi özetlemeye çalışır.

Bir işi alt parçalara bölmek hem işin takibini kolaylaştırır hem de aynı işin iki kez yapılmasını engeller. Bilgisayar programlamada da bu böyledir. Birden fazla kez kullanılacak işler bir çatı altında fonksiyon yazılarak toplanır. Bu sayede;

1. Program takibi kolaylaşır.
2. Hata çözümü kolaylaşır.
3. Tek noktadan değişiklik yapılır.

Fonksiyonlar işleri bölerek daha az satır kod yazmamızı sağlar. Örneğin; yazdığımız programın 10 farklı noktasında 5 satırlık işlem yaptırmamız gerekiyor. Eğer bunun için fonksiyon kullanmazsak 50 satır kod oluşacaktır. Fonksiyon kullanarak satır sayısı 15'e düşecektir. 5 satır fonksiyon için 10 satır da fonksiyon çağırma için kullanılacaktır.

Eğitmene Öneriler: Yukarıdaki etkinlik bitimiyle beraber öğrencilerden gelen yanıtlar neticesinde fonksiyonların özellikleri aşağıdaki gibi çıkarılmaya çalışılır. Biz yazılımcılar programlarımızda fonksiyonları kullanarak;

1) Daha kolay hatalarımızı bulabiliriz. (Materyaldeki dördüncü örnek.)
2) Daha doğru çözüm üretebiliriz. (Materyaldeki beşinci örnek.)
3) İhtiyaç duyduğumuz anda belli bir görevi yapması için çağırırız. (Materyaldeki birinci örnek.)
4) İhtiyaç duyduğumuz anda belli bir görevi yapması için çağırırız ve kullanılmasını sağlarız (Materyaldeki üçüncü örnek.)
5) Daha az satır kod yazarız ve programın yönetimi kolaylaştırırız. (Materyaldeki ikinci ve beşinci örnek)
6) İstenilen yerlerde kullanıyorlar ve kod tekrarını önlüyorlar. (Materyaldeki ikinci ve beşinci örnek)

B2. Fonksiyonların Nasıl Kullanıldığını Keşfediyorum

Süre: 40 dk.

Kazanımlar: K2. Verilen problemin çözümü için fonksiyon içeren programlar geliştirir.

Materyaller: O8. B2.1 C++ Programlama Dilinde Fonksiyon Tanımlama Görevleri

O8. B2.2 Fonksiyonların Kullanım Afişi?

Hazırlık: Eğitmen birinci materyalin çıktısını alır. Fonksiyonların kullanımına yönelik hazırlanan afiş her öğrencinin görebileceği şekilde ÖYS üzerinden erişime açılır.

Uygulama: Öğrenciler dörderli gruplar hâlinde çalışır. Eğitmen her grupta iki öğrenciye bir çıktı olacak şekilde "O8. B2. 1 C++ Programlama Dilinde Fonksiyon Tanımlama Görevleri" adlı materyalin çıktısını alır. Öğrencilerden verilen iki göreve yönelik fonksiyon tanımlamaları istenir. Eğitmen destekleyici bilgi olarak döngüler için hazırlanan afişi (O8. B2. 2) öğrencilerle paylaşılır. Öğrencilerin ihtiyaç duyduğu aşamada eğitmenler konu ile ilgili yöntemsel bilgiyi öğrenciye verir.

Eğitmene Öneriler: Yukarıdaki etkinlikleri öğrenciler bitirdikten sonra fonksiyonlar aşağıdaki gibi özetlenir.

C++ programlama dilinde fonksiyon yazmak için üç kısım vardır. Bunlardan ilki geriye döndürülecek değişkenin tipi (dönüş tipi, eğer geriye değer dönmüyor ise **void** olarak

belirtilir.), İkinci olarak fonksiyonun ismi, son olarak fonksiyon içerisinde ihtiyaç duyulan bilgilere parametre denir. Fonksiyon işlemlerini tamamladıktan sonra, eğer çağrıldığı yere değer döndürecek ise **return** komutu ile belirtilir.

Instagram'daki her resmin bir numarası vardır ve bu numarayı kullanarak işlemler yapılır. Bu numarayı bir fonksiyona parametre olarak göndeririz. Bir fotoğrafa yorum göndermek için fonksiyon yazarsak; fonksiyonun ismi: yorum_yap, alacağı değer: yorum metni, geriye de işlemin başarılı olup olmadığı döndürülür.

```
dönüş_tipi fonksiyon_ismi(parametreler)
{
    Yapılacak işlemler
    return dönüş değeri;
}
```

Fonksiyonları amacını belirtecek şekilde isimlendirmeye özen gösterilmelidir. Çünkü başka biri fonksiyonu kullanmak istediğinde amacını kolayca anlayabilmelidir. Örneğin verilen sayıların ortalamasını alan bir fonksiyon yazıyorsak ismini ortalama_al veya OrtalamaAl şeklinde belirtebiliriz. Benzer şekilde kullanıcılara mail atacak bir fonksiyon için mail_at veya MailAt şeklinde isimlendirebiliriz.

B3. Fonksiyonları Kullanarak Kodlama Yapalım!

Süre: 60 dk.

Kazanımlar: K1. Verilen problemin çözümü için fonksiyon tanımlar.

K2. Verilen problemin çözümü için fonksiyon içeren programlar geliştirir.

K3. Verilen problemin çözümünde fonksiyon çağırma kodunu geliştirir.

Materyaller: O8.B3.1. Fonksiyonları Kullanarak Kodlama Yapma!

O8.B3.2. Fonksiyon Kodlama ile İlgili Destekleyici Bilgi Sunusu

Hazırlık: Eğitmen dersin bu bölümü için hazırlanan materyallerin her ikisini ÖYS üzerinden erişime açar. Code Blocks uygulaması öğrencilerin kodlama yapabilmesi için hazır duruma getirilir.

Uygulama: Eğitmenler fonksiyonlar kullanarak kodlama etkinliğindeki uygulamalarından dört tanesini öğrencilerin seçmesini sağlayarak, öğrencilerin bu derste kodlama yapmasını ister. Diğer görevler ise öğrencilere kodlanması için ev görevi olarak verilebilir. Öğrenciler kodlama esnasında verilen görevler için hazırlanan sunu destekleyici bilgi olarak kullanılırken, öğrencinin ihtiyaç duyduğu anda gerekli işlemsel bilgiyi eğitmenlerden biri sınıfı dolaşarak, biri de kodlamayı öğrencilerin de görebileceği şekilde bilgisayarda kodlayarak sağlayabilir.

Eğitmene Öneriler:

Fonksiyon kodlama görevleri ve cevapları aşağıdaki gibidir.

Görev 1: Ekrana 10 kez "Deneyap" ardından 2 kez "Merhaba!" yazan bir ekrana_yaz isimli bir fonksiyon yazalım.

Cevap 1:

```
void ekrana_yaz()
{
    for(int i=0; i<10;i++)
    {
        cout << "Deneyap" <<endl;
    }
    for(int i=0; i<2;i++)
    {
        cout << "Merhaba!" <<endl;
    }
}
```

Fonksiyonumuzdan geriye herhangi bir bilgi dönmeyeceği için "void" yani "boş" olarak belirtiyoruz. Eğer bir geri dönüş tipi belirtirsek (int, double vs.), kesinlikle geriye o türde bir değer döndürmemiz gerekiyor. Fonksiyon içerisinde ilk olarak 10 kez "Deneyap" yazdırıyoruz. Ardından da 2 kez "Merhaba!" yazdırıyoruz.

Şimdi bunu sadece fonksiyonun ismini yazarak ana programdan çağıralım:

```
int main()
{
    ekrana_yaz();
}
```

Görev 2: Dikdörtgen şeklinde olan büyük bir arazi üçgensel bölgelere ayrılmak istenmektedir. Bunun içinde araziye ne kadar üçgen sığabileceğini bulmak isteyen bir yazılımcı ihtiyaç duyduğu anda çağırabileceği üçgen alanının hesaplamasına yönelik bir fonksiyon yazmak istemektedir. Yazılımcı bu üçgen alan bulma fonksiyonunu nasıl kodlaması gerekmektedir?

Cevap 2:

```
void ucgen_alan_hesapla(double taban, double yukseklik)
{
    double alan = (taban*yukseklik) / 2;
    cout << alan << endl;
}
```

Fonksiyonumuz hazır hâle geldi. Artık kullanıma hazır, üçgen alanı hesaplayıp ekrana yazdıran bir fonksiyona sahibiz. Programın istediğimiz yerinde çağırıp kullanabiliriz. Tabanı 2 ve yüksekliği 4 olan bir üçgen için alan aşağıdaki gibi hesaplanır. Parametreleri doğrudan sayı olarak girebildiğimiz gibi değişken ismi de girebiliriz.

```
int main()
{
    ucgen_alan_hesapla(2,4);
}
```

Görev 3: Fonksiyona gönderilen sayı 5 ile tam bölünüyor ise ekrana "tam bölünür." aksi durumda kalanı yazdıran fonksiyonu yazalım.

Cevap 3:

```
#include <iostream>
using namespace std;
void bes_ile_bolme(int sayi)
{
    if(sayi % 5 == 0)
        cout << "tam bolunur."<<endl;
    else
        cout << sayi%5<<endl;
}
int main()
{
    bes_ile_bolme(11);
    bes_ile_bolme(15);
    bes_ile_bolme(12);
}
```

Görev 4: Fonksiyona gönderilen iki sayıdan küçük olanı geriye döndüren fonksiyonu yazalım.

Cevap 4:

```
#include <iostream>
using namespace std;
int hangisi_buyuk(int sayi1, int sayi2)
{
    if(sayi1>sayi2)
        return sayi1;
    else
        return sayi2;
}
```

```
int main()
{
    int sayi = hangisi_buyuk(10,20);
    cout << sayi <<endl;

    sayi = hangisi_buyuk(22,11);
    cout << sayi <<endl;
}
```

Görev 5: Fonksiyona gönderilen tam sayı tipindeki dizinin en büyük sayısını ekrana yazan fonksiyonu yazalım.

Cevap 5:

```
void en_buyuk(int dizi[5])
{
    int enbuyuk = dizi[0];
    for(int i=1;i<5;i++)
    {
        if(enbuyuk<dizi[i])
            enbuyuk=dizi[i];
    }
    cout << enbuyuk;
}
```

Ana programımız ise aşağıdaki şekilde olacaktır.

```
int main()
{
    int sayilar[] = {5,3,4,5,8};
    en_buyuk(sayilar);
}
```

Görev 6: İki dizi içerisindeki en büyük iki sayının toplamını bulan programı fonksiyon kullanarak yazalım.

Cevap 6:

Önceki örneği sadece 1 düzenleme ile kullanabiliriz. Ekrana yazdırmak yerine en büyük sayıyı ana programa göndermemiz gerekiyor. Böylece sonraki işlemlerde kullanabiliriz.

```
int en_buyuk(int dizi[5])
{
    int enbuyuk = dizi[0];

    for(int i=1;i<5;i++)
    {
        if(enbuyuk<dizi[i])
            enbuyuk=dizi[i];
    }
    return enbuyuk;
}
```

Ana programı da aşağıdaki gibi yazabiliriz. İlk olarak dizilerimizi tanımlıyoruz. Ardından dizi1'i fonksiyona gönderip en büyüğünü buluyoruz. Sonra dizi2'nin en büyük elemanını bulup ekrana yazdırıyoruz. Son olarak yapmak istediğimiz toplama işlemini gerçekleştiriyoruz.

```
int main()
{
    int dizi1[] = {5,3,4,5,8};
    int dizi2[] = {9,3,4,5,8};

    int en_buyuk_1 = en_buyuk(dizi1);
    cout << "1. dizinin en buyugu:" << en_buyuk_1 << endl;
    int en_buyuk_2 = en_buyuk(dizi2);
    cout << "2. dizinin en buyugu:" << en_buyuk_2 << endl;

    cout << en_buyuk_1 << "+" << en_buyuk_2 << "=" << en_buyuk_1 + en_buyuk_2;
}
```

C. Kısmi Öğrenme Görevleri

Süre: 60 dk.

Materyal: O8. C.1. Kısmi Öğrenme Görevleri Afişi

Hazırlık: Kısmi öğrenme görevleri afişi öğrencilere ÖYS ortamında süreli ödev olarak açılır.

Uygulama: Öğrenciler afişteki görevleri istedikleri sırada ve sayıda kendi tercihlerine bağlı olarak verilen süre içinde uygulamaya başlar. Her bir görevi tamamlayan öğrencilere, göreve ilişkin beceri rozeti verilir. Öğrenciler görev seçimlerini ve tamamladıkları görevleri eğitmene bildirmelidir. Eğitmenler ihtiyaç duydukları zaman öğrencilere görevi tamamlama aşamasında

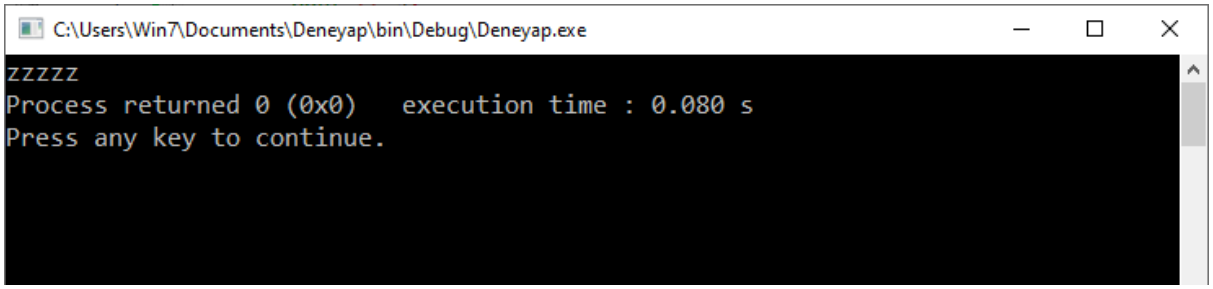
anlık geri bildirimlerde bulunur. Öğrenciler, ÖYS üzerinden süreli ödev olarak açılan kısmi öğrenme görevlerinden kendi tercihlerine göre istedikleri sayıda görev tamamlayıp eğitmene iletir. Süre bitiminde eğitmen görevleri ve yanıtlarını ÖYS ya da GitHub üzerinden öğrencilere gönderir.

Kısmi Öğrenme Görevleri Yanıtlar: Görevler, öğrenciye verilecek beceri rozetleri ile isimlendirilmiştir. Her bir rozetin altında ilgili görevin yanıtları verilmektedir. Eğitmen bu yanıtları uygulama süresinin sonunda GitHub üzerinden öğrencilerle paylaşır.

1. **KODLAYICI:** Parametre olarak gönderilen iki sayının toplamını geriye döndüren bir fonksiyonu oluşturmak isteseydiniz nasıl bir kod yazardınız?

```
#include <iostream>


using namespace std;
int topla(int sayi1, int sayi2)
{
    int toplam = sayi1 + sayi2;
    return toplam;
}
int main()
{
   int sonuc = topla(5,4);
   cout<< sonuc;
}
```

2. **KODLAYICI**: Parametre olarak gönderilen harfi, yine parametre olarak gönderilen sayı kadar ekrana yazdıran bir fonksiyon tanımlamak isteseydiniz nasıl kodlardınız? Örnek kullanım: tekrar_yaz(‘z’,5)



Resim 31. *Ekran çıktısı*

Cevap:

```
#include <iostream>
using namespace std;
void tekrar_yaz(char a, int adet)
{
    for(int i=0;i<adet;i++)
        cout << a;
}
int main()
{
   tekrar_yaz('z',5);
}
```

3. **KODLAYICI**: Bir kullanıcının sosyal medyada paylaştığı fotoğraflardaki toplam beğeni sayılarının bir dizide saklandığı düşünülürse; paylaşılan fotoğraflardaki beğenilerin toplamını geriye döndüren bir fonksiyonu nasıl kodlardınız?

Cevap:

```
#include <iostream>


using namespace std;
int  dizi_topla(int dizi[5])
{
    int toplam = 0;


    for(int i=0;i<5;i++)
        toplam = toplam + dizi[i];


    return toplam;
}
int main()
{
   int sayilar[5] = {5,6,9,3,2};
   int sonuc = dizi_topla(sayilar);
   cout << sonuc;
}
```

Hafta 8. Ders Materyalleri

O8. B1. 1 Fonksiyonların Özelliklerini Keşfediyorum!

1

Evimize katı meyve sıkacağı alıyoruz ve biz bunu canımız her ne zaman meyve suyu çektiğinde meyve sıkacağını kullanıyoruz. Yani bu makinenin görevi biz istediğimizde meyve suyu yapmak.

2

İnstagram da milyonlarca fotoğraf paylaşan insan var ve bu fotoğraflar on binlerce insan tarafından beğeniliyor. İnstagramı yazanlar her beğenilen fotoğraf için ayrı ayrı kodlar mı yazdılar acaba.

3

Her gün sabah uyanıp ekmek almaya gitmekten yoruldum. Keşke bir yardımcı robotum olsaydı onu her çağırdığımda gelip benim yerime ekmek alsaydı.

4

Türkçe öğretmenimin bana verdiği kompozisyon ödevini 20 sayfa yazarak bitirdim. Öğretmenim "yalnız" kelimesini yanlış yazdığımı söyledi. Şimdi tüm sayfalara teker teker gidip bu yanlışları düzeltmem gerekecek. Keşke yanlış yazdığım kelimenin birisini düzelttiğimde diğer yanlışlarımda düzelseydi.

5

Kodlamak istediğim web sitesinde sisteme her giriş yapan kullanıcıya Merhaba "kullanıcının ismi yazmak istiyorum. Ne yani binlerce kişi web siteme girerse her isim için ayrı ayrı merhaba mı diyeceğim.

O8. B2. 1 C++ Programlama Dilinde Fonksiyon Tanımlama Görevleri

1) Instagram'da işe başlayan bir yazılımcı, kullanıcıların fotoğraflara gönderdikleri yorumlara yönelik bir fonksiyon tanımlamak istiyor. Bu fonksiyonu oluşturacak olan siz olsaydınız nasıl tanımlardınız?
2) Dikdörtgen şeklinde olan büyük bir arazi üçgensel bölgelere ayrılmak istenmektedir. Bunun içinde araziye ne kadar üçgen sığabileceğini bulmak isteyen bir yazılımcı ihtiyaç duyduğu anda çağırabileceği üçgen alanının hesaplanmasına yönelik bir fonksiyon tanımlamak istemektedir. Yazılımcı bu alan fonksiyonunu nasıl tanımlamalıdır?

O8.B2. 2 Fonksiyonların Kullanım Afişi?

Resim 32. *Fonksiyon tanımlama afişi*

O8.B3. 1 Fonksiyonları Kullanarak Kodlama Yapma

O8.B3. 2 Fonksiyon Kodlama ile İlgili Destekleyici Bilgi Sunusu

Yukarıda verilen görevlerin her biri için destekleyici bilgi hazırlanmıştır. Bu destekleyici bilgilere ulaşmak için O8.B3.2 adlı sunumu açınız. Bu sunu öğrencilerin bilgisayarlara gönderilerek veya her öğrenci grubu için çıktı alınarak kullanılabilir.

O8. C. 1. Kısmi Öğrenme Görevleri Afişi

Kısmi öğrenme görevleri afişine ulaşmak için buraya tıklayınız.

Hafta 9. Nesneler

Kazanımlar

K1. C++ programlama dilinde nesne yönelimli programlama mantığını açıklar.

K2. Nesne yönelimli programlamayı prosedürel programlamadan ayırt eder.

K3. Nesne yönelimli programlamanın avantaj ve dezavantajlarını ayırt eder.

K4. Nesne ve sınıf kavramlarını analiz eder.

K5. Nesne yönelimli programlamada erişim belirteçlerini analiz eder.

K6. Günlük hayatta karşılaştığı problemlerle ilgili fonksiyon oluşturma işlemini gerçekleştirir.

K7. C++ programlamada sınıf ve nesne tanımlamasını farklı durumlarda uygular.

Amaç

Bu haftanın amacı, nesne yönelimli programlamanın temel felsefesini, yordamsal programlamadan farkını, avantaj ve dezavantajları ile nesne yönelimli programlamada sınıf, nesne ve fonksiyon tanımlama işlemlerini uygulamaktır.

Önerilen Ders Akışı

A. Öğrenme Görevleri

A1. Aynısını Çiz (25 dk.)

- ❏ Ders Arası (5 dk.)

A2. Sınıfın Özelliklerini Tanı! (60 dk.)

- ❏ Ders Arası (5 dk.)
- ❏ Motivasyon Oyunu (10 dk.)

A3. Kendi Sınıfını Afişle! (60 dk.)

- ❏ Ders Arası (5 dk.)
- ❏ Motivasyon Oyunu (10 dk.)

B. Kısmi Öğrenme Görevleri (60 dk.)

A. Öğrenme Görevleri

A1. Aynısını Çiz!

Sure: 25 dk.

Kazanımlar: K1. C++ programlama dilinde nesne yönelimli programlama mantığını açıklar.

K2. Nesne yönelimli programlamayı prosedürel programlamadan ayırt eder.

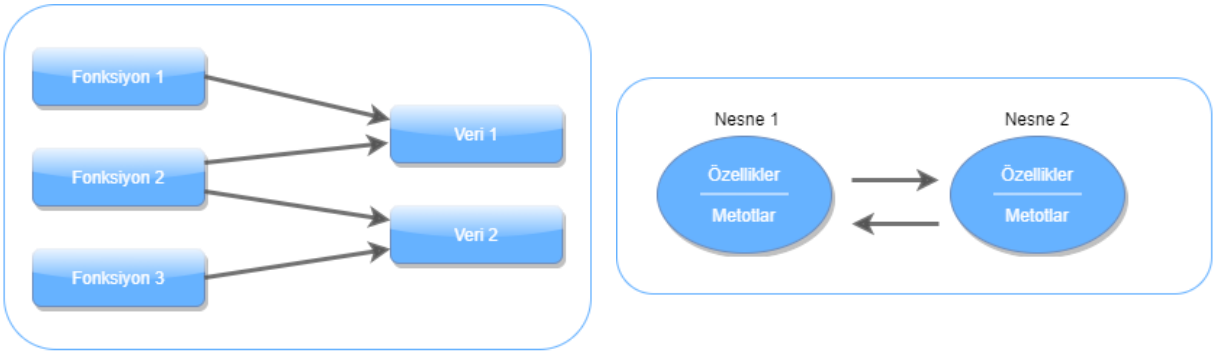
K3. Nesne yönelimli programlamanın avantaj ve dezavantajlarını ayırt eder.

Materyaller: Bir fanus ya da kura torbası

Hazırlık: Öğrenciler dörderli gruplar hâlinde otururlar. İçinde sınıf örneklerinin olduğu kura torbası hazırlanır.

Uygulama: Eğitmen öğrencilere bu öğrenme görevinde bir oyun oynayacaklarını söyler. Her grup meyve, çanta, taşıt, okul gibi sınıf örneklerinin yazılı olduğu kura torbasından bir kâğıt seçer. Grup üyeleri, bu sınıf kartına ait birer nesne düşünür ve bu nesnenin çizimini arkadaşlarına göstermeden çizer. Gruptan biri çizimini diğer arkadaşlarına tahmin ettirmeye çalışır. Grup üyeleri, çizimi yapan arkadaşlarına bu nesnenin özelliklerine ilişkin sorular sorar. Bu sorular; "Rengi ne renk?", "Canlı mı cansız mı?", "Eşya mı?", "Yenilebilir mi?" tarzında nesne hakkında çıkarımda bulunabilecekleri türde olmalıdır. Grup üyeleri bu şekilde arkadaşlarının çizdiği nesneyi tahmin etmeye çalışır. Her bir grupta üyelerin çizimlerini tahmin etme süresi 1 dk. olarak belirtilir. Tüm grup üyelerinin çizimlerinin tahmin süresi bittikten sonra, grupların çizim yaptıkları nesneler ve doğru tahmin edilen nesne sayısı tahtaya yazılır. Eğitmen bu şekilde kazanan grubu belirler. Oyun sonunda eğitmen sınıf ve nesne arasındaki farkı özetler. Eğitmen, grupların sınıf ve nesnelerinden örnekler verir. Buradan yola çıkarak nesne yönelimli programlamanın önemi, avantaj ve dezavantajları hakkında bilgi verir.

Konu İçeriği: C++ programlama dilinin amacı, kendi başına en güçlü programlama dillerinden biri olan C programlama diline nesne yönelimi eklemekti. Nesne yönelimli programlama, programları nesnelerin sınıfları açısından tanımlayan bir yazılım tasarım ve programlama yöntemidir. Aşağıdaki şekilde görüldüğü üzere yordamsal (prosedürel) programlama dillerinde veri ve fonksiyonlar birbirinden ayrı iken, nesne yönelimli programlama, veri ve fonksiyonları birleştirerek aralarındaki görevleri gerçekleştirmek için düzenli iletişim sağlayan nesneler kümesi şeklindedir.



Resim 33. *Yordamsal ve nesne yönelimli programlama*

Yordamsal programlama, veriler üzerinde işlemler gerçekleştiren fonksiyonlar yazmakla ilgiliyken, nesne yönelimli programlama ise hem verileri (özellikler) hem de fonksiyonları içeren nesneler oluşturmakla ilgilidir. Nesne yönelimli programlamanın, yordamsal programlamaya göre birçok avantajı vardır:

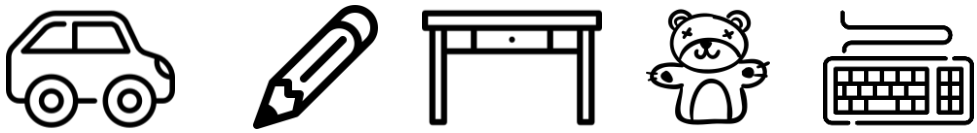
- Daha hızlıdır ve uygulanması daha kolaydır.
- Programlar için net bir yapı sağlar.
- C++ kodunun tekrar edilmemesine yardımcı olur ve kodun bakımını, değiştirilmesini ve hata ayıklamasını kolaylaştırır.
- Daha az kod ve daha kısa geliştirme süresiyle tam yeniden kullanılabilir uygulamalar oluşturmayı mümkün kılar.

Prosedürel Programlama	Nesne Yönelimli Programlama
Program, fonksiyon adı verilen küçük parçalara bölünmüştür.	Program, nesneler adı verilen küçük parçalara ayrılır.
Yeni veri ve fonksiyon eklemek kolay değildir.	Yeni veri ve fonksiyon eklemek koladır.
Verileri gizlemek için uygun bir yol yoktur, bu nedenle daha az güvenlidir.	Daha güvenli olmak için veri gizleme sağlar.
Fonksiyon veriden daha önemlidir.	Veri, fonksiyondan daha önemlidir.
Gerçek olmayan dünyaya dayanır.	Gerçek dünyaya dayanır.
Örnekler: C, Fortran, Pascal, Basic vb.	Örnekler: C++, Java, Python, C# vb.

C++ modülleri tasarlarken, tüm dünyayı nesne şeklinde görmeye çalışıyoruz. Örneğin araba; renk, kapı sayısı, hız limiti gibi belirli özelliklere sahip bir nesnedir. Farklı isimler ve markalara sahip birçok araba olabilir, ancak hepsi bazı ortak özellikleri paylaşmaktadır. Ayrıca, hızlanma ve yavaşlama gibi belirli yöntemlere de sahiptir.

Nesne (Object) Kavramı

Nesne yönelimli programlamanın temel birimi olan, bazı özellikleri ve davranışları olan tanımlanabilir bir varlıktır. Yani veriler üzerinde çalışan fonksiyonlar nesne olarak adlandırılan bir birim olarak oluşturulur. Örneğin: araba, kalem, masa, oyuncak ve klavye gibi.

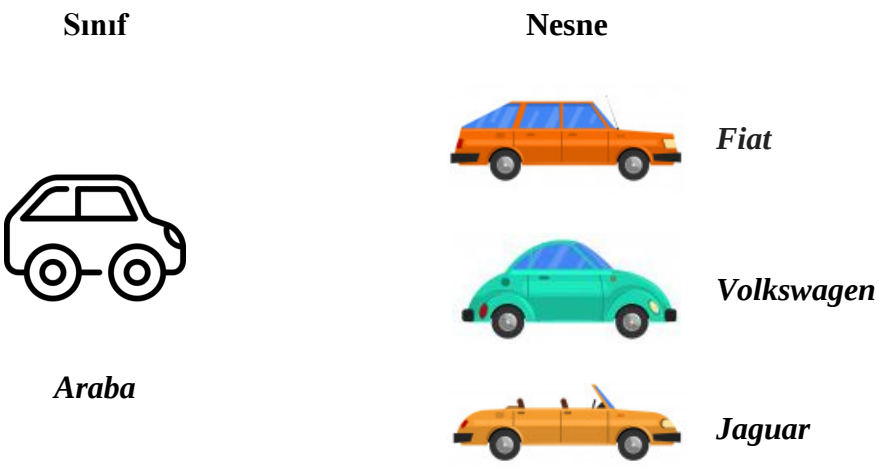


Resim 34. Nesne örnekleri

Nesne soyut bir veri örneğidir. Sınıf tanımlandığında, bellek ayrılmaz, ancak örnek oluşturulduğunda (yani bir nesne oluşturulduğunda) bellek ayrılır.

Sınıf (Class) Kavramı

Bir sınıf tanımladığınızda, nesne için bir taslak tanımlarsınız. Sınıf adının ne anlama geldiği, yani sınıftaki bir nesnenin ne içereceği ve böyle bir nesne üzerinde hangi işlemlerin gerçekleştirilebileceği tanımlanır. Araba bir sınıf ise bundan üretilen farklı araba çeşitleri de nesneleri ifade eder.



Resim 35. *Sınıf ve nesne kavramı*

Sınıf, veri üyeleri ve üye fonksiyonlarına sahip kullanıcı tanımlı bir veri türüdür. Veri üyeleri, veri değişkenleridir. Üye fonksiyonları ise bu değişkenleri işlemek için kullanılan fonksiyonlardır. Bunlar birlikte bir sınıftaki nesnelerin özelliklerini ve davranışını tanımlar.

A2. Sınıfın Özelliklerini Tanı!

Süre: 60 dk.

Kazanımlar: K4. Nesne ve sınıf kavramlarını analiz eder.

K5. Nesne yönelimli programlamada erişim belirteçlerini analiz eder.

K6. Günlük hayatta karşılaştığı problemlerle ilgili fonksiyon oluşturma işlemini gerçekleştirir.

Materyaller: O9. A2. 1. Grup Afişleri

Hazırlık: Materyaller her gruba 2'şer adet dağıtılmak üzere çıktı alınır.

Uygulama: Öğrenciler beşerli gruplar hâlinde çalışmaktadır. Eğitmen her gruba grup afişlerinden birini verir. Öğrencilerin gruplar hâlinde bu afişleri incelemeleri ve *sınıf, üye listesi, erişim belirteci, fonksiyon oluşturma* ve *nesne oluşturma* terimlerini tartışmaları istenir. Gruplara 5 dk. inceleme süresinin ardından, grupların afişlerini ve bu terimleri sırayla diğer arkadaşlarına açıklamaları istenir. Bunun için birinci gruptan birer üye diğer gruplara dağılır. Birinci grubun üyeleri kendi afişlerini dahil olduğu yeni gruptakilere 5 dk. süre ile açıklar ve kendi grubuna geri döner. Benzer şekilde ikinci 5 dk. başlatılır. İkinci grubun her bir üyesi diğer gruplara dağılır ve afişlerini açıklar. İkinci grubun üyeleri geri döndükten sonra, üçüncü grup üyeleri dağılır. Bu şekilde beş grubun da sırası tamamlanır. Eğitmen zaman planlaması için bir çan kullanabilir ya da "Grup 1 Dağılın!", "Grup 1 Geri Dönün" gibi emir ifadeleri ile süreci yönetebilir. Tüm grupların dağılma sürecinin ardından eğitmen öğrencilerden bu kavramlar hakkında ne öğrendiklerini özetleyen bir grup kâğıdı oluşturmalarını ve bunları tahtaya asmalarını ister. Eğitmen bu kâğıtlar üzerinden her bir kavramı açıklamaya çalışır.

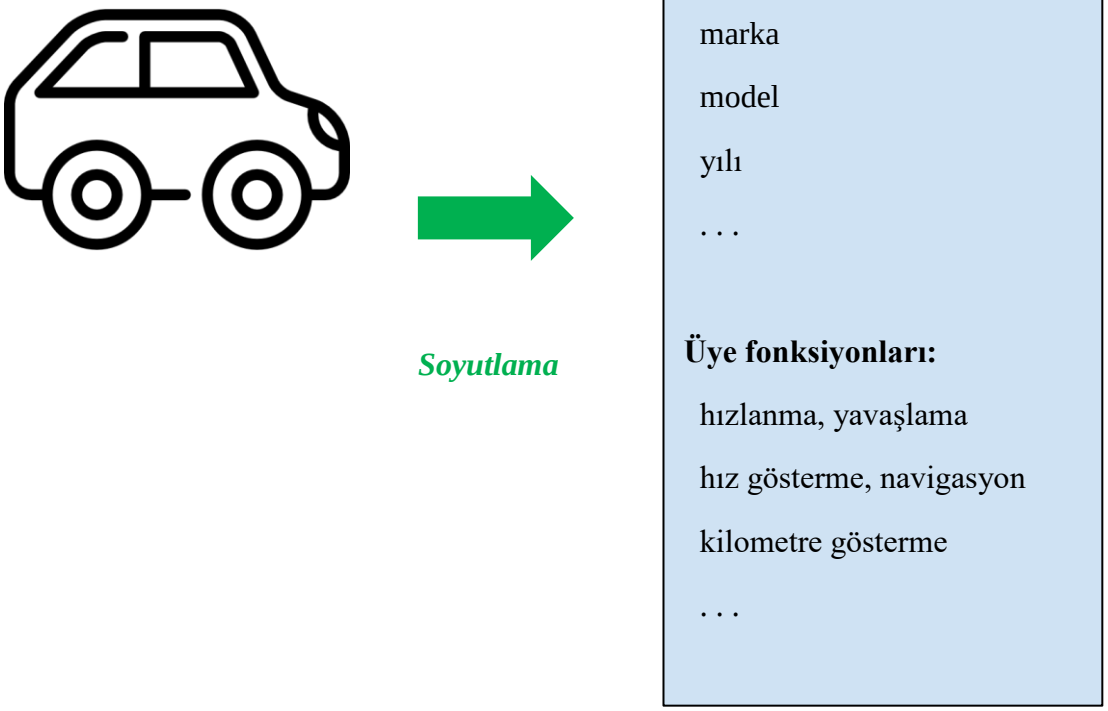
Eğitmene Öneriler: Eğitmen grup etkinliklerinin tamamlanmasını ardından kalan dakikalarda öğrencilere sınıf tanımlama, erişim belirteçleri, üye listesi, fonksiyonlar ve nesne tanımlama ile ilgili sorular yöneltir. Bu şekilde eksik ya da hatalı öğrenmeleri düzenlemeye çalışır.

Konu İçeriği: Sınıf Tanımlama

Sınıflar, nesne yönelimli programlamanın en önemli yapı elemanıdır. Bir sınıf, bir nesnenin özelliklerini ve kapasitelerini tanımlar. Pek çok farklı sınıf tanımı oluşturabilirsiniz. Bu bölümden itibaren artık nesne yönelimli programlama tasarımına giriş yapıyoruz. Başlangıçta biraz zorlu olacağı öngörülebilir fakat ne kadar değerli bir şey öğrendiğimizi unutmayalım. Kavramları anlamaya başladığınızda ve kod yazmayı ilerledikçe, nesne yönelimli programlamanın ilk bakışta göründüğü kadar zor olmadığını fark edeceksiniz. İlk aşamada sınıf kavramını anlamak için bir gerçek dünya nesnesi olan arabayı düşünebiliriz.

UYARI:

Her sınıf modeline yalnızca bir kavram (nesne bütünlüğü) sunmalıdır. Örneğin, Araba sınıfını Sürücü sınıfından ayrı tutun.



Resim 36. *Araba sınıfı örneği*

Yukarıda gördüğünüz üzere arabaya ait bir sınıf oluşturmak istediğimiz zaman bu sınıfa ait özellikleri ve fonksiyonları belirlememiz gerekmektedir. Özellikler sınıfın ayırt edici bilgilerinden oluşan marka, model ve yılı gibi verileri içermektedir. Fonksiyonlar ise sınıfın hızlanma, yavaşlama ve hız ve kilometre gösterme gibi fonksiyonlarını içermektedir. Sınıfın tasarımı tamamen bizim elimizde olup istediğimiz olası tüm yetenekleri sınıfa ekleyebiliriz. Bir sınıfı tanımlamak için aşağıdaki gibi genel bir sözdizimi kullanılmaktadır.

```
class SınıfAdı
{
public:
```

```
    üyeListesi
};
```

Burada **class**, sınıf tanımına ait anahtar kelime, **public** üye listesinin erişim belirteci, **üyeListesi**, sınıf üyelerinin listesi ve **SınıfAdı**, sınıfın adıdır. Sınıf adının büyük harfle başladığına dikkat edelim. Bu en yaygın kullanımdır ve kodunuzun okunabilir olması için önem arz eder. Sınıf bildiriminin fonksiyonlardaki gibi parantezle başlayıp bittiğine dikkat ediniz. Ayrıca kapanış parantezinden sonra noktalı virgül kullanıldığını unutmayınız. Noktalı virgülün unutulması derleyici tarafından yakalanacak bir hataya sebep olacaktır.

UYARI:

Bu eğitim kapsamında sınıfları ve sınıf üyelerini ayırt etmek için standart adlandırma kurallarına uyacağız. Sınıf adları büyük harfle ve üye adları küçük harfle başlatılacaktır. Bu kural sadece C++ 'da değil, aynı zamanda diğer tüm nesne yönelimli programlama dillerinde de kullanılır.

üyeListesi, üye bildirimlerini içeren listeden oluşur. Bunlar üye veri veya üye fonksiyon bildirimleri olabilir. Veri üyesi bildirimleri normal değişken bildirimleri ile aynıdır. Örneğin, sırasıyla karakter, tam sayı ve kesirli sayı türlerinde veri üyeleri oluşturmak istersek aşağıdaki tanımlamaları yaparız.

```
char a;
int b;
double c;
```

Ancak, veri üyelerini bildirdiğiniz yerde ilk değer ataması gerçekleştiremezsiniz. Değişkenlere ilk değer atamasının hazırlanan bir fonksiyonda ya da sınıfın dışında gerçekleştirilmesi gerekir. Tanımlanan bu veri üyelerinin kapsamı, sınıftan oluşturulan nesnenin kapsamı ile aynıdır. Bununla birlikte, veri üyelerine sınıfın dışından her zaman erişilebilmesi mümkün değildir. Yukarıda verdiğimiz araba sınıfında markanın, modelin ve yılın ne olduğu bu veri üyeleri tarafından saklanır. Bir sınıfın veri üyeleri, sınıf özelliklerini tanımlar ve açıklar. Bu nedenle, her bir araba nesnesinin markası, o nesnenin bir özelliğidir. Aşağıda araba sınıfına ait marka, model, yıl, renk ve fiyat veri üyelerinin sınıf içerisinde nasıl tanımlandığını görebilirsiniz.

```
class Araba
{
public:
    char marka[30];
```

```
        char model[30];
        int yıl;
        char renk[30];
        float fiyat;
    };
```

Erişim Belirteçleri

Erişim belirteçleri, C++ sınıfınızın içerisinde yer alan veri üyelerine nereden erişilebileceğini kontrol etmenizi sağlar. Bu kontrol sayesinde veri üyeleri üzerinde yetkisiz olarak değişiklik yapılmasının önüne geçilmesi sağlanır. Erişim belirteci, sınıftaki veri üyelerine erişimi kontrol eden bir kelimedir. Bir erişim tanımlayıcısının sözdizimi aşağıdaki gibidir:

```
class SınıfAdı
{
    üyeListesi
erişimBelirteci:
    üyeListesi
erişimBelirteci:
    üyeListesi
};
```

Bir erişim belirticisi tanımlandıktan sonraki üye listesindeki tüm veriler için geçerlidir. Sınıf içerisinde başka bir erişim belirticisi tanımlanırsa bu tanımlamadan sonraki veri üyeleri bu erişim türüne ait olur. Sınıfın sonuna ulaşıncaya kadar sınıfın tüm üyelerini en son tanımlanan erişim belirteci etkiler. Bir sınıfın sahip olabileceği farklı erişim belirteçlerinden ikisi:

1) public: Bu anahtar kelime ile tanımlanan tüm üyelere, sınıfın ulaşılabilir olduğu her yerden erişilebilir.

2) private: Bu anahtar kelime ile tanımlanan üyelere yalnızca aynı sınıfın diğer üye fonksiyonları tarafından erişilebilir.

UYARI:

Varsayılan olarak, tüm sınıf üyelerinin erişimi “private” olarak tanımlıdır. Bu nedenle, sınıf bildiriminde erişim belirticisi olmadan görünen tüm üyelerin erişimi “private” olur.

```
class Ceptelefonu
{
    char marka[30];
    int yil;
public:
    char model[30];
```

```
    char renk[30];
private:
    float fiyat;
    int imei_no;
};
```

Yukarıda verilen sınıf tanımlamasında fiyat ve imei no veri üyeleri ile birlikte marka ve yıl veri üyelerinin de "private" olduğunu unutmayınız. Diğer taraftan model ve renk "public" olarak tanımlanmıştır. Sınıf tanımlamalarınızda istediğiniz kadar erişim belirteciniz olabilir, ancak belirteçleri tek bir grup altında toplamak sınıfın anlaşılırlığını artıracaktır.

Fonksiyon Oluşturma

Bir sınıfın üye fonksiyonu, diğer herhangi bir değişken gibi sınıf içinde tanıma sahip olan bir fonksiyondur. Üyesi olduğu sınıfın herhangi bir nesnesi üzerinde çalışır ve bu nesne için bir sınıfın tüm üyelerine erişim sağlayabilir. Fonksiyon tanımları olmadan bir sınıfın tanımı tam olarak gerçekleştirilmiş sayılmaz. Fonksiyon tanımı yapıldıktan sonra tanımlanan bu fonksiyonlara yalnızca sınıfın bir nesnesi aracılığıyla erişilebilir.

Bir fonksiyonu iki şekilde tanımlayabilirsiniz. Birinci yol, sınıf bildirimi içinde bir fonksiyon bildirmek ve sonra onu dışarıda uygulamaktır. İkinci yol, fonksiyonu aynı zamanda sınıf bildirimi içinde bildirmek ve uygulamaktır. Genelde uygulamalarda birinci yol tercih edilir. Bir sınıf dışında gerçekleşen bir fonksiyon tanımının genel sözdizimi şu şekildedir:

```
dönüş_tipi SınıfAdı::fonksiyon_adi(parametre_listesi)
{
      fonksiyon_gövdesi
}
```

Tanımlanan fonksiyon içinde, sınıfa ait tüm üyelere adları kullanılarak erişilebilir. Sınıf üyeliği otomatik olarak sağlanır ve aynı sınıfa ait fonksiyonlar birbirini doğrudan çağırabilir. Belirli bir nesne için bir fonksiyon çağrıldığında, ilgili fonksiyon bu nesnenin verilerini işleyebilir. Artık Araba sınıfının hızlanma ve yavaşlama fonksiyonlarını uygulayabiliriz.

```
class Araba
{
public:
    char marka[30];
    char model[30];
    float fiyat;
    int hiz;
```

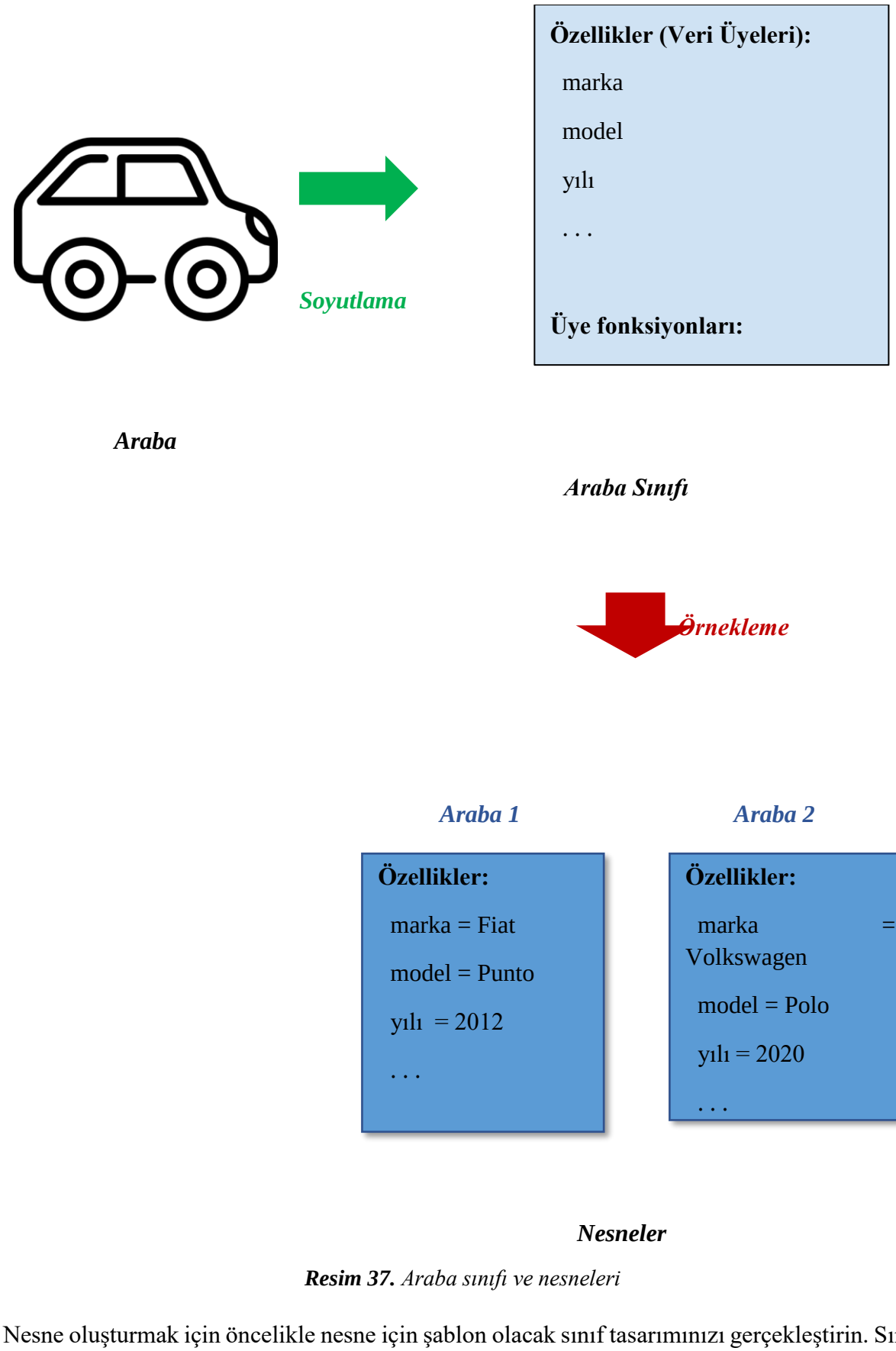
```
    void hizlanma();
    void yavaslama();
};

void Araba::hizlanma()
{
    hiz = hiz + 10;
    cout << "Araba hizlaniyor." << endl;
}
void Araba::yavaslama()
{
    hiz = hiz - 10;
    cout << "Araba yavasliyor." << endl;
}
```

UYARI:

Her yöntemin ve sınıfın başına, kullanıcıya tanımlanan üye veya sınıfın ne iş yaptığını belirten bir yorum satırı ekleyin.

Nesne Tanımlama

Nesne, sınıfın bir örneğidir. Bir sınıf tanımlandığında, bellek tahsis edilmez, ancak bir nesne oluşturulduğunda (somutlaştırıldığında) bellek ayrılır. Sınıfta tanımlanan verileri kullanmak ve fonksiyonlara erişmek için nesneler oluşturmamız gerekir. Daha önce, bir sınıf oluşturmanın bir veri türü oluşturmaya benzediğini belirtmiştik. Bu ilişki, ilerleyen aşamalarda nesneleri daha iyi öğrenmeye başladığınızda belirginleşecektir. Bir sınıfın adını temel bir veri türünün adı gibi (int, char, float) kullanabilirsiniz. Bir nesne değişkeni, belirli bir sınıfın nesnesini depolayan bir değişkendir. Aşağıdaki görselde araba sınıfı kullanılarak birden fazla nesnenin nasıl oluşturulduğunu görebilirsiniz.



Resim 37. *Araba sınıfı ve nesneleri*

Nesne oluşturmak için öncelikle nesne için şablon olacak sınıf tasarımınızı gerçekleştirin. Sınıf tanımlamanızı gerçekleştirdikten sonra nesne değişkeni için bir tanımlayıcı oluşturun ve hangi

sınıftan oluşacağını belirtiniz. Nesne oluşturmak için aşağıda verilen sözdizimlerinden birini kullanabilirsiniz:

```
SınıfAdı nesneDeğişkeni (parametre_listesi);


SınıfAdı nesneDeğişkeni = SınıfAdı (parametre_listesi);
```

A3. Kendi Sınıfını Afişle!

Süre: 60 dk.

Kazanımlar: K7. C++ programlamada sınıf ve nesne tanımlamasını farklı durumlarda uygular.

Hazırlık: Öğrenciler dörderli gruplar hâlinde bilgisayar başındadır.

Uygulama: Eğitmen A2 etkinliğinde grupların incelediği afişleri bu etkinlikte de materyal olarak kullanacaklarını belirtir. İlk olarak öğrencilere bir sınıf ve bu sınıfa ait nesneleri düşünmelerini ister. Öğrenciler burada A1 etkinliğinde çizdikleri nesne ve sınıflardan birine karar verebilirler. Belirledikleri sınıf ve nesneleri afişteki gibi kodlayarak, kodlarını çalıştırıp test etmeleri istenir. Eğitmen gruplara çalışan kodları, grup afişindekine benzer şekilde yeni bir afişe dönüştürün der. Bu afişte de grup afişindeki gibi kod satırlarını açıklayan bilgiler olması gerektiğini belirtir. Buradaki amacınız afiş aracılığıyla diğer arkadaşlarınıza konuyu öğretmenizdir, artık öğretmen sizsiniz diye de ekler. Eğitmen bu şekilde öğrencilerden doğru çalışan kodlar ile grup afişlerini yeniden tasarımlarını ister.

Eğitmene Öneriler: Eğitmen öğrencilerine yeni afiş tasarlama aşamasında “canva.com” afiş tasarlama programını kullandırabilir. Bu noktada afişin revize edilebilir dosyası öğrencilerle paylaşılabilir. Böylece afiş üzerindeki kodlar ve açıklamalar üzerinde öğrenciler değişiklik yapabilir. Diğer bir alternatif ise, Word ortamında afişi oluşturmaları istenebilir.

B. Kısmi Öğrenme Görevleri

Süre: 60 dk.

Materyal: O9. B.1 Kısmi Öğrenme Görevleri Afişi

Hazırlık: Kısmi öğrenme görevleri afişi öğrencilere ÖYS ortamında süreli ödev olarak açılır.

Uygulama: Öğrenciler afişteki görevleri istedikleri sırada ve sayıda kendi tercihlerine bağlı olarak verilen süre içinde uygulamaya başlar. Her bir görevi tamamlayan öğrencilere, göreve ilişkin beceri rozeti verilir. Öğrenciler görev seçimlerini ve tamamladıkları görevleri eğitmene bildirmelidir. Eğitmenler ihtiyaç duydukları zaman öğrencilere görevi tamamlama aşamasında anlık geri bildirimlerde bulunur. Öğrenciler, ÖYS üzerinden süreli ödev olarak açılan kısmi öğrenme görevlerinden kendi tercihlerine göre istedikleri sayıda görev tamamlayıp eğitmene iletir. Süre bitiminde eğitmen görevleri ve yanıtlarını ÖYS ya da GitHub üzerinden öğrencilere gönderir.

Kısmi Öğrenme Görevleri Yanıtlar: Görevler, öğrenciye verilecek beceri rozetleri ile isimlendirilmiştir. Her bir rozetin altında ilgili görevin yanıtları verilmektedir. Eğitmen bu yanıtları uygulama süresinin sonunda GitHub üzerinden öğrencilerle paylaşır.

Denetleyici: Kardeş olan iki peyzaj ustası kare şeklinde olan bahçelerine peyzaj yapmak istiyor. Her ikisi de kendi bahçelerine çit gerecektir. Çit için kare şeklindeki iki bahçenin kenar bilgilerini ölçerek, bahçelerin çevresini hesaplayan bir program tasarlarlar. Ancak program düzgün çalışmamaktadır. Programı düzeltmek zorundalar çünkü çevrelerini hesaplayacakları daha pek çok bahçe var. Bu iki kardeşe programı baştan tasarlamaları için yardım ediniz.

İpucu! Ustanın kullandığı hatalı programda sınıf içerisinde kullanılacak kenar bilgisi değer atama yoluyla gerçekleşirken, fonksiyon tanımlama sınıf dışında yazılmıştır. Tasarlayacağınız kodda bu detayların bulunmasına dikkat ediniz.

```
#include <iostream>
using namespace std;

class Kare {
private:
       float kenar;
public:
    void deger_atama(float);
    float cevre() {
        return 4 * kenar;
    }
};

void Kare::deger_atama (float k) {
    kenar = k;
}

int main () {
    Kare kare;
    kare.deger_atama (4.3);

    cout<<"Kare Cevresi: "<<kare.cevre()<<"\n";
    return 0;
}
```

Kodun Çıktısı:

```
Kare Cevresi: 17.2
```

Kodlayıcı: Ev sahipleri bahçelerine bir havuz yaptırmak istiyor. Bahçeyi inceleyen ustanın, havuzun yapılacağı alanı hesaplamaya ihtiyacı var. Ev sahiplerine istedikleri havuzun yarıçapını belirlemelerini istiyor. Ev sahiplerinden bu bilgiyi aldıktan sonra, daire şeklindeki

havuzun alanını hesaplayan bir programa bilgileri giriyor. Ustanın kullandığı programın kodlarını tasarlayınız (Dairenin alanı hesaplama formülü: $pi * r * r$ ve $pi = 3.14$).

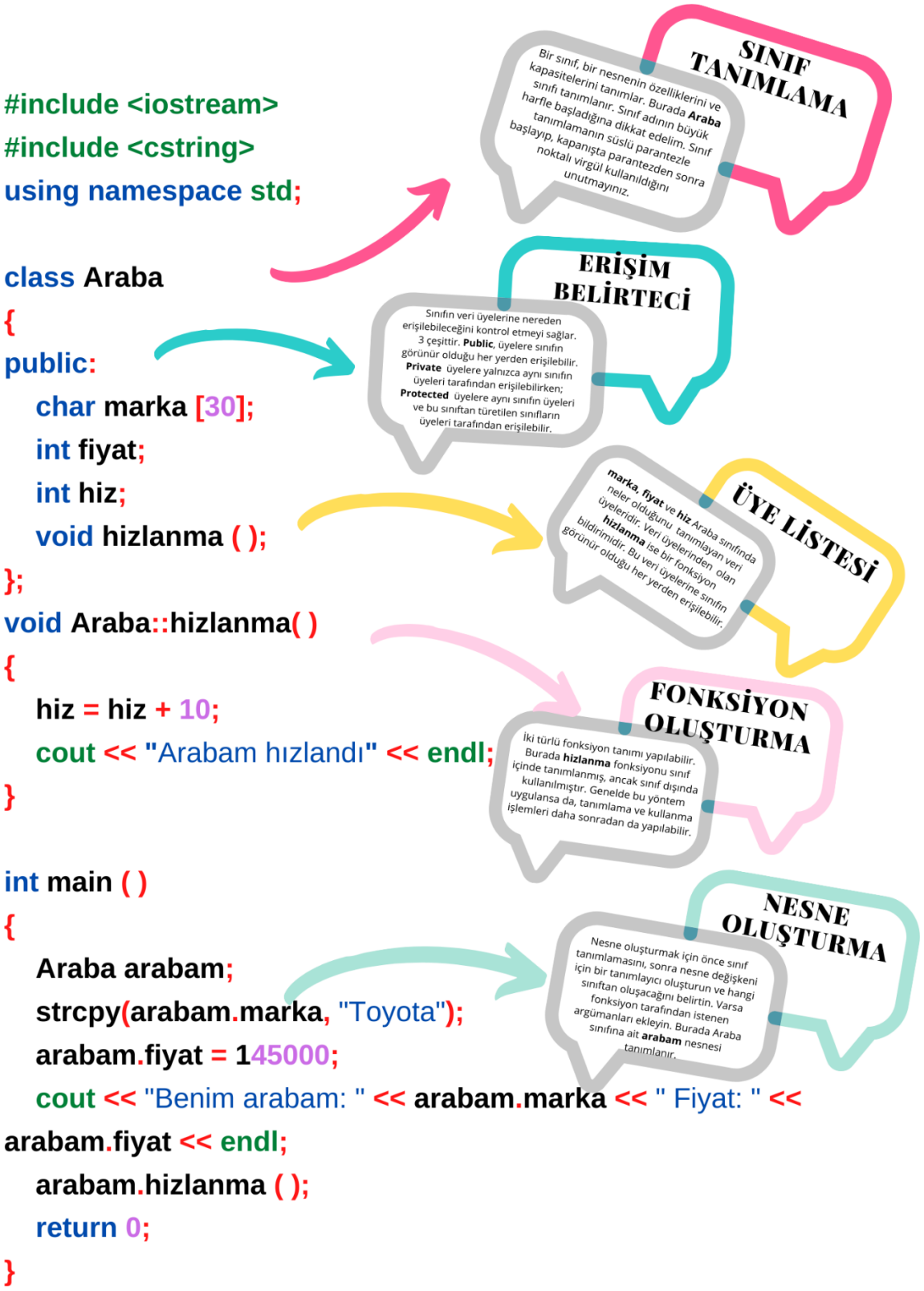
İpucu! Sınıf içerisinde tanımlanacak yarıçap bilgisinin usta tarafından erişilebilir olmasına dikkat ediniz.

```
#include <iostream>
using namespace std;

class Daire {
public:
    float yari_cap;
    float alan_bul(float yari_cap) {
        return 3.14 * yari_cap * yari_cap;
    }
};

int main () {
    Daire daire;
    cout << "Dairenin yaricapini giriniz: " << endl;
    cin >> daire.yari_cap;

    cout<<"Daire Alani: "<<daire.alan_bul(daire.yari_cap)<<"\n";

    return 0;
}
```

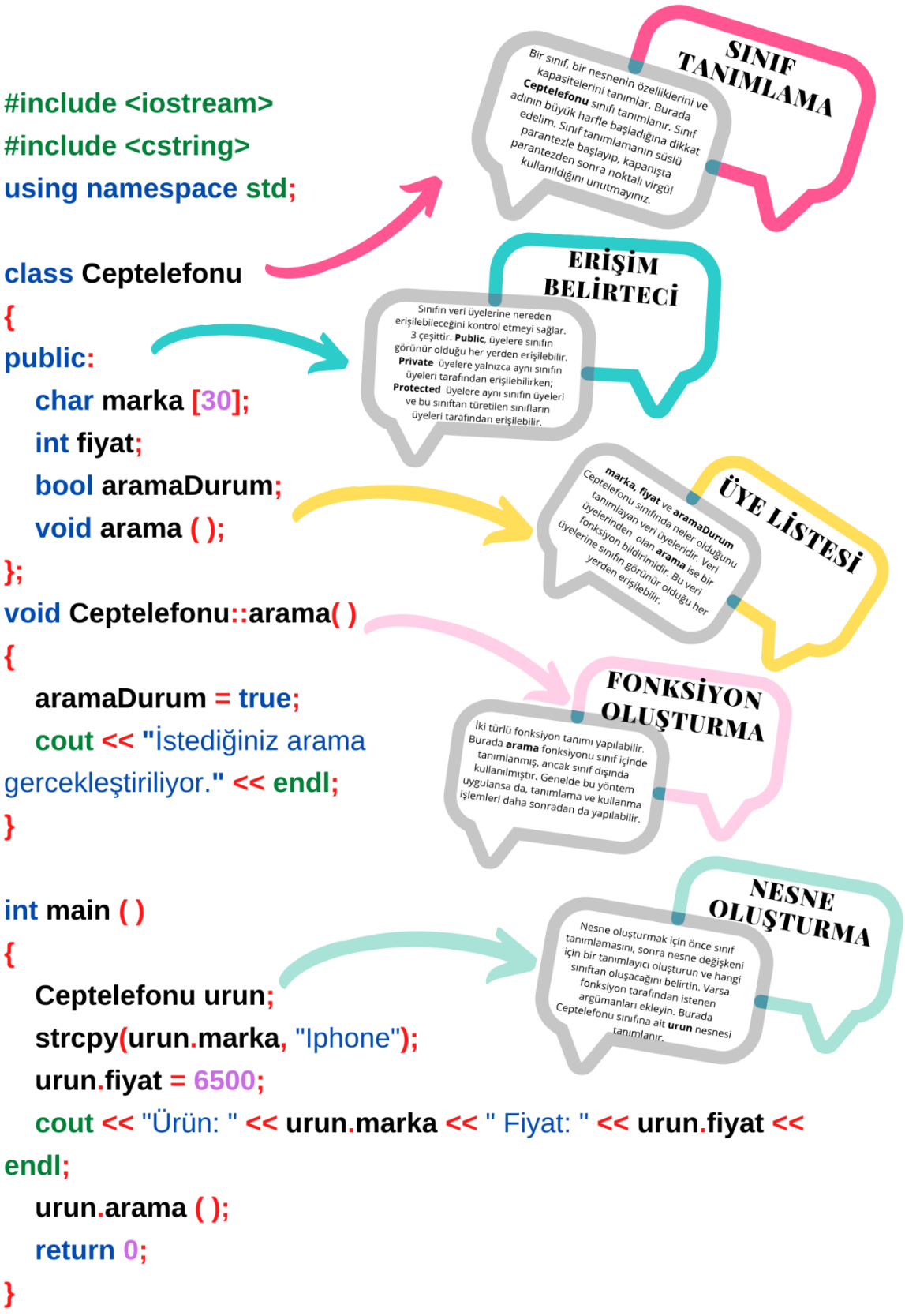
Kodun Çıktısı:

```
Dairenin yaricapini giriniz:
4.2

Daire Alani: 55.3896
```

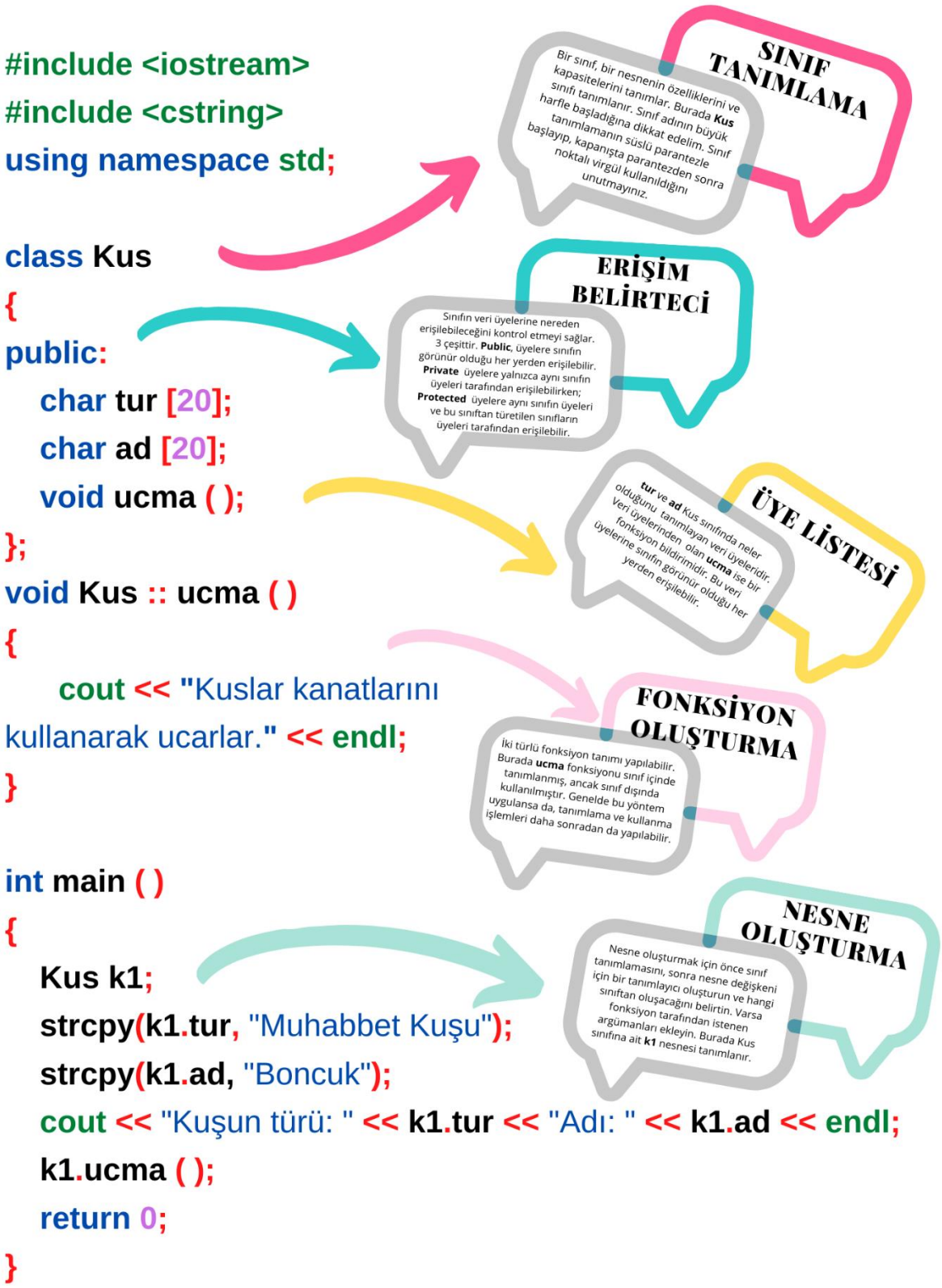
Hafta 9. Ders Materyalleri



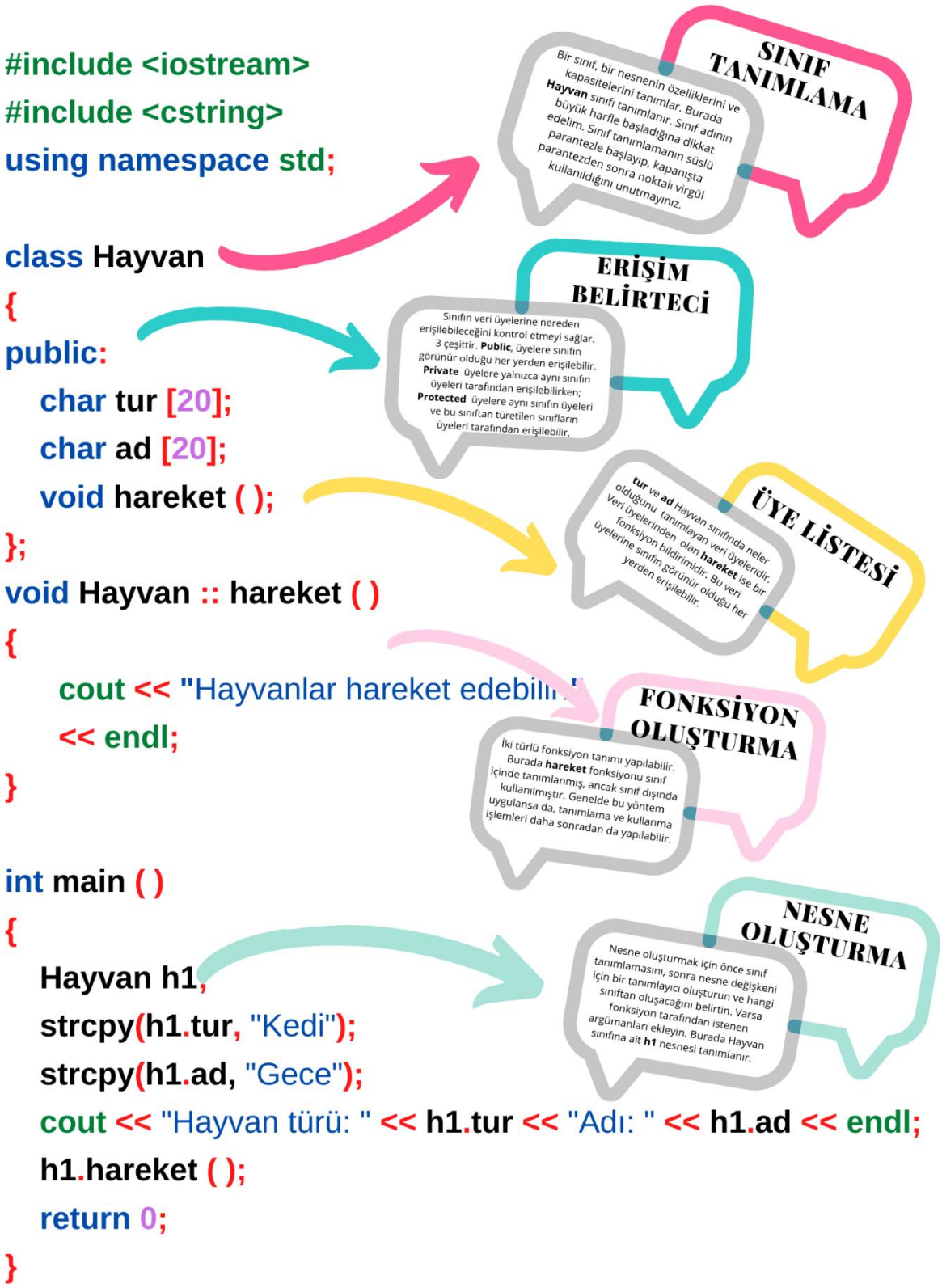
Resim 38. *Araba sınıfı afişi*



Resim 39. *Cep telefonu sınıfı afişi*



Resim 40. *Kuş sınıfı afişi*



Resim 41. *Hayvan sınıfı afişi*

O9. B. 1. Kısmi Öğrenme Görevleri Afişi

Kısmi öğrenme görevleri afişine ulaşmak için buraya tıklayınız.

Hafta 10. Nesne Yönelimli Programlama

Kazanımlar

K1. Yapıcı ve yıkıcı fonksiyon arasındaki farkı ayırt eder.

K2. Yapıcı fonksiyonları kullanarak program geliştirir.

K3. Yıkıcı fonksiyonları kullanarak program geliştirir.

Amaç

Bu haftanın amacı yapıcı, yıkıcı ve sabit fonksiyonları kullanmak, sınıf dosyaları ile büyük boyutlu projeler hazırlamayı anlamaktır.

Önerilen Ders Akışı

A. Öğrenme Görevleri

A1. Yarış Benimle! (40 dk.)

- ❏ Ders Arası (5 dk.)

A2. Kod Satırlarını Tamamla! (60 dk.)

- ❏ Ders Arası (5 dk.)
- ❏ Motivasyon Oyunu (10 dk.)

A3. Sınıfları Dosyala! (50 dk.)

- ❏ Ders Arası (10 dk.)

B. Kısmi Öğrenme Görevleri (60 dk.)

A. Öğrenme Görevleri

A1. Yarış Benimle!

Süre: 40 dk.

Kazanımlar: K1. Yapıcı ve yıkıcı fonksiyon arasındaki farkı ayırt eder.

Materyaller: O10. A1. 1. Yarışma Kartları

O10. A1. 2. Kod Örneği

Hazırlık: Bir kura torbasında yarışma kartları kesikli çizgilerden kesilerek kuraya hazırlanır. Kod örneği ÖYS'den öğrenci erişimine açılır ya da tahtaya yansıtılır.

Uygulama: Eğitmen öğrencilerine bu öğrenme görevinde bir oyun oynayacaklarını söyler. Sınıfı iki gruba ayırır. Her gruba kod örnekleri verilir. Bu örneklerde hem yapıcı fonksiyon hem de yıkıcı fonksiyon kodlarının nasıl kullanıldığı bulunmaktadır. Oyunda gruplara sırayla bir kart çektirilir ve yarışma kartı gruba okunur. Bu kartlarda ise, yapıcı ya da yıkıcı fonksiyon arasındaki farkı anlatan bir özellik yazmaktadır. Grup çektikleri yarışma kartındaki özelliğin, ellerindeki kod örneğine bakarak doğru ya da yanlış bir ifade olup olmadığına karar verecektir. Doğru bilinen her bir karta 10 puan eklenir. Grup puanları sınıfta tahtaya yazılır. Bu şekilde en çok puan toplayan grup, diğer gruptan bir ödül alır. Eğitmen oyun sırasında yapıcı ve yıkıcı fonksiyonları detaylandırır.

Eğitmene Öneriler: Eğitmen bu oyunu Web 2.0 teknolojilerinden yararlanarak hazırlayabilir. Oyun sonunda grup ödülüne öğrencilerle karar verilebilir ya da basit bir şekilde kazanan grubu alkışlamak gibi takdir hareketleri uygulanabilir. Her bir kartta eğitmen konu içeriğine bağlı olarak açıklamalarda bulunabilir.

Konu İçeriği: Yapıcı ve Yıkıcı Fonksiyonlar

Tanımladığınız sınıf içerisinde yapıcı (constructor) ve yıkıcı (deconstructor) olmak üzere iki özel fonksiyon türü olabilir. Her ikisi de isteğe bağlıdır ve diğer fonksiyonları sağlayamayacağı özel fonksiyonlar sağlarlar. Yapıcı fonksiyonlar sınıftan bir nesne oluşturulduğu zaman otomatik olarak çağrılır ve genellikle veri üyeleri için başlangıç değerlerini atamak için kullanılır. Örneğin cep telefonu sınıfı için düşünecek olursak yapıcı fonksiyon yıl veri üyesinin değerini sıfır yapabilir. Her zaman sınıfla aynı isme sahiptir ve int, void vb. gibi bir dönüş değerine sahip olamaz.

Yıkıcı fonksiyonlar ise, bir nesne yok edildiğinde otomatik olarak çağrılır ve gerekli tüm temizleme görevlerini gerçekleştirir. Yıkıcı fonksiyonlar, her zaman sınıfla aynı adla adlandırılır, ancak başında yaklaşık işareti (~) bulunur. Yıkıcı fonksiyonlar da bir dönüş değerine sahip olamaz. Hem yapıcı hem de yıkıcı fonksiyonlar diğer fonksiyonlar gibi tanımlanır ve uygulanır. Sınıf içinde eş zamanlı olarak tanımlamaları yapılacaksa aşağıdaki sözdizimi kullanılır.

```
class SınıfAdi
{
SınıfAdi (parametre listesi){
        yapıcı fonksiyon gövdesi
}
~SınıfAdi (){
        yıkıcı fonksiyon gövdesi
}
};
```

Yapıcı ve yıkıcı fonksiyonların tanımlamaları daha sonra yapılacaksa aşağıdaki sözdizimi kullanılır.

```
class SınıfAdi
{
        SınıfAdi (parametre listesi);
        ~SınıfAdi ();
};
SınıfAdi::SınıfAdi (parametre listesi){
        yapıcı fonksiyon gövdesi
}
SınıfAdi::~SınıfAdi (){
        yıkıcı fonksiyon gövdesi
}
```

Yapıcı fonksiyonun parametre alabileceğine dikkat ediniz. Parametre alabilen bir yapıcı fonksiyon oluşturursanız, sınıfın kullanımı sırasında nesne oluştururken bu parametreler için değerler sağlanmalıdır. Diğer taraftan, yıkıcı fonksiyonların argümanları olamaz. Otomatik olarak çağrıldığından, kullanıcının bağımsız değişken sağlama şansı yoktur.

```
class Ceptelefonu
{
public:
    char model[30];
    float fiyat;
    bool aramaDurum;
    bool mesajDurum;

    void arama();
    void mesaj_gonder();

    Ceptelefonu(){
```

```
        aramaDurum = false;
        mesajDurum = false;
    }
    ~Ceptelefonu(){
        cout << "Nesne yok edildi." << endl;
    }
};
```

A2. Kod Satırlarını Tamamla!

Süre: 60 dk.

Kazanımlar: K2. Yapıcı fonksiyonları kullanarak program geliştirir.

K3. Yıkıcı fonksiyonları kullanarak program geliştirir.

Materyaller: O10. A2. 1. Grup Görev Kartları

Hazırlık: Materyallerin birer çıktısı alınır ve gruplara dağıtmak için kesilir.

Uygulama: Öğrenciler beşerli gruplar hâlinde çalışmaktadır. Bu etkinlikte eğitmen istasyon tekniği kullanmaktadır. Bunun için sınıfta da dört farklı grup masası oluşturulur. Her masada bir görev kartı bulunmaktadır. Gruplar 5 dk. süre içinde grup görev kartını bilgisayarda kodlayarak tamamlamaya çalışır. Süre bitiminde grupların saat yönünde bir sonraki grup masasına geçmesi istenir. Diğer grubun kaldığı noktadan grup masasındaki görev kartı tamamlanmaya çalışılır. Bu şekilde tüm gruplar her bir grup masasına geçişini tamamladıktan sonra dönme işlemi bitirilir. Dönme işleminin sonunda eğitmen sırayla ilk grubun görevini ve masada oluşan çözümü arkadaşlarına açıklamalarını ister. Varsa hatalı noktalar giderilir. Bu şekilde tüm grup masalarına hak verilir ve tüm kodlardaki görevin amacı eğitmen tarafından açıklanır.

Eğitmene Öneriler: Eğitmen grup yanıtlarını sadece bir arkadaşın aktarması için gruplara bir moderatör belirlemelerini isteyebilir. Bu şekilde bir ya da iki arkadaşın bilgisayar başında diğer arkadaşların da katkısıyla kodu yazmaları hızlandırılmış olur. Grup oluşturma aşamasında bu moderatörlerin seçilmesi sağlanmalıdır. Diğer bir nokta ise, tüm grupların son kod satırlarını padlet.com gibi bir dijital tartışma panosuna göndermeleri istenebilir. Bu şekilde grupların yaptıklarının tüm öğrenciler tarafından erişilebilir olması sağlanır.

Konu İçeriği: Grup görevlerinin amaçlarını ve yanıtlarını aşağıda bulabilirsiniz.

Grup 1: Basit bir Araba sınıfı oluşturarak, birden fazla nesne tanımlaması yapmak.

```cpp
#include <iostream>
#include <cstring>
using namespace std;
// Bazi niteliklere sahip bir Araba sinifi olusturun
class Araba {
public:
    char marka[30];
    char model[30];
    int yil;
    int fiyat;
};
int main() {
    // Ilk Araba nesnesini olusturun
    Araba araba1;
    strcpy(araba1.marka, "Suzuki");
    strcpy(araba1.model, "Vitara");
    araba1.yil = 2016;
    araba1.fiyat = 225000;
    // Ikinci Araba nesnesini olusturun
    Araba araba2;
    strcpy(araba2.marka, "Volkswagen");
    strcpy(araba2.model, "T-Roc");
    araba2.yil = 2020;
    araba2.fiyat = 360000;
    // Nesnelerin ozelliklerini yazdirin
    cout << "Araba 1: " << araba1.marka << ", " << araba1.model << ", " << araba1.yil << ", " << araba1.fiyat << " \n";
    cout << "Araba 2: " << araba2.marka << ", " << araba2.model << ", " << araba2.yil << ", " << araba2.fiyat << " \n";
}
```

Kodun Çıktısı:

```
Araba 1: Suzuki, Vitara, 2016, 225000
Araba 2: Volkswagen, T-Roc, 2020, 360000
```

Grup 2: Araba sınıfına bir fonksiyon ekleme ve nesne tanımlama.

```
#include <iostream>
#include <cstring>
using namespace std;
class Araba {
public:
    char marka[30];
    char model[30];
    int hiz;
    int hizlan(int x);
};
int Araba::hizlan(int x) {
  return hiz + x;
}
int main() {
    Araba araba1; // Araba nesnesini olusturun
    strcpy(araba1.marka, "Opel");
    strcpy(araba1.model, "Astra");
    araba1.hiz = 120;
    cout << "Araba: " << araba1.marka << " " << araba1.model << " \n";
    cout << "Yeni Hiz: " << araba1.hizlan(30); // Fonksiyonu parametre ile cagirin
}
```

Kodun Çıktısı:

```
Araba: Opel Astra
Yeni Hiz: 150
```

Grup 3: Araba sınıfına sınıf içi bir yapıcı fonksiyon ekleme ve nesne tanımlama.

```
#include <iostream>
#include <cstring>
using namespace std;
class Araba {
public:
    string marka;
    string model;
    int yil;
    int fiyat;
    //Parametreli sinif ici yapıcı fonksiyon
    Araba(string x_marka, string x_model, int x_yil, int x_fiyat) {
        marka = x_marka;
        model = x_model;
        yil = x_yil;
        fiyat = x_fiyat;
    }
};

int main() {
    // Yapıcı fonksiyonu farkli degerlerle cagirarak Araba nesneleri olusturma
    Araba araba1("Suzuki", "Vitara", 2016, 225000);
    Araba araba2("Volkswagen", "T-Roc", 2020, 360000);

    // Degerleri yazdirma
    cout << araba1.marka << " " << araba1.model << " " << araba1.yil << " " << araba1.fiyat << "\n";
    cout << araba2.marka << " " << araba2.model << " " << araba2.yil << " " << araba2.fiyat << "\n";
}
```

Kodun Çıktısı:

```
Suzuki Vitara 2016 225000
Volkswagen T-Roc 2020 360000
```

Grup 4: Araba sınıfına sınıf dışı bir yapıcı fonksiyon ekleme ve nesne tanımlama.

```
#include <iostream>
#include <cstring>
using namespace std;
class Araba {
public:
    string marka;
    string model;
    int yil;
    int fiyat;
    // Yapici fonksiyon bildirimi
    Araba(string x_marka, string x_model, int x_yil, int x_fiyat);
};
//Sinif disinda yapıcı fonksiyon tanimlama
Araba::Araba(string x_marka, string x_model, int x_yil, int x_fiyat) {
    marka = x_marka;
    model = x_model;
    yil = x_yil;
    fiyat = x_fiyat;
}
int main() {
    // Yapıcı fonksiyonu farkli degerlerle cagirarak Araba nesneleri olusturma
    Araba araba1("Suzuki", "Vitara", 2016, 225000);
    Araba araba2("Volkswagen", "T-Roc", 2020, 360000);
    // Degerleri yazdirma
    cout << araba1.marka << " " << araba1.model << " " << araba1.yil << " " <<
araba1.fiyat << "\n";
    cout << araba2.marka << " " << araba2.model << " " << araba2.yil << " " <<
araba2.fiyat << "\n";
}
```

Kodun Çıktısı:

```
Suzuki Vitara 2016 225000
Volkswagen T-Roc 2020 360000
```

A3. Bayrak Yarışı!

Süre: 50 dk.

Kazanımlar: K2. Yapıcı fonksiyonları kullanarak program geliştirir.

K3. Yıkıcı fonksiyonları kullanarak program geliştirir.

Materyal: O10. A3. 1. Bayrak Yarışı Kartları

Hazırlık: Materyallerin çıktısı alınır ve bayrak yarışında kademeli olarak gruplara dağıtmak için kesilir. Öğretmen masasında kırmızı ve mavi iki farklı renkte bayrak asılıdır.

Uygulama: Sınıf iki kırmızı ve mavi takım olmak üzere iki gruba ayrılır. Her iki takımda ikişerli alt gruplar hâlinde çalışacaktır. Eğitmen bu etkinlikte öğrencilere üç aşamalı bayrak yarışı yapacaklarını söyler. Takımdaki alt gruplardan herhangi ikili grup ilk görevi eksiksiz ve başarılı bir şekilde tamamlarsa, hemen öğretmen masasındaki bayrağı kaldıracak ve o takım sonra ikinci göreve erişecektir. İkinci görev eğitmen tarafından bayrağı kaldıran takıma başlamaları için verilir. Örneğin kırmızı takımdan bir oyuncu bile bayrağı kaldırmayı başardıysa, kırmızı takımın diğer tüm oyuncuları birinci görevi bırakıp ikinci göreve başlayacaktır. İkinci bayrak kaldıran takım ise üçüncü aşamaya geçecektir. Bayrağın kaldırılabilmesi için kodların sorunsuz çalışması ve eğitmenin kontrolünü gerçekleştirmesi gereklidir. Bu nedenle oyunda öğrencilerin zamanla yarışacaklarını açıklayan eğitmen, aynı zamanda öğrencilerin ikişerli bir şekilde görevleri tamamlamaya çalışmalarını ister. Böylece üçüncü görevi de tamamlayan takım üçüncü kez bayrağı kaldırdığında oyun sona erer. Oyunun sonunda bayrağı kaldıran öğrenciler görevleri nasıl yaptıklarını sınıftaki diğer arkadaşlarına açıklar.

Eğitmene Öneriler: Eğitmen bayrağın bir takım tarafından kaldırıldığını diğer öğrencilere duyurmak için emir ifadeleriyle "Kırmızı takımın bayrağı kalktı, Kırmızı takım ikinci göreve başla" şeklinde duyurularda bulunmalıdır. İkinci görevin takım içinde hızla yayılması için eğitmen, takım alt oyuncuları kadar görev çıktısı alabilir. Örneğin 10 kişilik bir kırmızı takım için beş görev kartı çıktısı hazırlanır.

Birinci Bayrak Görevi: Öğrenci sınıfını tanımlama ve iki öğrenci nesnesi oluşturma.

```
#include <iostream>
#include <cstring>
using namespace std;
class Ogrenci {
    int ogr_no;
    char ogr_ad[20];
    char ogr_soyad[20];
    void deger_ata(int no, char ad[], char soyad[])
    {
        ogr_no = no;
        strcpy(ogr_ad, ad);
```

```
        strcpy(ogr_soyad, soyad);
    }
    void goster(){
        cout<<"Ogrenci Bilgi: " << ogr_no <<" "<< ogr_ad << " " << ogr_soyad <<
endl;
    }
};
int main(void) {
    Ogrenci ogr1;
    Ogrenci ogr2;
    ogr1.deger_ata(372, "Arda", "Ozcan");
    ogr2.deger_ata(624, "Duru", "Ozen");
    ogr1.goster();
    ogr2.goster();
    return 0;
}
```

İkinci bayrak görevi: Cep telefonu sınıfını kullanarak nesne oluşturma.

```
#include <iostream>
using namespace std;
class Ceptelefonu{
public:
    char marka[30];
    char model[30];
    int fiyat;
    bool aramaDurum;
    bool mesajDurum;

    Ceptelefonu(int x_fiyat){
        fiyat = x_fiyat;
    }
    ~Ceptelefonu(){
        cout << "Nesne yok edildi." << endl;
    }
    void arama();
    void mesaj_gonder();
```

```
};
void Ceptelefonu::arama()
{
    aramaDurum = true;
    cout << "Istediginiz arama gerceklestiriliyor." << endl;
}
void Ceptelefonu::mesaj_gonder()
{
    mesajDurum = true;
    cout << "Istediginiz mesaj gonderiliyor." << endl;
}

int main(){
    Ceptelefonu urun1(4500);
    Ceptelefonu urun2 = Ceptelefonu(3750);
    cout << "Urun 1 baslangic fiyati: " << urun1.fiyat << endl;
    cout << "Urun 2 baslangic fiyati: " << urun2.fiyat << endl;
    return 0;
}
```

Kodun Çıktısı:

```
Urun 1 baslangic fiyati: 4500
Urun 2 baslangic fiyati: 3750
```

Üçüncü bayrak görevi: Oda sınıfına ait tasarladığınız bir suit odanın ölçü bilgilerini kullanıcıya sorup, suit odanın alan ve hacmini hesaplayan programı yazınız.

```
#include <iostream>
using namespace std;

class Oda {
private:
    double uzunluk;
    double genislik;
    double yukseklik;
```

```
public:
    void veriAl(double uzn, double gns, double yks) {
        uzunluk = uzn;
        genislik = gns;
        yukseklik = yks;
    }

    double alanHesapla() {
        return uzunluk * genislik;
    }

    double hacimHesapla() {
        return uzunluk * genislik * yukseklik;
    }
};

int main() {
    Oda suitOda;
    suitOda.veriAl(3.5, 4.3, 3.2);
    cout << "Suit Oda Alan: " << suitOda.alanHesapla() << endl;
    cout << "Suit Oda Hacim: " << suitOda.hacimHesapla() << endl;
    return 0;
}
```

B. Kısmi Öğrenme Görevleri

Süre: 60 dk.

Materyal: O10. B. Kısmi Öğrenme Görevleri Afişi

Hazırlık: Kısmi öğrenme görevleri afişi öğrencilere ÖYS ortamında süreli ödev olarak açılır.

Uygulama: Öğrenciler afişteki görevleri istedikleri sırada ve sayıda kendi tercihlerine bağlı olarak verilen süre içinde uygulamaya başlar. Her bir görevi tamamlayan öğrencilere, göreve ilişkin beceri rozeti verilir. Öğrenciler görev seçimlerini ve tamamladıkları görevleri eğitmene bildirmelidir. Eğitmenler ihtiyaç duydukları zaman öğrencilere görevi tamamlama aşamasında anlık geri bildirimlerde bulunur. Öğrenciler, ÖYS üzerinden süreli ödev olarak açılan kısmi öğrenme görevlerinden kendi tercihlerine göre istedikleri sayıda görev tamamlayıp eğitmene iletir. Süre bitiminde eğitmen görevleri ve yanıtlarını ÖYS ya da GitHub üzerinden öğrencilere gönderir.

Kısmi Öğrenme Görevleri Yanıtlar: Görevler, öğrenciye verilecek beceri rozetleri ile isimlendirilmiştir. Her bir rozetin altında ilgili görevin yanıtları verilmektedir. Eğitmen bu yanıtları uygulama süresinin sonunda GitHub üzerinden öğrencilerle paylaşır.

Tasarlayıcı: Bir futbol takımının teknik direktörü, futbolcuların ad-soyad, forma numarası ve attığı gollerin sayısını tutmak için bir program hazırlamak ister. Teknik direktör bunun için örnek bir program yazar. Programda örnek olarak Futbolcu sınıf yapısı ve bu sınıfa ait 2 futbolcunun bilgileri nesne olarak tutulmaktadır. Bu programın kodlarını tasarlayınız.

```
#include <iostream>
#include <string>
using namespace std;
class Futbolcu{
public:
	string ad_soyad;
	int forma_no;
	int gol_sayisi;
	Futbolcu(string x_ad_soyad, int x_forma_no, int x_gol_sayisi){
	ad_soyad = x_ad_soyad;
	forma_no = x_forma_no;
	gol_sayisi = x_gol_sayisi;
	}
};
int main()
{
	Futbolcu f1("Arda Alp", 21 , 13);
	Futbolcu f2("Burakcem", 23, 17);
	cout << f1.ad_soyad << " " << f1.forma_no << " " << f1.gol_sayisi << "\n";
	cout << f2.ad_soyad << " " << f2.forma_no << " " << f2.gol_sayisi << "\n";
	return 0;

}
```

Kodun Çıktısı:

```
Arda Alp 21 13
Burakcem 23 17
```

Kodlayıcı: Grafik programlarında kullanmak üzere nokta nesnelerini tanımlamak için bir Nokta sınıfı oluşturalım. Noktalar iki boyutlu düzlemde yer alacağından özellik olarak x ve y

koordinatları olmak üzere iki adet koordinat bilgisine sahiptir. Programınızda noktaların sahip olması gereken yetenekler (davranışlar) ise şunlar olmalıdır:

- Noktalar, düzlemde herhangi bir yere konumlanabilmeli: git fonksiyonu
- Noktalar bulundukları koordinatları ekranda gösterebilmeli: goster fonksiyonu
- Noktalar, sıfır (0,0) koordinatında olup olmadıkları sorusunu yanıtlayabilmeli: sifir_mi fonksiyonu

```
#include <iostream>
using namespace std;
class Nokta{
    int x,y;
    public:
    void git(int, int);
    void goster();
    void sifir_mi();
};
void Nokta::git(int yeni_x, int yeni_y)
{
    x = yeni_x;
    y = yeni_y;
}
void Nokta::goster()
{
    cout << "X noktasi: " << x << ", Y noktasi: " << y << endl;
}
void Nokta::sifir_mi()
{
    if ((x == 0) && (y == 0))
        cout << "n1 su anda sifir noktasindadir." << endl;
    else
        cout << "n1 su anda sifir noktasinda degildir." << endl;
}
int main() {
    Nokta n1,n2;
    n1.git(78,34);
    n1.goster();
    n1.git(61,35);
```

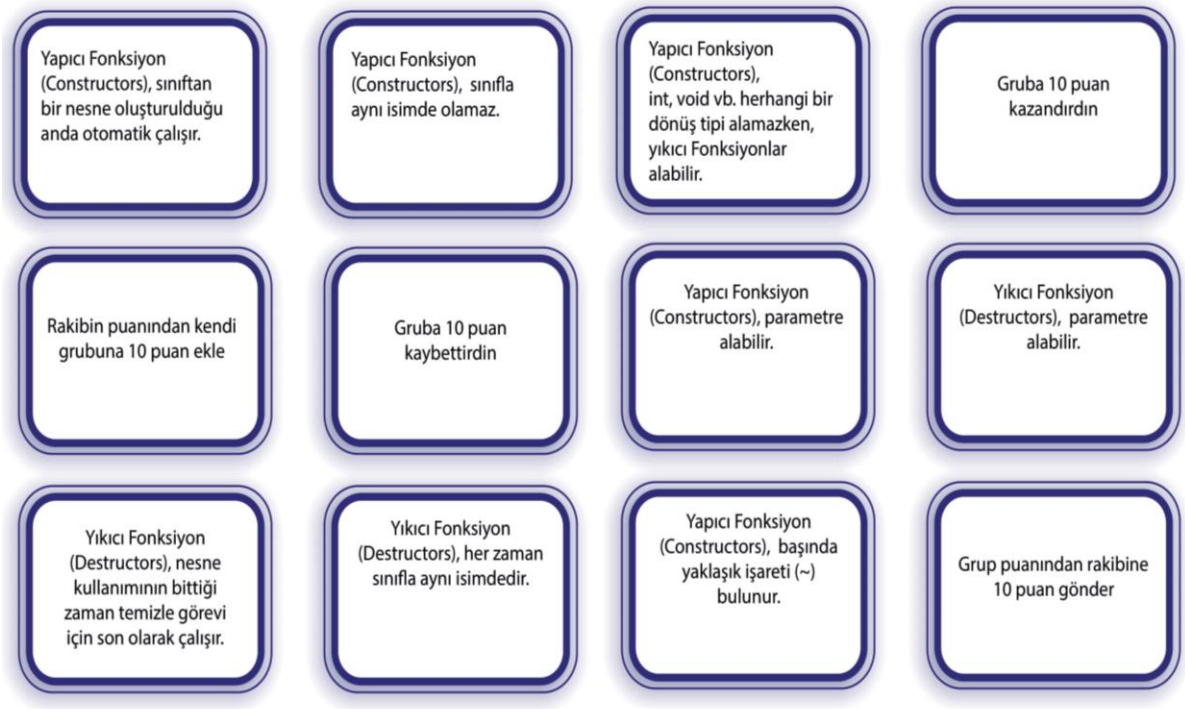
```
    n1.goster();
    n1.sifir_mi();
    n2.git(0,0);
    n2.sifir_mi();
    return 0;
}
```

Kodun Çıktısı:

```
X noktasi: 78, Y noktasi: 34
X noktasi: 61, Y noktasi: 35
n1 su anda sifir noktasinda degildir.
n1 su anda sifir noktasindadir.
```

Hafta 10. Ders Materyalleri

O10. A1. 1. Yarışma Kartları



O10. A1. 2. Kod Örneği

```
class Ceptelefonu
{
public:
   char model[30];
   float fiyat;
   bool aramaDurum;
   void arama();

   Ceptelefonu(){
      aramaDurum = false;
   }

   ~Ceptelefonu(){
   }

};
```

Yapıcı Metot (Constructs)

Yıkıcı Metot (Deconstructs)

O10. A2. 1. Grup Görev Kartları

```
#include <iostream>
#include <cstring>
using namespace std;
class Araba {
public:
    char marka[30];
    char model[30];
    int yil;
    int fiyat;
};
int main() {
    Araba araba1;
    strcpy(araba1.marka, "Suzuki");
    araba1.fiyat = 225000;
    strcpy(araba2.model, "T-Roc");
    araba2.yil = 2020;
    cout << "Araba 1: " << araba1.marka << ", " << araba1.model << ",
" << araba1.yil << ", " << araba1.fiyat << " \n";
    return 0;
}
```

Kodun Çıktısı:

```
Araba 1: Suzuki, Vitara, 2016, 225000
Araba 2: Volkswagen, T-Roc, 2020, 360000
```

Görev: Grup 1

Yukarıdaki kod çıktısının aynısını ekranda görmek için verilen koddaki eksik satırları tamamlayınız.

```
#include <iostream>

#include <cstring>

using namespace std;
class Araba {
public:
    char marka[30];
    char model[30];
    int hiz(int max_hiz);
};
int Araba::
int main() {
    strcpy(araba1.model, "Astra");
    cout << "Araba: " << araba1.marka << " " << araba1.model << "
\n";
    cout << "Hiz: " << araba1.hiz(30); // Fonksiyonu parametre ile
cagirin
    return 0;
}
```

Kodun Çıktısı:

```
Araba: Opel Astra

Hiz: 150
```

Görev: Grup 2

Yukarıdaki kod çıktısının aynısını ekranda görmek için verilen koddaki eksik satırları tamamlayınız.

```
#include <iostream>
#include <cstring>
using namespace std;
class Araba {
public:
    string marka;
    string model;
    int yil;
    int fiyat;
    Araba(string) {
    }
};
int main() {
    Araba araba2("Volkswagen", "T-Roc", 2020, 360000);
    cout << araba1.marka << " " << araba1.model << " " << araba1.yil
<< " " << araba1.fiyat << "\n";
    cout << araba2.marka << " " << araba2.model << " " << araba2.yil
<< " " << araba2.fiyat << "\n";
    return 0;
}
```

Kodun Çıktısı:

```
Suzuki Vitara 2016 225000
Volkswagen T-Roc 2020 360000
```

Görev: Grup 3

Yukarıdaki kod çıktısının aynısını ekranda görmek için verilen koddaki eksik satırları tamamlayınız.

```
#include <iostream>
#include <cstring>
using namespace std;
class Araba {
public:
    string marka;
    string model;
    int yil;
    int fiyat;
    Araba(string x_marka, string x_model, int x_yil, int x_fiyat);
};
{
 }
int main() {
    Araba araba1("Suzuki", "Vitara", 2016, 225000);
    Araba araba2("Volkswagen", "T-Roc", 2020, 360000);
    cout << araba2.marka << " " << araba2.model << " " << araba2.yil
<< " " << araba2.fiyat << "\n";
    return 0;
}
```

Kodun Çıktısı:

```
Suzuki Vitara 2016 225000
Volkswagen T-Roc 2020 360000
```

Görev: Grup 4

Yukarıdaki kod çıktısının aynısını ekranda görmek için verilen koddaki eksik satırları tamamlayınız.

O10. A3. 1. Bayrak Yarışı Kartları

Birinci Bayrak

"Ogrenci" isimli bir sınıf tanımlayarak, bu sınıfta iki öğrenci nesnesi oluşturmak istenmektedir. Sınıf tanımı içerisinde öğrenci numarası, ad ve soyad üyeleri tutulacaktır. Sınıf içerisinde ana fonksiyondan gelen bilgileri atamak için "deger_ata" ve atanan bilgileri göstermek için de "goster" isimli fonksiyon oluşturulması istenmektedir. Ana fonksiyonda iki öğrenci nesnesi tanımlayarak "deger_ata" fonksiyonu ile iki adet öğrenci bilgisini gönderin ve "goster" fonksiyonu kullanarak da öğrenci bilgilerini ekrana yazdırınız.

İkinci Bayrak

"Ceptelefonu" isimli bir sınıf tanımlayarak, bu sınıfta iki cep telefonu nesnesi oluşturmak istenmektedir. Sınıf tanımı içerisinde marka, model, fiyat, arama durum ve mesaj durum üyeleri tanımlanacaktır. Arama durum ve mesaj durum değişkenleri bool olarak tanımlanacaktır. Sınıf içerisinde ana fonksiyondan gelen fiyat bilgisini atamak için bir yapıcı fonksiyon tanımlayınız. Yine sınıf içerisinde tanımlayacağınız "arama" ve "mesaj" isimli iki fonksiyon ile cep telefonu arama ve mesaj durumu bilgisini ekrana yazdırınız. Ana fonksiyonda iki cep telefonu nesnesi tanımlayarak ürün fiyatlarını ekrana yazdırınız.

Üçüncü Bayrak

Oda sınıfına ait tasarladığınız bir suit odanın ölçü bilgilerini kullanıcıdan alarak suit odanın alan ve hacmini hesaplayan programı yazınız. Program içerisinde tanımlayacağınız Oda sınıfının uzunluk, genişlik ve yükseklik isimli üyeleri olacaktır. Sınıf içerisinde tanımlayacağınız "veriAl" fonksiyonu ile ana fonksiyondan gelen verileri alacaksınız. Yine sınıf içerisinde yazacağınız "alanHesapla" fonksiyonu ile alanı, "hacimHesapla" fonksiyonu ile hacmi hesaplayınız. Ana fonksiyonda bir adet Oda nesnesi oluşturarak uzunluk, genişlik ve yükseklik bilgilerini "veriAl" fonksiyonunu çağırarak atayınız. Daha sonra odanın alanını ve hacmini ekrana yazdırınız.

O10. C. 1. Kısmi Öğrenme Görevleri Afişi

Kısmi öğrenme görevleri afişine ulaşmak için buraya tıklayınız.

Hafta 11. C++ Programlama Dilinde Kütüphane Kullanımı ve Dosyalama İşlemleri

Kazanımlar

K1. Program içinde karakter kütüphanesi kullanarak sonucu test eder.

K2. Program içinde katar kütüphanesi kullanarak sonucu test eder.

K3. Dosyalama işleminin gerekliliğini açıklar.

K4. Dosya kütüphanesi kullanarak program geliştirir.

K5. Dosya okuma işlemlerini içeren program tasarlar.

K6. Dosya yazma işlemlerini içeren program tasarlar.

Amaç

Bu haftanın amacı öğrencilerin C++ programlama dili içerisinde bulunan kütüphaneleri kullanma ve dosyalama işlemleri yapabilmesini sağlamaktır.

Önerilen Ders Akışı

A. Giriş (10 dk.)

B. Öğrenme Görevleri

B1. C++ Programlama Dilinde Yerleşik Kütüphaneleri Keşfediyorum (20 dk.)

- ❑ Ders Arası (10 dk.)

B2. Kütüphanelerdeki Bazı Fonksiyonları Kullanarak Kodluyorum (50 dk.)

- ❑ Ders Arası (10 dk.)

B3. Neden Dosyalama İşlemleri Yaparız? (30 dk.)

B4. C++ Dilinde Dosyalama İşlemleri Yapıyorum (20 dk.)

- ❑ Ders Arası (10 dk.)

B5. Dosyalama İşlemleri ile İlgili Verilen Görevleri Kodluyorum (30 dk.)

C. Kısmi Öğrenme Görevleri (50 dk.)

A. Giriş

Süre: 10 dk.

Uygulama: Sınıfta bulunan her öğrenci kendine bir oyun arkadaşı seçer. Herkes seçtiği oyun arkadaşı ile taş, kâğıt ve makas oyununu karşılıklı oynar, oyunu kaybeden kazandığı kişinin arkasına geçerek onun takipçisi olur. Kazananlar sürekli bir diğer kazananla karşılaşarak aynı şekilde taş, kâğıt ve makas oyununu tekrar eder, kaybeden her kişi kazananın arkasına geçmektedir ve kazananın takipçisi olmaktadır. En uzun kuyruğa ulaşan bir başka ifadeyle hiç kaybetmeyen kişi oyunu kazanır.

Eğitmene Öneriler: Oyundaki amaç kazananı belirlemekten çok grup dinamiğini canlı tutmaktır. Oyunu kaybeden öğrenci, kazananın arkasına geçmektedir ve kaybettiği arkadaşı bir sonraki diğer kazananla yarıştığında arkasında bulunduğu için aslında farkında olmadan kaybettiği arkadaşına destek vermektedir.

B. Öğrenme Görevleri

B1. C++ Programlama Dilinde Yerleşik Kütüphaneleri Keşfediyorum

Süre: 20 dk.

Kazanımlar: K1. Program içinde karakter kütüphanesi kullanarak sonucu test eder.

K2. Program içinde katar kütüphanesi kullanarak sonucu test eder.

Materyaller: O11.B1.1 C++ Programlama Dilinde Kütüphaneler

O11.B1.2 Karakter Kütüphanesi

O11.B1.3 Metin Kütüphanesi

Hazırlık: Birinci materyal ÖYS üzerinden erişime açılır. Diğer iki materyalin ise 10'ar tane çıktısı alınır.

Uygulama: Eğitmen öncelikle her bir grup dörderli olacak şekilde sınıfı beş gruba böler. Materyalleri dağıtır ve ÖYS üzerinden erişilecek materyali açtırır. Bu materyallerden "O11.B1.1 C++ Programlama Dilinde Kütüphaneler" öğrenciye ve eğitimcilere kütüphanelerin ne olduğuna yönelik derse giriş yapılması için hazırlanmıştır. Diğer iki materyalde ise noktalı boş olan yerleri öğrenci gruplarının doldurarak kütüphane ve o kütüphanedeki kullanılabilecek hazır fonksiyonları öğrencilerin gerek materyalle gerek akranlarıyla etkileşime girerek öğrenmesi amaçlanmaktadır. Eğitmenlerin bu etkinlikteki görevi öğrenci gruplarını sırayla gezerek ihtiyaç duydukları anda geri bildirimler vererek onları desteklemektir.

Eğitmene Öneriler: Öğrencilere verilen görevlerin cevapları aşağıdaki gibidir. Öğrenci gruplarının doldurdukları görev kâğıtlarının aşağıdaki gibi olması beklenir. Cevaplar kırmızı renkte verilmiştir.

Fonksiyon	İşlevi	Örnek Kullanım	Sonuç
isalpha(c)	c karakteri eğer bir harf ise geriye **true** değilse **false** döndürür.	isalpha('2')	F
isdigit(c)	c karakteri eğer bir rakam ise geriye **true** değilse **false** döndürür.	isdigit('2')	T
isalnum(c)	c karakteri eğer bir rakam veya harf ise geriye **true** değilse **false** döndürür.	isalnum('*')	F
islower(c)	c karakteri eğer bir küçük harf ise geriye **true** değilse **false** döndürür.	islower ('d')	T
isupper(c)	c karakteri eğer bir büyük harf ise geriye **true** değilse **false** döndürür.	isupper('H')	T
tolower(c)	c karakterini küçük harfe çevirir.	tolower('E')	e
toupper(c)	c karakterini büyük harfe çevirir.	toupper('g')	G
strlen (s1)	s1 katarının uzunluğunu döndürür.	s1 = "Merhaba" strlen (s1)	7
strcpy (s1, s2)	s2 katarını s1 katarına kopyalar.	s1= "Merhaba" s2= "Dunya" strcpy (s1, s2)	Dunya
strcat (s1, s2)	s2 katarını s1 katarının sonuna ekler.	s1= "Merhaba" s2= "Dunya" strcat (s1, s2)	MerhabaDunya
strcmp (s1, s2)	s1 ve s2 aynı ise 0 değerini döndürür; Eğer alfabetik olarak s1, s2 metninden önce geliyorsa -1, sonra geliyorsa 1 değerini döndürür.	s1= "Merhaba" s2= "Dünya" strcmp (s1, s2)	M harfi D harfinden sonra geldiği için 1 değerini döndürür.

B2. Kütüphanelerdeki Bazı Fonksiyonları Kullanarak Kodluyorum

Süre: 50 dk.

Kazanımlar: K1. Program içinde karakter kütüphanesi kullanarak sonucu test eder.

K2. Program içinde katar kütüphanesi kullanarak sonucu test eder.

Materyaller: O11.B2.1 C++ Programa Dilinde Bulunan Kütüphaneleri ve Fonksiyonları Kullanma

Hazırlık: Eğitmen dersin bu bölümü için hazırlanan materyali ÖYS üzerinden paylaşır. Code Blocks uygulaması öğrencilerin kodlama yapabilmesi için hazır duruma getirilir.

Uygulama: Eğitmenler C++ programlama dilindeki bazı kütüphane ve hazır fonksiyonları kullanıp uygulaması için hazırlanan görev etkinliğinde en az iki tanesini öğrencilerin seçmesini sağlayarak, öğrencilerin bu derste kodlama yapmasını ister. Diğer görevler ise öğrencilere kodlanması için ev görevi olarak verilebilir. Öğrenciler kodlama esnasında verilen görevler için ilk derste verilen matematik, katar ve karakter kütüphaneleri ilgili materyalleri destekleyici bilgi olarak kullanılırken, öğrencinin ihtiyaç duyduğu anda gerekli işlemsel bilgiyi eğitmenlerden biri sınıfı dolaşarak, biri de kodlamayı öğrencilerin de görebileceği şekilde bilgisayarda kodlayarak sağlayabilir.

Eğitmene Öneriler: Fonksiyon kodlama görevleri ve cevapları aşağıdaki gibidir.

Görev 1: Klavyeden karakterleri sırasıyla girilen “Arda” ismini “a” değişken adıyla karakter dizisinde, “Duru” ismini ise “b” değişken adıyla katarda tutarak ekrana yazdıran kodu oluşturunuz.

Cevap 1:

```
#include<iostream>
using namespace std;
int main()
{
    char a[4];
    char b[5];
    int i;
    cout << "Ilk ismin karakterlerini giriniz: " << endl;
    for(i=0; i < 4; i++) {
        cin >> a[i];
    }
    cout << "Ikinci ismin karakterlerini giriniz: " << endl;
    for(i=0; i < 4; i++) {
        cin >> b[i];
```

```
    }
    b[4] = '\0';
    cout << "Ilk isim: ";
    for(i=0; i < 4; i++) {
        cout << a[i];
    }
    cout << "\nIkinci isim: ";
    cout << b;
    return 0;
}
```

Görev 2: Büyük küçük karışık hâlde yazılmış bir cümleyi her harfi büyük olacak şekilde yazdıralım. "BuGun Hava Cok GUZEl!" cümlesini "Bugun Hava Cok Guzel!" şekline getirelim.

Cevap 2:

```
#include <iostream>
#include <cstring>

using namespace std;

int main()
{
    char mesaj[] = "BuGun Hava Cok GUZEl!";

    for(int i=0; i<strlen(mesaj);i++)
    {
        if(i==0 || mesaj[i-1] ==' ')
            mesaj[i] = toupper(mesaj[i]);
        else
            mesaj[i] = tolower(mesaj[i]);
    }
    cout << mesaj;
    return 0;
}
```

Görev 3: Yazdığınız bir programda kullanıcıya bir karakter katarı içerisinde, kaç tane kelimeden oluştuğunu sayabilecek bir program yazınınız.

Cevap 3:

```
#include <iostream>
#include <cstring>

using namespace std;

int main()
{
    char mesaj[] = "Bugun hava cok guzel!";
    int kelimeSayisi = 0;
    for(int i=0; i<strlen(mesaj);i++)
    {
        if(mesaj[i] == ' ')
            kelimeSayisi++;
    }
    cout << "Bu cumlede " << kelimeSayisi+1 << " kelime bulunmaktadir.";
    return 0;
}
```

B3. Neden Dosyalama İşlemleri Yaparız?

Süre: 30 dk.

Kazanımlar: K3. Dosyalama işleminin gerekliliğini açıklar.

K4. Dosya kütüphanesi kullanarak program geliştirir.

Materyaller: O11.B3.1 Dosyalama İşlemleri Görev Yaprağı

O11.B3.2 C++ Programlama Dilinde Dosyalama İşlemleri Afişi 1

O11.B3.3 C++ Programlama Dilinde Dosyalama İşlemleri Afişi 2

O11.B3.4 C++ Programlama Dilinde Dosyalama İşlemleri Afişi 3

O11.B3.5 Dosyalama İşlemleri

Hazırlık: Eğitmen birinci materyal hariç diğer tüm materyalleri ÖYS üzerinden erişime açar ve öğrencilerin göreceği şekilde projeksiyon ile yansıtır. Bu afiş öğrencilere destekleyici bilgi

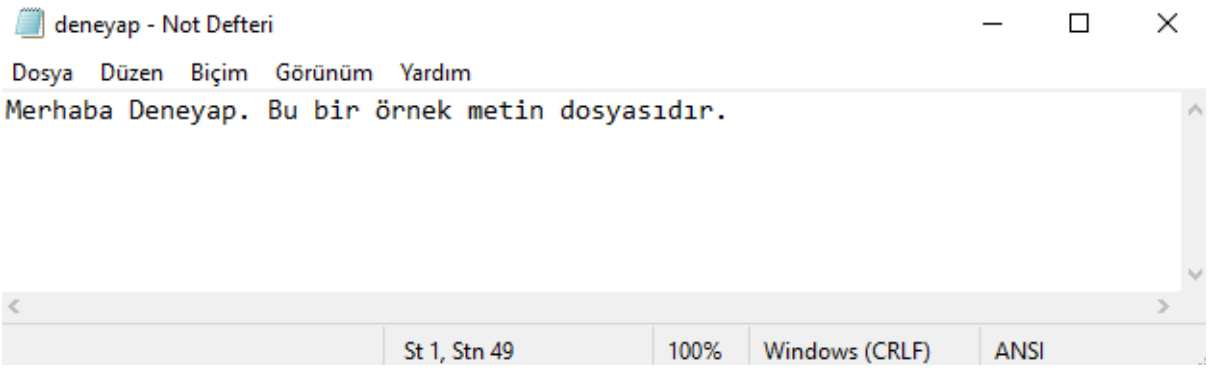
sağlamak için kullanılacaktır. Eğitmen dersin bu bölümü için hazırlanan materyal olan "O11.B3.1 Dosyalama İşlemleri Görev Yaprağı" adlı görev kâğıdını her gruba bir tane verecek şekilde 5 adet hâlinde çıktı alarak derse gelir.

Uygulama: Eğitmen öncelikle her bir öğrenciye "O11.B3.1. Dosyalama İşlemleri Görev Yaprağı" adlı görev kâğıdını dağıtır. Öğrencilerden bu ders için hazırlanan afişteki destekleyici bilgileri kullanarak bu görev kâğıdındaki boş yerleri doldurması istenir. Eğitmenlerin bu etkinlikteki görevi, öğrenci gruplarını sırayla gezerek onların ihtiyaç duydukları anda geri bildirimler, işlemsel bilgileri vererek öğrencileri desteklemesidir.

Eğitmene Öneriler: Afişlerde geçen bilgiler aşağıdaki gibi özetlenebilir. Öğrencilere verilen görevlerin cevaplarına ise bu bölümün sonunda ulaşılabilir.

DOSYALAMA

Programlama dilleri için dosya, verilerin kalıcı olarak saklanması için kullanılır. Dosya, program yardımıyla veya kullanıcılar tarafından oluşturularak depolama biriminde tutulur. Not defteri ile kolayca oluşturabiliriz.



Program verilerinin aksine, dosyadaki veriler bilgisayar kapansa bile silinmez. Bu sebeple ihtiyaç duyulan önemli bilgiler veya kullanıcılardan alınan bilgiler dosyalar yardımıyla tutulur. Genellikle "txt" uzantılı metin dosyaları kullanılır. "txt" uzantılı dosyalar hem kullanıcılar tarafından hemde program tarafından kolayca oluşturulabilir, okunabilir ve üzerine yazılabilir.

Var olan dosyalar üzerinde yapılacak metinsel işlemler okuma veya yazmadır. Dosya işlemleri ise, yeni dosya oluşturma ve mevcut dosyanın silinmesidir. Tüm bu işlemler C++ programlama dili ile hızlıca yapılmaktadır. Dosya işlemleri için C++ içerisinde üç temel sınıf bulunmaktadır.

ifstream	Okuma amaçlı açılacak dosya işlemleri için kullanılır.
ofstream	Yazma amaçlı açılacak dosya işlemleri için kullanılır.
fstream	Hem okuma hem de yazma amaçlı dosya işlemleri için kullanılır.

Dosyalar üzerinde yapılacak temel işlemler ve fonksiyonları ise aşağıdaki gibidir.

Dosyayı Aç	open()
Dosyadan veri oku	read()
Dosyaya veri yaz	write()
Dosyayı kapat	close()

Dosyayı açma sırasında eğer dosya mevcut değil ise, varsayılan olarak boş bir dosya oluşturulacaktır. Dosyayı farklı modlarda açabiliriz. Bu modlar;

Açıklama	mod
Normal dosya okuma modudur. Dosya en baştan okunmaya başlanır. Bu mod ifstream için varsayılan moddur.	ios::in
Normal dosya yazma modudur. Dosyaya en baştan yazılmaya başlanır. Bu mod ofstream için varsayılan moddur.	ios::out
Dosya yazma modudur. Dosyaya yazım işleminde, veriler dosyanın son karakterinden sonra eklenir.	ios::app
Dosya açıldığında içindeki tüm veriler silinir.	ios::trunc
Sadece dosya mevcut ise dosya açılacaktır. Eğer yoksa dosya oluşturulmayacaktır.	ios::nocreate

Öğrencilere verilen görevlerin cevapları aşağıdaki gibidir. Öğrenci gruplarının doldurdukları görev kâğıtlarının aşağıdaki gibi olması beklenir.

Dosya Açma Kipi	Dosya açılma kontrolü kodu	Dosya mevcutsa ne olur?	Dosya yoksa ne olur?
ifstream sınıfının varsayılan modu okuma için olan **ios::in** modudur. ofstream sınıfının varsayılan modu yazma için olan **ios::out** modudur. fstream ise varsayılan modu **ios::in ve ios::out** modlarıdır.	Bir dosyanın açılıp/açılmadığı "is_open()" fonksiyonu ile kontrol edilir. Eğer hatalı bir durum olduysa "0" değeri döndürecektir. Bu şekilde dosyanın düzgün biçimde açıldığı kontrol edilebilir.	Dosya mevcut ise açma moduna göre içerik silinebilir. ios::out modunda dosya içeriği silinecektir. ios::app modunda ise dosya içeriği korunacaktır.	Eğer dosya mevcut değil ise belirtilen isimde yeni dosya oluşturulacaktır. ios::nocreate modunda ise yeni dosya oluşturulmayacak sadece dosya mevcut ise açılacaktır.

Örnek Kod: fstream dosya; dosya.open(“deneyap.txt”, ios::in | ios::out);	Örnek Kod: if(!dosya.is_open()) cout << “Dosya acilamadi!”;		Örnek Kod: dosya.open(“deneyap.txt”, ios::nocreate); if(!dosya.is_open()) cout << “Dosya mevcut değil!”;

B4. C++ Dilinde Dosyalama İşlemleri Yapıyorum

Süre: 20 dk.

Kazanımlar: K5. Dosya okuma işlemlerini içeren program tasarlar.

K6. Dosya yazma işlemlerini içeren program tasarlar.

Materyaller: O11.B3.5 Dosyalama İşlemleri

O11. B4.1 Kodlar Arasındaki Farkı Bulma

Hazırlık: Eğitmen dersin bu bölümü için hazırlanan “O11. B3.1 Dosyalama İşlemleri” adlı afişi ÖYS üzerinden erişime açar ve öğrencilerin göreceği şekilde projeksiyon ile yansıtır. Bu afiş öğrencilere destekleyici bilgi sağlamak için kullanılacaktır. Eğitmen dersin bu bölümü için hazırlanan diğer materyal olan “O11. B4.1 Kodlar Arasındaki Farkı Bulma” adlı görev kâğıdını öğrencilerle 5 grubun her birine birer adet olacak şekilde getirir.

Uygulama: Eğitmenler sınıfı dörder kişilik beş gruba ayrılır. Her grubun görebileceği şekilde bu ders için hazırlanan afişi sınıfın belirli bölgelerine asar ve afişi projeksiyon üzerinden yansıtır. Her gruba “O11. B4.1 Kodlar Arasındaki Farkı Bulma” adlı materyali dağıtmadan önce aşağıdaki gibi derse ister bilgisayar ister tahta üzerinden destekleyici bilgiyi sunarak başlar.

Destekleyici Bilgi:

C++ programlama dili üzerinde dosya işlemleri <fstream> kütüphanesi aracılığıyla yapılmaktadır. ifstream ve ofstream sınıfları bu amaçlar için kullanılmaktadır. Dosya okuma işlemleri için ifstream, dosyaya yazma işlemleri için ofstream kullanılır.

Boş dosya oluşturma: İlk olarak kütüphaneyi projemize dahil ettik. Ardından dosya isimli bir nesne oluşturduk. Parantez içerisine de dosyanın ismini verdik. Eğer dosya yok ise projenin bulunduğu klasörde boş bir metin dosyası oluşturulmuş oldu.

```
#include <iostream>
#include <fstream>
using namespace std;
int main()
{
    ofstream dosya("deneyap.txt");
}
```

İstersek yapıcı fonksiyonu kullanmadan dosya.open() fonksiyonu ile de dosyayı açabiliriz. Varolan dosyanın üstüne bilgi ekleyebiliriz bunun için dosya isminden sonra ikinci parametre olarak mod bilgisini (ios::app) vermemiz gerekmektedir.

Dosya içerisine yazı yazmak için;

```
dosya << "Merhaba Deneyap!";
```

Satırını ekleyelim. Böylece dosyamızın içerisine "Merhaba Deneyap!" yazmış olduk. Son olarak dosyamızı kullanmayı bitirmek için;

```
dosya.close();
```

yazarak dosyamızı kapatıyoruz. Dosyamızı kapatarak geçiçi hafızayı da temizlemiş oluyoruz.

Bu bilgiler ışığında dağıttığım materyal üzerindeki kod farklılıklarını grup arkadaşlarımızla inceleyip, daha sonra hep beraber üzerinde tartışalım.

Eğitmene Öneriler: Eğitmenlere bu derse yönelik öneriler yukarıda (uygulama başlığı altında) verilmiştir.

B5. Dosyalama İşlemleri ile İlgili Verilen Görevleri Kodluyorum

Süre: 30 dk.

Kazanımlar: K5. Dosya okuma işlemlerini içeren program tasarlar.

K6. Dosya yazma işlemlerini içeren program tasarlar.

Materyaller: O11.B3.2 C++ Programlama Dilinde Dosyalama İşlemleri Afişi 1

O11.B3.3 C++ Programlama Dilinde Dosyalama İşlemleri Afişi 2

O11.B3.4 C++ Programlama Dilinde Dosyalama İşlemleri Afişi 3

O11.B3.5 Dosyalama İşlemleri

O11. B5.1 Dosyalama İşlemleri ile İlgili Verilen Görevleri Kodluyorum

O11.B5. 2 Dosyalama İşlemleri ile İlgili Destekleyici Bilgi Sunusu

Hazırlık: Eğitmen dersin bu bölümü için hazırlanan afişleri ÖYS üzerinden gönderir. Bu afiş ve "O11.B5.2 Dosyalama İşlemleri ile İlgili Destekleyici Bilgi Sunusu" öğrencilere destekleyici bilgi sağlamak için kullanılacaktır. Eğitmen dersin bu bölümü için hazırlanan diğer materyal olan "**O11. B5. 1 Dosyalama İşlemleri ile İlgili Verilen Görevleri Kodluyorum"** adlı görev kâğıdını her iki öğrenciye bir çıktı olacak şekilde 10 adet çıktı alır. Her bilgisayarda 2 öğrenci oturması sağlanarak verilen görevleri iki kişilik grupların yapması sağlanır. Code::Blocks uygulaması öğrencilerin kodlama yapabilmesi için hazır duruma getirilir.

Uygulama: Eğitmenler C++ programlama dilindeki dosyalama işlemleri için hazırlanan görev kâğıdındaki tüm etkinlikleri her öğrenci grubunun yapmasını ister. Öğrenciler kodlama esnasında verilen görevler için bir önceki derste hazırlanan dosyalama işlemleri ile ilgili afişleri ve "**O11. B5. 2 Dosyalama İşlemleri ile İlgili Destekleyici Bilgi Sunusu**" destekleyici bilgi olarak kullanır. Sunu öğrencilere paylaşılır ve ÖYS üzerinden istedikleri görev sayfasındaki

destekleyici bilgiyi görmesi sağlanır. Öğrencinin ihtiyaç duyduğu anda gerekli işlemsel bilgiyi eğitmenlerden biri sınıfı dolaşarak, biri de kodlamayı öğrencilerin de görebileceği şekilde bilgisayarda kodlayarak sağlayabilir.

Eğitmene Öneriler:

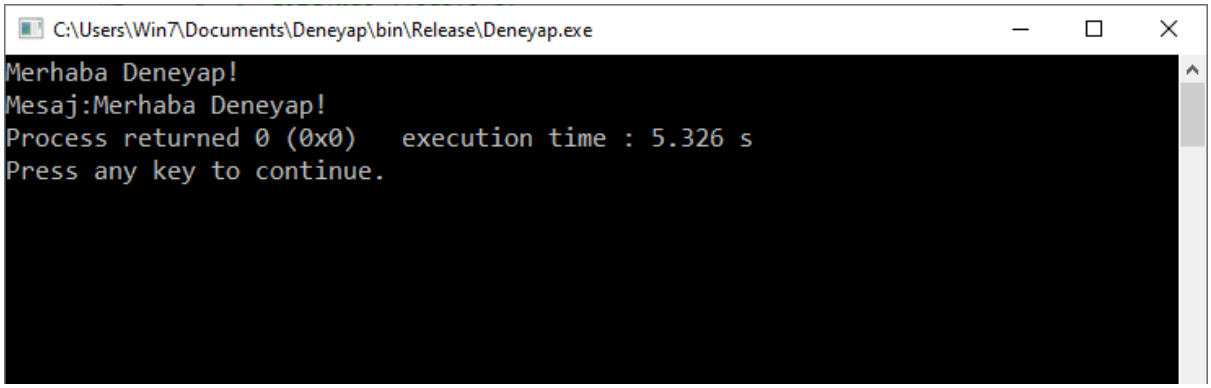
Dosyalama işlemleri ile ilgili verilen görevlerin cevapları aşağıdaki gibidir:

Görev 1: Klavyeden girilen "Merhaba Deneyap!" adlı cümleyi direkt string nesnesine aktarıp sonucu ekrana yazdıran kodu yazalım.

CEVAP 1:

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
    string cumle = "Merhaba Deneyap!";
    cout << "Mesaj:" << cumle;
}
```

Klavyeden okuduğumuz cümleleri doğrudan string nesnesine aktarabiliriz. Bunun için aşağıdaki örneğe bakalım.



Resim 42. *Ekran çıktısı*

```
#include <iostream>
#include <fstream>
#include <string>
using namespace std;
int main()
```

```
{
    string cumle;
    getline(cin,cumle);
    cout <<"Mesaj:" << cumle;
}
```

Cin komutu kullanıldığında hafızada tutulan önceki girişlerin temizlenmediği durumlarda sorun oluşabilmektedir. Bu yüzden *cin.ignore()* fonksiyonu kullanılarak giriş hafızası temizlenebilir.

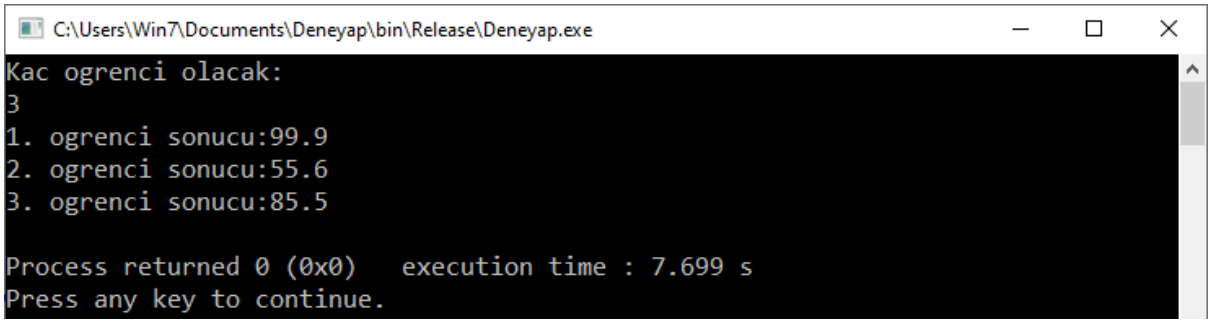
Görev 2: Klavyeden girilen 10 sayıyı dosyaya yazalım.

Cevap 2:

```
#include <iostream>
#include <fstream>
using namespace std;
int main()
{
    ofstream dosya("deneyap.txt");

    for(int i=0;i<10;i++)
    {
        int sayi;
        cin >> sayi;
        dosya << sayi << endl;
    }

    dosya.close();
}
```

Görev 3: Klavyeden girilen öğrenci sayısı kadar sınav notlarını klavyeden okuyup dosyaya yazdıran programı oluşturunuz.

Cevap 3:



Resim 43. *Ekran çıktısı*

```
#include <iostream>
#include <fstream>
using namespace std;
int main()
{
    int ogrenciSayisi;
    float sonuc;

    ofstream dosya("sinav.txt");
    if(!dosya.is_open())
    {
        cout << "Dosya Okunamadi!";
        return 0;
    }
    cout << "Kac ogrenci olacak:" << endl;
    cin >> ogrenciSayisi;
    for(int i=0;i<ogrenciSayisi;i++)
    {

        cout << i+1 <<". ogrenci sonucu:";
        cin >> sonuc;
        dosya << sonuc << endl;
    }
    dosya.close();
}
```

C. Kısmi Öğrenme Görevleri

Süre: 50 dk.

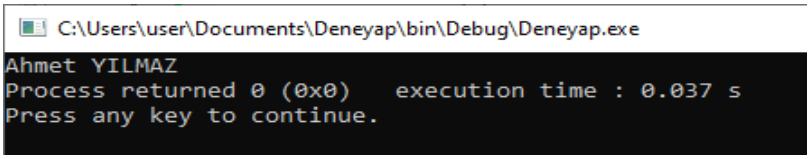
Materyal: O11. C.1. Kısmi Öğrenme Görevleri Afişi

Hazırlık: Kısmi öğrenme görevleri afişi öğrencilerin kolay ulaşabileceği noktalara asılır ya da öğrencilerin Öğrenme Yönetim Sistemi ortamında materyal olarak erişmeleri sağlanır.

Uygulama: Öğrenciler afişteki görevleri istedikleri sırada ve sayıda kendi tercihlerine bağlı olarak verilen süre içinde uygulamaya başlar. Her bir görevi tamamlayan öğrencilere, göreve ilişkin beceri rozeti verilir. Öğrenciler görev seçimlerini ve tamamladıkları görevleri eğitmene bildirmelidir. Eğitmenler ihtiyaç duydukları zaman öğrencilere görevi tamamlama aşamasında anlık geri bildirimlerde bulunur. Öğrenciler, ÖYS üzerinden süreli ödev olarak açılan kısmi öğrenme görevlerinden kendi tercihlerine göre istedikleri sayıda görev tamamlayıp eğitmene iletir. Süre bitiminde eğitmen görevleri ve yanıtlarını ÖYS ya da GitHub üzerinden öğrencilere gönderir.

Kısmi öğrenme görevinin ismi beceri rozetini tanımlamaktadır. Beceri rozetine ait görevlerin yanıtları ise aşağıda verilmektedir.

1. **KODLAYICI** Kişinin soyadını büyük harfe çevirelim.



Resim 44. Ekran çıktısı

```
#include <iostream>
#include <cstring>
using namespace std;
int main(){
    char mesaj[] = "Ahmet Yilmaz";
    bool bosluk = false;
    for(int i=0; i<strlen(mesaj);i++)     {
        if(bosluk)
            mesaj[i] = toupper(mesaj[i]);
        if(mesaj[i] == ' ')
            bosluk = true;
    }
    cout << mesaj ;
    return 0;
}
```

2. **KODLAYICI** Klavyeden girilen cümledeki ‘a’ veya ‘A’ karakterlerini sayan programı yazınız.

```
#include <iostream>

#include <cstring>

using namespace std;

int main()

{

char mesaj[100];

cin.getline(mesaj,100);
int sayac = 0;

    for(int i=0;i<strlen(mesaj);i++)

    {

        if(mesaj[i]=='a' || mesaj[i]=='A')

            sayac++;

    }

    cout << "Bu cumlede "<< sayac <<" adet a vardir.";

    return 0;

}
```

3. **KODLAYICI** 1-100 arası sayıların toplamını dosyaya yazınız.

```
#include <iostream>

#include <fstream>

using namespace std;

int main()

{

    int toplam = 0;

    for(int i=1;i<100;i++)

        toplam += i;

    ofstream dosya("sonuc.txt");

    if(dosya.is_open())

    {

        dosya << toplam;

    }

    else

    {

        cout << "Dosya Okunamadi!";

    }

}
```

Hafta 11. Ders Materyalleri

O11.B1.1 C++ Programlama Dilinde Kütüphaneler

Dikkat!

C++ programlama dili içerisinde geliştiricilerin kullanımına sunulmuş birçok fonksiyonların kütüphaneler içerisinde yer aldığını biliyor muydunuz?

Örneğin; Bu haftaya kadar kullandığımız ***cout*** ve ***cin*** fonksiyonları ***iostream*** isimli kütüphane içerisinde bulunmaktadır.

Dikkat!

Şimdi gelin bu derste C++ programlama dilinde kullanabileceğimiz diğer kütüphaneleri ve bu kütüphanelerdeki bazı fonksiyonları öğrenelim.

O11.B1.2 Karakter Kütüphanesi

Tek bir karakter için hazırlanmış fonksiyonlar cctype kütüphanesi içerisinde bulunur. Standart kütüphane olarak projemizde bulunmaktadır.

“

#include<cctype>

satırı ile projemize dahil ederiz.

”

Lütfen şimdi aşağıda yer alan noktalı yerleri grup arkadaşlarımızla dolduralım.

Fonksiyon	İşlevi	Örnek Kullanım	Sonuç
isalpha(c)	c karakteri eğer bir harf ise geriye true değilse false döndürür.	isalpha(11.4)	…..
isdigit(c)	c karakteri eğer bir rakam ise geriye true değilse false döndürür.	isdigit(11.4)	…..
isalnum(c)	c karakteri eğer bir rakam veya harf ise geriye true değilse false döndürür.	isalnum(/*)	…..
islower(c)	c karakteri eğer bir küçük harf ise geriye true değilse false döndürür.	islower (‘d’)	…..
isupper(c)	c karakteri eğer bir büyük harf ise geriye true değilse false döndürür.	isupper(‘H’)	…..
tolower(c)	c karakterini küçük harfe çevirir.	tolower(‘E’)	…..
toupper(c)	c karakterini büyük harfe çevirir.	toupper(‘g’)	…..

O11.B1.3 Metin Kütüphanesi

C++ programlama dilinde metinler üzerinde işlem yapan kullanıma hazır fonksiyonları içerisinde barındıran kütüphanenin adı **cstring** kütüphanesidir.

“

#include<cstring>

satırı ile projemize dahil ederiz.

”

Lütfen şimdi aşağıda yer alan noktalı yerleri grup arkadaşlarımızla dolduralım.

Fonksiyon	İşlevi	Örnek Kullanım	Sonuç
strlen (s1)	s1 katarının uzunluğunu döndürür.	s1 = “Merhaba” strlen (s1)	……..
strcpy (s1, s2)	s2 katarını s1 katarına kopyalar.	s1= “Merhaba” s2= “Dünya” strcpy (s1, s2)	……..
strcat (s1, s2)	s2 katarını s1 katarının sonuna ekler.	s1= “Merhaba” s2= “Dünya” strcat (s1, s2)	……..
strcmp (s1, s2)	s1 ve s2 aynı ise 0 değerini döndürür; s1 < s2 ise 0'dan küçük değer döndürür; s1 > s2 ise 0'dan büyük değer döndürür.	s1= “Merhaba” s2= “Dünya” strcmp (s1, s2)	……..

O11.B2.1 C++ Programa Dilinde Bulunan Kütüphaneleri ve Fonksiyonları Kullanma

Görev 1

Klavyeden karakterleri sırasıyla girilen "Arda" ismini "a" değişken adıyla karakter dizisinde, "Duru" ismini ise "b" değişken adıyla katarda tutarak ekrana yazdıran kodu oluşturunuz.

Görev 2

Büyük küçük karışık halde yazılmış bir cümleyi her harfi büyük olacak şekilde yazdıralım. "BuGun Hava Cok GUZEl!" cümlesini "Bugun Hava Cok Guzel!" şekline getirelim.

Görev 3

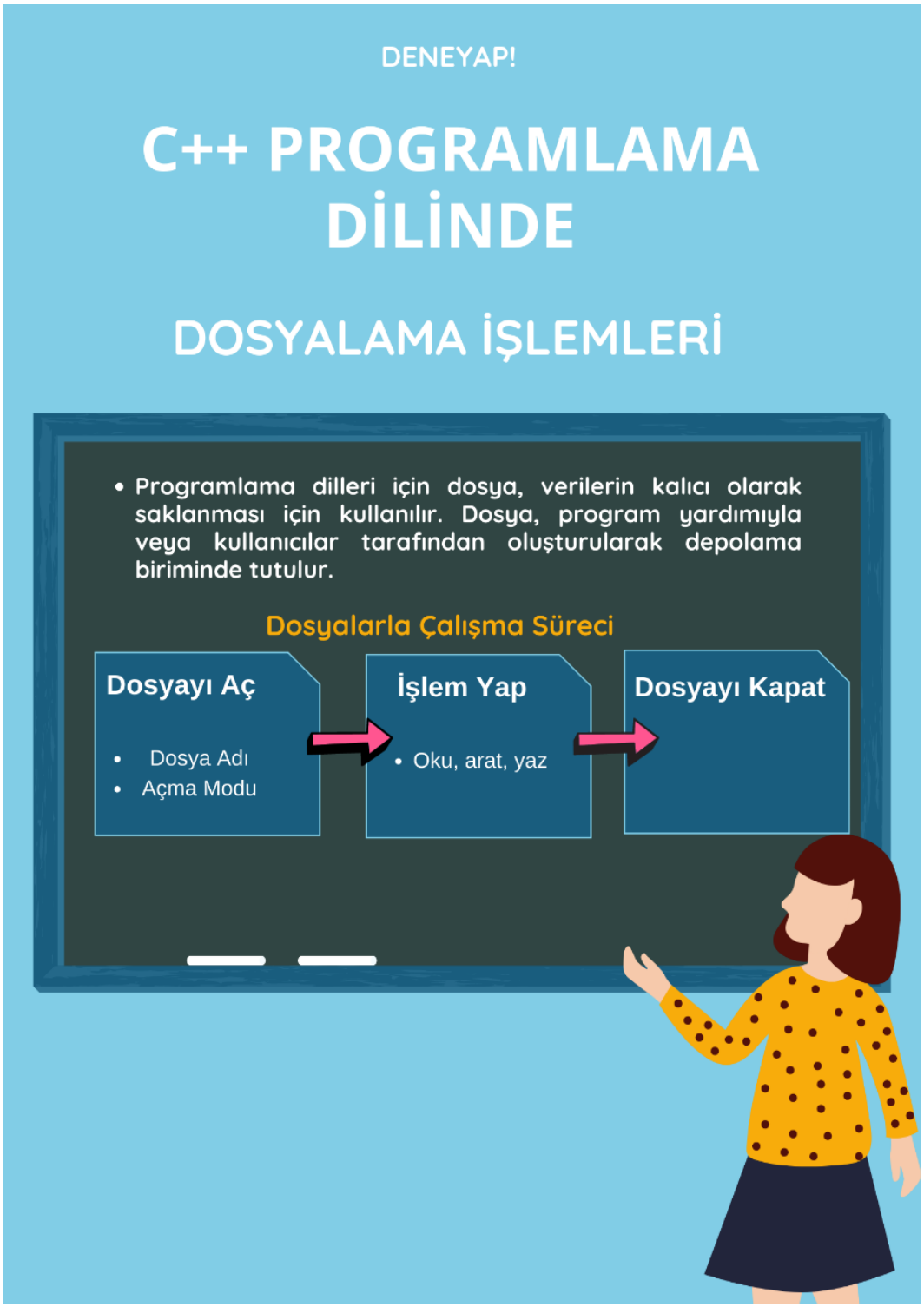
Yazdığınız bir programda kullanıcıya girişte karşılamak isteyeceğiniz mesajın, kaç tane kelimeden oluştuğunu sayabilecek bir program yazınınız.

011.B3.1 Dosyalama İşlemleri Görev Yaprağı

Dosya Açma Kipi	Dosya açılma kontrolü kodu	Dosya mevcutsa ne olur?	Dosya yoksa ne olur?
ifstream sınıfının varsayılan modu okuma için olan …….. modudur. ofstream sınıfının varsayılan modu yazma için olan ……. modudur. fstream ise varsayılan modu ……. ve ……. modlarıdır.	Bir dosyanın açılıp/açılmadığı …….. fonksiyonu ile kontrol edilir. Eğer hatalı bir durum olduysa "0" değeri döndürecektir. Bu şekilde dosyanın düzgün biçimde açıldığı kontrol edilebilir.	Dosya mevcut ise açma moduna göre içerik silinebilir. ……. modunda dosya içeriği silinecektir. ……. modunda ise dosya içeriği korunacaktır.	Eğer dosya mevcut değil ise belirtilen isimde yeni dosya oluşturulacaktır. …….. modunda ise yeni dosya oluşturulmayacak sadece dosya mevcut ise açılacaktır.

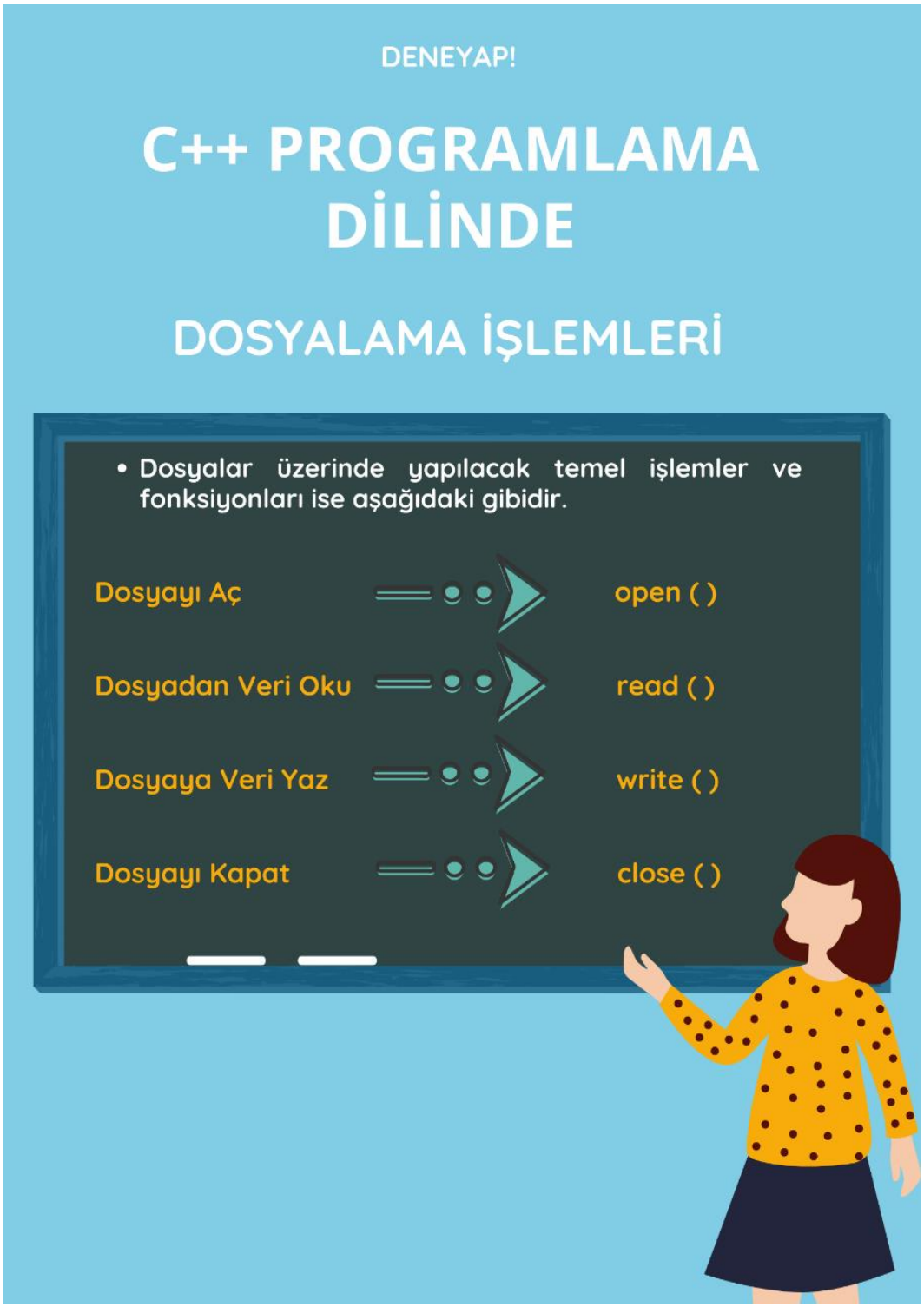
Örnek Kod:
`fstream dosya;`
 	
`dosya.open("deneyap.txt", ios::in | ios::out)`
	Örnek Kod:
	`if(!dosya.is_open())`
	`cout << "Dosya acilamadi!";`
			Örnek Kod:
			`dosya.open("deneyap.txt", ios::nocreate)`
			`if(!dosya.is_open())`
			`cout << "Dosya mevcut değil!";`

O11.B3.2 C++ Programlama Dilinde Dosyalama İşlemleri Yapıyorum Afişi 1



Resim 45. *Dosyalama işlemleri afişi 1*

O11.B3.3 C++ Programlama Dilinde Dosyalama İşlemleri Yapıyorum Afişi 2

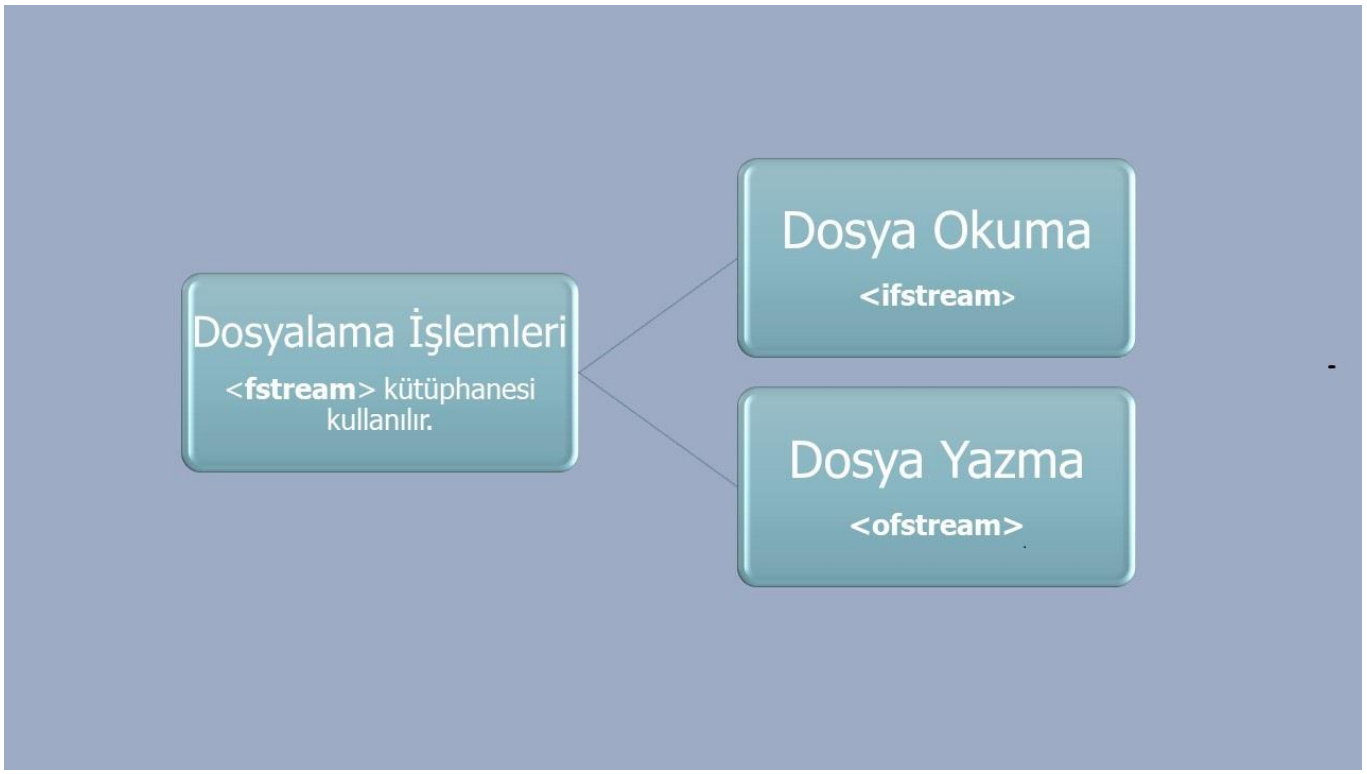


Resim 46. *Dosyalama işlemleri afişi 2*

O11.B3.4 C++ Programlama Dilinde Dosyalama İşlemleri Yapıyorum Afişi 3

Resim 47. *Dosyalama işlemleri afişi 3*

O11.B3.5 Dosyalama İşlemleri



Resim 48. *Dosyalama işlemleri*

O11. B4.1 Kodlar Arasındaki Farkı Bulma

İşlem

```
#include <iostream>
#include <fstream>

using namespace std;
int main()
{
    ofstream dosya;
    dosya.open("deneyap.txt");
    dosya << "Merhaba Deneyap!" << endl;
    dosya.close();
}
```

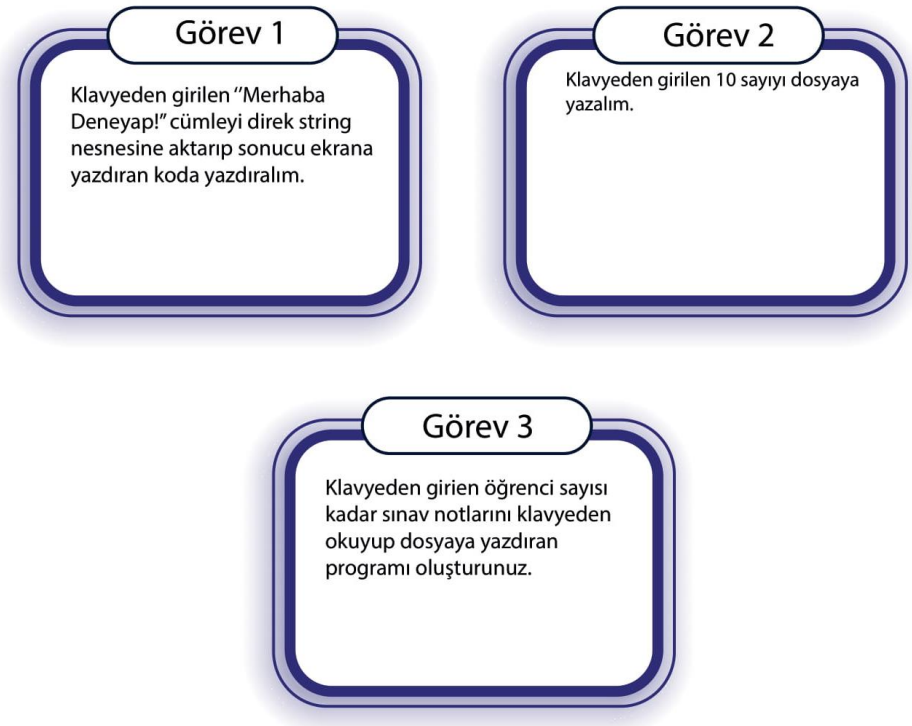
İşlem

```
#include <iostream>
#include <fstream>

using namespace std;
int main()
{
    ofstream dosya;
    dosya.open("deneyap.txt",ios::app);
    dosya<< "Merhaba Deneyap!" << endl;
    dosya.close();
}
```

Dikkat!

Yukarıda verilen kodları dikkatlice gözlemleyerek aralarında ne gibi bir fark olduğunu kodları bilgisayarda yazarak bulmaya çalışalım.

O11. B5.1 Dosyalama İşlemleri ile İlgili Verilen Görevleri Kodluyorum



O11. C. 1. Kısmi Öğrenme Görevleri Afişi

Kısmi öğrenme görevleri afişine ulaşmak için buraya tıklayınız.

Hafta 12. Proje Yarışması

Amaç

Bu haftanın amacı öğrencilerin yazılım teknolojileri dersi kapsamında öğrendiği bilgileri kullanarak yarışma içerisinde verilen soruları cevaplandırmasıdır.

Yarışma Öncesi Öğrenci Gruplarının Belirlenmesi

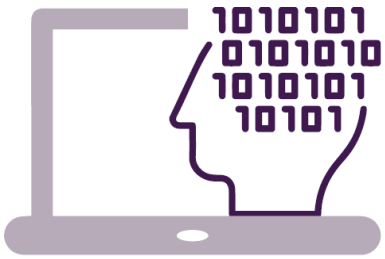
Yarışma için öğrenci gruplarının oluşturulmasında öğrencilerin eğitim boyunca kazandıkları beceri rozetleri dikkate alınmalıdır. Buradaki amacımız 4 farklı beceri rozetine farklı sayıda sahip olan öğrencilerin gruplara dengeli olarak dağıtılmasıdır.

Rozetler:

1. Analizci Rozeti (AR): Verilen problem için üretilen çözümlerin uygunluğunu kontrol eder ve varsa mantık hatalarının giderilmesini sağlar.

2. Tasarlayıcı Rozeti (TR): Probleme uygun çözümlerin uygulamaya geçebilmesi için kodlanmasını sağlar.

3. Kodlayıcı Rozeti (KR): Verilen probleme uygun çözümün nasıl olabileceği ile ilgili ön hazırlıkları yaparak gerekli algoritma ve akış diyagramlarının hazırlanmasını sağlar.



4. Denetleyici Rozeti (DR): Verilen problem için üretilen kodlamaların uygunluğunu kontrol eder ve varsa derleyici hatalarının giderilmesini sağlar.

Grupların belirlenmesi için aşağıdaki 5 adım izlenmelidir.

1. Listedeki tüm öğrenciler Kodlayıcı Rozeti (KR) sayılarına göre en büyükten en küçüğe doğru sıralanır ve ilk beş öğrencinin ismi KR1, KR2, KR3, KR4, KR5 olarak etiketlenir.

2. Listedeki kalan öğrenciler Tasarlayıcı Rozeti (TR) sayılarına göre en büyükten en küçüğe doğru sıralanır ve ilk beş öğrencinin ismi TR1, TR2, TR3, TR4, TR5 olarak etiketlenir.

3. Listedeki kalan öğrenciler Analizci Rozeti (AR) sayılarına göre en büyükten en küçüğe doğru sıralanır ve ilk beş öğrencinin ismi AR1, AR2, AR3, AR4, AR5 olarak etiketlenir.

4. Listedeki kalan öğrenciler Denetleyici Rozeti (DR) sayılarına göre en büyükten en küçüğe doğru sıralanır ve ilk beş öğrencinin ismi DR1, DR2, DR3, DR4, DR5 olarak etiketlenir.

5. Öğrenciler verilen etiket bilgileri dikkate alınarak aşağıdaki tabloya yerleştirilir.

Tablo 23. *Öğrenci grupları oluşturulması*

Grup 1	Grup 2	Grup 3	Grup 4	Grup 5
KR1	KR2	KR3	KR4	KR5
TR5	TR4	TR3	TR2	TR1

Grup 1	Grup 2	Grup 3	Grup 4	Grup 5
AR2	AR1	AR3	AR5	AR4
DR4	DR5	DR3	DR1	DR2

Yarışmadaki Görevlerin Değerlendirilmesi

Jüri değerlendirme formu için aşağıdaki linke tıklayınız:

https://forms.gle/DwNFffTRHBxgNdg47

- Her bir problem için verilmesi gereken süre 45 dakikadır. 4 problem için toplam süre ise 180 dakika olmaktadır.
- Problem çözümleri ekip tarafından daha kısa sürede teslim edilebilir.
- Problem çözümleri için kullanılan süre kayıt altına alınmalıdır.
- Problem çözümleri teslim edildikten sonra çözüm üzerinde bir değişiklik yapılamaz.
- Ekipler yeni problemi çözerken jüri teslim edilen problem çözümünü cevap kâğıdına uygun olarak değerlendirerek 0-100 arası puanı oluşturur.
- Dört problemin çözümünü tamamlayan gruplar istediği taktirde zamanları kalmışsa (henüz 4 soruda 180 dakika kullanılmamışsa) bu kalan zamanda Bonus problemi de çözebilirler.
- Bonus problemi için verilmesi gereken süre 30 dakikadır.
- Her bir ipucu kullanımında ekip 10 puan kaybedecektir.

Puan hesaplaması aşağıdaki gibi yapılacaktır.

$$GP = P1 + P2 + P3 + P4 + PB – (İP * 10) + TS – (S1 + S2 + S3 + S4 + SB)$$

GP: Genel Puan

P1 = Problem 1 Çözümünden Alınan Puan (Min:0 Max:100)

P2 = Problem 2 Çözümünden Alınan Puan (Min:0 Max:100)

P3 = Problem 3 Çözümünden Alınan Puan (Min:0 Max:100)

P4 = Problem 4 Çözümünden Alınan Puan (Min:0 Max:100)

PB = Bonus Probleminin Çözümünden Alınan Puan (Min:0 Max:50)

İP = Tüm Sorularda Kullanılan Toplam İpucu Sayısı (Min:0 Max:80)

TS = 180 (Toplam Süre)

S1 = Problem 1 Çözümünde Harcanan Süre (Min:0 Max:45)

S2 = Problem 2 Çözümünde Harcanan Süre (Min:0 Max:45)

S3 = Problem 3 Çözümünde Harcanan Süre (Min:0 Max:45)

S4 = Problem 4 Çözümünde Harcanan Süre (Min:0 Max:45)

SB = Bonus Probleminin Çözümünde Harcanan Süre (Min:0 Max:30)

HAZİNE AVI

Maceracı arkadaşlar hazine arayışındadır. Bir gün çok eski bir harita bulurlar. Haritada etrafı sularla kaplı dört küçük hazine adasından bahsediliyor. Ancak bu adalarda hazine avlamak o kadar da kolay değil. Maceracıların hazineye ulaşabilmeleri için adalara giriş şifrelerini bulmaları gerekiyor. Adalar arasında ilerledikçe çeşitli problemlerle karşılaşırlar. Maceracılar bu problemleri çözdükçe adalara giriş şifresini de çözmüş olacaklar. Maceracıların her soruda gerekli kod görevlerini yaparak bir sonraki adıma geçmeleri gerekmektedir. Bununla birlikte herhangi bir sorunun koduna ilişkin çözüm tamamlanmasa da adalarda ilerleme için gerekli işlemler elle tamamlanabilir. Fakat bu durumda ilgili sorudan herhangi bir puan alınmayacaktır. Elle çözüm yapılabilmesinin amacı maceracıların hazine avını yarıda bırakmasını engellemektir.


HAZİNE AVI
Maceracıların her soruda gerekli kod görevlerini yaparak bir sonraki adıma geçmeleri gerekmektedir. Bununla birlikte herhangi bir sorunun koduna ilişkin çözüm tamamlanmasa da adalarda ilerleme için gerekli işlemler elle tamamlanabilir. Fakat bu durumda ilgili sorudan herhangi bir puan alınmayacaktır. Elle çözüm yapılabilmesinin amacı maceracıların hazine avını yarıda bırakmasını engellemektir.

TEKNE TAMİRİ

Maceracıların haritadaki ilk durağı Kaşık Adası. Kaşık Adası'na ulaşmaları için bir tekne bulmaları gerekiyor. Ellerindeki tek teknenin ise tadilata ihtiyacı var. Tadilat için 8 tane kısa ince tahta, 7 tane uzun ince tahta ve 12 tane uzun kalın tahta kullanacaklar. Bunun için ödemeleri gereken ücreti tahta fiyat listesinden hesaplamaları lazım. İşte maceralarımız ilk problemleriyle karşılaşmıştır. Bunun için bir akış diyagramı oluşturmaya karar verirler. Bu akış diyagramında tahtaların adedi, boyu ve kalınlığına ilişkin bilgilerin kullanıcı tarafından girilmesi isteniyor.

Maceracılar akış diyagramının sonunda ihtiyaç listesine ödenecek toplam değeri hesaplayınca, Kaşık Adası'na giriş şifresini çözmüş olacaklar. Bu görevde maceracılara yardım ediniz.

Tablo 24. *Tekne tamiri tablosu*

Tahta Boyu	Tahta Kalınlığı	Tahta Fiyatı	İhtiyaç Listesi
1- Kısa	1- Kalın	35 TL	8 Kısa ve İnce Tahta 7 Uzun ve İnce Tahta 12 Uzun ve Kalın Tahta
	2- İnce	23 TL	
3- Uzun	1- Kalın	78 TL	
	2- İnce	64 TL	

1. İpucu: Problemin çözümü için döngü yapısı kullanabilirsiniz.

2. İpucu: İhtiyaç listesi farklı sayıda olabilir. Bu yüzden tüm durumları kontrol etmelisiniz.

TEKNE TAMİRİ

Maceracıların haritadaki ilk durağı Kaşık adası. Kaşık adasına ulaşmaları için bir tekne bulmaları gerekiyor. Ellerindeki tek teknenin ise tadilata ihtiyacı var.

Tadilat için 8 tane kısa ince tahta, 7 tane uzun ince ve 12 tane uzun kalın tahta kullanacaklar. Bunun için ödemeleri gereken ücreti tahta fiyat listesinden hesaplamaları lazım. İşte maceralarımız ilk problemleriyle karşılaşmıştır. Bunun için bir akış diyagramı oluşturmaya karar verirler. Bu akış diyagramında tahtanın adedi, boyu ve kalınlığına ilişkin bilgilerin kullanıcı tarafından girilmesi isteniyor.

Maceracılarımız akış diyagramının sonunda ihtiyaç listesine ödenecek toplam değeri hesaplayınca, Kaşık adasına giriş şifresini çözmüş olacaklar.

BU GÖREVDE MACERACILARA YARDIM EDİNİZ.

TAHTA FİYAT LİSTESİ

BOY	KALINLIK	FİYATI
KISA	KALIN	35 YTL
KISA	İNCE	23 YTL
UZUN	KALIN	78 YTL
UZUN	İNCE	64 YTL

XOX OYNA

Kaşık Adası'na hoş geldiniz.

İkinci sırada Balıkçı Adası vardır. Balıkçı Adası'nın şifresini kazanmak için maceracılarımız ada yerlilerine XOX oyunu yazmak için yardım etmelidir.

XOX oyunu iki kişi ile oynanan 3x3 bir tahta oyunudur. Bir kişi tahtaya 'X' koyduğunda rakibi 'O' koymaktadır. Oyunun amacı 3 adet 'X' ve 'O' harflerini yan yana, üst üste veya çapraz olarak yerleştirmektir. Maceracılar XOX tahtasının durumuna göre bir oyunun bitip bitmediğini tüm olası durumlar için kontrol eden programı yazarlar. Verilen tahta durumuna göre eğer oyun bitti ise ekrana XOX yazdırılması, bitmediyse DEVAM yazdırılması gerekmektedir.

Maceracılar yazdıkları bu programda aşağıdaki XOX tahta durumunu test eder. Program çıktısı Balıkçı adasının şifresidir.

X		
X		
O		

1. İpucu: Tahtayı char tahta[3][3] şeklinde tanımlayabilirsiniz.

tahta[0][0] = 'X', tahta[1][0] = 'X', tahta[2][0] = 'O'

2. İpucu: Satır ve sütun kontrolü yaparken 1 döngü ve 1 if koşulu yeterli olacaktır.


XOX OYNA
Kaşık adasına hoşgeldiniz.
İkinci sırada Balıkçı adası vardır. Balıkçı adasının şifresini kazanmak için maceracılarımız ada yerlilerine XOX oyunu yazmak için yardım etmelidir.
XOX oyunu iki kişi ile oynanan 3x3 bir tahta oyunudur. Bir kişi tahtaya 'X' koyduğunda rakibi 'O' koymaktadır. Oyunun amacı 3 adet 'X' ve 'O' harflerini yan yana, üst üste veya çapraz olarak yerleştirmektir. Maceracılar XOX tahtasının durumuna göre bir oyunun bitip bitmediğini tüm olası durumlar için kontrol eden programı yazarlar. Oyun bitmesi durumunda ekrana XOX bitmediyse, DEVAM yazdırılması gerekmektedir.
Maceracılar yazdıkları bu programda aşağıdaki XOX tahta durumunu test eder. Program çıktısı Balıkçı adasının şifresidir.
BU GÖREVDE MACERACILARA YARDIM EDİNİZ.
XOX TAHTASI
X
O
O

BALIKÇI ADASI

Balıkçı Adası'na hoş geldiniz.

Maceracılarımız bu adada yemek molası verir. Menüde 3 farklı balık türü bulunmaktadır. Bunlar; Levrek, Palamut ve Çupra'dır. Maceracılarımız sınıf içerisinde yazılacak bir hesapla fonksiyonu ile ana fonksiyondan gönderilen adet bilgisine göre ilgili balık için ödenmesi gereken ücreti hesaplamak ister. Programda tüm balıklar için ödenmesi gereken hesap bulunarak oluşturulacak bir "hesap.txt" dosyasına kaydedilmelidir. Oluşturulacak Balık sınıf yapısında balık türü ve fiyat bilgileri sınıf üyelerini oluşturacaktır. Bu bilgiler sınıf içerisinde tanımlanacak ve ana fonksiyondan parametre olarak gönderilecektir.

Toplam 3 adet levrek, 6 adet palamut ve 5 adet çupra yenildiğine göre ödenmesi gereken toplam hesap ne kadar olacaktır? Maceracıların ödeyecekleri hesap miktarı, son durak olan hazine adasının şifresidir.

Tablo 25. *Balık bilgisi tablosu*

Balık Türü	Fiyat
Levrek	15 TL
Palamut	9 TL
Çupra	16 TL

1. İpucu: Balık sınıfı 2 adet üye, 1 adet yapıcı fonksiyon ve 1 adet hesapla fonksiyonu içermelidir.

2. İpucu: Ana fonksiyonda nesne tanımlaması yapılırken tanımlama sırasında balık türü ve fiyat bilgisi de parametre olarak verilmelidir.


BALIK YEME
Balıkçı adasına hoşgeldiniz.
Maceracılarımız bu adada yemek molası verir. Menüde 3 farklı balık türü bulunmaktadır. Bunlar; Levrek, Palamut ve Çupra'dır. Maceracılarımız sınıf içerisinde yazılacak bir hesapla fonksiyonu ile ana fonksiyondan gönderilen adet bilgisine göre ilgili balık için ödenmesi gereken ücreti hesaplamak ister. Programda tüm balıklar için ödenmesi gereken hesap bulunarak oluşturulacak bir "hesap.txt" dosyasına kaydedilmelidir. Oluşturulacak Balık sınıf yapısında balık türü ve fiyat bilgileri sınıf üyelerini oluşturacaktır. Bu bilgiler sınıf içerisinde tanımlanacak ve ana fonksiyondan parametre olarak gönderilecektir.
Toplam 3 adet levrek, 6 adet palamut ve 5 adet çupra yenildiğine göre toplam ödenmesi gereken hesap ne kadar olacaktır? Maceracıların ödeyecekleri hesap miktarı, son durak olan hazine adasının şifresidir.
BU GÖREVDE MACERACILARA YARDIM EDİNİZ.
FİYAT LİSTESİ
BALIK FİYAT
LEVREK 15 YTL
PALAMUT 9 YTL
ÇUPRA 16 YTL

HAZİNE ADASI

Hazine Adası'na hoş geldiniz.

Maceracılar, hazine sandığının anahtarını açmak için 20 şans topu numarasından oluşan bir dizi ile karşılaşır. Bu dizideki numaralar rastgele dağılmış rakamlardan oluşmaktadır.

Maceracılar, şifreyi çözmek için bu rakamların kullanım miktarını (sayısını) en azdan çoğa doğru sıralayan kodu yazar. Çünkü kodun çıktısı hazine sandığının şifresidir.

3	1	4	6	3	9	5	0	6	9	2	8	2	9	5	8	6	8	9	2

1. İpucu: İç içe döngü yapısı kullanılabilir.

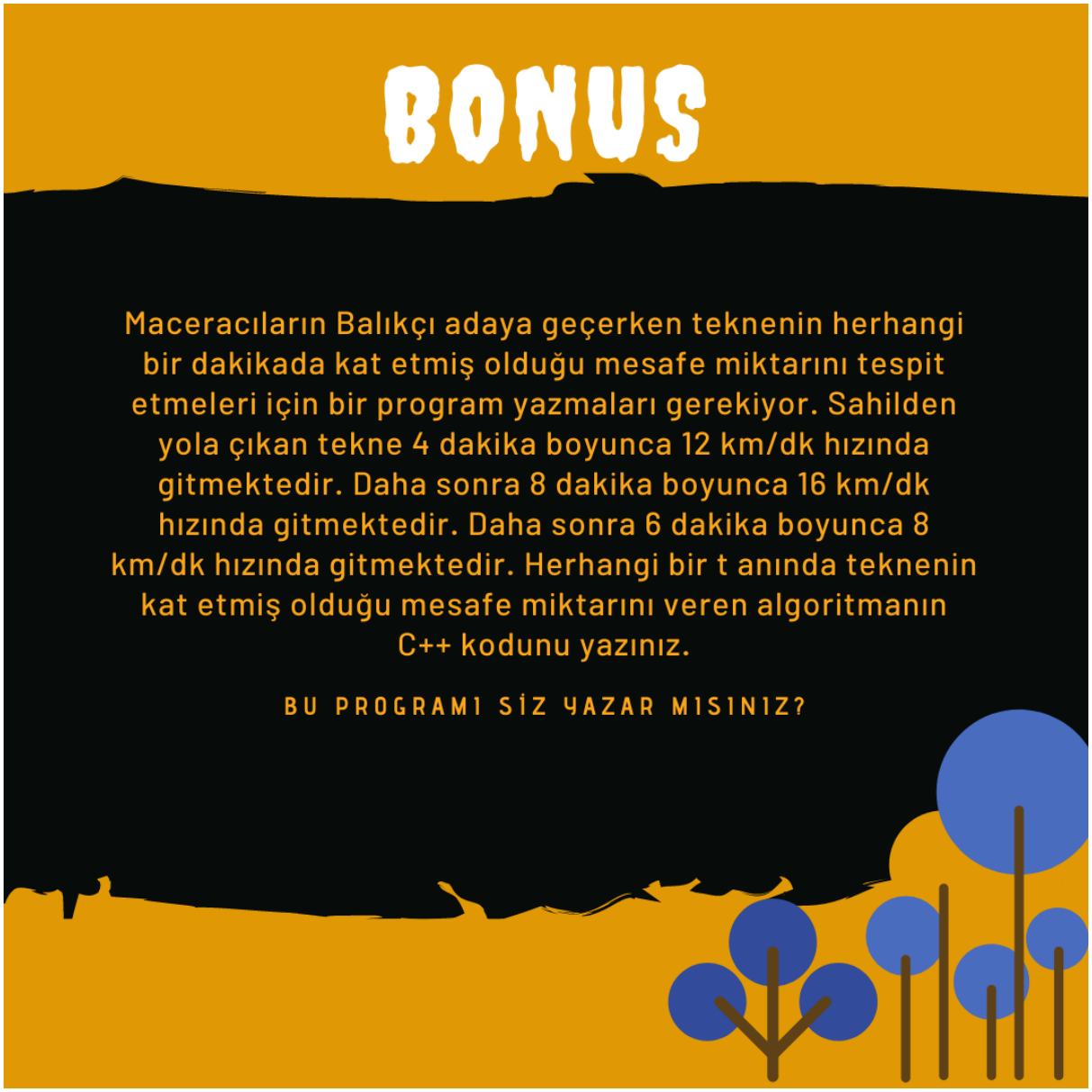
2. İpucu: Farklı rakam sayısı boyutunda bir dizi oluşturup tekrar sayılarını bu dizide tutabilirsiniz.

BONUS PROBLEMİ

Tekneyle birinci adadan ikinci adaya geçiş sırasında teknenin herhangi bir dakikada kat etmiş olduğu mesafe miktarının tespit edilmesi için bir program hazırlanması gerekmektedir. Sahilden yola çıkan tekne 4 dakika boyunca 12 km/dk hızında gitmektedir. Daha sonra 8 dakika boyunca 16 km/dk hızında gitmektedir. Daha sonra 6 dakika boyunca 8 km/dk hızında gitmektedir. Herhangi bir t anında teknenin kat etmiş olduğu mesafe miktarını veren algoritmanın C++ kodunu yazınız.

1. İpucu: Yol formülü X = V*T olarak bilinmektedir. (X: yol, V: hız, T: zaman)

2. İpucu: 3 farklı if kontrol yapısı kullanılarak çözülebilir.



GÖREVLERİN ÇÖZÜMLERİ

ÇÖZÜM 1:

Puanlama: Genel Yapı 20 Puan

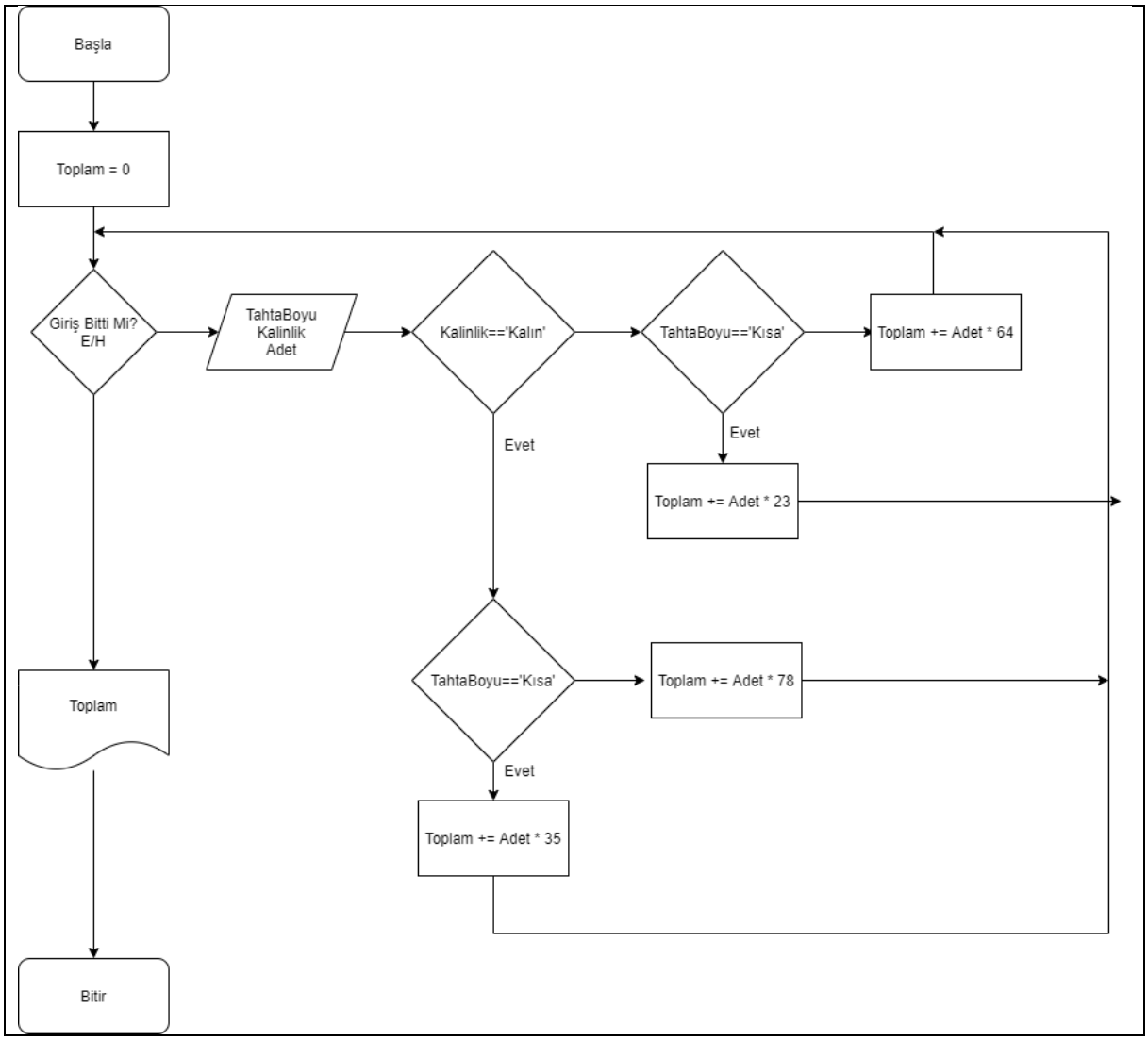
Değişken Tanımlama 20 Puan

Koşulların Tanımlanması 40 Puan

Toplam Hesaplama 20 Puan

Çıktı: 1568

Akış Diyagramı:



ÇÖZÜM 2:

Puanlama: Değişken Tanımlama Bölümü: 10 Puan

Satır kontrolü: 20 Puan

Sütun kontrolü: 20 Puan

Sağ çapraz kontrolü: 20 Puan

Sol çapraz kontrolü: 20 Puan

Ekrana sonuç yazdırma: 10 Puan

Çıktı: DEVAM

Kod:

```
// YARIŞMA PROBLEMI 2:
#include <iostream>
#include <stdlib.h>
using namespace std;
int main()
{
    char tahta[3][3];
    tahta[0][0] = 'X';
    tahta[1][0] = 'X';
    tahta[2][0] = '0';
    bool bitti = false;
    //Satir KontrolÃ¼
    for(int i=0;i<3;i++)
    {
        if(tahta[i][0] == tahta[i][1] && tahta[i][1] == tahta[i][2] )
        {
            bitti = true;
        }
    }
    //SÃ¼tun KontrolÃ¼
    for(int i=0;i<3;i++)
    {
        if(tahta[0][i] == tahta[1][i] && tahta[1][i] == tahta[2][i] )
        {
            bitti = true;
        }
    }
    //SaÃ° Capraz KontrolÃ¼
    if(tahta[0][0] == tahta[1][1] && tahta[1][1] == tahta[2][2] )
    {
        bitti = true;
    }
    //Sol Çapraz KontrolÃ¼
    if(tahta[0][2] == tahta[1][1] && tahta[1][1] == tahta[2][0] )
    {
        bitti = true;
    }
    if(bitti == true)
        cout <<"XOX";
    else
        cout <<"DEVAM";
}
```

ÇÖZÜM 3:

Puanlama: Balık Sınıfı Üyeleri Bölümü: 20 Puan

Hesaplama Fonksiyonu: 40 Puan

Ana Fonksiyon Bölümü: 40 Puan

Çıktı: 179

Kod:

```
// YARIŞMA PROBLEMI 3:
#include <iostream>
#include <string.h>
#include <fstream>
using namespace std;
class Balik{
public:
    char tur[20];
    float fiyat;
    Balik(char balik_tur[], float fiy){
        strcpy(tur, balik_tur);
        fiyat = fiy;
    }
    int hesapla(int adet){
        return adet * fiyat;
    }
};
int main() {
    ofstream dosya;
    dosya.open("hesap.txt");
    Balik levrek("Levrek", 15);
    Balik palamut("Palamut", 9);
    Balik cupra("Cupra", 16);
    int h = levrek.hesapla(3) + palamut.hesapla(6) + cupra.hesapla(5);
    dosya << h << endl;
    dosya.close();
    cout << "Toplam hesap = " << h;
    return 0;
}
```

ÇÖZÜM 4:

Puanlama: Değişken Tanımlama Bölümü: 20 Puan

Tekrar Sayılarını Bulma: 40 Puan

Azdan Çoğa Doğru Rakamları Yazdırma: 40 Puan

Çıktı: 7014352689

Kod:

```
// YARIŞMA PROBLEMI 4:
#include <iostream>
using namespace std;

int main()
{
    // Tanimlama Bolumu
    int dizi[20] ={3,1,4,6,3,9,5,0,6,9,2,8,2,9,5,8,6,8,9,2};
    int m=20;
    int tekrar[10]={0};

    // Tekrar Sayılarını Bulma
    for(int i=0;i<m;i++){
        for(int j=0; j<10; j++){
            if(dizi[i] == j){
                tekrar[j]++;
            }
        }
    }
    /*
    //Tekrar Sayılarını Bulma Hizli Yol
    for(int i=0; i<m; i++){
        tekrar[dizi[i]]++;
    }
    */

    //Azdan Çoğa Doğru Rakamlari Yazdirma
    for(int i=0; i<10; i++){
```

```
        for(int j=0; j<10; j++){
                if(tekrar[j]==i)
                    cout << j<<;
        }


    }
    return 0;
}
```

ÇÖZÜM BONUS:

Puanlama: Değişken Tanımlama ve Değer Okuma Bölümü: 25 Puan

Yol Kontrolü Bölümü 1: 25 Puan

Yol Kontrolü Bölümü 2: 25 Puan

Yol Kontrolü Bölümü 3: 25 Puan

Kod:

```
// YARIŞMA PROBLEMI BONUS:
#include <iostream>
using namespace std;
int main() {
    float zaman;
    float yol;
    cout << "Zaman değerini giriniz (0-16 arasında olmali): " << endl;
    cin >> zaman;

    if(zaman <= 4.0){
        ivme = (12-0) / (4-0);
        hiz = ivme * zaman;
        cout << "Teknenin hizi: " << hiz << endl;
    }
    else if(zaman<=12.0){
        hiz = 12.0;
        cout << "Teknenin hizi: " << hiz << endl;
    }else if (zaman<=16.0){
        ivme = (0-12) / (4-0);
        hiz = 12 + ivme * (zaman-12);
        cout << "Teknenin hizi: " << hiz << endl;
    }else{
        cout << "Hatali giris!!" << endl;
    }
  return 0;
}
```