ÖZEL EĞİTİM VE REHBERLİK HİZMETLERİ
GENEL MÜDÜRLÜĞÜ

BİLİM VE SANAT MERKEZLERİ

YAZ OKULU DESTEKLEME VE YETİŞTİRME KURSU PROGRAMI

YAZILIM ATÖLYESİ

ÖZEL EĞİTİM VE REHBERLİK HİZMETLERİ
GENEL MÜDÜRLÜĞÜ

Genel Yayın Yönetmeni	Cemal ÖZDEMİR
Yayın Koordinatörü	M. Ramazan BARIN
Koordinasyon	Dr. Serdar ÖZMEN Jale GÜNEŞ Seydihan YİĞİT
Yazarlar	Bekir ÇELEN Betül CİHANGİR Esra KIDIMAN DEMİRHAN Hidayet KILCAN Mehmet ÖZASLAN Mehmet İlkay KAYA Muhammet Murat YAMAN Mustafa Çağlar YORULMAZ Savaş ÖZBEY Serkan ÇAM Şadiye AYSİN TEKCAN
Program Geliştirme Uzmanı	Dr. Halil BOLAT
Ölçme ve Değerlendirme Uzmanı	Dr. Öğr. Üyesi Görkem CEYHAN
Dizgi ve Tasarım	Sude Ajans Reklam Org. Tan. Ltd. Şti. www.sudeajans.com.tr 0312 467 09 96
Genel Yayın Dizin Yayın No: Yardımcı ve Kaynak Kitaplar Dizi Yayın No:	9010 2768
ISBN	978-975-11-7164-1

ÖN SÖZ

Bilim ve teknolojinin hızla değiştiği günümüzde öğrencilerimize 21. yüzyıl becerilerini kazandırmak ve onları geleceğin dünyasında etkin ve saygın bireyler olarak yetiştirmek Bakanlığımızın öncelikli hedefleri arasında yer almaktadır. Bu süreçte öğrencilerimizin bireysel yeteneklerinin farkında olmaları ve kapasitelerini geliştirerek en üst düzeyde kullanmalarını sağlamak amacıyla farklı etkinlikler yoluyla beceri kazanmalarına yönelik çalışmalar ön plana çıkmaktadır. Bu bakış açısıyla bilim ve sanat merkezlerinde verilen proje temelli atölye çalışmalarının öğrencilerin geleceğin dünyasına hazırlanmaları noktasında önemli bir rolü olduğunu ifade edilebiliriz.

Özel yetenekli öğrencilerimizin eğitim aldığı bilim ve sanat merkezleri imkânlarından örgün eğitimde öğrenim gören bütün öğrencilerimizin yararlanması amacıyla 2022 yaz döneminden itibaren BİLSEM yaz okulu destekleme ve yetiştirme kursu kapsamında eğitimler devam etmektedir. Eğitimde fırsat eşitliğini sağlamak amacıyla bilim ve sanat merkezlerinde eğitim alan öğrencilerimiz ile birlikte diğer öğrencilerimizin de bu merkezlerden yararlanabilmelerini sağlanmıştır. Bu süreçte öğrencilerimizin ilgi, istek ve yetenekleri doğrultusunda bilim ve sanat merkezlerindeki atölye programlarından yararlanmaları ve atölye etkinlikleri yoluyla farklı deneyimler kazanmaları hedeflenmiştir.

BİLSEM yaz okulu atölye programları öğrencilerin devam ettiği eğitim kademeleri dikkate alınarak ilkokul, ortaokul ve lise kademeleri için altı haftayı kapsayacak şekilde ayrı ayrı hazırlanmıştır. Öğrencilerimizin atölyelerde aldıkları eğitim ile bilimsel düşünce ve davranışlarla estetik değerleri birleştiren, üretken, sorun çözen, kendini gerçekleştirmiş bireyler olarak yetişmeleri, yetenekleri ve yaratıcılıklarını erken yaşta fark ederek en üst düzeyde kullanmaları ve atölye deneyimleri doğrultusunda farklı beceriler edinmeleri hedeflenmektedir.

BİLSEM yaz okulu atölye programlarının uygulanmasında öğretmenlerimize büyük ölçüde kolaylık sağlamasını ve öğrencilerimiz için faydalı olmasını temenni ediyorum.

Cemal ÖZDEMİR
Genel Müdür

İÇİNDEKİLER

ORTAOKUL

LİSE

TÜRKİYE CUMHURİYETİ MİLLÎ EĞİTİM BAKANLIĞI

GİRİŞ

Günümüzde Bilgi ve iletişim teknolojileri alanında yaşanan gelişimler toplum hayatından iş hayatına önemli değişiklikleri beraberinde getirmiştir. Bu değişimden önemli ölçüde etkilenen alanlardan biri de eğitim olmuştur. Bu bağlamda eğitimde uygulanan yöntem-teknikler, kullanılan araç gereçler çeşitlenmiş teknolojinin eğitime entegrasyonu, teknolojik araçların kullanımı ve öğretimi gibi hususlar ön plana çıkmıştır. Eğitimde teknolojik araçların kullanımı ve öğretimi bağlamında önemini gittikçe arttıran konulardan biri de bilgisayar bilimlerinin çalışma alanı olan yazılım geliştirme teknolojileri olmuştur.

Yazılım geliştirme teknolojileri; Bilişim Teknolojileri alanında geliştirilen işletim sistemleri, yazılımlar/programlar, robotik sistemler, yapay zekâ uygulamaları, masaüstü ve mobil uygulamalar, siber güvenlik, web tasarım, dijital oyunlar gibi birçok alanda kritik öneme sahiptir. Yazılım geliştirme teknolojilerinin Bilişim Teknolojileri alanında elde ettiği bu önem temel bilgisayar okuryazarlığı yeterliliğini yazılım okuryazarlığına dönüştürmeyi başarmıştır.

Dolayısı ile bu durum toplumda bu alana duyulan ilgiyi arttırmış, bireylerde öğrenme ihtiyacı oluşturmuştur. Bu ihtiyaçtan hareketle yazılım geliştirme eğitimleri gerek örgün eğitim kapsamında gerekse örgün eğitime ilave olarak okul dışı ortamlarda da yapılmaya başlanmıştır.

Yazılım geliştirme süreci, günlük yaşam probleminin tespit edilmesi ve problem durumunun iyi analiz edilmesi, problem durumunun algoritma ve akış diyagramları ile modellenmesi, amaca en uygun yazılım geliştirme dilleri kullanılarak probleme çözüm üretilebilmesi olarak tanımlanabilir. Bu tanımdan hareketle yazılım geliştirme eğitiminin birçok bilgiyi ihtiva eden kompleks bir yapısı olduğu söylenebilir. Dolayısı ile yazılım geliştirme eğitimi sahip olduğu bu yapı itibari ile öğrencilerin eleştirel düşünme, akıl yürütme, problem çözme, algoritmik düşünme, matematiksel modelleme gibi becerilerinin gelişmesine imkân sağlamaktadır.

Yazılım Geliştirme Yaz Okulu Programının kapsamı belirlenirken ilkokul, ortaokul ve lise düzeyinde hazırlanmış farklı programlama dilleri incelenmiş, kapsamın belirlenmesi hususunda alan uzmanları ile istişareler yapılmıştır. Yaz okulu programının genel amacı, hedef kitlesi, süresi de gözetilerek programın hedef kitleye uygun, esnek, erişilebilir ve ulaşılabilir olmasına özen gösterilmiştir. Bu minvalde öğrenme alanı, alt öğrenme alanı ve kazanımlara uygun etkinlikler hazırlanmıştır. Etkinliklerde salt alan bilgisi vermekten ziyade ilgi çekici, günlük yaşam ile ilişkili, alan-beceri gelişimini sağlayacak, merak duygusunu harekete geçiren, özgün ürünler geliştirebilmelerine katkı sağlayıcı uygulamalar yapmalarına önem verilmeye çalışılmıştır.

Yazılım Geliştirme Yaz Okulu programı etkinliklerinin hazırlanmasında kurumların fiziksel altyapısı, öğrencilerin seviyeleri, ilgi ve yetenek alanları da dahil olmak üzere süreci etkileyebilecek birçok unsur dikkate alınmıştır. Uygulayıcılara kolaylık sağlaması açısından etkinlik öncesi yapılması gerekenler listelenmiş, ilkokul, ortaokul ve lise düzeyinde eğitim alan tüm öğrencilerin gelişim düzeylerine uygun etkinlikler düzenlenmiştir. Etkinlikler için öngörülen süreler; zenginleştirmelere, etkinliğin aşamalarına, öğrencilerin düzeylerine ve ihtiyaçlarına göre değerli öğretmenlerimiz tarafından artırılabilir/azaltılabilir.

Etkinliklerin sonunda verilen ölçme ve değerlendirme araçları ile öğretmenin, öğrenme öğretme süreçlerini takip etmesi, ürünün değerlendirilmesi ve öğrencinin gelişiminin izlenmesi amaçlanmıştır. Değerlendirme bölümlerinde verilen ölçme ve değerlendirme araçları etkinlik içeriğine ve sürecine bağlı olarak değişiklik göstermektedir.

Yüzyıl becerileri dikkate alınarak hazırlanmış olan bu çalışmanın tüm öğrencilerimize katkı sağlaması ve yeni fikirlere ışık tutmasını temenni ediyor, yaz okulu sürecinde öğrencilerle buluşan tüm değerli öğretmenlerimize teşekkürlerimizi sunuyoruz.

Yazılım Geliştirme Atölyesi Materyal Geliştirme Komisyonu

Haziran, 2023

PROGRAMIN TEMEL YAKLAŞIMI VE İLKELERİ

Yazılım Geliştirme Atölyesi Bilim ve Sanat Merkezi (BİLSEM) Destekleme ve Yetiştirme Yaz Okulu Atölye Programı'nda ilerlemecilik felsefesi ilkeleri benimsenerek katı bir disiplin anlayışından ziyade aktif öğrenme yaklaşımlarını temel alan ve bireysel farklılıkları gözeten bir yapıda hazırlanmıştır.

Program içeriğinin oluşturulmasında konu temelli yaklaşımlardan olan süreç tasarım modeli benimsenmiştir. Bu bağlamda yaparak yaşayarak öğrenmeyi hedefleyen, bilgi edinmeyi ve öğrenme sürecini merkeze alan beceri temelli yaklaşımlara başvurulmuştur.

PROGRAMIN YAPISI

Programın Genel Amaçları

Bu program Bilim ve Sanat Merkezi (BİLSEM) Destekleme ve Yetiştirme Yaz okuluna devam edecek öğrencilerin Bilgi ve İletişim teknolojileri alt alanlarından olan Yazılım geliştirme alanında beceri gelişimlerini destekleyecek şekilde hazırlanmıştır. Programın genel amacı öğrencilerin yazılım geliştirme alanında temel bilgi ve beceri sahibi olmalarına, ilgi ve yetenek alanlarını keşfedebilmelerine odaklanarak üretken bireyler olarak kariyer planlamalarına katkı sağlamaktır.

Bu genel amaç doğrultusunda Bilim ve Sanat Merkezi (BİLSEM) Destekleme ve Yetiştirme Yaz Okulunda Yazılım Geliştirme Atölye çalışmalarına katılacak 2-12. sınıf öğrencilerine aşağıdaki hedeflerin kazandırılması planlanmıştır.

1. Yazılım geliştirmenin günlük hayattaki önemini anlama,
2. Yazılım geliştirme sürecini günlük hayatta kullanabilme,
3. İhtiyacına uygun yazılımlar tasarlayabilme,
4. Bilgi ve iletişim teknolojileri alanında geliştirilen yeni bir ürünlerin çalışma yapılarını kavrama,
5. Yerli ve milli teknoloji geliştirmenin önemini kavrama.
6. Teknolojik ürünler geliştirmeye istek duyma.

Kazanımlar

Bilim ve Sanat Merkezi (BİLSEM) Destekleme ve Yetiştirme Yaz Okulu Yazılım Geliştirme Programı'nda yer alan kazanımlar ilkokul, ortaokul ve lise düzeyinde bilgi-işlemsel düşünme becerilerinden oluşmaktadır. Kazanımların sıralanmasında aşamalılık ve ardışıklık ilkesi gözetilerek basitten karmaşığa, kolaydan zora, somuttan soyuta olacak şekilde hazırlanmıştır. Ayrıca kazanımların güncellik, öğrenciye görelik, hedefe uygunluk, yaşama yakınlık, transfer edilebilirlik gibi öğretimsel ilkelere uygun olarak hazırlanmasına da dikkat edilmiştir. Kazanımlar, öğrenme alanı, alt öğrenme alanı ve kazanımların sırası şeklinde verilmiştir. Bazı kazanımların gerçekleştirilmesinde dikkat edilmesi gereken hususlar, ilgili kazanımların altında açıklama olarak belirtilmiştir.

Atölyenin Öğrenme Alanı /Temalar: Yazılım Geliştirme Atölyesi

1. İlkokul Düzeyi

1.1. Programlamaya Giriş

1.1.1. Farklı programlama dillerini tanır.

1.1.2. Program yazmak için gerekli hazırlıkları yapar.

1.2. Değişkenler

1.2.1. Problem durumuna uygun algoritma hazırlar

1.2.2. Bir algoritmadaki değişkenleri belirler.

1.2.3. Kodlamada değişkenleri kullanır.

1.3. Operatörler

1.3.1. Değişkenleri aritmetiksel işlemlerde kullanır.

1.3.2. Karşılaştırma operatörleri kavramını açıklar.

1.4. Koşul yapıları

1.4.1. Koşul yapılarını bir problemin çözümünde kullanır.

1.5. Döngüler

1.5.1. Verilen bir örnek durumdaki döngüleri tespit eder.

1.5.2. Bir problemin çözümünde döngü yapısını kullanır.

1.6. Algoritmik Problem Çözümü

1.6.1. Gerçek yaşam problemi için algoritmik çözüm üretir.

1.6.2. Belirlenen gerçek yaşam problemini çözecek projeyi geliştirir.

1. Ortaokul Düzeyi

1.1. Programlamaya Giriş

1.1.1. Farklı programlama dillerini kullanım alanlarına göre listeler.

1.1.2. Temel girdi-çıktı fonksiyonları ile işlem yapar.

1.2. Değişkenler

1.2.1. Değişken kavramını açıklar.

1.2.2. Kodlamada değişkenleri kullanır.

1.3. Operatörler

1.3.1. Operatör kavramını açıklar.

1.3.2. Aritmetiksel, mantıksal ve karşılaştırma operatörlerini kullanır.

1.4. Koşul yapıları

1.4.1. Koşul yapılarını bir problemin çözümünde kullanır.

1.5. Döngüler

1.5.1. Farklı döngü türlerini bir problemin çözümünde kullanır.

1.6. Fonksiyonlar

1.6.1. Fonksiyon kavramını açıklar.
1.6.2. Kullandığı programlama dilinde var olan hazır fonksiyonları kullanır.
1.6.3. Kullandığı programlama dilinde belirli bir amaca yönelik fonksiyon yazar.

2. Lise Düzeyi

2.1. Programlamaya Giriş

2.1.1. Programlama dillerinin amacını keşfeder.
2.1.2. Kod yazmak için gerekli hazırlıkları yapar.
2.1.3. Derleyici ve yorumlayıcı arasındaki farkı açıklar.
2.1.4. Temel girdi-çıktı fonksiyonları ile işlem yapar.

2.2. Değişkenler

2.2.1. Değişken kavramını açıklar.
2.2.2. İhtiyaca uygun değişken türlerini belirler.
2.2.3. Değişkenleri kurallarına uygun olarak oluşturur.
2.2.4. Bir kod örneğindeki değişkenleri tespit eder.

2.3. Listeler

2.3.1. Liste kavramını açıklar.
2.3.2. Liste ile değişken arasındaki farkı açıklar.
2.3.3. Listeler üzerinde indeksleme ve dilimleme işlemlerini yapar.
2.3.4. Listeler üzerinde farklı metot işlemlerini gerçekleştirir.

2.4. Operatörler

2.4.1. Operatör kavramını açıklar.
2.4.2. Aritmetik operatörleri kullanır.
2.4.3. Atama operatörlerini kullanır.
2.4.4. Mantıksal ve Karşılaştırma operatörlerini kullanır.

2.5. Koşul yapıları

2.5.1. Mantıksal ifadeleri ihtiyacına göre seçer.
2.5.2. Çoklu koşul yapısını kullanır.
2.5.3. İç içe koşul yapısını kullanır.

2.6. Döngüler

2.6.1. Döngü kavramını ve türlerini açıklar.
2.6.2. Döngüye ihtiyaç duyulan durumları fark eder.
2.6.3. Bir problemin çözümünde amacına uygun döngüyü kullanır.
2.6.4. İç içe döngü yapısını kullanır.
2.6.5. Listelerle döngüleri bir arada kullanır

2.7. Fonksiyonlar

2.7.1. Fonksiyon kavramını açıklar.

2.7.2. Kullandığı programlama dilinde var olan hazır fonksiyonları kullanır.

2.7.3. Parametre alan fonksiyonları yazar.

2.7.4. Geriye değer döndüren ve döndürmeyen fonksiyonları yazar.

2.7.5. Özyinelemeli fonksiyonların (recursive) çalışma mantığına uygun durumları tespit eder.

İçerik

Bilim ve Sanat Merkezi (BİLSEM) Destekleme ve Yetiştirme Yaz Okulu Yazılım Geliştirme Programı'nda konular farklı sınıf seviyelerinde derinleştirilerek, daha önce kazandırılan bilgi ve beceriler genişletilerek tekrar edecek şekilde içerik düzenlenmiştir.

Program içeriğinde programlamaya giriş, değişkenler, operatörler, koşul yapıları, döngüler ve algoritmik problem çözümüne yer verilmiştir. İlkokul, ortaokul ve lise seviyesinde benzer öğrenme alanları, alt öğrenme alanları ve kazanımlar yer almakla birlikte etkinlik boyutunda öğrencilerin gelişim özellikleri dikkate alınarak etkinlik tasarımları gerçekleştirilmiştir.

Etkinlikler

Bilim ve Sanat Merkezi (BİLSEM) Destekleme ve Yetiştirme Yaz Okulu Yazılım Geliştirme Programı'nın etkinlikleri ilkokul, ortaokul ve lise düzeyinde hazırlanmıştır. Program kapsamında ilkokul için 12, ortaokul için 12 ve lise için 12 etkinlik yer almaktadır. Hazırlanan etkinlikler ile öğrencilerin yazılım geliştirme alanında temel bilgi ve beceri sahibi olmaları, ilgi ve yetenek alanlarını keşfedebilmeleri amaçlanmıştır. Bu bağlamda öğrencilerin aktif katılım sağlayabilecekleri, günlük yaşamla ilişkili, aktivite temelli etkinlikler tasarlanmış olup, uygulayıcıya yardımcı olması bakımından dikkat çekme, güdüleme, derse geçiş, dersin işlenişi, özet bölümleri açık, anlaşılır ve net bir şekilde sunulmuştur. Biliyoruz ki bu amaçlarımıza, etkinlikleri uygulayan değerli öğretmenlerimizin bilgi, birikim, deneyim ve motivasyonlarının da sürece dahil olması ile ulaşılacaktır.

Değerlendirme

Ölçme ve değerlendirme araçları; öğretmenin, öğrenme öğretme süreçlerini takip etmesine, öğrencinin gelişimini izlemesine yardımcı olacak niteliktedir. Değerlendirme bölümlerinde verilen ölçme ve değerlendirme araçları etkinlik sürecine bağlı olarak değişiklik göstermektedir. Öğrenci çalışmalarının değerlendirilmesi için örnek olarak verilen ölçme değerlendirme araçlarına uygulayıcı tarafından eklemeler yapılabilecektir.

Etkinliklerin değerlendirilmesinde boşluk doldurma, soru cevap, kontrol listesi, dereceli puanlama anahtarı, öz değerlendirme ölçme araçları kullanılmıştır.

PROGRAMIN UYGULANMASI

Bilim ve Sanat Merkezi (BİLSEM) Destekleme ve Yetiştirme Yaz Okulu Yazılım Geliştirme Programı'nın uygulanmasında aşağıdaki hususlara dikkat edilmesi önerilmektedir.

Öğretmen Yeterlilikleri: Hazırlanan Yazılım Geliştirme Yaz Okulu Programı teknik alan bilgisi temelli öğrenme alanları içermektedir. Bu nedenle öğretmenlerin yazılım geliştirme konu alanlarına yönelik bilgi ve becerilerinin yeterli düzeyde olması gerekmektedir.

Öğrenme Çevresi: Yazılım Geliştirme Yaz Okulu Programı uygulama temelli olduğundan; alt yapı olanakları ve öğrenme çevresinin hazırlanması bağlamında belirli koşulların hazır bulundurulmasını gerektirmektedir.

Yöntem: Yazılım Geliştirme Yaz Okulu Programı'nda yer alan kazanımlar öğretim sürecinde kuramsal bilginin yanısıra uygulama olanakları ile zenginleştirilmelidir. Etkinlikler sürecinde öğrencilerin bağımsız uygulamalar yapmalarına imkân tanınması öğrenme verimliliğine katkı sağlayacaktır.

Öğretim Süreci ve Öğrenme Çıktılarının Değerlendirilmesi: Kazanımlar temel alınarak hazırlanan öğrenme-öğretme etkinliklerinde örnek değerlendirme araçları sunulmuştur. Etkinlik sürecinde uygulayıcı tarafından uygun görülen ek değerlendirme araçları kullanılabilir. Etkinlik sürecinde süreç ve ürün odaklı değerlendirilmenin yapılması öğrencilere geri dönütler verilmesi, süreç içerisinde öğrencilerin gelişiminin takip edilmesine imkân sağlayacaktır.

PROGRAMIN DEĞERLENDİRİLMESİ

Bilsem yaz okulu atölye programlarının değerlendirilmesinde süreç odaklı değerlendirme ve sonuç değerlendirmesi kullanılabilir. Süreç odaklı değerlendirmede ölçme araçları etkinliklerin uygulanmasının her aşamasında kullanılabileceği gibi etkinlikler uygulandıktan sonra öğrencilerin belirlenen kazanımları ne düzeyde elde ettiklerini belirlemeye yönelik olarak da kullanılabilir. Bu süreçte ölçekler, kontrol listeleri, anketler vb. ölçme araçları kullanılabileceği gibi öğrencilerin kazanıma ilişkin davranışları sergilemesini sağlayacak sorular, bulmacalar, boşluk doldurma, cümle-örnek olay tamamlama vb. yöntemler de kullanılabilir. Sonuç değerlendirme ise program tamamlandıktan sonra programın öğrenciler üzerindeki etkileri incelemek amacıyla gerçekleştirilir. Bu süreçte yarı yapılandırılmış görüşme formları kullanılarak bireysel ve odak grup görüşmeleri gerçekleştirilebilir. Ayrıca program değerlendirme anket formları kullanılarak öğrencilerin programa ilişkin görüşleri değerlendirilebilir. Bununla birlikte öğrencilerin belirlenen kazanımları ne derece elde ettiklerine yönelik görüşlerini değerlendirmek amacıyla program çıktı değerlendirme formları da kullanılabilir.

ÖZEL EĞİTİM VE REHBERLİK HİZMETLERİ
GENEL MÜDÜRLÜĞÜ

BİLİM VE SANAT MERKEZLERİ

YAZ OKULU DESTEKLEME VE YETİŞTİRME KURSU PROGRAMI

YAZILIM ATÖLYESİ

ETKİNLİKLER

İLKOKUL

TÜRKİYE CUMHURİYETİ MİLLÎ EĞİTİM BAKANLIĞI

ETKİNLİK NO	**1.1**
ETKİNLİK ADI	DİLLER
SINIF/KADEME	İlkokul
SÜRE	40+40=80 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Programlamaya Giriş
KAZANIMLAR	**1.1.1.** Farklı programlama dillerini tanır.
TEMEL BECERİLER	Araştırma yapma , Keşfetme
YÖNTEM VE TEKNİKLER	Doğrudan anlatım, Soru-cevap, Benzetim
ARAÇ-GEREÇLER	Bilgisayar, Uygulama geliştirme ortamı
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında blok tabanlı, üç boyutlu oyun geliştirmeye uygun uygulama geliştirme ortamı olması gerekir.
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma.
SÜREÇ	Öğrencilerden *"Dünyada kaç tane konuşma dili olduğunu"* tahmin etmeleri istenir. Bu sayı yaklaşık olarak 7000'i geçmektedir (Uzun, 2012). Öğrencilerden 5 ayrı dilde "merhaba" nasıl deniyor araştırmaları istenir. Dikkat çekme etkinliğinden sonra bilgisayarın çalışması için onunla iletişime geçmemiz gerektiği ve bunu bir programlama dili sayesinde yapabildiği belirtilir. Öğrencilerin internetten şu konularda araştırma yapmaları istenir. 1) İlk programlama dilinin adı nedir? Ne zaman ortaya çıkmıştır? 2) Kaç farklı programlama dili vardır? 3) Dünyada en çok kullanılan programlama dili nedir? 4) 3 farklı programlama dilinde ekrana "Merhaba Dünya" hangi komutla yazdırılır? Sorulara cevap verildikten sonra öğretmen tarafından dersin hedeflerini vurgulayan, etkinliği özetleyen ve sonraki etkinlik hakkında kısa bilgi veren ders sonu konuşması yapılır: *"Çocuklar bugün sizinle farklı programlama dilleri ve bunların kullanım alanlarını öğrendik. Bir sonraki derste bir programlama ortamını tanıyacağız."* denilir ve ders sonlandırılır

DEĞERLENDİRME	Öğrencilerin aşağıdaki sorulara cevap vermesi sağlanır. ✓ Öğrencilerle neden fazla sayıda programlama dili olduğu tartışılır. En sevdikleri bilgisayar uygulamasının hangi dilde yazıldığını öğrenmeleri istenir. ✓ Neden farklı programlama dillerinin olduğu sorgulanır.
KAYNAKÇA	Uzun, N. E. (2012). Türkçenin dünya dilleri arasındaki yeri üzerine. *Türkoloji Dergisi*, *19*(2), 115-134.

ETKİNLİK NO	**1.2**
ETKİNLİK ADI	KURULUMLAR
SINIF/KADEME	İlkokul
SÜRE	40+40=80 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Programlamaya Giriş
KAZANIMLAR	**1.1.2.** Program yazmak için gerekli hazırlıkları yapar.
TEMEL BECERİLER	Problem çözme, Araştırma, Gözlem
YÖNTEM VE TEKNİKLER	Göster-yap, Soru-cevap
ARAÇ-GEREÇLER	Bilgisayar, İnternet
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında internet bağlantısının olması gerekir. Etkinliklerin yürütüleceği uygulama geliştirme ortamının indirilmesi yada çevrimiçi bir ortam ise üyelik, sınıf oluşturma gibi ön hazırlıklar yapılmalıdır.
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma.
SÜREÇ	Öğrencilere daha önce program kurup kurmadıkları sorulur. Program kurulumu yapan öğrenci varsa nasıl kurduklarını anlatmaları istenir. Kurmadılarsa öğretmen kısaca programları kurmak için gerekli adımları anlatır. Etkinliğe geçildiğinde program geliştirme ortamının nasıl kurulduğu gösterilir. Bu çalışmada çocuklar için üç boyutlu oyun geliştirme ortamı kullanılarak hazırlanacak etkinlikler sunulmuştur. Bu uygulamanın indirilmesi ve kurulması gerekmektedir. Öğrencilere indirme sayfası gösterilir. İndirilmesi ve kurulması sağlanır. Kurulum esnasında Dil seçeneği, masaüstüne kısayol oluşturma seçeneği gibi sorular gelebileceği belirtilir. Etkinliğin tamamlanmasının ardından öğretmen tarafından dersin hedeflerini vurgulayan, etkinliği özetleyen ve sonraki etkinlik hakkında kısa bilgi veren ders sonu konuşması yapılır: “Çocuklar bugün sizinle bir programlama dillini kullanmak için gerekli ortamın kurulumunu ı öğrendik. Bir sonraki derste kurduğumuz programlama dili ortamını kullanmaya başlayacağız.” denilir ve ders sonlandırılır.
DEĞERLENDİRME	✓ Kurulan uygulama geliştirme ortamının çalıştırılması ve yeni bir uygulama ekranının (yeni dünya oluştur gibi) açılması istenir.

ETKİNLİK | 3

ETKİNLİK NO	**1.3**
ETKİNLİK ADI	DÜNYA KIRMIZI ELMA GÜNÜ
SINIF/KADEME	İlkokul
SÜRE	40+40+40+40=160 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Programlamaya Giriş
KAZANIMLAR	**1.2.2.** Bir algoritmadaki değişkenleri belirler. **1.2.3.** Kodlamada değişkenleri kullanır. **1.3.1.** Değişkenleri aritmetiksel işlemlerde kullanır. **1.3.2.** Karşılaştırma operatörleri kavramını açıklar.
TEMEL BECERİLER	Problem çözme, Analitik düşünme, Keşfetme
YÖNTEM VE TEKNİKLER	Göster-yap, Problem çözme
ARAÇ-GEREÇLER	Bilgisayar, Uygulama geliştirme ortamı
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında blok tabanlı, üç boyutlu oyun geliştirmeye uygun uygulama geliştirme ortamı olması gerekir. ✓ Aşağıdaki operatörler etkinlik öncesi gözden geçirilir. **Karşılaştırma operatörleri:** Değişkenlerin aldığı değerlerin karşılaştırmalarını yapmak için kullandığımız operatörlerdir. Yani bir değişkenin başka bir değişkenden büyük, küçük, eşit değil veya eşit olup olmadığını belirleyen operatörlerdir. Bu operatörlerin çoğunu günlük hayatımızda ve matematikte kullanmaktayız. == Eşittir (x == y) != Eşit Değildir (x != y) > Büyüktür (x > y) < Küçüktür (x < y) >= Büyük Eşittir (x >= y) <= Küçük Eşittir (x <= y) Karşılaştırma operatörleri bize boolean tipinde (True ya da False) veri gönderirler.
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma.na sahip olma

SÜREÇ

Dersin başında öğrencilere *"Elma sever misiniz? En çok hangi renk elmayı seversiniz?"* soruları sorularak cevaplar alınır. Ardından *"Dünya'da bazı ülkelerde 1 Aralık Dünya Kırmızı Elma Yeme Günü olarak kutlanır. Bugünde kırmızı elmalarla çeşitli lezzetler hazırlanır ve yenilir, bunu biliyor muydunuz? Hatta günün sloganı bile vardır. Her gün bir elma doktoru uzak tutar!"* bilgileri verilerek öğrencilerin dikkati çekilir.

Dikkat çekme sorularını takiben öğrencilere:

"Hepinizin bilgisayar oyunlarını oynamayı çok sevdiğinden eminim. Oynadığınız oyunlarda kuralları kendiniz koymak ister miydiniz? Bu oyunlarda kuralları değiştirme şansınız olsaydı hangi kuralı değiştirirdiniz?" soruları sorulur ve cevaplar üzerinde konuşulur.

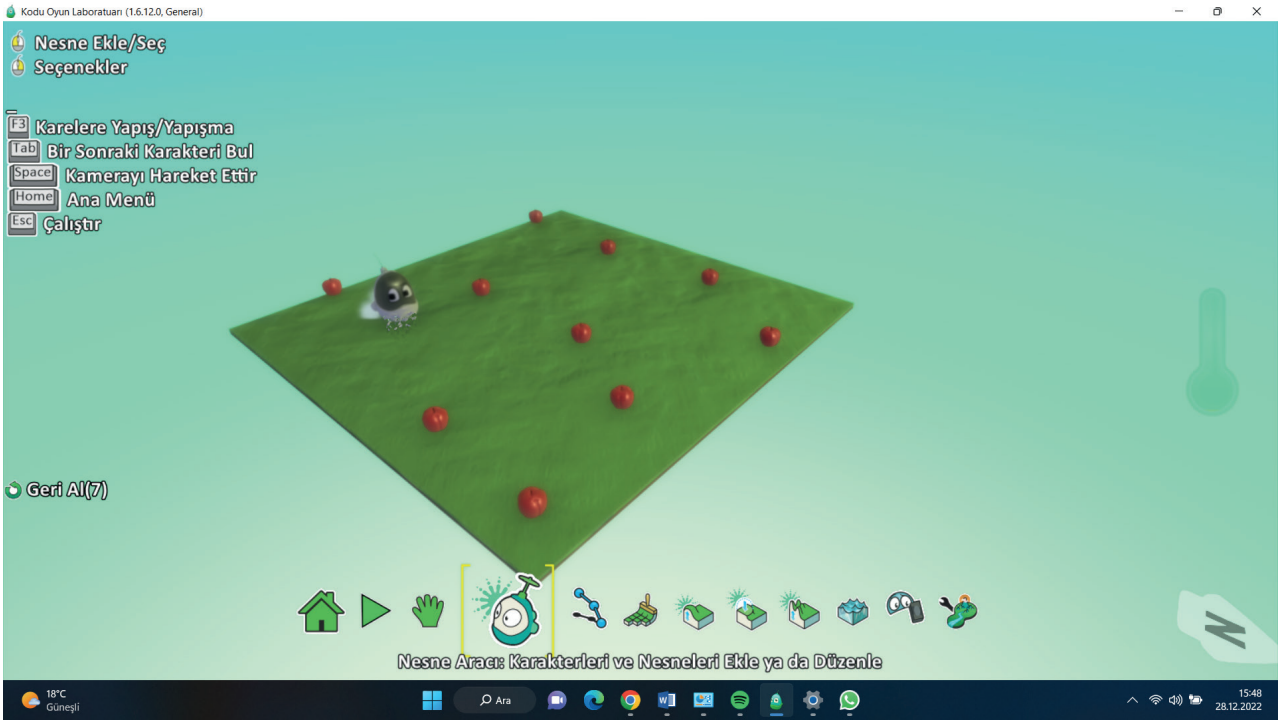
Ardından "*Bugün sizlerle oynarken çok eğleneceğiniz bir oyun kodlayacağız. Bu oyunun tasarımcısı ve yazılımcısı siz olacağınız için görsel olarak nasıl görünmesini isterseniz oyununuz da öyle görünecek. Aynı zamanda tüm kuralları siz belirleyeceksiniz."* denilir ve devam edilir.

"İlk olarak birlikte Dünya Kırmızı Elma Gününe özel bir elma toplama oyunu kodlayacağız. Birlikte tasarımını ve kodlamasını yaptığımız oyunun tamamlanmasının ardından sizden verdiğim yönergeye göre oyunun kendinize ait versiyonlarını geliştirmenizi isteyeceğim. İlk olarak oyunumuzun algoritmasını çıkaralım!" denilerek oyunun algoritması çıkarılır.

Oyunun algoritmasının çıkarılırken öğretmen öğrencilere karşılaştırma operatörleri kavramını açıklar. Ayrıca karşılaştırma operatörleri kullanımı hakkında örnekler verir. Örneğin puan artışının hedeflenen puanla karşılaştırması ve iki skorun birbirine eşitlenmesi durumu.

1. Yeni bir Dünya oluşturun.
2. Nesne Aracı menüsünden oyun karakterimiz olan koduyu ve istenen sayıda elmayı ekleyin.
3. Oyuncu karakterimiz olan koduyu programlamaya başlayın.
4. Kodunun nasıl hareket edeceğini kodlayın.
5. Kodunun kırmızı elmalara çarptığında elmaları yemesini ve puan artışını kodlayın.
6. Puan artışının hedeflenen puanla karşılaştırmasını ve iki skorun birbirine eşitlenmesi durumunda oyunu kazandıran kodu yazın.

Oyunun algoritmasının çıkarılmasının ardından tasarım aşaması öğrencilerle birlikte yapılır. Nesne Aracı menüsünden oyun karakterimiz olan kodu ve istenen sayıda elma Görsel 1.3.1' deki gibi Dünya'ya eklenir. Ardından yun karakterimiz olan kodunun hareket, elmaya çarptığında yeme, ve puan kazanma, toplanan puan ve hedeflenen skor eşitlenince kazanılan kodlar Görsel 1.3.2 deki gibi yazılır.



Görsel 1.3.1: Nesne Aracı menüsünden oyun karakterimiz olan kodu ve istenen sayıda elmanın Dünya'ya eklenmiş hali

Algoritma çıkarılırken değişken kullanımına dikkat çekilir. Elmaları topladığımızda puan artışının değişkenlerle aritmetiksel işlemler yardımıyla gerçekleştiğinin farkına varır. Aynı zamanda belirli bir puana ulaşınca kazanmasını karşılaştırma operatörleriyle yapıldığı belirtilir. Karşılaştırma operatörlerinin tanımı yapılır ve ilgili örnekler verilir.

Görsel 1.3.2: Oyun karakterimiz olan kodunun hareket, elmaya çarptığında yeme, puan kazanma, toplanan puan ve hedeflenen skor eşitlenince kazanılan kodların ekran görüntüsü

Oyunun öğretmen ile birlikte tamamlanmasının ardından öğrencilere *"Siz olsaydınız neleri farklı yapardınız?, Karşılaştırma operatörleri nedir? Karşılaştırma operatörleri için bir kullanım örneği verebilir misiniz?"* soruları sorulur ve cevaplar üzerinde konuşulur. Varsa eksik öğrenmeler giderilir. Son olarak "Öyleyse kendi oyununuzu yapmanın zamanı geldi. Aşağıdaki yönergeye uygun olarak elma toplama oyununun kendi versiyonlarını yapınız." denilerek kendi oyunlarını tasarlamaları ve kodlamaları beklenir. Bu yapılırken öğrencilerin aşağıdaki yönergeyi takip etmeleri istenir.

Yönerge

1. Yeni bir Dünya oluşturun.
2. Nesne Aracı menüsünden oyun karakterimiz olan koduyu ve istenen sayıda elmayı ekleyin.
3. İstenirse Dünya genişletilebilir ve ağaçlar ya da farklı nesneler eklenerek tasarım zenginleştirilebilir.
4. Oyuncu karakterimiz olan koduyu programlamaya başlayın.
5. Kodunun nasıl hareket edeceğini kodlayın. Bu kez yön tuşları yerine WASD tuşlarını tercih edebilirsiniz.
6. Kodunun tercih edilen renkte elmalara çarptığında elmaları yemesini ve puan artışını kodlayın.
7. Kodunun tercih edilmeyen renkte elmalara çarptığında elmaları yemesini ve puan azalmasını kodlayın. Tercih edilmeyen elma sayısı ve kodunun çarpması durumunda ne kadar puan azalacağı oyunu kodlayan kişiye göre değişebilir.
8. Toplanan skorun hedeflenen puanla karşılaştırmasını ve iki skorun birbirine eşitlenmesi durumunda oyunu kazandıran kodu yazın.

Etkinliğin tamamlanmasının ardından öğretmen tarafından dersin hedeflerini vurgulayan, etkinliği özetleyen ve sonraki etkinlik hakkında kısa bilgi veren ders sonu konuşması yapılır:

"Çocuklar bugün bir algoritmadaki değişkenleri belirlemeyi, bu değişkenleri aritmetiksel işlemlerde kullanmayı ve karşılaştırma operatörleri ile karşılaştırmayı öğrendik. Bir sonraki derste değişkenleri kullanarak farklı uygulamalar yapmayı öğreneceğiz." denilir ve ders sonlandırılır.

DEĞERLENDİRME

Tablo 1.3.1: Kontrol listesi

Kontrol Listesi	Evet	Hayır
Bir algoritmadaki değişkenleri belirler		
Kodlamada değişkenleri kullanır.		
Değişkenleri aritmetiksel işlemlerde kullanır.		
Karşılaştırma operatörleri kavramını açıklar.		

ETKİNLİK | 4

ETKİNLİK NO	**1.4**
ETKİNLİK ADI	BUZ HOKEYİ
SINIF/KADEME	İlkokul Düzeyi
SÜRE	40+40=80 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Programlamaya Giriş
KAZANIMLAR	**1.3.1.** Değişkenleri aritmetiksel işlemlerde kullanır. **1.3.2.** Karşılaştırma operatörleri kavramını açıklar. **1.4.1.** Karşılaştırma operatörlerini koşul yapılarında kullanır. **1.4.2.** Koşul yapılarını bir problemin çözümünde kullanır.
TEMEL BECERİLER	Problem çözme
YÖNTEM VE TEKNİKLER	Doğrudan anlatım, Gösterip-Yaptırma, Soru-cevap
ARAÇ-GEREÇLER	Bilgisayar, Uygulama geliştirme ortamı
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında blok tabanlı, üç boyutlu oyun geliştirmeye uygun, uygulama geliştirme ortamı olması gerekmektedir.
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma.

SÜREÇ

Öğrencilere dikkat çekmek amacıyla şu soru sorulur:

"Kimler Buz Hokeyi oyununu biliyor ya da daha önce izledi?"

Öğrencilerden gelen cevaplar dinlendikten sonra internetten kısa bir buz hokeyi maç görüntüsü izletilir. Ardından şu şekilde buz hokeyi hakkında kısaca bilgi verilir:

Hokey, 28 Olimpik spor branşından biri olup futboldan sonra en çok seyirci kitlesine sahip olan branştır. Ortaya çıktığı zaman bilinmese de yaklaşık 3000 yıl önce Mısır'da M.Ö 1000 yılında Etopya'da oynandığına dair belgeler bulunmaktadır. "Hokey" sözcüğü Fransızca "çoban sopası" anlamına gelen "Hocquet"-den geldiği sanılmaktadır (Türkiye Hokey Federasyonu, erişim Aralık 2022).

Hokeyin buz üstünde oynanan ve 19. yüzyılın ilk yıllarında Kanada'da ortaya çıkmıştır. Avrupa'da var olan benzer birkaç sporu baz alan sporda 1860'lı yıllarda buz hokey diski (puck) normal topun yerini almış ve 1879'da iki McGill Üniversitesi öğrencisi, Robertson ve Smith, oyunun ilk kurallarını oluşturulmuştur (Türkiye Olimpiyat Komitesi, erişim Aralık 2022*).*

Öğrencilere:

"Haydi bilgisayara karşı oynanan bir buz hokeyi bilgisayar oyununu tasarlayalım" denilerek etkinliğe giriş yapılır.

Oyun ile ilgili aşağıdaki sorulara cevap verilerek oyun hakkında bilgiler toplanır.

1. Oyunun amacı nedir?
2. Oyun nasıl bir dünyada gerçekleşmektedir?
3. Oyundaki karakterler kimlerdir?
4. Oyunda kontrol edeceğimiz değişkenler nelerdir?
5. Değişkenlerde değişim neye göre ve nasıl olacak?
6. Oyunda sabit sayı var mı?
7. Karakterler nasıl kontrol edilir?
8. Oyunun kuralları nedir?

CEVAPLAR

1. **Oyunun Amacı:**

   Oyuncu yön tuşlarını kullanarak ileri geri hareket edebilecek, ortadaki diske vurmaya ve bilgisayarın alanına ortadaki diski göndermeye çalışacak.
2. **Oyun Dünyası:**

Buz Hokeyi Sahası

3. **Oyundaki Karakterler:**

1. Oyuncunun karakteri
2. Bilgisayarın karakteri
3. Ortadaki disk
4. Hakem (Gerekirse)

4. **Değişkenler:**

   Bilgisayarın skoru

   Oyuncunun skoru

5. **Değişkenlerde Değişim Neye Göre ve Nasıl Olacak**

Ortadaki diskin hareketine göre artacaklar.

6. **Sabitler:**

   Oyunu kazanma puanı

7. **Oyunun Kontrolleri:**

   Bilgisayar kendi kendine, belirli bir alanda gidip gelecek.

   Oyuncu karakterini klavyeyi kullanarak kontrol edecek. (Yön tuşlarını ya da WASD tuşlarını kullanarak)

8. **Oyunun kuralları:**

   Ortadaki disk kimin alanına girerse karşı takım +1 puan alır.

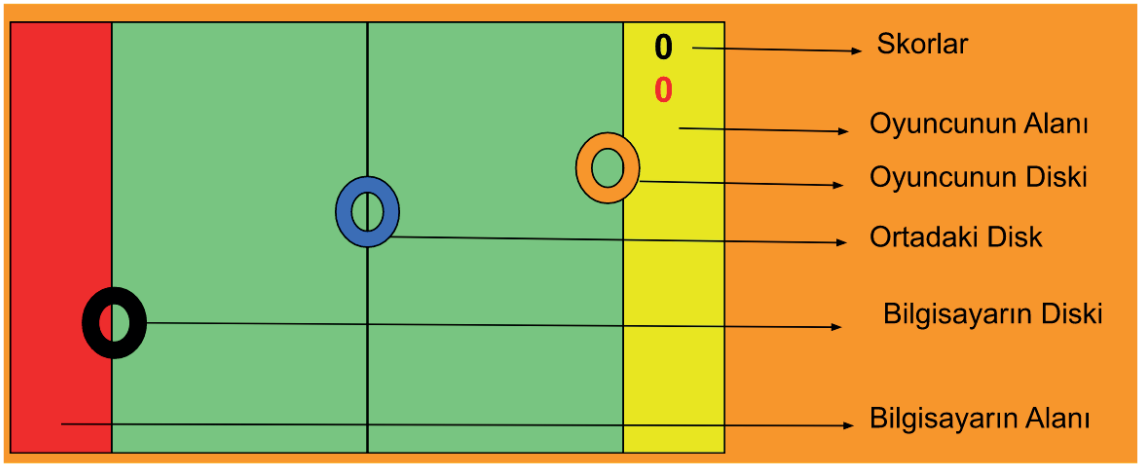
   10 puana ilk ulaşan oyunu kazanır.

Disk ilk oyuna başlarken veya puan alındıktan sonra orta sahaya gelerek oyunu başlatır.

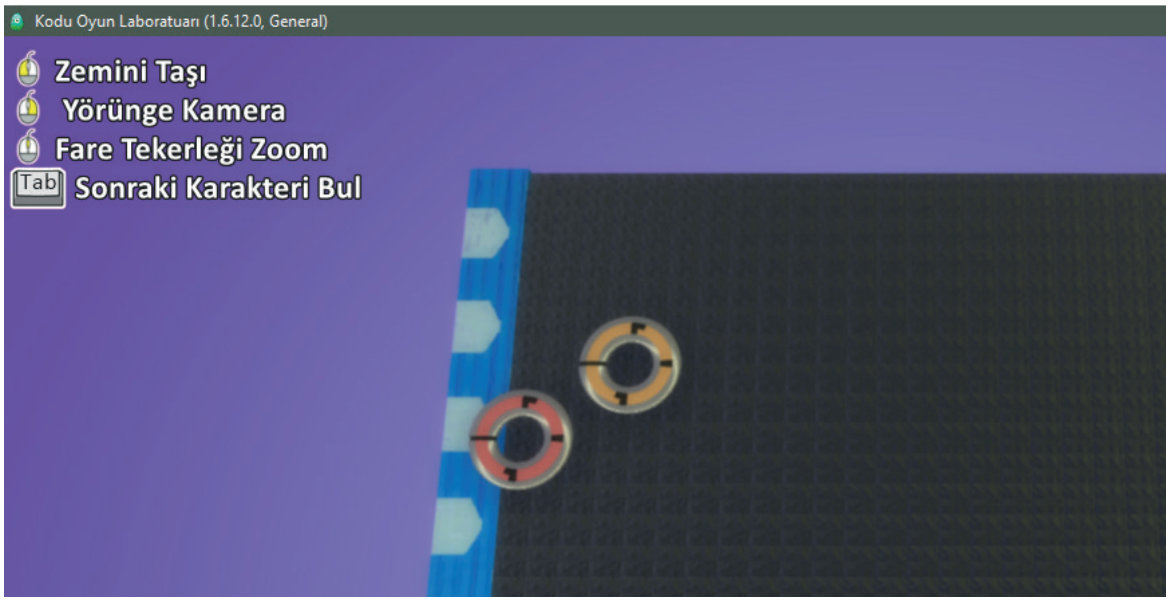
OYUN TASARIMI

1) Dünya Tasarımı

Oyun dünyasının taslak görünümü Görsel 1.4.1'de verilmiştir. Bu taslağa uygun olarak hazırlanmış dünya görünümü de Görsel 1.4.2'de sunulmuştur.



Görsel 1.4.1: Örnek Oyun Ekranı



Görsel 1.4.2: Görsel 1.4.1'de verilmiş örnek oyun ekranının uygulamada tasarlanmış hali. Bilgisayarın oyuncu karakteri olan diskin takip etmesi için bir patika aracı eklenmiştir.

1) Oyunun Kodları

Hakem görevini oyundaki herhangi bir karakter gerçekleştirebilir. Aşağıdaki kodlara göre hakem, disk 3'tür. Eğer hakem olarak yeni bir nesne tercih edilirse Disk 3'e ait olan 3 ve 4 nolu kodlar o nesnede olmalıdır.

Oyun içindeki her karakter için belirlenen nesneler şu şekilde kodlanır:

Disk 1

Sayfa 1

1 EĞER klavye Oklar -- YAP hareket et D/B

2 EĞER klavye değil -- YAP hareket et don

Disk 2

Sayfa 1

1 EĞER always -- YAP hareket et patikada

Disk 3

Sayfa 1

1 EĞER karada tür 47 -- YAP skor kırmızı 1 puan bir kez

2 EĞER karada tür 10 -- YAP skor mavi 1 puan bir kez

3 EĞER skor yaptı kırmızı eşit 10 puan -- YAP kazan kırmızı

4 EĞER skor yaptı mavi eşit 10 puan -- YAP kazan mavi

5 EĞER always -- YAP çal kodu

6 EĞER klavye WASD -- YAP hareket et

Etkinliğin tamamlanmasının ardından öğretmen tarafından öğrencilere: *"Operatör kavramı size neyi anlatıyor?Karşılaştırma operatörü nelerdir?"* soruları sorulur ve öğrencilerden alınan cevaplar üzerinde konuşulur. Gerekli dönütler sağlanır. Öğretmen daha sonra dersin hedeflerini vurgulayan, etkinliği özetleyen ve sonraki etkinlik hakkında kısa bilgi veren ders sonu konuşması yapılır:

"Bu etkinlikte buz hokeyi sporuna benzer bir bilgisayar oyunu hazırlanmıştır. Oyuna farklı seviyeler ekleyerek, iki oyunculu hale getirerek veya oyun dünyasını değiştirerek oyunu geliştirmek mümkündür." denilir ve ders sonlandırılır.

DEĞERLENDİRME

Öğrencilerin aşağıdaki sorulara cevap vermesi sağlanır.

- ✓ Bu oyunda Eğer ile kontrol ettiğimiz her şey bir koşul mudur.?
- ✓ Oyun sürekli bir tekrar içindedir ve bu tekrar bir koşul ile sonlanır. Oyunu sonlandıran bu koşul nedir?
- ✓ Bu oyundaki skorun artması hangi koşula bağlıdır?
- ✓ Oyuncuların hareketi hangi koşula bağlıdır?
- ✓ Patikada hareketi sağlayan döngünün çalışma mantığı nedir?

KAYNAKÇA

Türkiye Olimpiyat Komitesi (2022). Eyof sporlarını tanıyalım – buz hokeyi. https://www.olimpiyatkomitesi.org.tr/Detay/57/1 (Erişim tarihi: 27.10.2022).

Türkiye Hokey Federasyonu (2022). TARİHÇE. https://www.turkhokey.gov.tr/sayfalar/2/tarihce (Erişim tarihi: 27.10.2022).

ETKİNLİK | 5

ETKİNLİK NO	**1.5**
ETKİNLİK ADI	HIRSIZ-POLİS
SINIF/KADEME	İlkokul
SÜRE	40+40=80 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Programlamaya Giriş
KAZANIMLAR	**1.4.2.** Koşul yapılarını bir problemin çözümünde kullanır**.**
TEMEL BECERİLER	Problem çözme, Analitik düşünme, Keşfetme
YÖNTEM VE TEKNİKLER	Doğrudan anlatım, Soru-cevap, Eğitsel oyun
ARAÇ-GEREÇLER	Bilgisayar, İnternet, Projeksiyon cihazı/ Etkileşimli tahta
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında blok tabanlı, üç boyutlu oyun geliştirmeye uygun uygulama geliştirme ortamı olması gerekir.
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma. ✓ Koşul yapılarını bilir.

SÜREÇ

Dersin başında öğrencilere *"Koşul yapılarının neler olduğunu öğreneceğiz"*, denilerek öğrencilerin hedeften haberdar edilmesi sağlanır.

Öğrencilere;

"Hırsız polis oyununu daha önce oynadınız mı ya da duydunuz mu?"

diye sorulur. Oyunu bilen öğrencilerden nasıl oynandığını anlatması istenir. Cevaplar dinlendikten sonra iki farklı hırsız- polis oyunu olduğundan bahsedilir. Bunlardan biri masada oynanan diğeri hareket ederek oynanan oyunlardır.

İlk önce masada oynanan hırsız polis oyunun adımları öğrencilerle paylaşılır.

OYUNUN ADIMLARI

1. Öğrenci sayısı kadar eşit şekilde kesilmiş boş kâğıt alınır. Kâğıtlardan birine POLİS, birine HIRSIZ, birine HÂKİM ve diğerlerine HALK yazılır. Kâğıtlar ikiye katlanarak karıştırılır. Her öğrenciden bir kâğıt seçmesi istenir.
2. Hırsız yazan kâğıdı seçen oyuncu polis kâğıdını seçen oyuncuya yakalanmadan diğer oyunculara göz kırparak onları sobelemelidir.

3. Sobelenen oyuncu yüksek sesle "ben sobelendim diyerek oyun dışı olur. Bu arada polis karakteri çok dikkatli olmalıdır.
3. Eğer oyunun sonuna kadar hırsız herkesi sobelerse polis kaybetmiş olur.
4. Polis eğer hırsızı birini sobelerken yakalar ve kim olduğunu bulursa hırsız oyunu kaybetmiş olur.

Öğrencilerle bu oyun oynanarak etkinlik öncesi öğrencilerin kaynaşması ve motive olması sağlanır. Daha sonra öğrencilere *"Oyun oynamayı herkes çok sever. Oyun oynamak kadar eğlenceli bir şey de kendi oyunumuzu tasarlamaktır.* Şimdi de polisin hırsızı kovaladığı hırsız polis oyununu bilgisayarda tasarlayalım" denilerek etkinliğe giriş yapılır.

Oyun ile ilgili aşağıdaki sorulara cevap verilerek oyun hakkında bilgiler toplanır.

1. Oyunun amacı nedir?
2. Oyun nasıl bir dünyada gerçekleşmektedir?
3. Oyundaki karakterler kimlerdir?
4. Karakterler nasıl kontrol edilir?
5. Oyunun kuralları nedir?

CEVAPLAR

1. Oyunun Amacı:

Hırsız karakteri polis karakterinden kaçacak; polis karakteri hırsız karakterine çarpınca oyunu kazanacak.

2. Oyun Dünyası:

Oyun Sahası

3. Oyundaki Karakterler:

1. Polis karakteri
2. Hırsız karakteri

4. Oyunun Kontrolleri:

Hırsız karakteri hareket edecektir

Polis karakteri klavyenin WASD tuşlarını kullanarak hareket edecektir.

5. Oyunun kuralları:

Polis karakteri hırsız karakteri ile çarpışınca oyunu kazanır.

OYUN TASARIMI: Öğrencilerin de bu tasarımları kendi bilgisayarlarında yapmaları istenir.

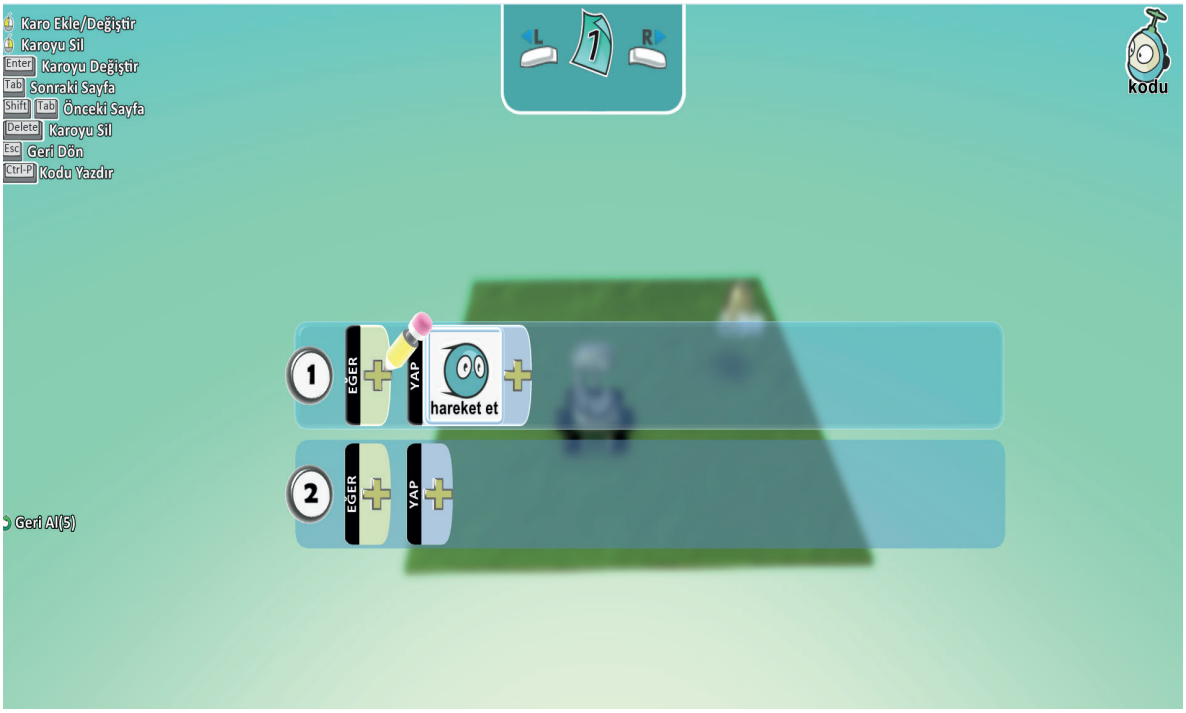
1) Dünya Tasarımı

Görsel 1.5.1 Örnek Oyun Ekranı

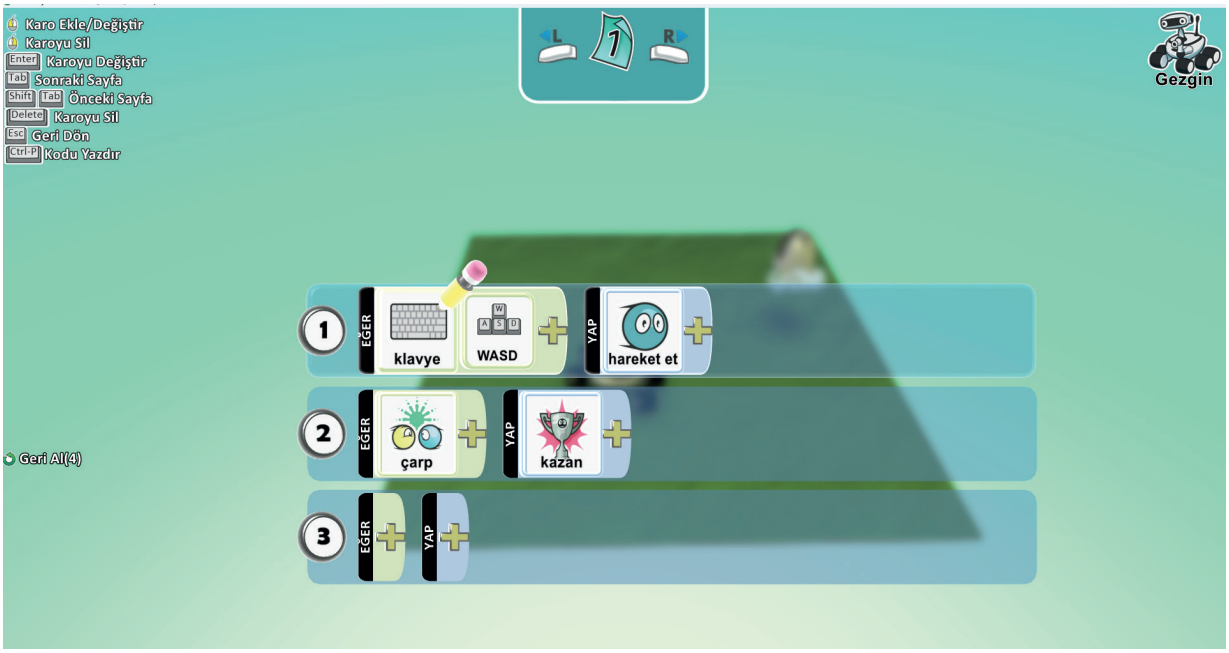
Görsel 1.5.1'deki gibi yeni bir dünya eklendikten sonra kodu karakteri eklenip Hırsız adı verilir. Gezgin karakteri eklenerek polis adı verilir.

2) Oyunun Kodları

Oyun içindeki her karakter için belirlenen nesneler şu şekilde kodlanır:



Görsel 1.5.2 Hırsız karakterinin kodların ekran görüntüsü



Görsel 1.5.3 Polis karakterinin kodların ekran görüntüsü

Öğrencilerin oyunlarını tamamlamasının ardından öğretmen öğrencilere:

"Çocuklar bugün sizinle koşul yapılarını kullandığımız bir oyun tasarladık. Bir sonraki derste döngü yapılarını kullanarak yeni bir oyun tasarlayacağız." diyerek sonraki etkinlik hakkında kısa bilgi verir ve dersi sonlandırır.

DEĞERLENDİRME

Öğrencilerin aşağıdaki sorulara cevap vermesi sağlanır.

- ✓ Bu oyunda Eğer ile kontrol ettiğimiz her şey bir koşuldur. Oyunu sonlandıran bu koşul nedir?
- ✓ Oyuncuların hareketi hangi koşullara bağlıdır?

ETKİNLİK | 6

ETKİNLİK NO	**1.6**
ETKİNLİK ADI	MOTOSİKLET YARIŞI
SINIF/KADEME	İlkokul
SÜRE	40+40=80 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Programlamaya Giriş
KAZANIMLAR	**1.5.1.** Verilen bir örnek durumdaki döngüleri tespit eder. **1.5.2.** Bir problemin çözümünde döngü yapısını kullanır.
TEMEL BECERİLER	Problem çözme, Gözlem
YÖNTEM VE TEKNİKLER	Göster-yap, Soru-cevap
ARAÇ-GEREÇLER	Bilgisayar, Internet
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında blok tabanlı, üç boyutlu oyun geliştirmeye uygun uygulama geliştirme ortamı olması gerekir.
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma.

SÜREÇ

Dersin başında öğrencilere *"Bugün sizinle tasarlayacağımız bir oyunla döngülerin ne olduğunu öğreneceğiz"*, denilerek öğrencilerin hedeften haberdar edilmesi sağlanır.

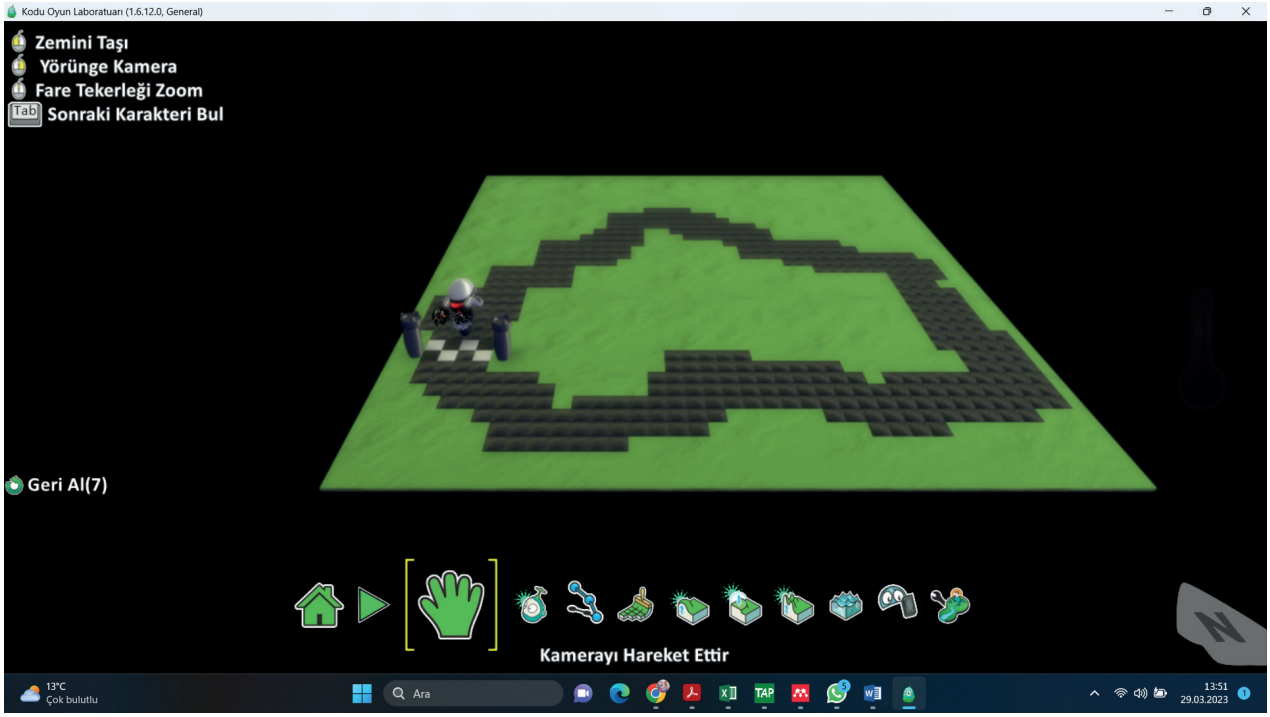
Daha sonra öğrencilere *"Çocuklar aranızda araba ya da motosiklet yarışlarına merakı olan var mı?"* soruları sorularak cevaplar alınır. Ardından *"Dünya' da birçok araba ya da motor yarışları sporları tutkunu vardır. Tabii bu tehlikeli sporlarda güvenliği sağlamak en önemli unsurdur. Motosiklet sürücülerinin kaza yaparak hayatını kaybetme oranı diğer sürücülerden yaklaşık 30 kat daha fazla olduğunu biliyor muydunuz? Hatta 2000 yılında dünyanın önde gelen motosiklet şirketleri, hız kapasitesinin sürekli arttırılması ve bunun sonunun gelmeyeceği ve hızın insanlar için tehlikeli olacağı düşüncesiyle bir anlaşma yaptılar. Bu anlaşma sonucunda 186 mph limitinde anlaştılar."* bilgileri verilerek öğrencilerin dikkati çekilir.

Dikkat çekme sorularını takiben öğrencilere *"Peki bilgisayarınızda, tabletinizde ya da telefonunuzda araba ve motosiklet yarışları oyunları oynuyor musunuz? Bu tarz oyunlar ilginizi çekiyor mu? Favori oyunlarınız hangileri? Bu oyunlarda değiştirmek istediğiniz herhangi bir şey var mı"* soruları sorulur ve cevaplar üzerinde konuşulur.

"Bugün sizlerle bir motosiklet yarışı oyunu tasarlayacağız. Öncelikle birlikte oyunumuzu tek kişilik olarak tasarlayacağız. Ardından sizlerden oyunu iki kişi oynayacak şekilde tasarlamanızı isteyeceğim. O yüzden dikkatle izleyelim. İlk olarak oyunumuzun algoritmasını çıkaralım!", denilerek oyunun algoritması uygulayıcı ve öğrencilerle birlikte çıkarılır.

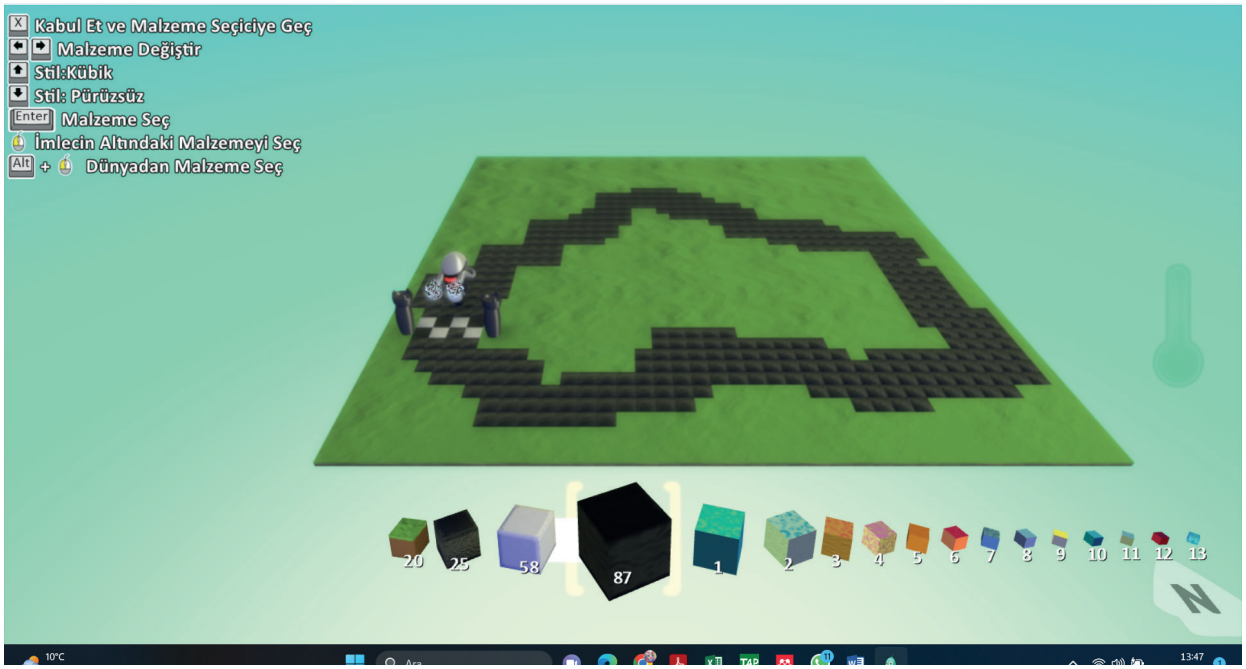
1. Yeni bir Dünya oluşturun.
2. Dünyanın tasarımını tamamlayın.
3. Nesne Aracı menüsünden oyun karakterimiz olan koduyu ekleyin.
4. Oyuncu karakterimiz olan koduyu programlamaya başlayın.

5. Kodunun nasıl hareket edeceğini kodlayın.
6. Kodunun yarış pistinde olduğu anlar için daha hızlı olmasını sağlayacak kodları yazın.
7. Kodunun yarış pistinde olmadığı anlar için daha yavaş olmasını sağlayacak kodları yazın.
8. Kodunun oyunu tamamlayarak bitiş çizgisine gelmesi durumunda oyunu kazandıran kodu yazın.



Görsel 1.6.1: Nesne Aracı menüsünden oyun karakterimiz olan kodu ve bitiş çizgisinin eklenmiş ve oyunumuzun tasarımının tamamlanmış halinin ekran görüntüsü

Oyunun tasarım aşamasının tamamlanması ve Görsel 1.6.1' de yer alan görünümü elde edebilmemiz için Görsel 1.6.2' de yer alan Zemin Fırçası ile istenen fırçalar ve yüzeyler seçilir.



Görsel 1.6.2: Oyunun tasarım aşaması için Zemin Fırçası ile istenen fırçalar ve yüzeyler seçimi ekran görüntüsü

Kodlar yazılırken döngü kullanımına dikkat çekilir. Görsel 1.6.3 de yer alan son kod olan, koduyu takip eden kameranın "hep" komutu ile oyun içerisinde sürekli görevine devam edeceği belirtilir. Döngüler ve döngülerin kullanımı hakkında öğrencilere bilgi verilir. Burada kullanılan döngünün sonsuz döngü türüne örnek olduğu vurgulanır. Öğrencilerden günlük hayattan döngüye örnek bir durum vermeleri istenir.

Görsel 1.6.3: Oyun karakterimiz olan kodunun hareket etme, hızlanma, yavaşlama, kazanma ve kamera tarafından takip edilme kodlarının ekran görüntüsü

Oyunun uygulayıcı ile birlikte tamamlanmasının ardından öğrencilere "*Siz olsaydınız neleri farklı yapardınız?*" sorusu sorulur ve cevaplar üzerinde konuşulur. "*Sizlerden az önce birlikte tamamladığımız tek kişilik motosiklet yarışı oyununun iki kişilik olan versiyonunu yapmanızı istiyorum. Kodunun biri sizin kontrolünüzde yarışırken diğer koduyu bilgisayar kontrol edecek şekilde kodlayacaksınız.*"

Yönerge:

1. Yeni bir Dünya oluşturun.
2. Nesne Aracı menüsünden oyun karakterimiz olan koduları ekleyin.
3. İstenirse Dünya genişletilebilir ve ağaçlar ya da farklı nesneler eklenerek tasarım zenginleştirilebilir.
4. Oyuncu karakterlerimiz olan koduları programlamaya başlayın.
5. Bizim kontrolümüzde olan kodunun nasıl hareket edeceğini kodlayın. Bu kez yön tuşları yerine WASD tuşlarını tercih edebilirsiniz.
6. Bilgisayar kontrolünde olacak olan kodunun kodlarını yazın. Burada koduya belirli bir yol çizmeli ve hız ayarını yapın.
7. Koduları kodlarken döngülerden yardım alın.
8. Oyunun hangi koşullar altında biteceğini ve puan ayarlamasını yapın.

Etkinliğin tamamlanmasının ardından öğretmen tarafından dersin hedeflerini vurgulayan, etkinliği özetleyen ve sonraki etkinlik hakkında kısa bilgi veren ders sonu konuşması yapılır:

"Çocuklar bugün sizinle döngü yapılarını öğrendik" denilir ve ders sonlandırılır.

DEĞERLENDİRME

Tablo 1.6.1: Kontrol listesi

Kontrol Listesi	Evet	Hayır
Verilen bir örnek durumdaki döngüleri tespit eder.		
Kodlamada döngüleri kullanır.		
Bir problemin çözümünde döngü yapısını kullanır.		

ETKİNLİK | 7

ETKİNLİK NO	**1.7**
ETKİNLİK ADI	OYUNUMU KODLUYORUM
SINIF/KADEME	İlkokul
SÜRE	40+40+40+40+40+40+40+40=320 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Programlamaya Giriş
KAZANIMLAR	**1.6.1.** Gerçek yaşam problemi için algoritmik çözüm üretir**.** **1.6.2.** Belirlenen gerçek yaşam problemini çözecek projeyi geliştirir.
TEMEL BECERİLER	Doğrudan anlatım, Soru-cevap, Eğitsel
YÖNTEM VE TEKNİKLER	Proje Tabanlı Öğrenme, Gösterip Yaptırma
ARAÇ-GEREÇLER	Bilgisayar, İnternet, Projeksiyon cihazı/ Etkileşimli tahta
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında blok tabanlı, üç boyutlu oyun geliştirmeye uygun uygulama geliştirme ortamı olması gerekir. ✓ **Ek 1.7.1:** Oyun Değerlendirme Formu
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma. ✓ Değişken kullanır. ✓ Koşul yapılarını bilir.

SÜREÇ

Dersin başında öğrencilere *"Bugün şu ana kadar öğrendiğiniz tüm bilgileri ve hayal gücünüzü kullanarak kendi oyununuzu tasarlayacağınız bir proje hazırlayacak ve arkadaşlarınıza projelerinizi sunacaksınız"* denilerek öğrencilerin hedeften haberdar edilmesi sağlanır.

Daha sonra öğrencilere şu sorular sorulur:

"Oynadığınız dijital oyunlar nelerdir? Bu oyunları oynamak size ne kazandırdı?"

Öğrencilerden gelen cevaplar dinlendikten sonra dijital oyun oynamanın olumlu etkileri hakkında kısaca bilgi verilir:

- ✓ *Verilen görevleri yerine getirip sorumluluk alabilme ve takip edebilme,*
- ✓ *Problem çözebilme yeteneği ve analitik düşünceye sahip olma,*
- ✓ *Motor yeteneklerinin gelişmesi ve koordinasyon yetenekleri,*
- ✓ *Lojistik gibi dağıtım ve kaynak araştırma yönetimi becerilerinin gelişmesi,*
- ✓ *Aynı zamanda birden çok iş yapabilme, takip edebilme ve yönetebilme,*
- ✓ *Pratik zekânın gelişmesi, analitik ve pratik düşünme, kararsızlığın azalması,*

- ✓ *Strateji oluşturabilme ve tahmin yürütebilme yeteneğinin artması,*
- ✓ *Okuma-yazma ve matematik becerilerinin gelişmesi,*
- ✓ Örüntü (*Pattern) tanıma,*
- ✓ *Tümevarımsal muhakeme yeteneğinin artması,*
- ✓ *Harita okuyabilme ve yön bulabilme yeteneklerinin gelişmesi,*
- ✓ *Neden sonuç ilişkisine bağlı karar verebilme,*
- ✓ *Takım çalışmasına uyum sağlayabilme ve işbirliği yapabilme yeteneklerinin artması,*
- ✓ *Dijital bilgi ve yeteneğinin artması,*
- ✓ *Eğlenceye bağlı olarak mutluluğun oluşturduğu avantajlar ve bir şeyleri başarma karşılığında aldığı ödül motivasyonun artışı,*
- ✓ *Yaratıcılık ve buna bağlı olarak özgüvenin artması,*
- ✓ *Sinir, stres, negatif düşünme gibi olumsuz dürtülerin azaltılması gibi güzel olumlu örnekler yer almaktadır (*Dinç, 2010, aktaran Soyöz-Semerci ve Balcı, 2020).

Bu açıklamalardan sonra öğrencilere ;

"Şimdi sıra sizde: yaratıcılığınızı, hayal gücünüzü ve oyun deneyimlerinizi kullanarak kendi oyunuzu tasarlamak ister misiniz?" denir.

Bu etkinlikte öğrencilerin önceki etkinliklerde öğrendikleri bilgileri kullanarak kendi keşiflerini artırmalarına yardımcı olacak bir proje çalışması hazırlamaları istenir. Proje konusu olarak öğrencilerin kurallarını ve tasarımını kendilerinin belirledikleri bir oyun yapmaları istenir. Öğrenciler projede tasarlayacakları oyunlarını bireysel ya da 2 kişilik gruplar halinde yapabilirler.

Proje yapılırken her grubun yapması gereken işlem adımları öğretmen tarafından öğrencilere açıklanır. Bu işlem adımlarının ayrıntılı olarak açıklanması önemlidir.

Proje yapımında takip edilmesi gereken işlem adımları:

- ✓ Projelerinde geliştirecekleri oyun ile ilgili aşağıdaki sorulara cevap verilerek oyunun tasarımına başlamaları istenir.

1. Oyunun amacı nedir?
2. Oyun nasıl bir dünyada gerçekleşmektedir?
3. Oyundaki karakterler kimlerdir?
4. Karakterler nasıl kontrol edilir?
5. Oyunun kuralları nedir

- ✓ Projede tasarlanacak oyun planının gözden geçirilmesi, yazılımda gerçekleştirilemeyecek işlem adımlarının çıkartılması, eksik kısımlar varsa eklenmesi.
- ✓ Projenin tasarlanması ve kodlanması
- ✓ Projenin çalıştırılması ve hatalar varsa giderilmesi
- ✓ Projelerin sunulması

Proje gruplarına çalışmaları için uygun ortam sağlanır ve gerekli süre verilir. Öğretmen, zaman zaman grupların yanlarına giderek bir sorun olmadığından emin olur ve gerekliyse destek ve yönlendirme sağlar. Çalışmaları konusunda ihtiyaç duyarlarsa zaman zaman arkadaşlarından destek almaları konusunda teşvik edilir. Öğrencilere uygulamayı tasarlamaları, kodları oluşturmaları için yeterli süre verilir. Öğretmen, öğrencilerin projeyi sürdürmeleri ve ürünle sonlandırmaları için onları motive eder. Süre sonunda projede tasarladıkları oyunun sunumuna geçilir.

Sınıf mevcuduna göre dersin bir bölümü öğrencilerin sunumlarına ayrılmalıdır. Sunumlar sonrasında gruplar, diğer grupların geliştirdiği oyunlarını Ek 1.7.1 de yer alan Değerlendirme formunu kullanarak puanlarlar. Arkadaşlarının değerlendirmesi sonucunda yüksek puan alan projeler ödüllendirilebilir.

Öğretmen öğrencilere:

"Çocuklar bu etkinlikte hepiniz kendi oyunlarınızı tasarlayıp kodladınız. Emek vererek hazırladığınız bu oyunları arkadaşlarınıza sundunuz." diyerek dersi sonlandırır.

DEĞERLENDİRME

Öğrencilerin aşağıdaki sorulara cevap vermesi sağlanır.

- ✓ İlk başta yapmayı hayal ettiğiniz proje nasıldı?
- ✓ Proje fikrinizi gerçekleştirirken değişiklikler yapmak durumunda kaldınız mı? Evetse neler değişti?
- ✓ Yapmakta zorlandığınız durumlar oldu mu?
- ✓ Oyununu geliştirmeye devam edecek misin? Edeceksen neler yapmak istersiniz?

EK 1.7.1: OYUN DEĞERLENDİRME FORMU:

	PUANLAMA				
	1	**2**	**3**	**4**	**5**
Oyunun adı güzeldi					
Oyun özgündü. (Daha önce oynamadığım bir oyundu)					
Oyunun amacı anlaşılırdı.					
Oyundaki karakterlerin tasarımı başarılı idi.					
Oyun dünyasının tasarımı başarılı idi.					
Oyunda kodlamalar doğru yapılmıştı.					
Oyunu oynamak keyifliydi.					
Oyun sunumu güzeldi.					
Toplam Puan:					

Oyunu iyileştirmek için önerileriniz:

KAYNAKÇA

Yeşilay. (2020). *Oyun bağımlılığı nelere yol açıyor?. https://www.yesilay.org.tr/tr/makaleler/* oyun-bagimliligi-nelere-yol-aciyor (Erişim tarihi: 17.04.2023).

Soyöz-Semerci, Ö. U. ve Balcı, E. V. (2020). Lise öğrencilerinde dijital oyun bağımlılığı üzerine bir alan araştırması: Uşak Örneği. *Journal of Humanities and Tourism Research, 10(3), 538-567. https://doi.org/10.14230/johut869*

ÖZEL EĞİTİM VE REHBERLİK HİZMETLERİ
GENEL MÜDÜRLÜĞÜ

BİLİM VE SANAT MERKEZLERİ

YAZ OKULU DESTEKLEME VE YETİŞTİRME KURSU PROGRAMI

YAZILIM ATÖLYESİ

ETKİNLİKLER

ORTAOKUL

TÜRKİYE CUMHURİYETİ MİLLÎ EĞİTİM BAKANLIĞI

ETKİNLİK NO	**2.1**
ETKİNLİK ADI	PROGRAMLAMA DİLİNE İLK ADIM
SINIF/KADEME	Ortaokul
SÜRE	40+40= 80 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Programlamaya Giriş
KAZANIMLAR	**2.1.1.** Farklı programlama dillerini kullanım alanlarına göre listeler.
TEMEL BECERİLER	Problem çözme, Analitik düşünme, Keşfetme
YÖNTEM VE TEKNİKLER	Doğrudan anlatım, Soru-cevap, Benzetim
ARAÇ-GEREÇLER	Bilgisayar, İnternet, Projeksiyon cihazı/ Etkileşimli tahta
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında yüklü ya da çevrimiçi kullanabileceği bir uygulama geliştirme ortamı olması gerekir. ✓ Programlama dili kavramlarına ilişkin farklı tanımlar öğretmen tarafından incelenir. ✓ Dersin dikkat çekme aşamasında kullanılmak üzere farklı dillerde yazılmış kod görselleri öğretmen tarafından sınıf ortamına getirilir. ✓ **Ek 2.1.1:** Kullanım alanları ve programlama dilleri
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma.

SÜREÇ

Dersin başında öğrencilere *"Arkadaşlar bugün sizlerle programlama dili kavramını ve programlama dillerinin çeşitlerinin öğreneceğiz", denilerek öğrencilerin hedeften haberdar edilmesi sağlanır.*

Öğretmen öğrencilerin dikkatlerini etkinlik içeriğine çekebilmek amacıyla öğrencilerine, *"Çocuklar sizce yazılımlar nasıl yapılır? Aranızda programlama dili kavramını daha önce duyan var mı?"* gibi sorular sorar. Öğrencilerden gelen cevaplar değerlendirilir, cevap alınamaması durumunda ise farklı dillerde yazılmış kod örnekleri ekrana yansıtılarak öğrencilere ipucu verilebilir.

Dikkat çekme sorularını takiben aşağıdaki sorularla etkinliğe giriş yapılır:

"Aranızda evinizin ihtiyaçlarını karşılamak için markete, manava giden var mı? Varsa bize bu süreci anlatabilir mi?" soruları yöneltilerek, öğrenci cevapları alınır. Gelen cevaplar Görsel 2.1.1.'deki gibi bir grafik haline getirilerek öğrencilerle paylaşılır.

Görsel 2.1.1. Mesaj transferini etkileyen unsurlar

Grafik öğrencilerle beraber incelenerek, öğrencilere "*Arkadaşlar sizce anne tarafından iletilen mesajın çocuk tarafından doğru bir şekilde anlaşılabilmesini sağlayan unsurlar nelerdir?*" sorusu yöneltilir. Öğrencilerden cevap gelmemesi durumunda öğretmen tarafından öğrencilere "mesajın anlaşılırlığını sağlayan en önemli unsurun ortak bir konuşma dili olduğu" ile ilgili ipucu verilir. Gelen cevaplara dönütler sağlandıktan sonra "annenin evin ihtiyaçlarını çocuğunun anlayabileceği bir dilde iletmesi gerektiği aksi takdirde çocuğun annesine yardımcı olmakta zorluk yaşayacağı" vurgulanır. Ardından öğrencilere *"bilgisayarlarla da konuşabilmek, istediğimiz bazı işlemleri bilgisayara yaptırabilmek için de bilgisayarın konuşma dili olan programlama dillerinden en az bir tanesine hâkim olmamız gerektiği"* belirtilir.

"Günümüzde bilgisayar ile iletişim kurmamızı sağlayacak yüzlerce programlama dili mevcut" denilir. Ardından öğrencilerden interneti kullanarak farklı programlama dillerini ve kullanım alanlarını araştırmaları istenir. EK 2.1.1 dağıtılarak, öğrencilerden kullanım alanlarına göre programlama dillerine örnekler vermeleri istenir. Araştırma sonunda EK 2.1.1 formu dolduran öğrencilerden formu sınıf ortamında arkadaşlarına sunmaları istenir.

Etkinliğin tamamlanmasının ardından öğretmen tarafından dersin hedeflerini vurgulayan, etkinliği özetleyen ve sonraki etkinlik hakkında kısa bilgi veren ders sonu konuşması yapılır:

"Çocuklar bugün sizinle farklı programlama dilleri ve bunların kullanım alanlarını öğrendik. Bir sonraki derste programlama dilinde temel girdi ve çıktı işlemlerini gerçekleştireceğiz." denilir ve ders sonlandırılır

DEĞERLENDİRME

✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma.

Tablo 2.1.1: Kontrol listesi

Kontrol Listesi	Evet	Hayır
Programlama dili kavramını açıklar.		
Programlama diline örnek verir.		
Farklı programlama dillerini kullanım alanlarına göre listeler.		

EKLER:

Ek 2.1.1- Kullanım alanları ve programlama dilleri

Kullanım alanları	Programlama dilleri
Web uygulamaları	
Mobil uygulamalar	
Gömülü sistemler	
Masaüstü uygulamaları	
Yapay zekâ uygulamaları	
Oyun tasarımı	
Siber güvenlik	

ETKİNLİK | 2

ETKİNLİK NO	**2.2**
ETKİNLİK ADI	GİRDİ-İŞLEM-ÇIKTI
SINIF/KADEME	Ortaokul
SÜRE	40+ 40= 80 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Programlamaya Giriş
KAZANIMLAR	**2.1.2.** Temel girdi-çıktı fonksiyonları ile işlem yapar.
TEMEL BECERİLER	Problem çözme, Analitik düşünme, Keşfetme
YÖNTEM VE TEKNİKLER	Doğrudan anlatım, Soru-cevap, Gösterip-yaptırma
ARAÇ-GEREÇLER	Bilgisayar, Uygulama geliştirme ortamı, İnternet, Projeksiyon cihazı/ Etkileşimli tahta
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında yüklü ya da çevrimiçi kullanabileceği bir uygulama geliştirme ortamı olması gerekir. ✓ Ders içerisinde kullanılacak Girdi, İşlem, Çıktı, Bilgisayar kavramlarına ilişkin farklı tanımlar öğretmen tarafından incelenir. ✓ Her öğrencinin bir bilgisayar karşına oturması sağlanır. ✓ **Ek 2.2.1:** Problemin uygulama geliştirme ortamında kod örneği ve ekran çıktıları
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma.

SÜREÇ

Dersin başında öğrencilere *"Arkadaşlar bilgisayarı olan var mı?", "Bilgisayar nedir? Bilgisayarı nasıl tanımlayabiliriz?"* soruları yöneltilerek, cevapları alınır. Alınan cevaplar ile öğrencilerin konu hakkındaki ön bilgileri tespit edilir. Ardından öğrencilere Görsel 2.2.1 gösterilerek, bilgisayarın; veriyi alan, işleyen ve çıktı üreten cihazlar olduğu üzerinde durulur.

Görsel 2.2.1 Bilgisayar temel işlem süreci

Dikkat çekme soruları ve Görsel 2.2.1'in öğrencilere sunulmasını takiben öğrencilere:

"Çocuklar, bilgisayarın temel işlem süreci içerisinde yer alan 'Veri', 'İşlem' ve 'Çıktı' kavramları yazılım geliştirme sürecinde de önemli kavramlardır." denilir. Ardından kavramlar öğretmen tarafından açıklanarak yazılım geliştirme sürecindeki önemi vurgulanır. Python programlama dilinde yapılan işlemler sonucunda elde edilen çıktıyı görebilmek amacıyla "print" fonksiyonu; kullanıcıdan veri alabilmek için ise "input fonksiyonunun kullanıldığı belirtilir. Ardından öğrenciler bilgisayar başına oturtularak "print" fonksiyonunu örnekleyebilmek için Kod 2.2.1 yazdırılarak kodları çalıştırmaları sağlanır. Tüm öğrencilerin Ekran çıktısı 2.2.1' deki gibi ekran çıktıları elde etmeleri beklenir.

Kod 2.2.1. Print fonksiyonu kodu

```
1. print(3+5)
2. print("Merhaba Dünya")
3. print(8/4)
4. print(9-6)
5. print(8*9)
```

Ekran çıktısı 2.2.1. Print fonksiyonu ekran çıktısı

```
1. 8
2. 'Merhaba Dünya'
3. 2.0
4. 3
5. 72
```

Görevi tamamlayan öğrencilere gerekli dönütler verildikten sonra, "input" fonksiyonunu örnekleyebilmek için öğrencilere Kod 2.2.2 yazdırılarak kodları çalıştırmaları istenir. Tüm öğrencilerin Ekran çıktısı 2.2.2'deki gibi ekran çıktıları elde etmeleri beklenir.

Kod 2.2.2. Input fonksiyonu kodu

```
1. input("Adınızı giriniz:")
2. input("Soyadınızı giriniz:")
3. input("Yaşınızı giriniz:")
```

Ekran çıktısı 2.2.2. Input fonksiyonu ekran çıktısı

```
1. Adınızı giriniz: Aras
2. Soyadınızı giriniz: Yorulmaz
3. Yaşınızı giriniz:6
```

Görevi tamamlayan öğrencilere gerekli dönütler verildikten sonra öğrencilerden, kullanıcıların pizza malzemelerini girebilecekleri sipariş yazılımı yapmaları istenir. Öğrenciler ile yapılacak yazılımın genel özellikleri paylaşılır.

1. Eklemek istediğiniz birinci malzemenin ismini giriniz.
2. Eklemek istediğiniz ikinci malzemenin ismini giriniz.
3. Eklemek istediğiniz üçüncü malzemenin ismini giriniz.
4. Ekrana "**'Siparişiniz alındı. Sadece 30 dk. Sonra pizzanız hazır hale gelecektir.' olacaktır**" yazdır.

Yazılımın genel özelliklerini inceleyen öğrencilere soruları olup olmadığı sorulur ve ardından:

"Şimdi sıra geldi kodlamaya..." denilerek öğrencilerden programı kodlaması istenir. Yardım edilmeden algoritmasını ve programını doğru olarak tamamlayanlara küçük bir ödül verilebilir (çikolata veya gofret gibi). Algoritma kısmında eksik veya hatalı olan kısımlar düzeltilmez; öğrencinin kod yazma sırasında hatasını fark etmesi sağlanır; hatasını fark edemez ise, yönlendirmelerle hatasını düzeltmesi sağlanır.

Pizza oluşturma deneyiminden yola çıkarak verilen örnekteki girdi ve çıktı kavramları açıklanarak etkinlik tamamlanır.

Etkinliğin tamamlanmasının ardından öğretmen tarafından dersin hedeflerini vurgulayan, etkinliği özetleyen ve sonraki etkinlik hakkında kısa bilgi veren ders sonu konuşması yapılır:

"Çocuklar bugün sizinle temel girdi-çıktı fonksiyonları ile işlemleri öğrendik ve uygulamalar yaptık. Bir sonraki derste değişken kavramını öğrenip, kodlamada değişkenleri kullanacağız." denilir ve ders sonlandırılır.

DEĞERLENDİRME

Tablo 2.2.1: Kontrol listesi

Kontrol Listesi	Evet	Hayır
Girdi kavramını açıklar.		
İşlem kavramını açıklar.		
Çıktı kavramını açıklar.		
Girdi ve çıktı fonksiyonlarını kullanarak kodlama yapar.		

ETKİNLİK NO	**2.3**
ETKİNLİK ADI	HAMBURGER SİPARİŞİ
SINIF/KADEME	Ortaokul
SÜRE	40 +40= 80 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Değişkenler
KAZANIMLAR	**2.2.1.** Değişken kavramını açıklar. **2.2.2.** Kodlamada değişkenleri kullanır.
TEMEL BECERİLER	Problem çözme, Analitik düşünme, Keşfetme
YÖNTEM VE TEKNİKLER	Anlatım, Soru Cevap, Tartışma, Grup çalışması
ARAÇ-GEREÇLER	Bilgisayar, Uygulama geliştirme ortamı,
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında yüklü ya da çevrimiçi kullanabileceği bir uygulama geliştirme ortamı olması gerekir. ✓ Ders içerisinde kullanılacak Veri, Bilgi, Değişken, Sabit kavramlarına ilişkin farklı tanımlar öğretmen tarafından incelenir. ✓ Her öğrencinin bir bilgisayar karşına oturması sağlanır. ✓ **Ek 2.3.1:** Problemin uygulama geliştirme ortamında kod örneği ve ekran çıktıları
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma. ✓ Öğrenciler bir programlama dilinde var olan temel girdi ve çıktı fonksiyonlarının kullanımını bilir.

SÜREÇ

Dersin başında öğrencilere *"Daha önce hiç hamburger siparişi verdiniz mi?", "En beğendiniz hamburger çeşidi hangisidir?" ve "Bir hamburgerden diğerine değişen özellikler nelerdir?", "Bir hamburgerden diğerine değişmeyen özellikler nelerdir"* soruları yöneltilerek, cevapları alınır.

Dikkat çekme sorularını takiben aşağıdaki senaryo anlatılır:

"Zeynep Bade, kardeşi Aras için bir sürpriz yapmak istemektedir. Aras'ın seveceğini düşündüğü bir hamburger sipariş vermeye karar verir. Hamburgerciye sipariş vermek için gittiğinde, hamburgerin yapımında kullanılacak bazı malzemelerin sabit olduğunu, bazı malzemelerin ise isteğe bağlı olduğunu ve 3 adet zorunlu malzeme seçmesi gerektiğini öğrenir. Zeynep Bade'ye sipariş vermesinde ve seçim yapmasında yardımcı olabilir misiniz?" sorusu sorulur. Tahtaya hamburger için sabit olan ve isteğe bağlı olan malzemeler öğrencilerden de fikir alınarak yazılır. Tercihen Tablo 2.3.1' de yer alan liste projeksiyondan yansıtılabilir.

Tablo 2.3.1. Hamburger malzeme listesi

Hamburger İçin Sabit Malzemeler	Hamburger İçin İsteğe Bağlı Malzemeler
Hamburger ekmeği	**Tavuk köfte**
	Et köfte
	Mantar
	Kaşar
	Marul
	Turşu
	Hardal
	Ketçap
	Mayonez

Bu aşamada öğrenciler, hamburger sipariş programı için bir algoritma yazmaya yönlendirilir. Öğrencilerin yazması gereken algoritmanın aşağıdaki adımları içermesi gereklidir.

1. Başla.
2. Eklemek istediğiniz birinci malzemenin ismini giriniz.
3. Eklemek istediğiniz ikinci malzemenin ismini giriniz.
4. Eklemek istediğiniz üçüncü malzemenin ismini giriniz.
5. Ekrana "'**Hamburgerinizin içinde birinci malzeme ismi, ikinci malzeme ismi ve üçüncü malzeme ismi'** olacaktır" yazdır.
6. Bitir.

Algoritma adımlarını tamamladıktan sonra öğrencilere:

"Algoritmaları oluşturduğunuzu görüyorum. Şimdi sıra geldi kodlamaya..." denilerek öğrencilerden programı kodlaması istenir. Yardım edilmeden algoritmasını ve programını doğru olarak tamamlayanlara gerekli dönütler verilir. Algoritma kısmında eksik veya hatalı olan kısımlar düzeltilmez; öğrencinin kod yazma sırasında hatasını fark etmesi sağlanır; öğrencinin hatasını fark edemediği durumlarda, yönlendirmelerle hatasını düzeltmesi sağlanır.

Hamburger oluşturma deneyiminden yola çıkarak verilen örnekteki değişken ve sabit kavramları açıklanır. Öğretmen tarafından günlük hayat örnekleri üzerinden sabit ve değişken değerlere örnek verilir. *"Örneğin: otobüs bir varlık, yolcu sayısı ise bir duraktan diğerine veya bir otobüsten diğerine (yolcu sayısının artıp azalması) değişen bir özelliktir"* denilerek örneklendirilir. Ardından öğretmen öğrencilere, *"Şimdi sıra sizlerde değişken ve sabit kavramlarına örnekler vererek bu kavramları bana açıklayabilir misiniz?"* der ve gelen cevaplara uygun olarak öğretmen öğrencilerine dönütler sağlayarak etkinliği sonlandırır.

Etkinliğin tamamlanmasının ardından öğretmen tarafından dersin hedeflerini vurgulayan, etkinliği özetleyen ve sonraki etkinlik hakkında kısa bilgi veren ders sonu konuşması yapılır:

"Çocuklar bugün sizinle değişken kavramını öğrendik ve uygulamalar yaptık. Bir sonraki derste operatör kavramını öğreneceğiz." denilir ve ders sonlandırılır.

DEĞERLENDİRME

Öğrencilerin kavramları içselleştirmeleri için günlük yaşamdan sabit ve değişkenlere örnekler vermeleri istenir. Daha sonra öğrencilerden değişken ve sabit kavramlarını açıklamaları beklenir. Kavram yanılgıları varsa düzeltilir.

Öğrencilere *"Acaba değişken kullanmadan pizza siparişi ile ilgili programı yazabilir miydik?", "Sizce değişkenler bir programlama dilinde program yazmak için gerekli midir? Neden?" soruları* yöneltilir. Bu sorular öğrencilerin değişken kullanımının gerekliliğini fark edip, etmediklerini anlamak için önemlidir.

Öğretmen, öğrencilerden gelen cevaplara göre ek etkinlik yapılabilir.

Yapılan kodlamalar ve soruya verilen cevaplar Tablo B.2.3.1'de verilen kontrol listesi kullanılarak değerlendirilir.

Tablo 2.3.1: Kontrol listesi

Kontrol Listesi	Evet	Hayır
Değişken kavramını açıklar.		
Sabit kavramını açıklar.		
Günlük yaşamdan değişkenlere örnekler verir.		
Günlük yaşamdan sabitlere örnekler verir.		
Sabit ve değişkenleri kullanarak kodlama yapar.		

EKLER:

EK 2.3.1: Problemin uygulama geliştirme ortamında kod örneği ve ekran çıktıları

Algoritmanın kodlanmış örneği Kod 2.1.2'de, ekran çıktısı ise Ekran çıktısı 2.2.1'de verilmiştir.

Kod 2.3.1. Algoritmanın kodlanmış örneği

```
1. secenek1=input("Hamburgerinize eklemek istediğiniz birinci malzemeyi giriniz:")
2. secenek2=input("Hamburgerinize eklemek istediğiniz ikinci malzemeyi giriniz:")
3. secenek3=input("Hamburgerinize eklemek istediğiniz üçüncü malzemeyi giriniz:")
4. print("Hamburgeriniz:", secenek1, ",", secenek2, "ve", secenek3, "malzemelerinden oluşturulacaktır.")
```

Ekran çıktısı 2.3.1. Print fonksiyonu ekran çıktısı

```
1. Hamburgerinize eklemek istediğiniz birinci malzemeyi giriniz: et köfte
2. Hamburgerinize eklemek istediğiniz ikinci malzemeyi giriniz: ketçap
3. Hamburgerinize eklemek istediğiniz üçüncü malzemeyi giriniz: hardal
4. Hamburgeriniz: et köfte, ketçap ve hardal malzemelerinden oluşturulacaktır.
```

ETKİNLİK NO	**2.4**
ETKİNLİK ADI	OPERATÖRLER
SINIF/KADEME	Ortaokul
SÜRE	40 +40= 80 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Operatörler
KAZANIMLAR	**2.3.1.** Operatör kavramını açıklar **2.3.2.** Aritmetiksel, mantıksal ve karşılaştırma operatörlerini kullanır.
TEMEL BECERİLER	Problem çözme, Analitik düşünme, Keşfetme
YÖNTEM VE TEKNİKLER	Doğrudan anlatım, Soru-cevap, Gösterip-yaptırma
ARAÇ-GEREÇLER	Bilgisayar, Uygulama geliştirme ortamı
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında yüklü ya da çevrimiçi kullanabileceği bir uygulama geliştirme ortamı olması gerekir. ✓ Her öğrencinin bir bilgisayar karşına oturması sağlanır. ✓ Her öğrencinin kendi algoritmasını ve kodlamasını yapması gerekmektedir. **Operatörler:** Programlama dillerinde aritmetiksel, mantıksal veya karşılaştırma ile ilgili bir görevi gerçekleştirmek için önceden tanımlanmış özel sembol veya karakterlerdir.

Operatör Çeşidi	Sembol
İşaret operatörleri	+, -
Aritmetik operatörler	+, -, *, /, %,
Mantıksal operatörler	&, |, ^, !, ~ ,&&, ||, true, false
String birleştirme operatörü	+
Arttırma / Azaltma	++, --
Kaydırma operatörü	<<, >>
Karşılaştırma operatörleri	==, !=,<,>, <=, >=
Atama operatörleri	=, +=, -=, *=, /=, %=, &=, |=, ^=, <<=, >>=

	İşaret Operatörleri: Tanımlanan değerlerin negatif veya pozitif olduğu belirlemek için kullanılır. **Aritmetik operatörler:** Matematiksel işlemlerin yapılmasında kullanılan operatörlerdir. **Mantıksal operatörler:** Koşullu ifadelerin birlikte değerlendirilmesi gerektiği durumlarda kullanılır. **Atama Operatörleri:** Değişkenlere değer ataması işlemlerinde kullanılır. Kullanıcının klavyeden girilen değeri değişkene atama işlemini yapabileceği gibi başka bir işlemin sonucunun başka bir değişkene atama işlemi de gerçekleşebilir. **Karşılaştırma Operatörleri:** Birden fazla değişkeni karşılaştırma yapmak amacıyla kullanılır.
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma. ✓ Temel düzeyde programlama bilgisi (değişken atamaları ve girdi-çıktı işlemleri)

SÜREÇ

Dersin başında öğrencilere temel matematik işlemlerinin neler olduğu sorulur. *"Arkadaşlar matematiksel hesaplamaları gerçekleştirirken temelde hangi işlemleri kullanıyorsunuz? Matematiksel işlemlerinizin arasında en sık kullandığınız 6 işlem hangisidir?"* soruları yöneltilerek, cevapları alınır. Alınan cevaplar üzerine ön bilgileri tespit edilir.

Dikkat çekme sorularının sorulmasının ardından öğrencilerden cevaplar alınır. Cevaplar üzerinde konuşulur. Daha sonra öğretmen programlamada kullanılan operatörlerin neler olduğunu açıklar ve her operatöre birer örnek verilir. Uygulayıcı programlamada kullanılan operatörlerin nasıl kullanıldığını Python programlama dilinde uygulamalı olarak gösterir.

Öğretmen öğrencilerden aritmetik operatörlere örnek olarak aşağıdaki basit kodları Python programlama ortamında yazmalarını ister. Öğrencilerin kodu çalıştırmalarının ardından verilen ekran çıktıları ile aynı sonuçları görmeleri beklenir.

Görev sırasında öğrencilere her bir aşamada gerekli dönütler verilir ve görevi tamamlanmasının ardından bir sonraki operatör işlemine geçilir.

Kod 2.4.1 Toplama İşlemi Kodu

```
a=10
b=25
print(a+b)
```

Ekran Çıktısı 2.4.1. Toplama İşlemi Ekran Çıktısı

```
35
```

Kod 2.4.2 Çıkarma İşlemi Kodu

```
x=103
y=25
print(x-y)
```

Ekran Çıktısı 2.4.2. Toplama İşlemi Ekran Çıktısı

```
78
```

Öğretmen öğrencilere yukarıda basit kod parçacıklarını verdikten sonra *"Arkadaşlar yukarıda ki kod parçacıklarında programlamada kullanılan aritmetik işlemlerden toplama ve çıkarma işlemlerinin nasıl kullanıldığını gördünüz.*

Şimdi *yeni göreviniz; kullanıcının klavyeden 2 sayı girdiği ve ardından girilen sayıların 'toplama', 'çıkarma', 'çarpma, 'bölme', 'tam bölüm' ve 'kalan' işlemlerini yapmanızı istiyorum ve yazılan kod parçacıklarının üstüne açıklama satırı olarak hangi operatörün ne amaçla kullanıldığını yazınız."* der.

Ardından öğrencilerden istenen kodları Python programlama ortamına yazmaları istenir. İşlemi tamamlayan öğrenciler kontrol edilir ve gerekli dönütler öğrencilere verilerek tüm öğrencilerin süreci tamamlamaları sağlanır.

Kod 2.4.3 Çıkarma İşlemi Kodu

```
a = int(input ('a sayısını giriniz '))
b = int(input ('b sayısını giriniz '))
toplam = a + b
fark = a - b
carpim = a * b
bolum = a // b
tambolum = a / b
kalan = a % b
print(toplam)
print(fark)
print(carpim)
print(bolum)
print(tambolum)
print(kalan)
```

Ekran Çıktısı 2.4.3. Toplama İşlemi Ekran Çıktısı

```
a sayısını giriniz 20
b sayısını giriniz 5
25
15
100
```

```
4
4.0
0
```

Etkinliğin tamamlanmasının ardından öğretmen süreç sonunda öğrencilerden operatörler kavramının kullanılabileceği farklı uygulama alanları ile alakalı örnek vermelerini ister. Kavram yanılgıları var ise bunlar düzeltilir. Gelen cevaplar üzerine ek etkinlik planı yapılabilir. Öğretmen bir sonraki derste *aritmetiksel, mantıksal ve karşılaştırma operatörlerin kullanımını öğreneceğiz* der ve ders sonlandırılır.

DEĞERLENDİRME

Öğretmenin süreç içerisinde sorduğu sorulara verilen cevaplar ve yapılan uygulamadan alınan çıktılara göre Tablo 2.4.1.'de verilen kontrol listesi kullanılarak öğrenciler değerlendirilir.

Tablo 2.4.1: Kontrol listesi

Kontrol Listesi	Evet	Hayır
Operatör kavramını açıklar.		
Aritmetik operatörler kavramını açıklar.		
Aritmetik operatörlere örnekler verir.		

KAYNAKÇA

Algan, S. (2008). Her Yönüyle C#. İstanbul: Pusula Yayıncılık.

Ergün, E. (2022). *Programlama Eğitiminde Soru Tabanlı Sistem Yaklaşımı.* Ankara: Pegem Akademi.

ETKİNLİK NO	**2.5**
ETKİNLİK ADI	PROGRAMLAMADA OPERATÖRLERİN KULLANIMI
SINIF/KADEME	Ortaokul
SÜRE	40+40+40+40=160 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Operatörler
KAZANIMLAR	**2.3.2.** Aritmetiksel, mantıksal ve karşılaştırma operatörlerini kullanır.
TEMEL BECERİLER	
YÖNTEM VE TEKNİKLER	Doğrudan anlatım, Soru-cevap, Gösterip-yaptırma
ARAÇ-GEREÇLER	Bilgisayar, Uygulama geliştirme ortamı
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında yüklü ya da çevrimiçi kullanabileceği bir uygulama geliştirme ortamı olması gerekir. ✓ Her öğrencinin bir bilgisayar karşına oturması sağlanır. ✓ Her öğrencinin kendi algoritmasını ve kodlamasını yapması gerekmektedir. **Operatörler:** Programlama dillerinde aritmetiksel, mantıksal veya karşılaştırma ile ilgili bir görevi gerçekleştirmek için önceden tanımlanmış özel sembol veya karakterlerdir. İşlem basamakları oluşturulurken operatörlerden yararlanılır. Toplama işleminde iki sayının toplamını almak için, bölümden kalanı bulmak için veya tam bölüm işlemini bulmak için üssünü veya karekökünü hesaplamak için gibi örneklendirebiliriz. Ya da verilen sayıların hangisinin büyük hangisinin küçük olduğuna karar vermek ya da verilen sayıların aralarındaki eşitliği bulmak için kullanıldığını söyleyebiliriz. Operatörler istenen işlemlerin yerine getirilmesi sırasında girdilere ihtiyaç duyarlar. Örneğin “+” operatörünün toplama yapabilmesi için iki tane sayıya ihtiyaç duyması buna örnek verilebilir. Bu sayılara da operand(sabit veya değişken) denilmektedir. Operandalar(sabit veya değişken) ilgili operatörler tarafından işlem sırasında kullanılan değerlerdir. Operatörler işlevlerine göre 6 kısımda incelenebilir. Bunlar; ✓ Aritmetik operatörler. ✓ Karşılaştırma operatörleri. ✓ Bitsel operatörler.

	✓ Mantıksal operatörler. ✓ Atama ve işlemli atama operatörleri. ✓ Özel amaçlı operatörler.
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma. ✓ Temel düzeyde programlama bilgisi (değişken atamaları, girdi-çıktı işlemleri, operatör kavramı)

SÜREÇ

Dersin başında öğrencilere fiziksel veya dijital ortamda bir hesap makinesi gösterilerek operatörlerin hesap makinelerinde nasıl kullanıldığını sorar. Alınan cevaplar üzerine ön bilgiler hatırlatılır ve etkinliğin uygulamasına geçilir.

Öğretmen tanımlarını verdiği operatörler ile alakalı örnekler vererek etkinliğine başlar.

Kod 2.5.1. Mantıksal Operatör Kodu

```
x = 5
sonuc = not(x > 3 and x < 10)
print(sonuc)
```

Ekran Çıktısı 2.5.1. Kodun Ekran Çıktısı

```
False
```

```
Kod 2.5.2. Mantıksal Operatör Kodu
x = 12
sonuc = not(x > 3 and x < 10)
print(sonuc)
```

Ekran Çıktısı 2.5.2. Kodun Ekran Çıktısı

```
True
```

Öğrencilere mantıksal operatörlerin kullanımına yönelik kodlar verilir ardından öğrencilerinde mantıksal operatörlerin kullanılabileceği alanlara yönelik örnek vermelerini ve ilgili kodları çalıştırmaları istenir.

Kod 2.5.3. Karşılaştırma Operatör Kodu

```
x=3
y=3
sonuc=(x!=y)
print(sonuc)
```

Ekran Çıktısı 2.5.3. Kodun Ekran Çıktısı

```
False
```

Kod 2.5.4. Karşılaştırma Operatör Kodu

```
x=3
y=3
sonuc=(x==y)
print(sonuc)
```

Ekran Çıktısı 2.5.4. Kodun Ekran Çıktısı

```
True
```

Öğrencilere karşılaştırma operatörlerinin kullanımına yönelik kodlar verilir ardından öğrencilerinde karşılaştırma operatörlerinin kullanılabileceği alanlara yönelik örnek vermelerini ilgili kodları çalıştırmaları istenir.

Operatörler ile alakalı örnek uygulamalar yapıldıktan sonra öğrencilere aşağıdaki senaryo okunur.

"Bilgehan ve Yiğit Ankara'da bir üniversitede bilgisayar mühendisliği bölümünde okumaktadırlar. Okudukları bölüm itibari ile alanları ile alakalı sürekli fikir alışverişinde bulunurlar. Bilgehan ve Yiğit ailelerinin kendilerine gönderdikleri paraları çekmek için ATM'ye giderler. ATM'ye para çekmeye geldikleri zamanlarda ATM'nin kendilerine sürekli olarak farklı adette banknot verdiklerini fark ederler. Yine bir gün ATM'ye para çekmeye gittiklerinde aralarında şöyle bir konuşma geçer: İkimizde aynı tutarı çekelim bakalım banknot adetleri yine mi farklı gelecek. İkisi de 385 TL para çekerler ATM Bilgehan'a 7 adet banknot vermişken Yiğit'e ise 12 adet banknot vermiştir(ATM'de yeteri kadar tüm banknotların olduğu varsayılmıştır). Yiğit Bilgehan'a bankamatikten bu parayı en az kaç banknotla alabileceklerini sorar bunun üzerine çekilen tutarın 6 adet banknotla tamamlanması gerektiğini hesaplarlar. Bunun üzerine Bilgehan ve Yiğit bu hafta bölümde programlama dersinde gördükleri operatörler konusunda öğrendikleri bilgi ile bu sorunu çözebileceklerini düşünürler. Oluşturdukları algoritma ile bu soruna ATM'ye girilen tutarı en az banknot verecek şekilde çözüm önerisi getirecek bir yazılım geliştirirler."

Öğretmen senaryoyu okuduktan sonra Bilgehan ve Yiğit'in geliştirdikleri yazılımın bir benzerini kendilerinin geliştirmelerini ister ve her öğrenciyi bir bilgisayara oturtur.

Önce öğrencilerin en az banknotu verecek algoritmayı oluşturmalarını ister. Bu aşamada öğrencilere şöyle bir ipucu verebilir; "***en az banknotu alabilmek için tutar girildikten sonra 200'e bölüm sonucu oluşan sayının tam kısmı (noktadan önceki kısmı) alınmalıdır. Bunun için "//"(tam bölüm) operatörü kullanılabilir. Örneğin 1300/200= 6,5 olmasına rağmen, 1300//200=6 sonucunu verecektir. Ayrıca her aşamadaki banknot adedi hesaplandıktan sonra kalan ödeme için işlemler devam ettirilmelidir."*** Ardından süreci adım adım takip ederek ilgili noktalarda öğrencilere dönütler verir ve sürecin tamamlanmasını sağlar.

Kod 2.5.5. ATM En Az Banknot Kodu

```
cekilecektutar = int(input("Ödenecek tutarı giriniz "))
ikiyuz = cekilecektutar // 200
tutar = cekilecektutar - ikiyuz*200
```

```
yuz = tutar // 100
tutar = tutar-yuz*100
elli = tutar // 50
tutar = tutar - elli*50
yirmi = tutar // 20
tutar = tutar - yirmi*20
on = tutar // 10
tutar = tutar - on*10
bes = tutar // 5
bir = tutar - bes*5
print("cekilecektutar 200lük banknot=",ikiyuz)
print("cekilecektutar 100lük banknot=",yuz)
print("cekilecektutar 50lik banknot=",elli)
print("cekilecektutar 20lik banknot=",yirmi)
print("cekilecektutar 10luk banknot=",on)
print("cekilecektutar 5lik banknot=",bes)
print("cekilecektutar 1lik para=",bir)
```

Ekran Çıktısı 2.5.5. ATM En Az Banknot Kodu Ekran Çıktısı

```
Ödenecek tutarı giriniz 385
cekilecektutar 200lük banknot= 1
cekilecektutar 100lük banknot= 1
cekilecektutar 50lik banknot= 1
cekilecektutar 20lik banknot= 1
cekilecektutar 10luk banknot= 1
cekilecektutar 5lik banknot= 1
cekilecektutar 1lik para= 0
```

Öğretmen oluşturulan algoritma ile yazılan kodları öğrencilere anlatır ve ipucunda vermiş olduğu "//" operatörünün kullanımını açıklar.

Burada kullanılan "//" operatörü iki sayının bölümü sonucu bölümün tamsayı kısmını verir. Örneğin 970 TL çekilmek istendiğinde ikinci satırdaki hesaplama sonucu ***ikiyuz*** *değişkeninin değeri 4 olarak bulunur (970/200=4,85 olsa da "//" operatörü bize 4 rakamını verecektir). Üçüncü satırda ise ödenen miktar yani 4x200=800 TL* ödencek *toplam miktardan düşülmüştür. Böylelikle 170 TL ödenecek şekilde program devam edecektir. Diğer satırlardaki hesaplamalarda da aynı mantık kullanılmıştır.*

Öğretmen kodları açıkladıktan sonra ekran çıktısını öğrencilere gösterir ve öğrencilerin de kendi bilgisayarlarında yazdıkları kodları çalıştırmalarını ister.

Etkinliğin tamamlanmasının ardından öğretmen süreç sonunda öğrencilerden aritmetiksel, mantıksal ve karşılaştırma operatörlerinin kullanılabileceği farklı uygulama alanları ile alakalı örnek vermelerini ister. Kavram yanılgıları var ise bunlar düzeltilir. Gelen cevaplar üzerine ek etkinlik planı yapılabilir. Öğretmen bir sonraki derste "*koşul yapılarının kullanımını öğreneceğiz*" der ve ders sonlandırılır.

DEĞERLENDİRME

Öğretmenin süreç içerisinde sorduğu sorulara verilen cevaplar ve yapılan uygulamadan alınan çıktılara göre Tablo 2.5.1.'de verilen kontrol listesi kullanılarak öğrenciler değerlendirilir.

Tablo 2.5.1: Kontrol listesi

Kontrol Listesi	Evet	Hayır
Aritmetiksel operatörleri açıklar.		
Mantıksal operatörleri açıklar.		
Karşılaştırma operatörlerini açıklar.		
Verilen probleme yönelik çözüm önerileri sunar.		
Verilen probleme yönelik aritmetiksel, mantıksal ve karşılaştırma operatörlerini kullanarak bir yazılım gerçekleştirir.		
Geliştirdiği yazılımı test eder.		

KAYNAKÇA

Algan, S. (2008). Her Yönüyle C#. İstanbul: Pusula Yayıncılık.

Ergün, E. (2022). *Programlama Eğitiminde Soru Tabanlı Sistem Yaklaşımı.* Ankara: Pegem Akademi.

ETKİNLİK | 6

ETKİNLİK NO	**2.6**
ETKİNLİK ADI	KARAR KONTROL YAPILARI (KOŞULLU İFADELER)
SINIF/KADEME	Ortaokul
SÜRE	40+40=80 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Koşul Yapıları
KAZANIMLAR	**2.4.1.** Koşul yapılarını bir problemin çözümünde kullanır.
TEMEL BECERİLER	Problem çözme, Analitik düşünme, Keşfetme
YÖNTEM VE TEKNİKLER	Doğrudan anlatım, Soru-cevap, Gösterip-yaptırma
ARAÇ-GEREÇLER	Bilgisayar, Uygulama geliştirme ortamı
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında yüklü ya da çevrimiçi kullanabileceği bir uygulama geliştirme ortamı olması gerekir. ✓ Her öğrencinin bir bilgisayar karşına oturması sağlanır. ✓ Her öğrencinin kendi algoritmasını ve kodlamasını yapması gerekmektedir. **Koşullu İfadeler:** Programlama dillerinde herhangi bir karar kontrol yapısı kullanmadan tüm işlem satırları çalıştırmak mümkündür. Fakat bazı durumlarda istenen durumlara özel koşullar sağlandığında ilgili satırların çalışmasını sağlayacak yapılara ihtiyacımız olur. Buradaki koşul yapılarına karar kontrol yapıları adı verilir. Bu yapılar tanımlanan koşullar sağlandığında veya sağlanmadığında ilgili satırın çalışmasını veya çalışmamasını belirleyecek olan yapılardır. Karar kontrol yapıları: ✓ If deyimi ✓ Else If (Elif) deyimi ✓ Else deyimi ✓ İç İçe If yapıları olarak sıralanabilir. **If Deyimi:** If deyimi bir programın akışını kontrol etmek için kullanılır. Belirli bir koşula göre yapılması istenen işlemler, If deyimi kullanılarak gerçekleştirilir. Koşul, doğru ya da yanlış değer alabilen mantıksal bir ifadedir. **Else If (Elif) Deyimi:** Koşullu ifadelerde zaman zaman birden fazla koşula bağlı durumlar olabilir. Aynı anda sadece bir tek koşulun sağlanabileceği bu gibi durumlarda, her bir koşulu ayrı ayrı yazdığımız Else If yapılarını kullanabiliriz.

	Else Deyimi: Koşul yapılarında If şartının sağlanmadığı durumda gerçekleştirilecek işlemleri belirlemek için kullanılır. "Aksi hâlde" anlamındadır. İç içe **If Yapıları:** Bir if yapısının içine tekrardan bir if ekleyerek oluşturulan yapılardır. Burada dikkat edilmesi gereken nokta, içerideki if yapısının dışardaki if yapısının koşulunun sağlanma durumuna bağlı olduğudur. Program eğer dıştaki if döngüsüne girmezse içteki if döngüsüne bakılmaksızın atlanır ve program akışı dıştaki if yapısının sonlandığı yerden devam eder
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma. ✓ Temel düzeyde programlama bilgisi (değişken atamaları, girdi-çıktı işlemleri, operatör kavramı, aritmetiksel, mantıksal ve karşılaştırma operatörleri)

SÜREÇ

Öğretmen dersin başında öğrencilere sınıfta bulunan klimayı göstererek klimanın çalışma modları üzerine birkaç soru yöneltir. "*Arkadaşlar sizce sınıfta bulunan klimanın çalışma modları neler olabilir?*" sorusunu yönelttikten sonra gelen cevaplar arasından otomatik mod cevabının gelmesi beklenir. Otomatik mod cevabı gelmemiş ise ipucu vererek ilgili cevabı vermeleri sağlanır. Öğretmen ardından klimanın otomatik modunun nasıl çalıştığı hususunda bilgiler verir. Burada klimanın otomatik modda iken belirli bir koşula bağlı olarak çalıştığı üzerinde durulur ve etkinliğe geçilir.

Öğretmen dikkat çekme aşamasından sonra karar kontrol yapıları hakkında kavramsal bilgileri detaylı olarak öğrencilere aktarır. Her öğrencinin bir bilgisayar başına oturtulması sağlandıktan sonra karar kontrol yapılarından **if, elif ve else** komutlarının nasıl kullanıldığı hakkında küçük bir örnek uygulama yaptırır.

Kod 2.6.1 Koşul Yapıları Kodu

```
bilgehanyas = int(input("Bilgehan'ın yaşını giriniz:"))
yigityas = int(input("Yiğit'in yaşını giriniz:"))
if yigityas > bilgehanyas:
   print("Yiğit, Bilgehandan'dan büyüktür.»)
elif bilgehanyas == yigityas:
   print("Bilgehan ile Yiğit'in yaşları aynıdır.")
else:
    print("Bilgehan, Yiğit'ten büyüktür")
```

Verilen kod sayesinde Bilgehan ve Yiğit'in yaşlarının karşılaştırması yapılır, girilen yaşların koşul yapılarına göre ilgili işlem satırlarının işletilmesi ile sonucun ekrana yazdırılması sağlanır.

Öğretmen kodlar ile ilgili gerekli açıklamaları yaptıktan sonra öğrencilerin kodları çalıştırmasını ister ve ekran çıktılarının sonuçları üzerine öğrenciler ile konuşur. Öğretmen bu aşamada öğrenciler kodları yazarken değişkenler, değişkenlerin kullanımı, operatörler ve kullanımı hakkında kısa hatırlatmalar yapabilir.

Ekran Çıktısı 2.6.1. Koşul Yapıları Kodunun Ekran Çıktısı

```
Bilgehan'ın yaşını giriniz:20
Yiğit'in yaşını giriniz:21
Yiğit, Bilgehandan'dan büyüktür.
```

Ekran Çıktısı 2.6.2. Koşul Yapıları Kodunun Ekran Çıktısı

```
Bilgehan'ın yaşını giriniz:23
Yiğit'in yaşını giriniz:20
Bilgehan, Yiğit'ten büyüktür
```

Ekran Çıktısı 2.6.3. Koşul Yapıları Kodunun Ekran Çıktısı

```
Bilgehan'ın yaşını giriniz:21
Yiğit'in yaşını giriniz:21
Bilgehan ile Yiğit'in yaşları aynıdır.
```

Kod çalıştırıldıktan sonra öğrencilerin, yaşların karşılaştırılmasındaki 3 durumu da görmeleri sağlanır.

Öğretmen karar kontrol yapıları ile alakalı kavramları detaylı bir şekilde verip ardından küçük bir uygulama yaptıktan sonra aşağıdaki senaryoyu okur.

"Bilgehan ve Yiğit Ankara'da bir üniversitede bilgisayar mühendisliği bölümünde okumaktadırlar. İkisinin de bölümde çalışkan ve özverili olduklarını gözlemleyen bölüm başkanı ikisini de yanına çağırır. Bilgehan ve Yiğit ile küçük bir sohbet gerçekleştirdikten sonra bölümdeki özverili çalışmalarından dolayı kendilerini de alan ile alakalı geliştirebileceklerini düşünerek rektörlük bilgi işlem daire başkanlığında yarı zamanlı çalışmaları için teklifte bulunur. Bilgehan ve Yiğit bu fırsatı kaçırmadan ilgili birimde hemen işe başlarlar. Birimdeki adaptasyon sürecini tamamladıktan sonra bilgi işlem daire başkanı Prof. Dr. M. Akif Sözer ikisini de yanına çağırır. Üniversitenin web sayfasında bulunan öğrenci yönetim sisteminin yenilendiğini ve hala arayüzün geliştirilmesi için çalışmaların devam ettiği söyler. Bunun üzerine Bilgehan ve Yiğit'e ilgili arayüzün geliştirilmesi aşamasında öğrenci notlarının harf karşılıklarını ve öğrencinin dersten geçip kaldığını sistem üzerinde görebilmeleri için kod parçasını yazmalarını ister. Bunun üzerine Bilgehan ve Yiğit vakit kaybetmeden işe başlarlar. Yazacakları kod parçasının öncelikle algoritmasını oluşturduktan sonra kısa sürede de kod parçasını tamamlarlar."

Öğretmen senaryoyu okuduktan sonra öğrencilere dönerek sizce Bilgehan ve Yiğit verilen problem durumunu nasıl çözmüş olabilirler diyerek öğrencilerinde notların harf karşılıklarını veren kodları yazmalarını ister. Kod yazımına geçmeden önce kalma ve geçme notları ayrıca notların harf karşılıklarını öğretmen öğrenciler ile beraber belirler ve belirlenen tabloya göre kod yazımına geçilir.

Tablo 2.6.1 Geçme Kalma Not Aralığı

Ortalama	Durum
Ortalama < 60	Kaldınız
Ortalama >= 60	Geçtiniz

Tablo 2.6.2 Harf Notu Aralığı

Ortalama	Harf Notu
Ortalama < 50	FF
Ortalama <60	DC
Ortalama <70	CC
Ortalama <75	CB
Ortalama <85	BB
Ortalama <90	BA
Ortalama >=90	AA

Kod 2.6.2 Harf Notu Dönüşüm Kodu

```
print("Harf Notu Belirleme")
vize = int(input("vize notunuzu giriniz "))
final = int(input("final notunuzu giriniz "))
ortalama = vize*0.4 + final*0.6
print("dönem sonu notunuz=", ortalama)
if ortalama>=60:
    print("GEÇTİNİZ")
else:
    print("Maalesef KALDINIZ")
if ortalama<50:
  harfnotu="FF"
elif ortalama<60:
  harfnotu="DC"
elif ortalama<70:
  harfnotu="CC"
elif ortalama<75:
  harfnotu="CB"
elif ortalama<85:
  harfnotu="BB"
elif ortalama<90:
  harfnotu="BA"
else:
  harfnotu="AA"
print("Harf Notunuz =",harfnotu)
```

Ekran Çıktısı 2.6.4. Harf Notu Dönüşüm Kodunun Ekran Çıktısı

```
Harf Notu Belirleme
vize notunuzu giriniz 40
final notunuzu giriniz 50
dönem sonu notunuz= 46.0
Maalesef KALDINIZ
Harf Notunuz = FF
```

Ekran Çıktısı 2.6.5. Harf Notu Dönüşüm Kodunun Ekran Çıktısı

```
Harf Notu Belirleme
vize notunuzu giriniz 90
final notunuzu giriniz 80
dönem sonu notunuz= 84.0
GEÇTİNİZ
Harf Notunuz = BB
```

Öğretmen senaryoda verilen problem durumuna yönelik yazılacak kod parçasının yazımı esnasında öğrencilere ipucu verebilir. Süreci tamamlayan öğrencilere gerekli dönütler verildikten sonra yazdıkları kod parçasının eksiksiz bir şekilde çalıştırmalarını sağlar.

Etkinliğin tamamlanmasının ardından öğretmen süreç sonunda öğrencilerden koşul yapılarının kullanılabileceği farklı uygulama alanları ile alakalı örnek vermelerini ister. Kavram yanılgıları var ise bunlar düzeltilir. Gelen cevaplar üzerine ek etkinlik planı yapılabilir. Öğretmen bir sonraki derste "*döngülerin kullanımını öğreneceğiz*" der ve ders sonlandırılır.

DEĞERLENDİRME

Öğretmenin süreç içerisinde sorduğu sorulara verilen cevaplar ve yapılan uygulamadan alınan çıktılara göre Tablo 2.6.1.'de verilen kontrol listesi kullanılarak öğrenciler değerlendirilir.

Tablo 2.6.1: Kontrol listesi

Kontrol Listesi	Evet	Hayır
Karar kontrol yapılarını açıklar.		
Verilen probleme yönelik koşul yapılarını kullanarak bir yazılım gerçekleştirir.		
Geliştirdiği yazılımı test eder.		

KAYNAKÇA

Ergün, E. (2022). *Programlama Eğitiminde Soru Tabanlı Sistem Yaklaşımı.* Ankara: Pegem Akademi.

ETKİNLİK NO	**2.7**
ETKİNLİK ADI	PROGRAMLAMADA DÖNGÜLERİN KULLANIMI
SINIF/KADEME	Ortaokul
SÜRE	40+40+40+40=160 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Döngüler
KAZANIMLAR	**2.5.1.** Farklı döngü türlerini bir problemin çözümünde kullanır.
TEMEL BECERİLER	Problem çözme, Analitik düşünme, Keşfetme
YÖNTEM VE TEKNİKLER	Doğrudan anlatım, Soru-cevap, Gösterip-yaptırma
ARAÇ-GEREÇLER	Bilgisayar, Uygulama geliştirme ortamı
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında yüklü ya da çevrimiçi kullanabileceği bir uygulama geliştirme ortamı olması gerekir. ✓ Her öğrencinin bir bilgisayar karşına oturması sağlanır. ✓ Her öğrencinin kendi algoritmasını ve kodlamasını yapması gerekmektedir. **Döngü:** Bazı problemlerin çözümünde aynı işi/görevi birçok defa tekrarlamanız gerekmektedir. Bu durumda ilk akla gelen çözüm görevi yapan kodun kopyalanıp alta tekrar yapıştırılması olsa da bu çözüm çok pratik ve uygulanabilir değildir. Kodu kaç kere kopyalayıp yapıştırdıysanız o kadar çalışacaktır (daha az veya daha fazla çalışması mümkün değildir), işlemin kaç kez yapılacağını en baştan bilemeyeceğiniz durumlar olabilir veya yapılan işte en ufak bir değişiklik yaptığınızda tüm kopyalanan blokları değiştirmeniz gerekir. Döngü yapıları kullanılarak karmaşık ve uzun işlem alacak problem çözümleri birkaç satırda yapılacak şekilde basitleştirilebilir. Döngüler bir program içerisinde bir işlem birden fazla yapılması gerektiğinde kodları tek tek yazmak yerine kullanılan yapılardır. Programlama dillerinde bu tip işlemlerin yapılabilmesi için çok sık kullanılan 2 tane döngülerin türü vardır. **Sayaçlı Döngü (For** Döngüsü): Başlangıçta tekrar sayısının belirli olduğu durumlarda kullanılan döngü türüdür. Döngü başlangıcında kullanılan değişkene döngü içinde müdahale edilerek tekrar sayısı değiştirilebilir. **Koşullu Döngü (While** Döngüsü): Koşul sağlandığı sürece tekrar eden döngü türüdür. Koşulun sağlandığı her kontrolden sonra döngü içindeki işlemler bir kez yapılır. Koşul sağlanmadıktan sonraki ilk kontrolde döngüden çıkılır.

	Kaprekar Sabiti: Hintli bir matematikçi tarafından matematik literatürüne eklenen kaprekar sabiti, 6174 sayısını ifade etmektedir. Dört basamaklı herhangi bir sayısının rakamlarının önce büyükten küçüğe sonra küçükten büyüğe sıralanması ile elde edilen yeni sayıların birbirinden çıkarılması mantığına dayanmaktadır. İşlem sonucunda elde edilen yeni sayı kullanılarak aynı mantık tekrarlanır. Bu işlemin en fazla yedi defa tekrarlanması ile sonuç ya sıfır ya da kaprekar sabiti olan 6174 sayısı olmaktadır.
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma. ✓ Temel düzeyde programlama bilgisi (değişken atamaları, girdi-çıktı işlemleri, operatör kavramı, aritmetiksel, mantıksal ve karşılaştırma operatörleri, karar kontrol yapıları)

SÜREÇ

Öğretmen dersin başında öğrencilerle bir oyun oynamak istediğini söyler. Öğretmen oyun hakkında kısa bir bilgilendirme yapar.

"Arkadaşlar oyunumuz çok basit, bir kişinin tuttuğu bir sayıyı bir diğeri tahmin etmeye çalışacak. Öğretmen, tahmin edilen sayıyı karşı taraf bildiğinde ***tebrikler bildin*** dönütü bilemediğinde ise ***yanlış tahmin*** *şeklinde dönüt verileceğini söyler. Öğretmen öğrencilere bu oyunun yazılımını geliştirecek olsanız hangi komut yapılarını kullanmanız gerekir."* diye sorar.

Alınan cevaplar üzerine ön bilgiler hatırlatılarak bir önceki etkinlikte kullanılan karar kontrol yapılarının kullanılması gerektiği fikrine ulaşılır. Ardından öğretmen oyunun yapısında bir değişiklik yapılacağını söyler ve oyunda tahmin edilecek sayının doğru cevabı bulunana kadar sürecin işlemesi gerektiği söyler. Öğretmen oyundaki bu değişikliği yapabilmek için hangi komut yapısında değişiklik yapılıp hangi komut yapısının kullanılması gerektiğini sorar ve etkinliğin uygulamasına geçer.

Öğretmen programlama dillerinde kullanılan döngü ve döngü türlerinin tanımlarını detaylı olarak açıklar. Ardında döngülerle alakalı küçük kod parçaları üzerinde uygulama yapar.

Kod 2.7.1. Döngülere Örnek Kod

```
toplam=0
i=1
while i<=10:
    toplam=toplam+i
    i=i+1
print("sayıların toplamı",toplam)
```

Ekran Çıktısı 2.7.1. Kodun Ekran Çıktısı

```
sayıların toplamı 55
```

Öğretmen verilen kodda 1'den 10'a kadar olan sayıların toplamını veren kod parçasını öğrencilere anlatır

öğrencilerin verilen kod parçasını kendi bilgisayarlarına yazmalarını ister ve aynı ekran çıktısını görmelerini sağlar. Süreç içerisinde öğrencilere dönütler vererek bir diğer örneğe geçer.

Kod 2.7.2. While ve For Döngülerine Örnek Kod

```
toplam=0
x = 0
while x < 10:
    x += 1
    if x % 2 == 0:
        print(x)
```

```
for i in range (1,11):
    if i%2==0:
        print(i)
```

Ekran Çıktısı 2.7.2. Kodun Ekran Çıktısı

```
2
4
6
8
10
```

Öğretmen while ve for döngüerinin kullanımına yönelik ayrı ayrı verilen kodlarda 1'den 10'a kadar olan sayılardan çift olanlarını listeleyen kod parçalarını öğrencilere anlatır. Ardından öğrencilerin verilen kod parçalarını kendi bilgisayarlarına yazmalarını ister ve aynı ekran çıktısını görmelerini sağlar. Süreç içerisinde öğrencilere dönütler vererek bir diğer örneğe geçer.

Kod 2.7.3.While ve For Döngülerine Örnek Kod

```
x = 0
while x < 10:
        x += 1
        if x % 2 != 0:
                print(x)
```

```
for i in range (1,11):
if i%2!=0:
        print(i)
```

Ekran Çıktısı 2.7.3. Kodun Ekran Çıktısı

```
1
3
5
7
9
```

Öğretmen while ve for döngüerinin kullanımına yönelik ayrı ayrı verilen kodlarda 1'den 10'a kadar olan

sayılardan tek olanlarını listeleyen kod parçalarını öğrencilere anlatır. Ardından öğrencilerin verilen kod parçalarını kendi bilgisayarlarına yazmalarını ister ve aynı ekran çıktısını görmelerini sağlar. Süreç içerisinde öğrencilere dönütler vererek bir diğer örneğe geçer.

Örnekte kaprekar sabitine ulaşmayı sağlayan bir kod yazımı gerçekleştirilecektir. Öncelikle öğrencilere kaprekar sabiti ile ilgili bilgilendirme yapılır. Öğrencilere, daha önceki etkinliklerde öğrendiklerinden hareket ederek;

- ✓ Sisteme girilen dört basamaklı sayının matematiksel operatörler kullanarak basamaklarına ayrılması,
- ✓ Elde edilen basamak değerlerinin koşul yapısı ve mantıksal operatörlerin kullanarak büyük - küçük tespitinin yapılması,
- ✓ Elde edilen sayıların matematiksel operatörler ile yeni dört basamaklı sayılara dönüştürerek sonuca ulaşılması işlemlerini gerçekleştirecekleri söylenir.

Kod 2.7.4. Kaprekar Sayısının Kodu

```
kaprekar = 6174
sonuc = 0
sayi1 = int(input("Dört basamaklı bir sayı giriniz: "))
while sonuc != kaprekar :
    binlik = sayi1 // 1000
    sayi2 = sayi1 - binlik*1000
    yuzluk = sayi2 // 100
    sayi2 = sayi2 - yuzluk*100
    onluk = sayi2 // 10
    sayi2 = sayi2 - onluk*10
    birlik = sayi2 // 1

    if binlik>=yuzluk:
        bin = binlik
        yuz = yuzluk
    else:
        bin = yuzluk
        yuz = binlik
    if onluk >= birlik:
        on = onluk
        bir = birlik
    else:
        on = birlik
        bir = onluk

    if bin >= on:
        bin= bin
        on = on
```

```
    else:
        binlik = bin
        onluk = on
        bin = onluk
        on = binlik

    if yuz >= bir:
        yuz = yuz
        bir = bir
    else:
        yuzluk = yuz
        birlik= bir
        yuz = birlik
        bir = yuzluk

    if yuz >= on:
        yuz = yuz
        on = on
    else:
        yuzluk = yuz
        onluk = on
        yuz = onluk
        on = yuzluk

    ilksayi= bin*1000 + yuz*100 + on*10 + bir
    print(ilksayi)
    ikincisayi = bir*1000 + on*100 + yuz*10 + bin
    print(ikincisayi)
    sonuc = ilksayi - ikincisayi
    print("Sonuc = ", sonuc)
    sayi1 = sonuc
print ("Sonuç, kaprekar sabitine eşitlendi")
```

Ekran Çıktısı 2.7.4. Kaprekar Sayısının Kodunun Ekran Çıktısı

```
Dört basamaklı bir sayı giriniz: 2023
3220
223
Sonuc = 2997
9972
2799
Sonuc = 7173
```

```
7731
1377
Sonuc = 6354
6543
3456
Sonuc = 3087
8730
378
Sonuc = 8352
8532
2358
Sonuc = 6174
Sonuç, kaprekar sabitine eşitlendi
```

Ekran Çıktısı 2.7.5. Kaprekar Sayısının Kodunun Ekran Çıktısı

```
Dört basamaklı bir sayı giriniz: 5532
5532
2355
Sonuc = 3177
7731
1377
Sonuc = 6354
6543
3456
Sonuc = 3087
8730
378
Sonuc = 8352
8532
2358
Sonuc = 6174
Sonuç, kaprekar sabitine eşitlendi
```

Öğretmen kaprekar sabitine ulaşmayı sağlayan kodun yazımı esnasında yazımı esnasında öğrencilere ipucu verebilir. Süreci tamamlayan öğrencilere gerekli dönütler verildikten sonra yazdıkları kod parçasının eksiksiz bir şekilde çalıştırmalarını sağlar.

Etkinliğin tamamlanmasının ardından öğretmen süreç sonunda öğrencilerden döngülerin kullanılabileceği farklı uygulama alanları ile alakalı örnek vermelerini ister. Kavram yanılgıları var ise bunlar düzeltilir. Gelen cevaplar üzerine ek etkinlik planı yapılabilir. Öğretmen bir sonraki derste "*fonksiyonlar ve hazır fonksiyonların kullanımını öğreneceğiz*" der ve ders sonlandırılır.

DEĞERLENDİRME:

Öğretmenin süreç içerisinde sorduğu sorulara verilen cevaplar ve yapılan uygulamadan alınan çıktılara göre Tablo 2.7.1.'de verilen kontrol listesi kullanılarak öğrenciler değerlendirilir.

Tablo 2.7.1: Kontrol listesi

Kontrol Listesi	Evet	Hayır
Döngü kavramını açıklar.		
Farklı döngü türlerini açıklar.		
Verilen probleme yönelik farklı döngü türlerini kullanarak bir yazılım gerçekleştirir.		
Geliştirdiği yazılımı test eder.		

KAYNAKÇA

Ergün, E. (2022). *Programlama Eğitiminde Soru Tabanlı Sistem Yaklaşımı.* Ankara: Pegem Akademi.

Nuez, F. (2022). Symmetries in Generalized Kaprekar's Routine. *Symmetry*, *14*(1), 37.

ETKİNLİK | 8

ETKİNLİK NO	**2.8**
ETKİNLİK ADI	YERLEŞİK FONKSİYONLAR
SINIF/KADEME	Ortaokul
SÜRE	40+40=80 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Fonksiyonlar
KAZANIMLAR	**2.6.1.** Fonksiyon kavramını açıklar. **2.6.2.** Kullandığı programlama dilinde var olan hazır fonksiyonları kullanır.
TEMEL BECERİLER	Problem çözme, Analitik düşünme, Keşfetme
YÖNTEM VE TEKNİKLER	Bilgisayar, Uygulama geliştirme ortamı, internet, Projeksiyon cihazı/ Etkileşimli tahta
ARAÇ-GEREÇLER	Bilgisayar, Uygulama geliştirme ortamı
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında yüklü ya da çevrimiçi kullanabileceği bir uygulama geliştirme ortamı olması gerekir. ✓ Ders içinde kullanılacak fonksiyon, parametre, fonksiyon çağırma, geri değer döndürme gibi kavramlar öğretmen tarafından incelenir. ✓ Her öğrencinin bir bilgisayar karşına oturması sağlanır.
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma.

SÜREÇ

Tahtaya aşağıdaki ev aletlerinin ismi yazılarak öğrencilere bu aletlerin görevlerinin/işlevlerinin ne olduğu sorulur.

Fırın, Elektrik süpürgesi, çamaşır makinesi, bulaşık makinesi, aspiratör.

Öğrencilerden cevaplar alındıktan sonra öğretmen öğrencilere "Çeşitli görevleri yerine getirmek için tasarlanmış bu ev aletlerinin hepsinin bizim hayatımızı kolaylaştıran işlevleri olduğundan söz edilir. Her kek yapmak istediğimizde zaten var olan fırınımızı kullanırız, yeni bir fırın alma ihtiyacı duymayız." der ve öğrencilere yazılımlarda da buna benzer sık kullanılan işlevler olduğu, kendimizin de belli bir işlevi olan kod bloklarını bir arada kullanarak fonksiyon adı verilen yapılar oluşturabileceğimiz ifade edilir.

Bu aşamada yine öğretmen öğrencilere "*Bugüne kadar sıklıkla kullandığımız print() ve input() komutları da birer fonksiyondur. print() işlev olarak ekrana çıktı olarak yazı yazdırmamızı sağlarken, input() ise kullanıcıdan bilgi almamızı sağlamaktadır.*" der

Bu aşamada öğrencilere bugün yapılacak etkinliğimizin fonksiyonlar ve Python programlama dilinde

hazır olarak gelen fonksiyonlar olduğundan bahsedilerek etkinliğe geçilir.

Etkinlikte öğrencilere kavramlarda verilen fonksiyon tanımı verildikten sonra öğrencilere *"Bugün sizler ile Python programlama dilinde hazır bulunan bazı fonksiyonlar ile ilgili uygulamalar yapacağız."* denilir.

Öğrencilere *"İlk önce göreceğimiz fonksiyonların işlevlerini/görevlerini inceleyelim. Bu fonksiyonlara yerleşik (Built-in) fonksiyonlar denir."* Denilerek aşağıdaki tablo gösterilir.

Tablo 2.8.1. Python programlama dilinde bulunan yerleşik fonksiyon örnekleri

Fonksiyon	Açıklaması	Fonksiyon	Açıklaması
abs()	Bir sayının mutlak değerini döndürür.	min()	Bir grup içinden içinde en küçük olanı döndürür
id()	Bir değişkenin kimlik bilgisini	round()	Sayının yuvarlanmış halini döndürür
max()	Bir grup içinden içinde en büyük olanı döndürür	type()	Bir nesnenin tipini döndürür.

Tüm öğrencilere *fonksiyonlar1.py* adında bir dosya açıp dosyaya Kod 2.8.1.'deki kodu yazmaları ve bilgi girişi olarak kendi isimlerini vermeleri istenir.

Kod 2.8.1. Geri dönüş işlemi

```
isim = input("isminizi giriniz:")
karakter_sayisi=len(isim)
print("isminizde",karakter_sayisi,"karakter var.")
```

Ekran çıktısı 2.8.1. geri döndürme işlemi ekran çıktısı

```
isminizi giriniz:eylül
isminizde 5 karakter var.
```

Bu ekranda öğrencilere *"print() fonksiyonu ile input() ve len() fonksiyonlarının kullanım açısından birbirlerinden olan farklılığı nedir?"* denilir. Alınan cevaplarda aranan farklılık iki fonksiyonun çağrıldıkları noktaya bir sonuç döndürürken print() fonksiyonunun bir değer döndürmemesidir. Bu noktada alınan cevaplardan sonra öğretmen öğrencilere geri değer döndürme kavramından bahseder.

Tüm öğrencilere *fonksiyonlar2.py* adında bir dosya açıp dosyaya Kod 2.8.2.'deki kodu yazmaları istenir. Kod yazıldıktan sonra çalıştırılarak çıktılar üzerinden öğrencilere her fonksiyonun çalışması anlatılır.

Kod 2.8.2. Yerleşik fonksiyonların kullanımı

```
s1=1
s2=15
s3=10.2258
s4=s1-s2

mutlak_s4=abs(s4)
```

```
kimlik_s1=id(s1)
kimlik_isim=id(s2)
en_buyuk=max(s1,s2,s3,s4)
en_kucuk=min(s1,s2,s3,s4)
yuvarlanmis_s3=round(s3,2)
tip_s1=type(s1)
tip_s4=type(s4)

print("s4 mutlak değeri:",mutlak_s4)
print("s1 ve isim kimlik:",kimlik_s1,kimlik_isim)
print("en büyük sayı:",en_buyuk,"en küçük sayı:",en_kucuk)
print("s3 yuvarlanmış:",yuvarlanmis_s3)
print("s1 tip:",tip_s1,"s4 tip:",tip_s4)
```

Ekran çıktısı 2.8.2. Yerleşik fonksiyonların çıktıları.

```
s4 mutlak değeri: 14
s1 ve isim kimlik: 2969723169072 2969723169520
en büyük sayı: 15 en küçük sayı: -14
s3 yuvarlanmış: 10.23
s1 tip: <class 'int'> s4 tip: <class 'int'>
```

Etkinliğin devamında öğrencilere "Kullanıcıdan aldığı 4 sayıdan en küçük olandan en büyük olanı çıkaran ve elde ettiği sonucun mutlak değerini ekrana yazdıran kodu Python programlama dilinde yazınız." Denilerek ilgili kodu yazmak için *fonksiyonlar3.py* adı ile yeni bir dosya açmaları istenir. Programın örnek bir çözümü Kod 2.8.3.'teki gibidir.

Kod 2.8.3. Problem çözümü

```
s1=int(input("sayı giriniz:"))
s2=int(input("sayı giriniz:"))
s3=int(input("sayı giriniz:"))
s4=int(input("sayı giriniz:"))
en_buyuk=max(s1,s2,s3,s4)
en_kucuk=min(s4,s2,s3,s1)
fark = en_kucuk - en_buyuk
mutlak = abs(fark)
print("sayıların en büyük en küçük mutlak farkı:",mutlak)
```

Etkinliğin tamamlanmasının ardından öğretmen öğrencilere *"Fonksiyon nedir? Fonksiyon kavramı neyi ifade ediyor"* soruları sorulur ve öğrencilerden alınan cevaplar üzerine konuşulur. Öğretmen daha sonra etkinliğin hedeflediği kazanım ve etkinliği özetleyen bir konuşma yapar:

"Sevgili öğrenciler bugün sizlerle temel olarak fonksiyon kavramını ve Python programlama dilinde var olan yerleşik fonksiyonların kullanımını ve neler olduklarını öğrendik. Bir sonraki dersimizde sizler ile kendi fonksiyonlarımızı yazmaya öğreneceğiz."

DEĞERLENDİRME:

Tablo 2.8.2: Kontrol listesi

Kontrol Listesi	Evet	Hayır
Fonksiyon kavramını açıklar		
Parametre kavramını açıklar.		
Yerleşik fonksiyon kavramını açıklar.		
Yerleşik fonksiyonları amacına uygun olarak kullanır.		

ETKİNLİK | 9

ETKİNLİK NO	**2.9**
ETKİNLİK ADI	KENDİ FONKSİYONUMU YAZIYORUM
SINIF/KADEME	Ortaokul
SÜRE	40 + 40 + 40 + 40=160 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Fonksiyonlar
KAZANIMLAR	**2.6.3.** Kullandığı programlama dilinde belirli bir amaca yönelik fonksiyon yazar.
TEMEL BECERİLER	Problem çözme, Analitik düşünme, Keşfetme
YÖNTEM VE TEKNİKLER	Doğrudan anlatım, Soru-cevap, Gösterip yaptırma
ARAÇ-GEREÇLER	Bilgisayar, Uygulama geliştirme ortamı, internet, Projeksiyon cihazı/ Etkileşimli tahta
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında yüklü ya da çevrimiçi kullanabileceği bir uygulama geliştirme ortamı olması gerekir. ✓ Ders içinde kullanılacak fonksiyon tanımlama, parametre, fonksiyon çağırma, geri değer döndürme gibi kavramlar öğretmen tarafından incelenir. ✓ Her öğrencinin bir bilgisayar karşına oturması sağlanır.
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma.

SÜREÇ

Öğretmen öğrencilere *"Meyve Sıkacağı aletinin işlevi/fonksiyonu nedir diye sorar?"* alınan cevaplardan sonra öğrencilere *"Suyu çıkarılmak istenen meyveleri meyve sıkacağının haznesine koymalıyız. Bunları meyve sıkacağının aldığı parametreler, çıkan meyve suyunu da dönüş değeri olarak düşünebiliriz. Peki sizde bir ev aleti tasarlamak ister misiniz? Tasarlarsanız işlevi/fonksiyonu ne olurdu? Hangi parametreleri alır ve ne gibi bir iş yapar veya geriye bir işlenmiş ürün döndürürdü?"* sorularını sorduktan sonra alınan cevaplar kısa bir süre sınıf için tartışılır.

Bu girişten sonra yazılım geliştirme yaparken kendimizin fonksiyonlar tanımlayabileceğimizden bahsedilerek etkinliğe geçilir.

Etkinlikte öğrencilere kavramlarda verilen fonksiyon tanımlama işleminden bahsedilir. Python programlama dilinde bir fonksiyon bloğu tanımlamak için *def* anahtar kelimesinin kullanıldığı öğrencilere anlatılır. Bu adımda öğrencilerden fonksiyon tanımlama kavramının tanımı öğrencilere sorularak, kendi cümleleri ile tanımlamaları istenir. Daha sonra etkinlik için öğrencilerden *fonksiyonlar4.py* isimli bir dosya oluşturmaları istenerek Kod 2.9.1.'de bulunan kodu bu dosyaya yazmaları için bir süre verilir.

Kod 2.9.1. Çöp adam çizen fonksiyon

```
def cop_adam_ciz():
```

```
    print(" O ")
    print(" /|\ ")
    print(" | ")
    print(" / \ ")
```

Ekran çıktısı 2.9.1. Çöp adam çizen fonksiyonun ekran çıktısı

Öğrenciler kod yazma işlemi bitirdikten sonra öğrencilerden yazdıkları kodu çalıştırmaları istenir. Öğrencilere *"Kodumuzu çalışmasına rağmen neden ekrana herhangi bir çıktı vermedi?"* diye sorularak cevaplar alınır. Alınan cevaplardan sonra bu yapılan adımın fonksiyonun *tanımlama* adımı olduğu her fonksiyonun görevini yerine getirmek için çağrılması gerektiği öğrencilere anlatılır. Burada öğrencilere fonksiyonun bir kere tanımlandıktan sonra istenildiği kadar çağırılabileceği anlatılır. Öğrencilere Kod 2.9.2.'de gösterildiği gibi kodlarına fonksiyon çağırma kodlarını da ekleyerek yazdıkları kodu çalıştırmaları istenir ve Ekran çıktısı 2.9.2. incelenir.

Kod 2.9.2. Çöp adam çizen fonksiyonun çağırmalar eklenmiş hali

```
def cop_adam_ciz():
    print(" O ")
    print(" /|\ ")
    print(" | ")
    print(" / \ ")

cop_adam_ciz()
cop_adam_ciz()
```

Ekran çıktısı 2.9.2. Parametre alan fonksiyonun çıktısı

```
 O
/|\
 |
/ \
 O
/|\
 |
/ \
```

Ekran çıktısı ve yazılan kod incelenirken öğrencilere, yazdığımız fonksiyonun bir parametre almadığı bir değer döndürmediği ifade edilir.

Parametre alan bir fonksiyon yazmak için *fonksiyonlar5.py* dosyası açılır ve içine Kod 2.9.3.'daki kodlar

yazılır ve 2.9.3.'deki çıktı elde edilir.

Kod 2.9.3. parametre alan fonksiyon örneği

```
def n_kez_mesaj_yaz(n:int,mesaj:str):
        for i in range(n):
                print(mesaj)
n_kez_mesaj_yaz(2,"merhaba")
n_kez_mesaj_yaz(3,"güle güle")
```

Ekran çıktısı 2.9.3. Parametre alan fonksiyon çıktısı

```
merhaba
merhaba
güle güle
güle güle
güle güle
```

Öğrencilere Kod 2.9.3. parametre alan fonksiyon olarak yazıldığı ve ihtiyaçlarımıza uygun olarak tanımlayacağımız fonksiyonlara tanımımıza uygun parametreler ekleyebileceğimiz anlatılır. Burada yine tanımlanan fonksiyonun geriye değer döndürmediğinden bahsedilir.

Öğrencilere geriye değer döndüren bir fonksiyon örneği yazmak için *fonksiyonlar6.py* isimli bir dosya açmaları istenir. Bu dosyanın içine Kod 2.9.4.'teki kodlar yazılır. Bu fonksiyonun boy ve ağırlık değerlerine göre boy-kilo endeksini hesapladığı öğrencilere anlatılır. Öğrenciler ile kod ve 2.9.4.'teki ekran çıktısı incelenir.

Kod 2.9.4. Geriye değer döndüren fonksiyon tanımı

```
def bki_hesapla(agirlik:float,boy:float):
        bki=agirlik/(boy*boy)
        return bki

by1=float(input("boyunuzu metre cinsinden giriniz:"))
ag1=float(input("ağırlığınızı kg cinsinden giriniz:"))
bki1=bki_hesapla(ag1,by1)
print("boy kilo indeksininiz:",bki1)
```

Ekran çıktısı 2.9.4. Geriye değer döndüren fonksiyon çıktısı

```
boyunuzu metre cinsinden giriniz:1.74
ağırlığınızı kg cinsinden giriniz:76
boy kilo indeksininiz: 25.102391333069097
```

Burada öğrencilere bu fonksiyonun çağrıldığı yere hesapladığı değeri *return* anahtar kelimesi ile döndürdüğü bunun içinde dönen değerin aynı *input()* fonksiyonun da olduğu gibi bir değişkene aktarılması gerektiği anlatılır.

Öğrencilere buraya kadar öğrendikleri bilgiler ile aşağıdaki tanımda verilen fonksiyonları yazmaları için *yeni bir çalışma dosyası* açmaları istenir. (Kod 2.9.5.'te örnek bir çözüm verilmiştir.)

- ✓ Parametre olarak bir dikdörtgene ait kısa ve uzun kenar bilgilerini alarak çevresini hesaplayıp sonucu geri döndüren bir fonksiyon yazınız.
- ✓ Parametre olarak saat ve dakika bilgisi alarak geriye saniye bilgisini döndüren fonksiyonu yazınız.
- ✓ Çalıştırıldığında çıktı olarak kolları havada çöp adam çıktısını veren fonksiyonu yazınız

Kod 2.9.5. Örneklerin çözümleri

```
def dikdortgen_cevresi(kisa:float,uzun:float):
    cevre=2*(kisa+uzun)

def saat_dakika_2_saniye(saat:int,dakika:int):
    saniye = saat*60*60+dakika*60
    return saniye

def kollari_havada_cop_adam_ciz():
    print(" O ")
    print(" \|/ ")
    print(" | ")
    print(" / \ ")
```

Etkinliğin tamamlanmasının ardından öğretmen öğrencilere etkinliğin hedeflediği kazanım ve etkinliği özetleyen bir konuşma yapar:

"Sevgili öğrenciler bugün sizlerle temel olarak kullanıcı tanımlı fonksiyonlar tanımlamayı ve belli bir amaca yönelik fonksiyon tanımlamayı öğrendik."

DEĞERLENDİRME:

Tablo 2.9.1: Kontrol listesi

Kontrol Listesi	Evet	Hayır
Fonksiyon tanımlama kavramını açıklar		
Geri değer döndürme kavramını açıklar.		
Amacına uygun fonksiyonun kodunu yazar.		

ÖZEL EĞİTİM VE REHBERLİK HİZMETLERİ
GENEL MÜDÜRLÜĞÜ

BİLİM VE SANAT MERKEZLERİ

YAZ OKULU DESTEKLEME VE YETİŞTİRME KURSU PROGRAMI

YAZILIM ATÖLYESİ

ETKİNLİKLER

LİSE

TÜRKİYE CUMHURİYETİ MİLLÎ EĞİTİM BAKANLIĞI

ETKİNLİK NO	**3.1**
ETKİNLİK ADI	İLK KOD'UM
SINIF/KADEME	Lise
SÜRE	40+40=80 Dakika
ÖĞRENME ALANI/KONU	Yazılım geliştirme / Programlama Dillerine Giriş
KAZANIMLAR	**3.1.1.** Programlama dillerinin amacını keşfeder. **3.1.2.** Kod yazmak için gerekli hazırlıkları yapar. **3.1.3.** Derleyici ve yorumlayıcı arasındaki farkı açıklar. **3.1.4.** Temel girdi-çıktı fonksiyonları ile işlem yapar.
TEMEL BECERİLER	Problem çözme, Analitik düşünme, Keşfetme
YÖNTEM VE TEKNİKLER	Doğrudan anlatım, Soru-cevap, Gösterip-yaptırma
ARAÇ-GEREÇLER	Bilgisayar, Uygulama geliştirme ortamı,
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında yüklü ya da çevrimiçi kullanabileceği bir uygulama geliştirme ortamı olması gerekir. ✓ Ders içerisinde kullanılacak Programlama dili, derleyici, yorumlayıcı, uygulama geliştirme ortamı kavramlarına ilişkin farklı tanımlar öğretmen tarafından incelenir. ✓ Her öğrencinin bir bilgisayar karşına oturması sağlanır. ✓ Çeşitli programlama dillerine ait yazılış örnek programları hazır eder.
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma. ✓ Öğrenciler bir programlama dilinde var olan temel girdi ve çıktı fonksiyonlarının kullanımını bilir.

SÜREÇ

Dersin başında öğrencilere:

"Bilgisayarlar bu kadar karmaşık işlemi nasıl yapabiliyor?" sorusu sorularak öğrencilerin soru hakkındaki fikirleri alınır. Gelen cevaplara göre öğrencilerin hazır bulunuşluk seviyeleri değerlendirilir. Cevaplar arasında bilgisayarların önceden programlanması ile ancak problemleri çözebileceği veya sorulan bir soruya cevap verebileceği cevabı ön plana çıkarılır.

Cevaplar arasında programlama ile ilgili kavramlar gelmezse öğretmen *"Bilgisayarların çok karmaşık işlemleri çözebilecek işlemleri yapabilecek donanıma sahip olduğu buna karşın ancak bir insan tarafından programlandığı zaman işlem yapabilme kabiliyetine ulaşır."* diyerek öğrencilerde temel bir alt yapı oluşturur.

Dikkat bölümünden sonra aşağıdaki örnek verilerek etkinliğe giriş yapılır:

"Uzman olamayan kişilere bir iş yaptırılmasının gerektiği durumlarda bu işin nasıl yapılacağını adım adım anlatan yönergeler kullanılır. Örneğin makarna yapmayı bilmiyorsunuz. İnternetten bulduğunuz bir yemek tarifi size ne yapmanız gerektiğini adım adım anlatıyor. Sizde bu adımları sırası ile uygulayarak makarnanızı hazırlıyorsunuz. Bilgisayarları bu örnekteki acemi aşçıya benzetebiliriz. Her türlü malzemesi var ama ne yapacağını bilmiyor.

Burada önemli olan bu hikâyedeki yemek tarifidir. Çünkü yemek tarifinde olduğundaki gibi bir bilgisayar da bir işi nasıl yapacağı adım adım verilen komutlar sayesinde yerine getirir. Bilgisayara bu şekilde komutların bilgisayarın anlayacağı bir dilde verilmesine programlama denir."

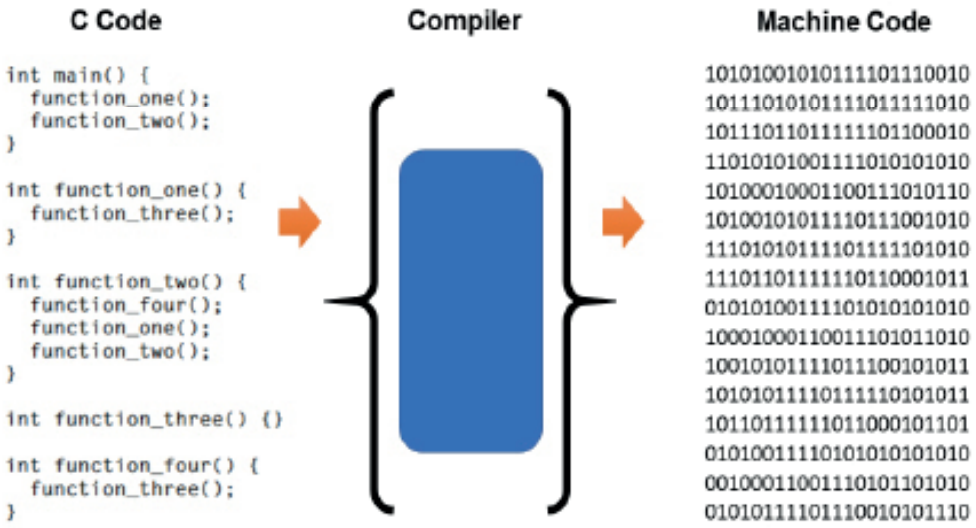
Öğrencilere *"Peki bilgisayarın anlayabileceği bir dil ne olabilir?"* sorusu yöneltilir. Gelen cevaplardan aşağıdaki bilgiler verilir.

"Bilgisayara yerine getirmesi gereken komutu herhangi bir dili kullanarak veremeyiz. Bilgisayarların anlayabileceği özel diller vardır. Bunu insanların konuştuğu çeşitli dillere ve bilmediğiniz dilde size söylenen bir şeyi anlamamanıza benzetebiliriz. Bilgisayara yerine getireceği komutları verdiğimiz bu bilgisayarlara özel dillere programlama dilleri denir."

Yukarıdaki açıklamalardan sonra öğrencilere C, C++, C#, python, java isimlerini duyup duymadıkları sorulur. Gelen cevaplar sonrasında bu kelimelerin programlama dilleri olduğu söylenir.

Dersin bu bölümünde öğretmen çeşitli dillerde yazılmış kod örneklerini gösterir. Ardında şu açıklamaları yapar.

"Çeşitli dillerde yazılmış kod örneklerini gördük. Kodların belli bir düzende yazıldığı, kendi içinde tıpkı dilimizde olduğu gibi kuralları olduğuna dikkat etmişsinizdir. Bu kodlar insanların anlayabileceği şekildedir. Ancak bilgisayarlar bu kodları olduğu gibi kullanamazlar. Öncelikler bu kodların makine diline dönüştürülmesi gerekmektedir. Makine diline dönüştürülmüş kodlar artık insanın anlayamayacağı şekildedir. Görsel 3.1.1'de bilgisayar dilinde yazılmış bir kodun derlendikten sonra temsili gösterimi yer almaktadır."



*Görsel 3.1.1 SEQ Görsel * ARABIC 1: Bir kodun derlenmesinin temsili gösterimi*

Öğretmen **derleyici** ile **yorumlayıcı** ait şu bilgileri sınıfla paylaşır:

"Kodlar makine diline dönüştürülürken iki program kullanılır. Bunlar derleyiciler ve yorumlayıcılardır. Derleyici ve yorumlayıcılar aynı görevi yerine getirirler. Tablo 3.1.1'de derleyici ve yorumlayıcılara ait bazı özellikler verilmiştir.

Tablo 3.1.1. Derleyici ve yorumlayıcılara ait özellikler tablosu

DERLEYİCİ	YORUMLAYICI
Derleyiciler kodlar çalıştırılmadan önce bütün kodları makine diline dönüştür.	*Yorumlayıcılar kodları satır-satır makine dönüştürülür.*
Derleyiciler bütün satırları makine diline çevirir.	*Yorumlayıcılar sadece çalıştırılan satırları makine diline çevirir.*
Derleyiciler sonucu bir dosyada saklar. Derlenmiş kodlar tekrar derlenmeye ihtiyaç duyulmadan çalıştırılabilir.	*Yorumlayıcı tarafında makine diline çevrilmiş kodlar her defasında tekrar yorumlanmalıdır.*
Hızlı çalışır	*Yavaş çalışır.*

Etkinliğin son bölümünde öğretmen;

"Şimdide yazılım geliştirme ortamlarını inceleyelim. Yazılım geliştirme ortamları programlamanın yapıldığı, kodların derlendiği ve/veya yorumlandığı, program çıktılarının gözlemlenebildiği, programcıya çeşitli yardımcı araçlar sunan programlar veya platformlardır. Günümüzde bilgisayar üzerinde çalışan yazılım geliştirme ortamlarının yanında internet üzerinde çalışan yazılım geliştirme ortamları da mevcuttur."

Bu tanımları yaptıktan sonra öğretmen dersinde kullanacağı yazılım geliştirme ortamını ana hatlarıyla tanıtır. Sonrasında "şimdide ilk kodlarımızı yazmaya ne dersiniz?" sorusunu sorar ve Görsel 3.1.2'de görülen temel çıktı kodunu kullanarak öğrencilere ilk kodlarını yazdırır.

```
1
2  print("Merhaba Dünya!")
```

Görsel 3.1.2: Örnek python kodu

Etkinliğin tamamlanmasının ardından öğretmen tarafından öğrencilere: *"derleyici ve yorumlayıcı arasındaki farklar nelerdir?"* sorusu ve öğrencilerden alınan cevaplar üzerinde konuşulur. Gerekli dönütler sağlanır. Öğretmen daha sonra dersin hedeflerini vurgulayan, etkinliği özetleyen ve sonraki etkinlik hakkında kısa bilgi veren ders sonu konuşması yapılır:

"Çocuklar bugün sizinle Programlama dili, derleyici, yorumlayıcı, uygulama geliştirme ortamı kavramlarını öğrendik ve uygulama yaptık. Bir sonraki derste değişken kavramını öğreneceğiz." denilir ve ders sonlandırılır.

DEĞERLENDİRME:

Öğrencilerin kavramları içselleştirmeleri için günlük yaşamdan sabit ve değişkenlere örnekler vermeleri istenir. Daha sonra öğrencilerden değişken ve sabit kavramlarını açıklamaları beklenir. Kavram yanılgıları varsa düzeltilir.

Öğretmen, öğrencilerden gelen cevaplara göre ek etkinlik yapılabilir.

Yapılan kodlamalar ve soruya verilen cevaplar Tablo 3.1.2'de verilen kontrol listesi kullanılarak değerlendirilir.

Tablo 3.1.2: Kontrol listesi

Kontrol Listesi	Evet	Hayır
Programlama dillerinin amacını açıklar.		
Bir yazılım geliştirme ortamını kullanır.		
Yorumlayıcı tanımlar		
Derleyiciyi tanımlar		
Yorumlayıcı ile derleyici arasındaki temel farkları açıklar.		
Print komutunu kullanır.		

ETKİNLİK NO	**3.2**
ETKİNLİK ADI	DEĞİŞKENLER
SINIF/KADEME	Lise
SÜRE	40+40=80 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Değişkenler
KAZANIMLAR	**3.2.1.** Değişken kavramını açıklar. **3.2.2.** İhtiyaca uygun değişken türlerini belirler. **3.2.3.** Değişkenleri kurallarına uygun olarak oluşturur. **3.2.4.** Bir kod örneğindeki değişkenleri tespit eder.
TEMEL BECERİLER	Problem çözme, Analitik düşünme, Keşfetme
YÖNTEM VE TEKNİKLER	Doğrudan anlatım, Soru-cevap, Gösterip-yaptırma
ARAÇ-GEREÇLER	Bilgisayar, Uygulama geliştirme ortamı, internet, Projeksiyon cihazı/ Etkileşimli tahta
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında yüklü ya da çevrimiçi kullanabileceği bir uygulama geliştirme ortamı olması gerekir. ✓ Ders içerisinde kullanılacak Veri, Bilgi, Değişken, Sabit kavramlarına ilişkin farklı tanımlar öğretmen tarafından incelenir. ✓ Her öğrencinin bir bilgisayar karşına oturması sağlanır.
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma. ✓ Öğrenciler bir programlama dilinde var olan temel girdi ve çıktı fonksiyonlarının kullanımını bilir.

SÜREÇ

Dersin başında öğrencilere *"Programlama da sıkça kullandığımız ve sizin matematik dersinden de bildiğiniz bir kavramdan bahsedeceğim. Biliyorsunuz matematikte değişken ve sabit değerler vardır. Bana değişken ve sabit değerler ile ilgili örnek verebilir misiniz?"* soruları yöneltilerek, cevapları alınır. Öğrencilerden yanıt alınmadığı takdirde öğretmen "*matematikte bir dairenin yarı çapı, daire büyüklüğüne göre değişkendir ama örneğin pi sayısının değeri 3.14'e sabittir.*" diyerek değişken ve sabit kavramlarına örnek vererek açıklar. Daha sonra "*Arkadaşlar bir programlama dilini öğrenmeye başladığımızda atacağımız ilk adımlardan birisi, o dil içerisinde yer alan veri tiplerini öğrenmektir. Bu etkinliğimiz de Python üzerinde farklı veri tipleri ile değişken tanımlama, kurallı ve doğru bir şekilde değişken adlandırılması gibi konularla çalışacağız.*" denilerek öğrencilerin hedeften haberdar edilmesi sağlanır.

Dikkat çekme sorularını takiben aşağıdaki sırayla etkinliğe giriş yapılır.

Etkinliğin iyi anlaşılması, düzenli ve disiplinli ilerlemesini sağlayabilmek adına öğrencilerin öncelikle etkinliği etkileşimli tahtada gösterilirken dikkatlice dinlemesi, anlatım bittikten sonra öğretmenin söylediği zamanda uygulamaya geçilerek yazılım geliştirme ortamında uygulaması istenir.

Bu aşama da öğrencilere etkileşimli tahtada gösterilen Tablo 3.2.1 incelemeleri için süre verilir. Daha sonra öğretmen önce Tablo 3.2.1'i örneklerle açıklar ve gerekli görürse tablodaki örnekler dışında tahtaya birkaç farklı örnek yazar ve örneğin hangi veri tipine ait olduğunu öğrencilere sorabilir.

Tablo 3.2.1. Python Veri tipleri listesi

Veri Tipi	Açıklama	Örnekler
int	Tam sayı	42, -100, 0
float	Ondalık sayı	3.14, -0.5, 2.0
bool	Boolean (Mantıksal)	True, False
str	Metin (karakter)	"Merhaba", 'Dünya'
list	Liste	[1, 2, 3], ["a", "b", "c"]
tuple	Demet	(1, 2, 3), ("a", "b", "c")
set	Küme	{1, 2, 3}, {"a", "b", "c"}
dict	Sözlük	{"anahtar": "değer", "ad": "Ali"}

Bu aşama da öğretmen: öğrencilere "*Tablo 3.2.1 'de veri tipleri Python'daki en yaygın veri tiplerini kapsar ancak tüm veri tiplerini içermez. Bununla birlikte, Python'da kullanabileceğiniz diğer veri tipleri arasında bytearray, bytes, complex, frozenset ve memoryview gibi tipler de vardır.*" diyerek örneklerdeki veri tiplerinin yazım kurallarını da dikkatli bir şekilde gösterilir.

Daha sonra öğretmen: "Tablo 3.2.2. deki değişken isimlendirme kurallarından bahsedip, şimdi de veri tiplerini bir değişkene atayarak yazılım geliştirme ortamında uygulayalım" diyerek Kod 3.2.1 de verilen kodlar, yazılım geliştirme ortamında yazılır ve öğrencilere açıklanır. Tüm öğrencilerin Ekran çıktısı 3.2.1'deki gibi ekran çıktıları elde etmeleri beklenir. Varsa anlaşılmayan kısımlar öğrencilere tekrar anlatılır.

Tablo 3.2.2 Değişken İsimlendirme Kuralları

Değişken İsimlendirme Kuralları
1. Değişken isimleri, harf veya alt çizgi (_) ile başlamalıdır. Rakamla başlayan bir değişken ismi kullanılamaz.
2. Değişken isimleri yalnızca harfler, rakamlar ve alt çizgi (_) karakterlerini içerebilir. Diğer özel karakterler kullanılamaz.
3. Değişken isimleri büyük/küçük harfe duyarlıdır. Örneğin, "degisken" ve "Degisken" iki farklı değişken ismidir.
4. Değişken isimleri açıklayıcı ve anlamlı olmalıdır. İsimlendirme yaparken değişkenin kullanım amacına göre uygun bir isim seçmek önemlidir.

Kod 3.2.1. Veri tipleri örnek kodları

```
#Veri Tipleri
#Tam Sayı (int)
x=17                # x adlı değişkene 17 tam sayı atanır
print(x)            # x değişkenin değerini ekrana yazar
print(type(x)) # Veri tipini ekrana yazar

#Ondalıklı (float)
x=17.23             # x adlı değişkene 17,23 ondalıklı değer atanır
print(x)            # x değişkenin değerini ekrana yazar
print(type(x)) # Veri tipini ekrana yazar

#Metin (str)
x = "Merhaba Dünya" # x adlı değişkene "Merhaba Dünya" metin değeri atanır
print(x)                  # x değişkenin değerini ekrana yazar
print(type(x))       # Veri tipini ekrana yazar

x="17"                  # x adlı değişkene 17 metin değeri atanır
print(x)            # x değişkenin değerini ekrana yazar
print(type(x)) # Veri tipini ekrana yazar
```

Ekran Çıktısı 3.2.1. Veri tipleri örnek kodları çıktısı

```
17
<class 'int'>

17.23
<class 'float'>

Merhaba Dünya
<class 'str'>

17
<class 'str'>
```

Öğrencilerin yazılım geliştirme ortamında kodları yazmanları ve ekrana yazdırmaları istenir. Tüm öğrenciler etkinliklerini tamamladıktan sonra kontrol edilir varsa öğrencilerin hataları düzeltilir ve geri dönütler verilir.

Bu aşama da öğrencilere: "*Kod 3.2.1.'te* ***x*** *adında bir değişken oluşturduk ve bu değişkene sırasıyla farklı veri tiplerinde farklı değerler verdik. Python programlama dilinde verdiğimiz değerlerin hangi veri tipine ait olduğunu ayrıca belirtmedik. Python hangi veri tipine ait olduğunu sizin yazmış olduğunuz veri tipi formatına göre kullanır. Örneğin tam sayı değeri olarak* ***x=17*** *belirtilirken,* ***x="17"*** *olarak belirttiğimiz*

değerindeki 17 bir metin ifadesi olarak algılanır. Çünkü metin ifadeleri ya çift tırnak içinde "metin ifadesi" ya da tek tırnak işareti 'metin ifadesi' şeklinde yazılması gerekir, bu durumu değişkenlerde verileri atarken Tablo 3.2.1 örnekler üzerinde detaylı bir şekilde bahsetmiştik." Eğer veri tiplerini değişkenlere atarken doğru formatta belirtmezsek hem işlem yaparken sorun yaşarız, hem de çalıştırdığımız zaman hata alabiliriz." denilir.

Son aşamada öğrencilerden "Boy Kilo" uygulaması geliştirmeleri istenir ve etkinlik başlangıç kod kısmı öğrencilerle paylaşılır uygulama geliştirirken istenen yönergeye göre tamamlamaları istenir.

Uygulama başlangıç kodu:

```
sınıfAdı= "Bilişim"
ad1=input("Kişi adı giriniz ")
kilo1=int(input("Kişinin Kilosunu giriniz "))
boy1=float(input("Kişinin Kilosunu giriniz "))
```

Uygulama Yönergesi

Öğrencilere her öğrenci verisi için kendisi ve 3 arkadaşı için veri toplaması gerektiği belirtilecektir;

1. Sınıf adı kodlama sırasında sabit değer olarak atanacaktır.
2. Her öğrenci verisi için örneğin "ad1" , "boy1" ve "kilo1" benzeri kurallara uygun ve anlaşılır bir şekilde değişkenler oluşturacak,
3. Toplam 4 kişi için oluşturulan değişkenlere dışarıdan veri girişi yapılacak,
4. Toplam 4 öğrencinin verilerini (ad, boy ve kilo) bilgileri sırasıyla ekrana alt alta yazılacak

şeklinde yönerge verildikten sonra kodu oluşturmaları ve ekrana yazdırmaları istenir. Tüm öğrenciler etkinliklerini tamamladıktan sonra kontrol edilir varsa eksik kısım öğretmen tarafından gösterilir.

Etkinliğin tamamlanmasının ardından öğretmen tarafından dersin hedeflerini vurgulayan, etkinliği özetleyen ve sonraki etkinlik hakkında kısa bilgi veren ders sonu konuşması yapılır:

"Çocuklar bugün sizinle değişken kavramını öğrendik ve uygulamalar yaptık. Bir sonraki derste listeler kavramını öğreneceğiz." denilir ve ders sonlandırılır.

DEĞERLENDİRME:

Öğrencilerin kavramları içselleştirmeleri için günlük yaşamdan sabit ve değişkenlere örnekler vermeleri istenir. Daha sonra öğrencilerden değişken ve sabit kavramlarını açıklamaları beklenir. Kavram yanılgıları varsa düzeltilir.

Öğretmen, öğrencilerden gelen cevaplara göre ek etkinlik yapılabilir.

Yapılan kodlamalar ve soruya verilen cevaplar Tablo B.3.2.1'de verilen kontrol listesi kullanılarak değerlendirilir.

Tablo 3.2.1: Kontrol listesi

Kontrol Listesi	Evet	Hayır
Değişken kavramını açıklar.		
Sabit kavramını açıklar.		
Değişkenlerde kullanılan farklı veri tiplerine örnekler verir.		
Değişken ve sabit değer arasında farkı bilir		
Bir kod örneğindeki değişkenleri tespit eder.		

ETKİNLİK | 3

ETKİNLİK NO	**3.3**
ETKİNLİK ADI	LİSTELERLE ÇALIŞMA
SINIF/KADEME	Lise
SÜRE	40 + 40 + 40 + 40=160 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Listeler
KAZANIMLAR	**3.3.1.** Liste kavramını açıklar. **3.3.2.** Liste ile değişken arasındaki farkı açıklar. **3.3.3.** Listeler üzerinde indeksleme ve dilimleme işlemlerini yapar.
TEMEL BECERİLER	Problem çözme, Analitik düşünme, Keşfetme
YÖNTEM VE TEKNİKLER	Doğrudan anlatım, Soru-cevap, Gösterip yaptırma
ARAÇ-GEREÇLER	Bilgisayar, Uygulama geliştirme ortamı, internet, Projeksiyon cihazı/ Etkileşimli tahta
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında yüklü ya da çevrimiçi kullanabileceği bir uygulama geliştirme ortamı olması gerekir. ✓ Ders içerisinde kullanılacak Veri, Liste, İndeks kavramlarına ilişkin farklı tanımlar öğretmen tarafından incelenir. ✓ Her öğrencinin bir bilgisayar karşına oturması sağlanır.
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma. ✓ Öğrenciler bir programlama dilinde var olan temel girdi ve çıktı fonksiyonlarının kullanımını bilir. ✓ Kodlamada değişken kullanması

SÜREÇ

Dersin başında öğrencilere:

"Tek bir metni ya da sayıyı değişken tanımlayarak hafızada tutmayı öğrendik. Peki sınıftaki tüm arkadaşlarınızın adını tanımlamak gerekirse nasıl yaparız?" sorusu yöneltilerek, öğrencilerden cevaplar alınır. Öğrencilerden öğrenci sayısı kadar değişken oluşturulur (değişken1, değişken2 vb.) yanıtları beklenir. Öğrencilerden yanıt alınmadığı takdirde öğretmen, "*Arkadaşlar daha önceki dersler de iki farklı veri tipi görmüştük. Bunlardan biri karakter dizileri, öteki ise sayılardı. Python'da karakter dizileri ve sayıların dışında, başka amaçlara hizmet eden, başka veri tipleri de vardır. Bunlardan bir tanesi de içerisinde hem aynı türden veriyi hem de farklı veri tipini tutabilen listeler",* denilerek öğrencilerin hedeften haberdar edilmesi sağlanır.

Dikkat çekme sorularını takiben aşağıdaki sırayla etkinliğe giriş yapılır.

Etkinliğin iyi anlaşılması, düzenli ve disiplinli ilerlemesini sağlayabilmek adına öğrencilerin öncelikle

etkinliği etkileşimli tahtada gösterilirken dikkatlice dinlemesi, anlatım bittikten sonra öğretmenin söylediği zamanda uygulamaya geçilerek yazılım geliştirme ortamında uygulaması istenir.

Kod 3.3.1. de verilen kodlar, yazılım geliştirme ortamında yazılır ve öğrencilere açıklanır. Tüm öğrencilerin Ekran çıktısı 3.3.1'deki gibi ekran çıktıları elde etmeleri beklenir. Varsa anlaşılmayan kısımlar öğrencilere tekrar anlatılır.

Kod 3.3.1. Liste oluşturma kodları

```
# boş liste örneği
bos=[ ]
print (bos)
# Karakter dizilerinden oluşan liste örneği
ad = ["Bekir", "Demir", "Aysel"]
print (ad)
# Tam sayılardan oluşan liste örneği
tam = [1, 2, 3]
print (tam)
# Ondalıklı sayılardan oluşan liste örneği
ondalık = [12.3, 123.45, 98.76]
print (ondalık)
# Farklı veri tiplerinden oluşan liste örneği
hepsi=["Demir",1,12.3,"Bekir",98.76]
print (hepsi)
```

Ekran Çıktısı 3.3.1. Liste oluşturma kodları çıktısı

```
[]
['Bekir', 'Demir', 'Aysel']
[1, 2, 3]
[12.3, 123.45, 98.76]
['Demir', 1, 12.3, 'Bekir', 98.76]
```

Bu aşamada öğrencilerden sırasıyla aile bireylerinin isimlerinin ve yaşlarının olduğu farklı veri tiplerinden oluşan bir listenin kodunu oluşturmaları ve ekrana yazdırmaları istenir. Tüm öğrenciler etkinliklerini tamamladıktan sonra kontrol edilir varsa eksik kısım öğretmen tarafından gösterilir.

Daha sonra öğrencilere:

"Python'da listeler üzerinde değişik türden verilerin bir arada olabileceğini öğrendik. Bu listelerde her bir eleman bir index (indeks) numarasına sahiptir ve bir listenin başlangıç indeks 0'dır. Listelerdeki her veriyi erişebilmek için indeks değerlerini kullanırız" denilerek Kod 3.3.2. de verilen kodlar yazılım geliştirme ortamında yazılır ve öğrencilere açıklanır. Tüm öğrencilerin Ekran çıktısı 3.3.2'deki gibi ekran çıktıları elde etmeleri beklenir. Varsa anlaşılmayan kısımlar öğrencilere tekrar anlatılır.

Kod 3.3.2 Liste indekslerine erişim

```
#Karakter dizilerinden oluşan liste örneği
ad = ["Bekir", "Demir", "Aysel"]
```

```
#Tüm listeti ekranda göstermek için
print (ad)
#Sadece ilk veriyi indeks 0 olanı "Bekir" göstermek için
print(ad[0])
#Sadece İkinci veriyi indeks 1 olanı "Demir" göstermek için
print(ad[1])
```

Ekran Çıktısı 3.3.2 Liste indekslerine erişim çıktısı

```
['Bekir', 'Demir', 'Aysel']
Bekir
Demir
```

Bu aşamada öğrencilerden sırasıyla aile bireylerinin isimlerinin ve yaşlarının olduğu farklı veri tiplerinden oluşturdukları kişileri ve yaşlarını indeks sırasıyla erişim sağlayan kodu oluşturmaları ve ekrana yazdırmaları istenir. Tüm öğrenciler etkinliklerini tamamladıktan sonra kontrol edilir varsa eksik kısım öğretmen tarafından gösterilir.

En son aşamada Python'da listeler üzerinde birden fazla veriye erişebilmek için listelerde dilimleme işlemlerini anlatmak amacıyla Kod 3.3.3. de verilen kodlar yazılım geliştirme ortamında yazılır ve öğrencilere açıklanır. Tüm öğrencilerin Ekran çıktısı 3.3.3'deki gibi ekran çıktıları elde etmeleri beklenir. Varsa anlaşılmayan kısımlar öğrencilere tekrar anlatılır.

Kod 3.3.3 Listelerde dilimleme işlemleri

```
sayı = [1,2,3,4,5,6,7,8,9]
print(sayı)
# 1. Eleman
print(sayı[0])
# 4. Eleman
print(sayı[3])
# 3.indeksten 8.indekse kadar (dahil değil)
print(sayı[3:8])
# Baştan 4. indekse kadar
print(sayı[:4])
# 6.indeksten en sona kadar
print(sayı[5:])
# Baştan sona bir aralıkla 2'şer
print(sayı[::2])
# En sonuncu indeks
print(sayı[-1])
# Sondan bir önceki indeks
print(sayı[-2])
```

Ekran Çıktısı 3.3.3 Listelerde dilimleme işlemleri çıktısı

```
[1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
1
4
[4, 5, 6, 7, 8]
[1, 2, 3, 4]
[6, 7, 8, 9]
[1, 3, 5, 7, 9]
9
8
```

Bu aşamada öğrencilerden sırasıyla aile bireylerinin isimlerinin ve yaşlarının olduğu farklı veri tiplerinden oluşturdukları kişileri ve yaşlarını öğretmenin belirteceği dilimleme sırasına göre erişim sağlayan kodu oluşturmaları ve ekrana yazdırmaları istenir. Tüm öğrenciler etkinliklerini tamamladıktan sonra kontrol edilir varsa öğrencilerin hataları düzeltilir ve geri dönütler verilir.

Etkinliğin tamamlanmasının ardından öğretmen tarafından öğrencilere: "*Liste nedir? Liste kavramı size neyi anlatıyor, değişken ile farkları ve benzerlikleri nelerdir?*" soruları sorulur ve öğrencilerden alınan cevaplar üzerinde konuşulur. Gerekli dönütler sağlanır. Öğretmen daha sonra dersin hedeflerini vurgulayan, etkinliği özetleyen ve sonraki etkinlik hakkında kısa bilgi veren ders sonu konuşması yapılır:

"*Çocuklar bugün sizinle liste kavramını, liste ve değişkenler arasındaki farkları ve listeler üzerinde indeksleme ve dilimleme işlemlerini öğrendik ve uygulamalar yaptık. Bir sonraki derste listelerde farklı metot işlemleriyle çalışma yapmayı öğreneceğiz.*" denilir ve ders sonlandırılır.

DEĞERLENDİRME:

Öğrencilerin kavramları içselleştirmeleri için uygulama geliştirirken bir problem çözümünde kullanılabilecekleri liste yapılarından örnekler vermeleri istenir. Daha sonra öğrencilerden liste yapısı, indeksleme ve dilimleme kavramını açıklamaları beklenir. Kavram yanılgıları varsa düzeltilir.

Öğretmen, öğrencilerden gelen cevaplara göre ek etkinlik yapılabilir.

Yapılan kodlamalar ve soruya verilen cevaplar Tablo B.3.3.1'de verilen kontrol listesi kullanılarak değerlendirilir.

Tablo 3.3.1: Kontrol listesi

Kontrol Listesi	Evet	Hayır
Liste kavramını açıklar.		
Liste ile değişken arasındaki farkı açıklar.		
İndeks kavramını açıklar.		
Listeler üzerinde indeksleme işlemlerini yapar.		
Listeler üzerinde dilimleme işlemlerini yapar.		

ETKİNLİK | 4

ETKİNLİK NO	**3.4**
ETKİNLİK ADI	LİSTELERDE FARKLI METOT İŞLEMLERİYLE ÇALIŞMA
SINIF/KADEME	Lise
SÜRE	40 + 40 =80 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Listeler
KAZANIMLAR	**3.3.4.** Listeler üzerinde farklı metot işlemlerini gerçekleştirir.
TEMEL BECERİLER	Problem çözme, Analitik düşünme, Keşfetme
YÖNTEM VE TEKNİKLER	Doğrudan anlatım, Soru-cevap, Gösterip-yaptırma
ARAÇ-GEREÇLER	Bilgisayar, Uygulama geliştirme ortamı, internet, Projeksiyon cihazı/ Etkileşimli tahta
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında yüklü ya da çevrimiçi kullanabileceği bir uygulama geliştirme ortamı olması gerekir. ✓ Ders içerisinde kullanılacak Liste, Metot kavramlarına ilişkin farklı tanımlar öğretmen tarafından incelenir. ✓ Her öğrencinin bir bilgisayar karşına oturması sağlanır.
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma. ✓ Öğrenciler bir programlama dilinde var olan temel girdi ve çıktı fonksiyonlarının kullanımını bilir. ✓ Kodlamada liste ve listelerde dilimleme ve indeksleme işlemlerini yapar.

SÜREÇ

Dersin başında öğrencilere:

"Bir önceki etkinlikte Liste ile değişken arasındaki farkları, Listeler üzerinde indeksleme ve dilimleme işlemlerini öğrendik ve uyguladık. Bir program yazarken kodlama sırasında oluşturduğumuz listeler sabit olan verilerdir. Peki uygulama çalıştırdıktan sonra, bir ihtiyacımız olduğunda boş ya da hazır verileri olan bir liste üzerinde hangi işlemleri yapmamız gerekli olabilir?" sorusu yöneltilerek, öğrencilerden cevaplar alınır. Öğrencilerden listeye *"yeni bir veri ekleyebiliriz, silebiliriz ya da değiştirebiliriz. Listelerde aradığımız verileri bulmak için arama yaparız vb."*yanıtlar beklenir. Öğrencilerden yanıt alınmadığı takdirde öğretmen öğrencilere:

"Arkadaşlar ister boş liste olsun, ister hazır bir liste, program çalıştığı zaman listelere yeni bir veri eklemek, var olan bir veriyi silmek ya da değiştirmek gerekebilir. Ya da bir liste içinde belli bir veriyi bulmak için arama yapmamız gerekebilir. Bugünkü etkinliğimizde size listeler üzerinde farklı metot işlemlerini gerçekleştireceğiz.", diyerek öğrencilerin hedeften haberdar edilmesi sağlanır.

Dikkat çekme sorularını takiben aşağıdaki sırayla etkinliğe giriş yapılır.

Etkinliğin iyi anlaşılması, düzenli ve disiplinli ilerlemesini sağlayabilmek adına öğrencilerin öncelikle etkinliği etkileşimli tahtada gösterilirken dikkatlice dinlemesi, anlatım bittikten sonra öğretmenin söylediği zamanda uygulamaya geçilerek yazılım geliştirme ortamında uygulaması istenir.

Öğretmen Python'da bütün veri tipleri bize birtakım metotlar sunar. Bu metotlar yardımıyla, ilgili veri tipi üzerinde önemli değişiklikler veya sorgulamalar yapabiliyoruz diyerek Tablo 3.4.1'ü gösterip açıklar. Daha sonra liste metotları örnek kodları Kod 3.4.1, Kod 3.4.2, Kod 3.4.3, Kod 3.4.4, Kod 3.4.5 sırasıyla yazılım geliştirme ortamında öğretmen tarafından gösterilir ve açıklanır daha sonra öğrencilerinde kodları yazması ve çalıştırması istenir. Varsa anlaşılmayan kısımlar öğrencilere tekrar anlatılır.

Tablo 3.4.1. Liste Metotları

append ( )	Bir listeye yeni bir öğe eklemek için kullanılır.
clear ( )	Bir listenin içeriğini siler.
copy ( )	Bir listeyi kopyalamak için kullanılır.
count ( )	Bir liste içinde aynı adı taşıyan öğenin kaç kez tekrar edildiğini belirtir.
extend ( )	Bir listeye başka bir liste ile birleştirerek tek liste olarak genişletir.
index ( )	Bir liste içinde öğenin indis sırasını belirtir.
insert ( )	Bir listede yeni bir öğeyi istediğimiz indis sırasına eklemek için kullanırız.
pop ( )	Bir listede indis sırasını bildiğimiz öğeyi silmemizi sağlar.
remove ( )	Bir listede adını bildiğimiz öğeyi silmemizi sağlar.
reverse ( )	Bir listenin içeriğini indis sırasına göre tersine çevirir.
sort ( )	Bir listede öğeleri A-Z ya da Z-A şeklinde sıralamak için kullanırız

Tüm öğrencilerin kodları çalıştırması sonucunda Ekran çıktısı 3.4.1, Ekran çıktısı 3.4.2, Ekran çıktısı 3.4.3, Ekran çıktısı 3.4.4, Ekran çıktısı 3.4.5'deki gibi ekran çıktıları elde etmeleri beklenir. Varsa anlaşılmayan kısımlar konu öğrencilere tekrarlanır.

Kod 3.4.1. Liste metotları örnek kodları

```
#Liste Metotları
# append()
sayı = [1,2,3,4]
harf = ["a","b","c","d"]
print(sayı)
print(harf)
sayı.append(5)
harf.append("e")
print ("append: Bir listeye yeni bir öğe eklemek için kullanılır. ")
print(sayı)
print(harf)
```

```
print ("------------------------------------------------ ")

#clear()
sayı = [1,2,3,4]
print(sayı)
sayı.clear()
print ("clear: Bir listenin içeriğini siler. ")
print(sayı)
print ("------------------------------------------------ ")
```

Ekran Çıktısı 3.4.1. Liste metotları örnek kodları çıktısı

```
[1, 2, 3, 4]
['a', 'b', 'c', 'd']
append: Bir listeye yeni bir öğe eklemek için kullanılır.
[1, 2, 3, 4, 5]
['a', 'b', 'c', 'd', 'e']
------------------------------------------------
[1, 2, 3, 4]
clear: Bir listenin içeriğini siler.
------------------------------------------------
```

Kod 3.4.2 Liste metotları örnek kodları

```
#Liste Metotları
#copy()
sayı = [1,2,3,4,2]
sayı2=[]
print(sayı)
print(sayı2)
sayı2=sayı.copy()
print ("copy: Bir listeyi kopyalamak için kullanılır. ")
print(sayı2)
print ("------------------------------------------------ ")

#count()
sayı = [1,2,3,4,2]
harf = ["a","b","a","d","a"]
print(sayı)
print(harf)
print ("count: Bir liste içinde aynı adı taşıyan öğenin kaç kez tekrar edildiğini belirtir. ")
print(sayı.count(2))
print(harf.count("a"))
```

```
print ("------------------------------------------------ ")
```

Ekran Çıktısı 3.4.2 Liste metotları örnek kodları çıktısı

```
[1, 2, 3, 4, 2]
[]
copy: Bir listeyi kopyalamak için kullanılır.
[1, 2, 3, 4, 2]
------------------------------------------------
[1, 2, 3, 4, 2]
['a', 'b', 'a', 'd', 'a']
count: Bir liste içinde aynı adı taşıyan öğenin kaç kez tekrar edildiğini belirtir.
2
3
------------------------------------------------
```

Kod 3.4.3 Liste metotları örnek kodları

```
#Liste Metotları

#extend()
sayı = [1,2,3,4]
harf = ["a","b","c","d"]
print(sayı)
print(harf)
sayı.extend(harf)
print("Bir listeye başka bir liste ile birleştirerek tek liste olarak genişletir. ")
print (sayı)
print ("------------------------------------------------ ")


#index()
harf = ["a","b","c","d"]
print(harf)
print("Bir liste içinde öğenin indis sırasını belirtir.")
print(harf.index("c"))
print ("------------------------------------------------ ")
```

Ekran Çıktısı 3.4.3 Liste metotları örnek kodları çıktısı

```
[1, 2, 3, 4]
['a', 'b', 'c', 'd']
Bir listeye başka bir liste ile birleştirerek tek liste olarak genişletir.
```

```
[1, 2, 3, 4, 'a', 'b', 'c', 'd']
------------------------------------------------
['a', 'b', 'c', 'd']
Bir liste içinde öğenin indis sırasını belirtir.
2
------------------------------------------------
```

Kod 3.4.4 Liste metotları örnek kodları.

```
#Liste Metotları

#insert()
sayı = [1,2,4,5]
print (sayı)
sayı.insert(2,3)
print("Bir listede yeni bir öğeyi istediğimiz indis sırasına eklemek için kullanırız.")
print (sayı)
print ("------------------------------------------------ ")

#pop()
harf = ["a","b","a","c","d"]
print (harf)
harf.pop(2)
print("Bir listede indis sırasını bildiğimiz öğeyi silmemizi sağlar.")
print(harf)
print ("------------------------------------------------ ")

#remove()
harf = ["a","b","a","c","d"]
print(harf)
harf.remove("a")
print("Bir listede adını bildiğimiz öğeyi silmemizi sağlar.")
print(harf)
print ("------------------------------------------------ ")
```

Ekran Çıktısı 3.4.4 Liste metotları örnek kodları çıktısı

```
[1, 2, 4, 5]
Bir listede yeni bir öğeyi istediğimiz indis sırasına eklemek için kullanırız.
[1, 2, 3, 4, 5]
------------------------------------------------
['a', 'b', 'a', 'c', 'd']
```

```
Bir listede indis sırasını bildiğimiz öğeyi silmemizi sağlar.
['a', 'b', 'c', 'd']
------------------------------------------------
['a', 'b', 'a', 'c', 'd']
Bir listede adını bildiğimiz öğeyi silmemizi sağlar.
['b', 'a', 'c', 'd']
------------------------------------------------
```

Kod 3.4.5 Liste metotları örnek kodları

```
#Liste Metotları
#reverse()
sayı = [1,2,3,4]
print (sayı)
sayı.reverse()
print("Bir listenin içeriğini indis sırasına göre tersine çevirir.")
print(sayı)
print ("------------------------------------------------ ")

#sort()
sayı = [3,2,1,4]
print (sayı)
sayı.sort()
print(sayı)
print("Bir listede öğeleri A-Z ya da Z-A şeklinde sıralamak için kullanırız ")
harf = ["a","b","c","d"]
print(harf)
harf.sort(reverse=True)
print(harf)
print ("------------------------------------------------ ")
```

Ekran Çıktısı 3.4.5 Liste metotları örnek kodları çıktısı

```
[1, 2, 3, 4]
Bir listenin içeriğini indis sırasına göre tersine çevirir.
[4, 3, 2, 1]
------------------------------------------------
[3, 2, 1, 4]
[1, 2, 3, 4]
Bir listede öğeleri A-Z ya da Z-A şeklinde sıralamak için kullanırız
['a', 'b', 'c', 'd']
['d', 'c', 'b', 'a']
```

Bu aşamada öğrencilerden "Telefon Rehberi" uygulaması geliştirmeleri istenir ve etkinlik başlangıç kod kısmı öğrencilerle paylaşılır uygulama geliştirirken istenen yönergeye göre tamamlamaları istenir.

Uygulama başlangıç kodu:

```
ad=["Bekir", "Demir", "Aysel" ]
telefon=[5051234567, 5321234567,5441234567 ]
isim=input("Kişi adı giriniz ")
tel=int(input("Telefon giriniz "))
```

Uygulama Yönergesi

Öğrencilere "ad" ve "telefon" isimli listelerde;

1. Örnek olarak verilen kullanıcıdan veri alarak ad ve telefon numaralarını listeye eklenecek,
2. "Bekir" adlı kişinin ismini ve telefonu listelerden silinecek,
3. İki listeyi de tersten sıralayacak,

kodu oluşturmaları ve ekrana yazdırmaları istenir. Tüm öğrenciler etkinliklerini tamamladıktan sonra kontrol edilir varsa öğrencilerin hataları düzeltilir ve geri dönütler verilir.

Etkinliğin tamamlanmasının ardından öğretmen tarafından dersin hedeflerini vurgulayan, etkinliği özetleyen ve sonraki etkinlik hakkında kısa bilgi veren ders sonu konuşması yapılır:

"B*u etkinliğimizde Python'da listeler üzerinde farklı metotlarla işlem yapmayı* öğrendik. *Bu sayede bir listeye öğe ekleme, öğe silme ve öğeler üzerinde düzenleme işlemi yaparak farklı uygulamalar için çalışmalar yaptık" Bir sonraki derste Operatörler ile* çalışma yapmayı öğreneceğiz." denilir ve ders sonlandırılır.

DEĞERLENDİRME

Öğrencilerin kavramları içselleştirmeleri için uygulama geliştirirken bir problem çözümünde kullanılabilecekleri liste metotlarından örnekler vermeleri istenir. Daha sonra öğrencilerden listeler kullanılan farklı metot kavramını açıklamaları beklenir. Kavram yanılgıları varsa düzeltilir.

Öğretmen, öğrencilerden gelen cevaplara göre ek etkinlik yapılabilir.

Yapılan kodlamalar ve soruya verilen cevaplar Tablo B.3.4.2'de verilen kontrol listesi kullanılarak değerlendirilir.

Tablo 3.4.2 Kontrol listesi

Kontrol Listesi	Evet	Hayır
Listelerde metot kavramını açıklar.		
Listeler üzerinde **append ( )** metoduyla işlemler yapar.		
Listeler üzerinde **clear ( )** metoduyla işlemler yapar.		
Listeler üzerinde **copy ( )** metoduyla işlemler yapar.		
Listeler üzerinde **count ( )** metoduyla işlemler yapar.		
Listeler üzerinde **extend ( )** metoduyla işlemler yapar.		
Listeler üzerinde **index ( )** metoduyla işlemler yapar.		
Listeler üzerinde **insert ( )** metoduyla işlemler yapar.		
Listeler üzerinde **pop ( )** metoduyla işlemler yapar.		
Listeler üzerinde **remove ( )** metoduyla işlemler yapar.		
Listeler üzerinde **reverse ( )** metoduyla işlemler yapar.		
Listeler üzerinde **sort ( )** metoduyla işlemler yapar.		
Listeler üzerinde metotları kullanarak uygulama geliştirir.		

ETKİNLİK | 5

ETKİNLİK NO	**3.5**
ETKİNLİK ADI	OPERATÖRLER
SINIF/KADEME	Lise
SÜRE	40 + 40 =80 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Operatörler
KAZANIMLAR	**3.4.1.** Operatör kavramını açıklar. **3.4.2.** Aritmetik operatörleri kullanır. **3.4.3.** Atama operatörlerini kullanır. **3.4.4.** Mantıksal ve Karşılaştırma operatörlerini kullanır.
TEMEL BECERİLER	Problem çözme, Analitik düşünme, Keşfetme
YÖNTEM VE TEKNİKLER	Doğrudan anlatım, Soru-cevap, Gösterip-yaptırma
ARAÇ-GEREÇLER	Bilgisayar, Uygulama geliştirme ortamı,
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında yüklü ya da çevrimiçi kullanabileceği bir uygulama geliştirme ortamı olması gerekir. ✓ Ders içerisinde kullanılacak Operatör, Aritmetiksel Operatör, Atama Operatörü, Mantıksal Operatör, Karşılaştırma Operatörü gibi kavramlarına ilişkin farklı tanımlar öğretmen tarafından incelenir. ✓ Her öğrencinin bir bilgisayar karşına oturması sağlanır. ✓ **Ek 3.5.1:** Operatör seçimi çalışma kağıdı ve cevapları, ✓ **Ek 3.5.2:** Mantıksal operatörler çalışma kağıdı ve cevapları
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma. ✓ Öğrenciler bir programlama dilinde var olan temel girdi ve çıktı fonksiyonlarının kullanımını bilir. ✓ Kodlamada değişken kullanması

SÜREÇ

Dersin başında öğrencilere *"operatörden ne anladıkları sorulur?" Daha sonra "matematikte kullandığınız operatör veya operatörler var mıdır? Hangileri vardır, kullanım şekli nasıldır ve görevleri nedir?"* soruları yöneltilir ve alınan cevaplar tahtaya aşağıdaki gibi tablo şeklinde yazılır.

Tablo 3.5.1 Aritmetiksel Operatörler

Operatör	Kullanımı	Görevi	Sonuç
A=5 B=3 ise			
+	A+B	Toplama	8
-	A-B	Çıkarma	2
*	A*B	Çarpma	15
/	A/B	Bölme	~1,67
**	A**B	Kuvvet	125
//	A//B	Tam Sayı Bölme	1
%	A%B	Mod Alma	2

Dikkat çekme soruları ve Tablo 3.5.1'in öğrencilerle beraber doldurulmasına takiben öğrencilere:

"Operatör kelimesini bana tanımlayabilir misiniz?" denilerek operatörün tanımı öğrencilerin verdiği cevaplar doğrultusunda yapılır. Ardından öğrencilere "*operatörlerin çeşitleri var mıdır? Varsa nelerdir?*" soruları sorulur. Alınan cevaplar tahtaya yazılır. Öğrencilere Ek 3.5.1 verilerek, operatörleri gruplamaları istenir. Daha sonra tercih edilen programlama dilindeki operatör çeşitleri (aritmetiksel, atama, karşılaştırma, mantıksal operatörler) öğrencilere aktarılır. Öğrencilerin verdiği cevaplar kontrol edilerek geri dönütler verilir. Ardından öğrencilere:

"Sizden klavyeden girilen iki sayının toplamını bulan programı yazmanızı istiyorum" denilerek öğrencilerden Kod 3.5.1'i yazmaları beklenir. Öğrencilerin yazdıkları program kontrol edilir. Varsa programlarındaki eksik kalan yerler tamamlanır. Tüm öğrencilerden Ekran çıktısı 3.5.1'deki gibi bir ekran çıktısı elde etmeleri beklenir.

Kod 3.5.1 Toplama işleminin kodları

```
1. A=int(input("Lütfen birinci sayıyı giriniz-> "))
2. B=int(input("Lütfen ikinci sayıyı giriniz-> "))
3. C=A+B
4. print("İki sayının toplamı = ",C)
```

Ekran çıktısı 3.5.1 Toplama işleminin ekran çıktısı

```
1. Lütfen birinci sayıyı giriniz-> 5
2. Lütfen ikinci sayıyı giriniz-> 3
3. İki sayının toplamı = 8
```

Görevi tamamlayan öğrencilere "*birinci sayının ikinci sayıya göre mod değerini ekrana yazdıran programı yazınız*" denilerek, öğrencilerden Kod 3.5.2'yi yazmaları beklenir. Öğrencilerin yazdıkları program kontrol edilir. Varsa programlarındaki eksik kalan yerler tamamlanır.

Kod 3.5.2 Mod alma işleminin kodu

```
1. A=int(input("Lütfen birinci sayıyı giriniz-> "))
2. B=int(input("Lütfen ikinci sayıyı giriniz-> "))
3. C=A%B
4. print("İşlemin sonucu = ",C)
```

Ekran çıktısı 3.5.2 Mod alma işleminin ekran çıktısı

```
1. Lütfen birinci sayıyı giriniz-> 5
2. Lütfen ikinci sayıyı giriniz-> 3
3. İşlemin sonucu = 2
```

Öğrencilere :

"Sizlerden ilk örneğimizde toplama işlemi yapmanızı istemiştim. Şimdi de sizden çıkarma işlemini yapmanızı istiyorum ancak bu sefer atama operatörlerini kullanmanızı istiyorum" denilerek öğrencilerden Kod 3.5.3'ü yazmaları beklenir.

Kod 3.5.3 Çıkarma işlemi kodu – Atama operatörü

```
1. A=int(input("Lütfen birinci sayıyı giriniz-> "))
2. B=int(input("Lütfen ikinci sayıyı giriniz-> "))
3. A-=B
4. print("İki sayının farkı = ",A)
```

Ekran çıktısı 3.5.3 Çıkarma işlemi ekran çıktısı – Atama operatörü

```
1. Lütfen birinci sayıyı giriniz-> 5
2. Lütfen ikinci sayıyı giriniz-> 3
3. İki sayının farkı = 2
```

Görevi tamamlayan öğrencilere "*birinci sayıyı ikinci sayıya tam sayı olarak bölen işlemin değerini ekrana yazdıran programı yazınız*" denilerek, öğrencilerden Kod 3.5.4'ü yazmaları beklenir. Öğrencilerin yazdıkları program kontrol edilir. Varsa programlarındaki eksik kalan yerler tamamlanır.

Kod 3.5.4 Tam Sayı Bölme operatörü kodları

```
1. A=int(input("Lütfen birinci sayıyı giriniz-> "))
2. B=int(input("Lütfen ikinci sayıyı giriniz-> "))
3. A//=B
4. print("İki sayının tam sayı bölme sonucu = ",A)
```

Ekran çıktısı 3.5.4 Tam Sayı Bölme operatörü ekran çıktısı

```
1. Lütfen birinci sayıyı giriniz-> 5
2. Lütfen ikinci sayıyı giriniz-> 3
3. İki sayının tam sayı bölme sonucu = 1
```

Öğrencilere :

*"Anneniz sizden marketten alınacaklar listesini, marketten almanızı istemektedir. Hazırlanan listede, Ekmek **veya** simit, peynir, kaşar, elma **veya** armut, zeytin, çay **veya** oralet **veya** kahve, salam ve sucuk almanızı gerektiği yazılıdır."* denildikten sonra Ek 3.5.2'deki çalışma kağıdı dağıtılır ve soruya uygun şekilde doldurulması istenir.

Öğrencilerin kağıtları doldurduktan sonra, doğru cevaplar verilir. Yanlış cevap veren öğrencilerin hataları düzeltilir ve geri dönütler verilir.

Etkinliğin tamamlanmasının ardından öğretmen tarafından dersin hedeflerini vurgulayan, etkinliği özetleyen ve sonraki etkinlik hakkında kısa bilgi veren ders sonu konuşması yapılır:

"Çocuklar bugün sizinle operatör, aritmetiksel operatör, atama operatörü, mantıksal operatör, karşılaştırma operatörü gibi kavramlarını öğrendik ve uygulamalar yaptık. Bir sonraki derste koşul yapıları kavramını öğreneceğiz." denilir ve ders sonlandırılır.

DEĞERLENDİRME

Öğrencilerin kavramları içselleştirmeleri için günlük yaşamda kullandıkları operatörlere örnekler vermeleri istenir. Daha sonra öğrencilerden operatör, aritmetiksel operatör, atama operatörü, mantıksal operatör, karşılaştırma operatörü kavramlarını açıklamaları beklenir. Kavram yanılgıları varsa düzeltilir.

Öğretmen, öğrencilerden gelen cevaplara göre ek etkinlik yapılabilir.

Yapılan kodlamalar ve soruya verilen cevaplar Tablo B.3.5.2'de verilen kontrol listesi kullanılarak değerlendirilir.

Tablo B.3.5.2 Kontrol listesi

Kontrol Listesi	Evet	Hayır
Operatör kavramını açıklar.		
Aritmetik operatör kavramını açıklar.		
Atama operatör kavramını açıklar.		
Mantıksal operatör kavramını açıklar.		
Karşılaştırma operatör kavramını açıklar.		
Verilen bir problemi aritmetik operatörlerini kullanarak çözer.		
Verilen bir problemi atama operatörlerini kullanarak çözer.		

EK 3.5.1- Operatör Seçimi Çalışma Kağıdı

OPERATÖRLER

Python programlama dilinde kullanılan operatörleri gruplamak istiyoruz.

Aşağıdaki kutucukların içinde bulunan operatörlerin rakamlarını, en altta bulunan gruplandırma kutucuklarının içerisine uygun yerlere yazınız

+ **1**	**==** **2**	**+=** **3**	**!=** **4**
<= **5**	*** *=** **6**	**//** **7**	**-=** **8**
****** **9**	**<** **10**	***** **11**	***=** **12**
OR (| |) **13**	**%** **14**	**//=** **15**	**/** **16**
>= **17**	**%=** **18**	**>** **19**	**AND (&&)** **20**
/= **21**	**-** **22**		

Aritmetik Operatörler	Karşılaştırma Operatörleri	Atama Operatörleri	Mantıksal Operatörler

OPERATÖRLER CEVAPLARI

Aritmetik Operatörler	Karşılaştırma Operatörleri	Atama Operatörleri	Mantıksal Operatörler
1, 7, 9, 11, 14, 16, 22	2, 4, 5, 10, 17, 19	3, 6, 8, 12, 15, 18 21	13, 20

EK 3.5.2- Mantıksal Operatörler Çalışma Kağıdı

MANTIKSAL OPERATÖRLER

*"Anneniz sizden marketten alınacaklar listesini, marketten almanızı istemektedir. Hazırlanan listede, Ekmek **veya** simit, peynir, kaşar, elma **veya** armut, zeytin, çay **veya** oralet **veya** kahve, salam ve sucuk almanızı gerektiği yazılıdır."*

Yukarıdaki yönerge doğrultusunda aşağıdaki boşlukları **mantıksal operatörleri** kullanarak doldurunuz

Bakkaldan elma ..... simit.....kaşar alabilirsin.
Bakkaldan zeytin..... çay.....armut alabilirsin.
Bakkaldan kahve..... çay.....oralet alabilirsin
Bakkaldan peynir..... armut.....elma alabilirsin.
Bakkaldan sucuk..... salam.....kaşar alabilirsin.
Bakkaldan simit..... ekmek.....salam alabilirsin.

MANTIKSAL OPERATÖRLER CEVAPLARI

Bakkaldan elma **and (&&)** simit **and (&&)** kaşar alabilirsin.
Bakkaldan zeytin **and (&&)** çay **and (&&)** armut alabilirsin.
Bakkaldan kahve **or (||)** çay **or (||)** oralet alabilirsin.
Bakkaldan peynir **and (&&)** armut **or (||)** elma alabilirsin.
Bakkaldan sucuk **and (&&)** salam **and (&&)** kaşar alabilirsin.
Bakkaldan simit **or (||)** ekmek **and (&&)** salam alabilirsin.

ETKİNLİK | 6

ETKİNLİK NO	**3.6**
ETKİNLİK ADI	KOŞUL YAPILARI
SINIF/KADEME	Lise
SÜRE	40 + 40 =80 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Koşul Yapıları
KAZANIMLAR	**3.5.1.** Mantıksal ifadeleri ihtiyacına göre seçer. **3.5.2.** Çoklu koşul yapısını kullanır. **3.5.3.** İç içe koşul yapısını kullanır.
TEMEL BECERİLER	Problem çözme, Analitik düşünme, Keşfetme
YÖNTEM VE TEKNİKLER	Doğrudan anlatım, Soru-cevap, Gösterip-yaptırma
ARAÇ-GEREÇLER	Bilgisayar, Uygulama geliştirme ortamı,
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında yüklü ya da çevrimiçi kullanabileceği bir uygulama geliştirme ortamı olması gerekir. ✓ Ders içerisinde kullanılacak koşul yapısı kavramına ilişkin farklı tanımlar öğretmen tarafından incelenir. ✓ Her öğrencinin bir bilgisayar karşına oturması sağlanır.
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma. ✓ Öğrenciler bir programlama dilinde var olan temel girdi ve çıktı fonksiyonlarının kullanımını bilir. ✓ Kodlamada değişkenleri kullanır. ✓ Temel koşul yapılarını kullanır.

SÜREÇ

Dersin başında öğrencilere:

"Kendinizi bir elektronik firmasında çalışan yazılımcı olarak düşünmenizi istiyorum. Kargo ücretlendirmelerini toplam sipariş tutarı 0-100 TL aralığında ise 25 TL; sipariş tutarı 101-250 TL aralığında ise 15 TL ve sipariş tutarı 251 TL ve 251 TL'den fazla ise ücretsiz olarak hesaplayıp kullanıcıyı bilgilendiren bir yazılım yapmanız istenmektedir." şeklinde açıklama yapılarak öğrencilerin dikkati çekilir.

Dikkat çekmeyi takiben öğrencilere:

"Hazırlayacağınız bu yazılım hakkında ilk yapmanız gereken şey nelerdir? Toplam sipariş tutarının kargonun hesaplanması için sisteme girilmesini nasıl sağlamayı düşünüyorsunuz?" denilerek öğrencilerin kodlama öncesinde yapmaları gerekenleri düşünmeleri, en iyi yolu ve yöntemi tartışmaları sağlanır. Öğrencilerden problemi çözecek işlem basamaklarını oluşturmaları beklenir. Belirlenen işlem basamakları aşağıdaki basamaklarla benzer olması beklenir:

1. Girilen tutar 0-100 TL arasında ise kargo ücreti 25 TL olacaktır. Ekrana ise, örneğin sipariş 45 TL ise;

"Sipariş tutarı : 45 TL
Kargo tutarı: 25 TL
Toplam ödenecek ücret : 70 TL"
şeklinde yazmalı.

2. Girilen tutar 101-250 TL arasında ise kargo ücreti 15 TL olacaktır. Ekrana ise, örneğin sipariş 145 TL ise;

"Sipariş tutarı : 145 TL
Kargo tutarı: 15 TL
Toplam ödenecek ücret : 160 TL"
şeklinde yazmalı.

3. Girilen tutar 251 TL ve üzeri ise kargo ücreti ücretsiz olacaktır. Ekrana ise, örneğin sipariş 450 TL ise;

"Sipariş tutarı : 450 TL
Kargo tutarı: 0 TL (Ücretsiz)
Toplam ödenecek ücret : 450 TL"
şeklinde yazmalı.

İşlem basamakları öğrencilerle birlikte çıkarıldıktan sonra, öğrencilere koşul ifadesinin ne olduğu ve kullanım şekli öğrencilere anlatılır.

If kullanımının önce basit bir programda nasıl kullanıldığı öğrencilere anlatılır. Örnek problem olarak;

"Klavyeden girilen ders notu 50'den büyükse ekrana "Dersten Geçti", küçükse "Dersten Kaldı" yazısını ekrana yazan program" verilir.

Kod 3.6.1 Girilen ders notunun 50'den büyükse ekrana "Dersten Geçti", küçükse "Dersten Kaldı" yazısını ekrana yazan programa ait kodlar

```
1. baraj = 50
2. dersnotu = int(input("Lütfen ders notunu giriniz->"))
3. if dersnotu < baraj:
4.         print("Dersten Kaldı")
5. else:
6.         print("Dersten Geçti")
```

Kod 3.6.1 tamamlandıktan sonra öğrencilere:

"Klavyeden girilen iki sayı arasındaki ilişkiyi kontrol eden bir program yazmak istiyoruz. Eğer ilk sayı ikincisinden küçük ise, ikinci sayının 15'ten büyük olup olmadığını kontrol edeceğiz ve sonuçlara göre çıktı vereceğiz. Eğer ikinci sayı 15'ten büyükse " ikinci sayı, 15'ten büyüktür" yazısını, küçükse " ikinci sayı, 15'ten küçüktür" yazısını ekrana yazdıracağız. İlk sayı ikinci sayıdan büyük veya eşitse, "İlk sayı, ikinci sayıya eşit veya büyüktür" yazısını ekrana yazan programı yazınız." denilerek, öğrencilerin kodlama yapması beklenir.

Kod 3.6.2 İki sayı arasındaki ilişkiyi ekrana yazan programa ait kodlar

```
1.  ilksayi = int(input("Lütfen ilk sayıyı giriniz->"))
2.  ikincisayi = int(input("Lütfen ikinci sayıyı giriniz->"))
3.  if ilksayi < ikincisayi:
4.          print("ilk sayi, ikinci sayıdan küçüktür")
5.          if ikincisayi > 15:
6.                  print("ikinci sayı, 15'ten büyüktür")
7.          else:
8.                  print("ikinci sayı, 15'ten küçüktür")
9.  else:
10.             print("İlk sayı, ikinci sayıya eşit veya büyüktür")
```

Yazılan kodlar kontrol edilerek programın doğru çalıştırılması sağlanır. Varsa eksik yerler tamamlanarak geri dönütler verilir. Ardından öğrencilere:

"Etkinliğin başındaki kargo ücreti problemine ait programı yazmanızı istiyorum" denilerek öğrencilerden Kod 3.6.3'ü yazmaları beklenir. Öğrencilerin yazdıkları program kontrol edilir. Varsa programlarındaki eksik yerler tamamlanır ve geri dönütler verilir.

Kod 3.6.3. Kargo ücreti, sipariş ücreti ve toplam ücreti yazan programa ait kodlar

```
1.  siparis = int(input("Lütfen sipariş tutarını giriniz->"))
2.  if siparis > 0 and siparis < 100:
3.          kargo = 25
4.  elif siparis > 100 and siparis < 250:
5.          kargo = 15
6.  else:
7.          kargo = 0
8.  print("Sipariş Tutarı :",siparis,"TL")
9.  print("Kargo Tutarı :",kargo,"TL")
10. print("Toplam Tutarı :",siparis+kargo,"TL")
```

Etkinliğin tamamlanmasının ardından öğretmen tarafından dersin hedeflerini vurgulayan, etkinliği özetleyen ve sonraki etkinlik hakkında kısa bilgi veren ders sonu konuşması yapılır:

"Çocuklar bugün sizinle koşul yapıları kavramını öğrendik ve uygulamalar yaptık. Bir sonraki derste döngü kavramını öğreneceğiz." denilir ve ders sonlandırılır.

DEĞERLENDİRME

Öğrencilerin kavramları içselleştirmeleri için günlük yaşamda kullandıkları koşul yapılarından örnekler vermeleri istenir. Daha sonra öğrencilerden koşul yapısı kavramını açıklamaları beklenir. Kavram yanılgıları varsa düzeltilir.

Öğretmen, öğrencilerden gelen cevaplara göre ek etkinlik yapılabilir.

Yapılan kodlamalar ve soruya verilen cevaplar Tablo B.3.6.1'de verilen kontrol listesi kullanılarak değerlendirilir.

Tablo 3.6.1 Kontrol listesi

Kontrol Listesi	Evet	Hayır
Koşul yapısı kavramını açıklar.		
if-else yapısını verilen problemin çözümünde kullanır.		
if-elif-else yapısını verilen problemin çözümünde kullanır.		
Mantıksal ifadelerden ihtiyacına uygun olanı seçerek, verilen problemin çözümünde kullanır.		

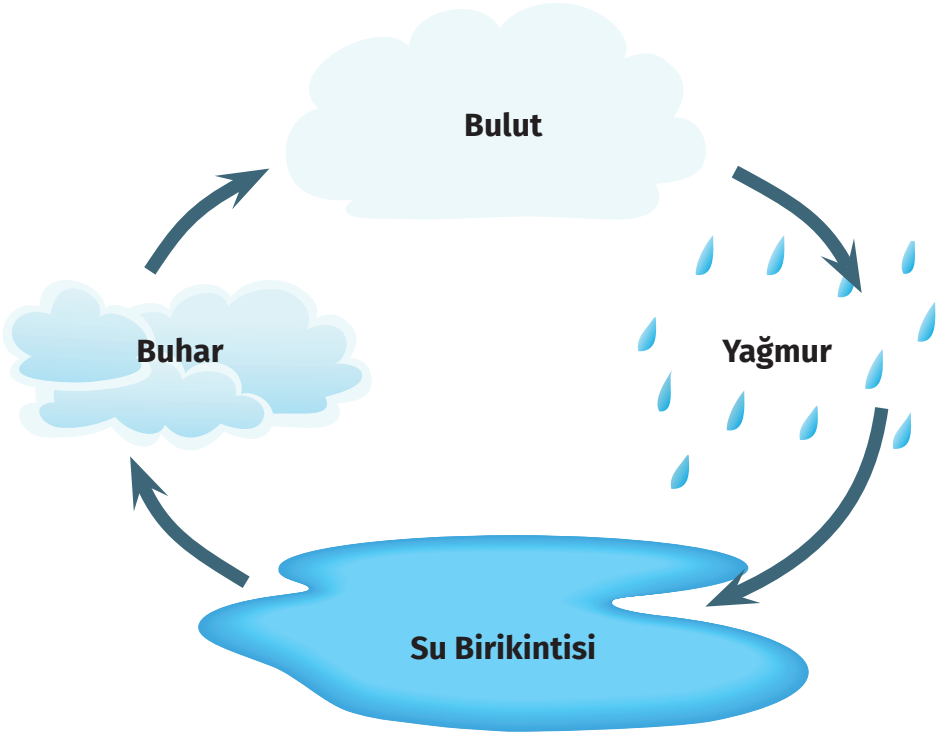
ETKİNLİK | 7

ETKİNLİK NO	**3.7**
ETKİNLİK ADI	TEKRAR EDEN KOD PARÇALARI-1
SINIF/KADEME	Lise
SÜRE	40 + 40 =80 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Döngüler
KAZANIMLAR	**3.6.1.** Döngü kavramını ve türlerini açıklar. **3.6.2.** Döngüye ihtiyaç duyulan durumları fark eder. **3.6.3.** Bir problemin çözümünde amacına uygun döngüyü kullanır. **3.6.4.** İç içe döngü yapısını kullanır.
TEMEL BECERİLER	Problem çözme, Analitik düşünme, Keşfetme
YÖNTEM VE TEKNİKLER	Doğrudan anlatım, Soru-cevap, Gösterip-yaptırma
ARAÇ-GEREÇLER	Bilgisayar, akıllı tahta veya projeksiyon, kodlama aracı
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında yüklü ya da çevrimiçi kullanabileceği bir uygulama geliştirme ortamı olması gerekir. ✓ Her öğrencinin bir bilgisayar karşına oturması sağlanır. ✓ Her öğrencinin kendi algoritmasını ve kodlamasını yapması gerekmektedir. ✓ **Döngü:** İşlemlerin belirli bir sayıda ya da sonsuz kere tekrar ettirilmesi durumudur.
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma. ✓ Temel düzeyde programlama bilgisi (değişken atamaları, girdi-çıktı işlemleri, operatör kavramı, aritmetiksel, mantıksal ve karşılaştırma operatörleri, karar kontrol yapıları)

SÜREÇ

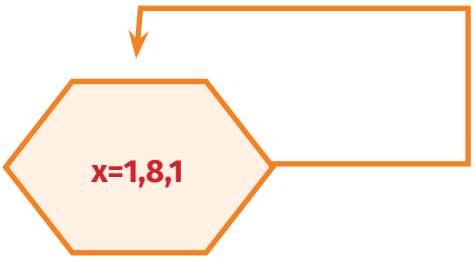
Öğrencilere döngüden ne anladıkları sorulur ve gerçek hayatta gerçekleşen döngülere örnek vermeleri istenir. Verilen cevaplar tahtaya yazılır. Tahtaya yazılan örneklerin bir döngü olup olmadığı öğrencilerle tartışılır.

Görsel 3.7.1 öğrencilere gösterilir ve öğrencilere görselden ne anladıkları sorulur. Öğrencilerden cevaplar alındıktan sonra verilen görselde bir su döngüsü olduğu belirtilir ve gerçek hayatta pek çok yerde döngünün olduğundan bahsedilir. Ek örnek olarak bir fabrikanın üretim bandından günlük olarak belirli bir sayıda ürün üretmesi de verilir. Ürün elde etme işleminde belirlenen bir sayıda ürün elde edilinceye kadar üretimin tekrar ettiği örnek verilir.



Görsel 3.7.1 Su döngüsü

Görsel 3.7.2 gösterilerek yazılımda da döngüler olduğundan bahsedilir. Döngüler tekrar eden işlemlerde gereğinden fazla kod yazmamak için kullanılan kod parçalarıdır. Verilen görselde döngünün üç adet parametresi vardır. Bunlar başlangıç, bitiş, artırma yada azaltma operatörleridir. Görsel 3.7.2'de *X* değişkeni döngünün kaç kez döneceğini belirleyen bir sayaçtır ve bu sayaç 1 ile başlatılmaktadır. *X* değeri 8'i geçtiğinde döngü sonlanacaktır. En sondaki 1 ise sayacın artım miktarıdır. *X* değişkeni döngü her bir sefer döndüğünde bir artırılacaktır. Buradaki 1 değeri yerine negatif bir sayı yazılırsa döngü geriye doğru sayacaktır. Ya da 1' den büyük bir sayı yazılırsa döngü belirlenen miktar kadar artar veya azalır. Derslerimizde *for* ve *while* olmak üzere iki tip döngü çeşidi kullanılacaktır. Görsel 3.7.2' de verilen akış şeması for döngüsüne aittir.



Görsel 3.7.2 for döngüsü akış gösterimi

Kod 3.7.1'de yazılımsal olarak for döngüsünün nasıl kullanılacağı gösterilmektedir. 1, 2 ve 3 numaralı satırlarda *x* değişkeni sayaç olarak kullanılmaktadır. 1 numaralı satırda döngünün sonlanma kriteri gösterilmektedir. Buna göre *x* değişkeni 1' den başlar 20' ye kadar gelir 21 olduğunda döngü sonlanır ve *x* değişkeni her seferinde 1 artar. Burada 1 ile *x*' in başlangıç değeri girilmektedir. 2. Satırdaki 1 değeri yerine başka bir değerde kullanılabilir. 3 numaralı satırdaki gösterimde ise *x* 1'den başlar 21'e kadar gelir. 21 olduğunda döngüye girmez. En sondaki 1 değeri *x* sayacının artma miktarıdır. Bu kısımda döngünün artma ve azalma miktarı belirlenir.

Kod 3.7.1 Yazılımsal olarak for döngüsünün kullanımı

```
1. for x in range(21)
2. for x in range(1,21)
3. for x in range(1,21,1)
```

Öğrencilerden print("bilsem") komutunu 10 defa alt alta yazarak yazdırmaları istenir. İşlem gerçekleştirildikten sonra bu işlemi 100 defa yazdırmak isteseydik 100 satır kod yazmamız gerektiği belirtilir. Kod 3.7.2 öğrencilere gösterilir ve aynı işlemleri tekrar tekrar kodlamanın yerine döngü ile işlemlerin daha az satır kod ile tamamlanabileceği belirtilir.

Kod 3.7.2 Örnek döngü kullanımı

```
1. for x in range(1,11):
2.         print("merhaba")
```

Ekran çıktısı 3.7.1. Örnek döngünün ekran çıktısı

```
1. merhaba
2. merhaba
3. merhaba
4. merhaba
5. merhaba
6. merhaba
7. merhaba
8. merhaba
9. merhaba
10. merhaba
```

Yeni örnek verilerek etkinlik devam ettirilir. Kod 3.7.3'te for döngüsünün farklı kullanım şekilleri ve Ekran çıktısı 3.7.2'de kodların çıktıları verilmektedir. *print* komutunun içerisine eklenen *end=" "* parametresi yan yana yazdırmayı sağlamaktadır. Sol tarafta *x* değişkeni 1'den başlar ve sonlanma kriterine kadar 1' er artar. Ortada *x* değişkeni 1' den başlar ve sonlanma kriterine kadar 2' şer artar. Sağda ise *x* değişkeni 7' den başlar ve sonlanma kriterine kadar 1'er azalır.

Kod 3.7.3 İleri 1' er sayan döngü, 2'şer sayan döngü ve geriye sayan döngü

```
1. for x in range(1,8,1):
       print(x, end=" ")

2. for x in range(1,8,2):
       print(x, end=" ")

3. for x in range(7,0,-1):
       print(x, end=" ")
```

Ekran çıktısı 3.7.2. 1'er, 2'şer ve geriye doğru sayan döngülerin ekran çıktıları

```
Çıktı-1    1 2 3 4 5 6 7
Çıktı-2    1 3 5 7
Çıktı-3    7 6 5 4 3 2 1
```

Kod 3.7.4 öğrencilere gösterilir ve burada 1' den 7' ye kadar olan sayıların toplamının alındığı belirtilir. Görsele göre döngü başlamadan önce *toplam* değişkeni oluşturulur ve başlangıç değeri olarak 0 atanır. *i* sayaç değişkenin değeri her değiştiğinde toplam değerinin üzerine aktarılır. *i* değişimine göre toplam değişkeninin değişimi Tablo 3.7.1 de gösterilmektedir.

Kod 3.7.4. 1' den 7' ye kadar olan sayıların toplamı.

```
1.  toplam=0
2.  for i in range(8):
3.          toplam+=i
4.  print(toplam)
```

Tablo 3.7.1 *i* değişkeninin değerine göre toplam değişkeninin değişimi.

i	0	1	2	3	4	5	6	7
toplam	**0**	**1**	**3**	**6**	**10**	**15**	**21**	**28**

Kod 3.7.5 öğrencilere gösterilir. Burada bizim belirlediğimiz belirli bir aralıkta 3'e ve 5'e tam bölünen sayıların toplamı hesaplanmaktadır. 3'e tam bölünen sayılar *sonuc1* değişkeninde 5'e tam bölünen sayılar ise *sonuc2* değişkeninde tutulmaktadır ve başlangıç değerleri 0 olarak atanır. *bas* ve *bit* değişkenleri ise döngünün hangi aralıkta çalışacağını belirleyen ve kullanıcı tarafından atanacak olan değişkenlerdir. Ekran çıktısı 3.7.3'e bakıldığında *bas* değişkenine 15, *bit* değişkenine 45 atandığı gözlemlenebilir. % operatörü ile *i* değişkeninin 3 ve 5 sayılarına tam bölünüp bölünmediği kontrol edilmektedir. 3'e tam bölünüyorsa *sonuc1* değişkeni *i* kadar artırılır; 5'e tam bölünüyorsa *sonuc2* değişkeni *i* kadar artırılır. Döngü tamamlandıktan sonra *sonuc1* ve *sonuc2* değişkenleri elde edilir ve ekrana yazdırılır. Öğrenciler bu işlemi tamamladıktan sonra onlardan hem 3'e hem de 5'e tam bölünen sayıların toplamını bulmaları istenir.

Kod 3.7.5 Başlangıç ve bitiş değeri belirlenen sayı aralığında 3' e veya 5'e tam bölünen sayıların toplamı

```
1.  sonuc1=0
2.  sonuc2=0
3.  bas=int(input("başlangıç değerini giriniz:"))
4.  bit=int(input("bitiş değerini giriniz: "))
5.  for i in range(bas,bit+1):
6.          if(i%3==0):
7.                  sonuc1=sonuc1+I
8.          if(i%5==0):
```

```
9.                  sonuc2=sonuc2+I
10.  print("üçe tam bölünen sayıların toplamı: ",sonuc1)
11.  print("beşe tam bölünen sayıların toplamı: ",sonuc2)
```

Ekran çıktısı 3.7.3 15 ve 45 aralığında 3' e veya 5'e tam bölünen sayıların toplamını gösteren ekran çıktısı

```
1. başlangıç değerini giriniz:15
2. bitiş değerini giriniz: 45
3. üçe tam bölünen sayıların toplmı: 330
4. beşe tam bölünen sayıların toplamı: 210
```

Etkinlik yeni bir problem ile devam ettirilir. Eleman sayısı bilinen bir kümenin istenilen eleman sayısı kadar toplam alt küme sayısının bulunmasını sağlayan kodlar Kod 3.7.6'da gösterilmektedir. Ekran çıktısı 3.7.4'te verilen örnekte 8 elemanlı bir kümenin 2 elemanlı alt küme sayısı gösterilmektedir. Bunun için lik bir kombinasyon hesaplanmalıdır ve şeklinde hesaplanır. Buna göre pay paydada belirlenen alt küme sayısı kadar geriye doğru götürülerek elde edilen değerler çarpılır. Paydanın ise faktoriyel hesabı yapılır (Örnek: 3!=3.2.1=6). Bunun için kodlarda başlangıç değerleri 1 olan *pay* ve *payda* değişkenleri oluşturulur. Bu değişkenler döngü içerisinde devamlı çarpma işlemine tabi tutulacağı için 0 değeri başlangıç olarak atanmaz. *sonuc* değişkeni ise elde edilecek olan *pay/payda* değerini tutacaktır ve başlangıç değeri 0 atanır. *eleman* değişkeni kümenin toplam eleman sayısıdır. *n* değişkeni ise belirlenecek olan alt küme sayısıdır ve her iki değişkende kullanıcı tarafından atanır. Döngü 1' den başlar ve alt eleman sayısı kadar döndürülür. *pay* değişkeni her seferinde eleman değişkeni ile çarpılır ve her döngü değişiminde eleman değeri bir azaltılır. *payda* değişkeni ise artan *i* değeri ile çarpılarak elde edilir. Döngü tamamlandıktan sonra *sonuc=pay/payda* şeklinde elde edilir ve ekrana yazdırılır.

Kod 3.7.6 Eleman sayısı verilen bir kümenin istenilen alt küme sayısının hesaplanması

```
1.  pay=1
2.  payda=1
3.  sonuc=0
4.  eleman=int(input("kümenin eleman sayısını giriniz: "))
5.        n=int(input("alt elemanlı küme sayısını giriniz: "))
6.        for i in range(1,n+1):
7.        pay=pay*eleman
8.        payda=payda*I
9.        eleman=eleman-1
10.  sonuc=pay/payda
11.  print("toplam alt küme sayısı: ",sonuc)
```

Ekran çıktısı 3.7.4 8 elemanlı bir kümenin 2 elemanlı alt küme sayısını gösteren ekran çıktısı

```
1.  kümenin eleman sayısını giriniz: 8
2.  alt elemanlı küme sayısını giriniz: 2
3.  toplam alt küme sayısı: 28.0
```

Etkinlik Kod 3.7.7' deki kodlar ile devam ettirilir. Buradaki problemde taban ve üs değeri girilen üslü bir sayının sonucu döngü ile hesaplanmak istenmektedir. *taban* ve *us* değerleri kullanıcı tarafından girilir ve *sonuc* başlangıç değeri 1 olarak atanır. Döngü *us* kadar tekrar ettirilir ve *sonuc* her döngü tekrarında *taban* ile çarpılır. Elde edilen sonuç döngü bitiminde ekrana Ekran çıktısı 3.7.5'deki gibi yazdırılır.

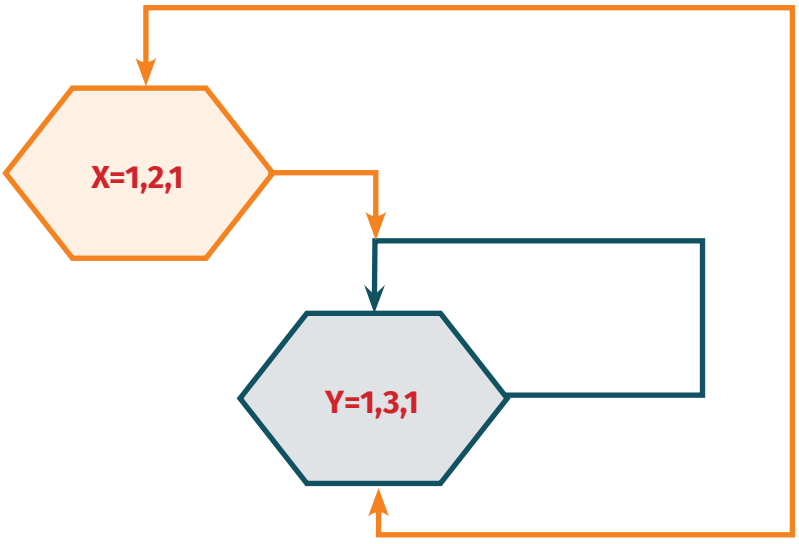
Kod 3.7.7 Döngü ile üs hesaplama

```
1.  taban=int(input("üssü alınacak sayıyı giriniz: "))
2.  us=int(input("üssü giriniz:"))
3.  sonuc=1
4.  for i in range(1,us+1):
5.          sonuc=sonuc*taban
6.  print("sonuç:",sonuc)
```

Ekran çıktısı 3.7.5 5 üssü 2 işlemini hesaplayan programın ekran çıktısı

```
1.  üssü alınacak sayıyı giriniz: 5
2.  üssü giriniz:2
3.  sonuç: 25
```

Etkinliğin devamında döngülerin iç içe kullanılacağından bahsedilir. Bazen tek bir döngü kullanmak bir problemin çözümünde yeterli olmayabilir. Bir döngü çalışırken içerisinde bu döngüye bağlı olarak çalışması gereken başka döngülerde olabilir. Bu durumun akışı örnek olarak Görsel 3.7.3'de gösterilmektedir.



Görsel 3.7.3 İç içe for döngüsü akış şeması

X değişkeni dışardaki döngünün sayacıdır ve 2'ye kadar 1'er artar, değişkeni ise içerideki döngünün sayacıdır ve 3'e kadar 1'er artar. Her bir *X* değerine bağlı olarak *Y* değişimi tablo 3.7.2 de gösterilmektedir.

Tablo 3.7.2 İç içe döngü *X* ve *Y* sayaçlarının değişimi

X	1	1	1	2	2	2
Y	1	2	3	1	2	3

Ekran çıktısı 3.7.6 öğrencilere gösterilir. Böyle bir ekran çıktısı elde edilebilmesi için tek bir döngünün yeterli olup olmayacağı öğrencilere sorulur. Cevaplar alındıktan sonra böyle bir çıktı elde edilebilmesi için iç içe bir döngü kullanılması gerektiği belirtilir.

Ekran çıktısı 3.7.6 Örnek ekran çıktısı

```
1.  !   !   !   !   !   !
2.  !   !   !   !   !
3.  !   !   !   !
4.  !   !   !
5.  !   !
6.  !
```

Kod 3.7.8'deki kodlar öğrencilerle paylaşılır. Problemin çözümü için iç içe iki adet for döngüsü kullanılmaktadır. Dışarıdaki döngünün sayacı *x* değişkeni, içerideki döngünün sayacı *y* değişkenidir. Dışarıdaki döngü sonlanmadan içerideki döngü çalışmaya devam edecektir. Dışarıdaki döngü satır sayısı kadar dönecektir. İçerideki döngü ise bir azalarak devam edecek ve bitiş değeri x değişkeni kadar olacaktır. Her bir satırda yazdırılan ! karakterlerinin arasına \t ile bir sekmelik boşluk konulmaktadır ve iç döngü her bittiğinde diğer satıra geçilebilmesi için \n ile bir alt satıra geçilir.

Kod 3.7.8 İç içe for döngüsü kodları

```
1.  for x in range(1,7):
2.      for y in range(6,x-1,-1):
3.          print("!",end="\t")
4.      print("\n")
```

Kodlar yazılıp çalıştırıldıktan sonra öğretmen: *"Bugün sizlerle tekrarlayan kod parçalarına giriş yaptık. for* döngüsünü ve iç içe *for* döngüsünün nasıl çalıştığını uyguladık. Bir sonraki dersimizde de bir başka döngü komutu olan *while* döngüsünü ve *while* döngüsünün iç içe nasıl kullanılacağını öğreneceğiz." diyerek etkinliği sonlandırır.

DEĞERLENDİRME

1. Tekrar eden işlemler için kullanılan kod parçasına ..... adı verilir.

2. Döngü komutlarının adını yazınız: ..... döngüsü ..... döngüsü

3. Tablo 3.7.3' te verilen durumlarda döngüye ihtiyaç olup olmadığını karşılarına belirtiniz.

Tablo 3.7.3 Durumlar tablosu

Durum	VAR	YOK
Klavyeden girilen bir sayının pozitif olduğunu bulan program.		
1' den 10000' e kadar olan sayıların toplamını bulan program.		
Faktoriyel hesabı yapan program.		
Bir kümenin n elemanlı alt küme sayısını bulan program.		
Belirli bir sayı aralığındaki asal sayıları bulan program.		

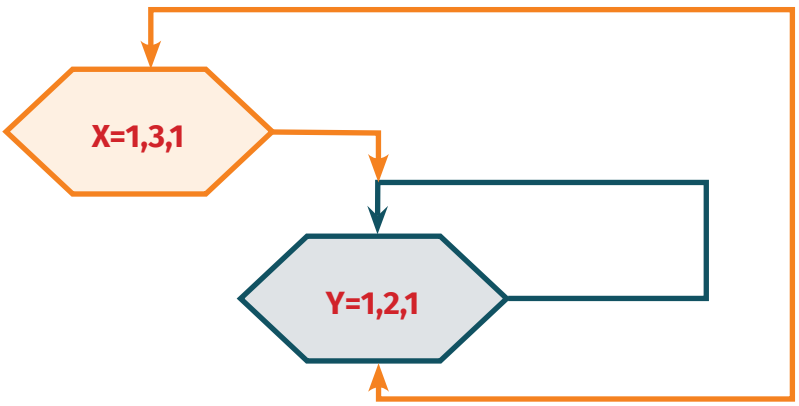
4. Kod 3.7.9'da faktöriyel hesabı yapan program verilmektedir. *n=5* için döngü üzerinde her bir *i* değişiminde elde edilecek olan *f* değerlerini Tablo 3.7.4' e işleyiniz.

Kod 3.7.9 Faktöriyel hesabı yapan kodlar

```
1.  n=int(input("faktöriyeli alınacak sayıyıyı giriniz: "))
2.  f=1
3.  for i in range(1,n+1):
4.          f=f*i
5.          print(f)
```

Tablo 3.7.4 *f* değişkeni değişim tablosu

i	1	2	3	4	5
f					



5. Görsel 3.7.4'de örnek olarak verilen iç içe döngünün *X* ve *Y* değişimlerini Tablo 3.7.5'e yazınız.

Görsel 3.7.4 Örnek bir iç içe for döngüsü akış diyagramı

Tablo 3.7.5 *X* ve *Y* değişkenleri değişim tablosu

X						
Y						

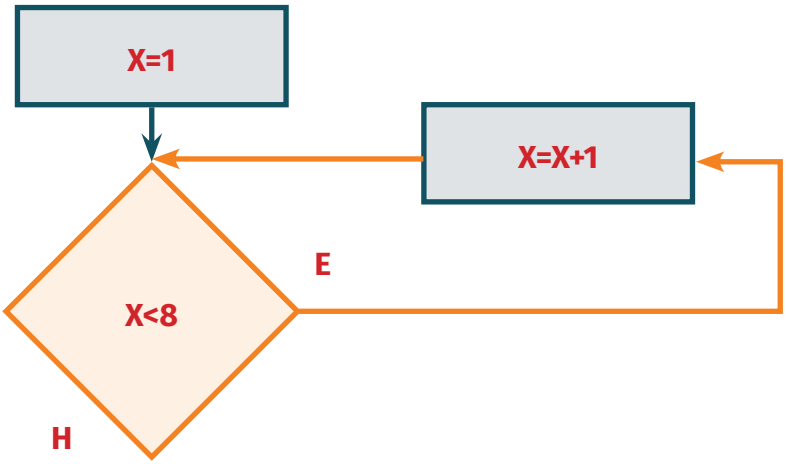
ETKİNLİK | 8

ETKİNLİK NO	**3.8**
ETKİNLİK ADI	TEKRAR EDEN KOD PARÇALARI-2
SINIF/KADEME	Lise
SÜRE	40 + 40 =80 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Döngüler
KAZANIMLAR	**3.6.3.** Bir problemin çözümünde amacına uygun döngüyü kullanır. **3.6.4.** İç içe döngü yapısını kullanır.
TEMEL BECERİLER	Problem çözme, Analitik düşünme, Keşfetme
YÖNTEM VE TEKNİKLER	Doğrudan anlatım, Soru-cevap, Gösterip-yaptırma
ARAÇ-GEREÇLER	Bilgisayar, akıllı tahta veya projeksiyon, kodlama aracı
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında yüklü ya da çevrimiçi kullanabileceği bir uygulama geliştirme ortamı olması gerekir. ✓ Her öğrencinin bir bilgisayar karşına oturması sağlanır. ✓ Her öğrencinin kendi algoritmasını ve kodlamasını yapması gerekmektedir. ✓ **Döngü:** İşlemlerin belirli bir sayıda ya da sonsuz kere tekrar ettirilmesi durumudur.
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma. ✓ Temel düzeyde programlama bilgisi (değişken atamaları, girdi-çıktı işlemleri, operatör kavramı, aritmetiksel, mantıksal ve karşılaştırma operatörleri, karar kontrol yapıları)

SÜREÇ

Öğrencilere önceki derste öğrendikleri *for* döngüsünün çalışma prensibi hatırlatılır. Bu derste öğrenecekleri *while* döngüsünün çalışma prensibi olarak *for* döngüsüne kıyasla farklılıklarının neler olabileceği sorulur.

Görsel 3.8.1'deki *while* döngüsünün akış diyagramı öğrencilere gösterilir. *while* döngüsün belirli bir koşul sağlandığı sürece tekrar eden döngü olduğu belirtilir. Görsele göre *x* değişkeni 8'den küçük olduğu sürece döngü tekrar edecektir. *x*'in başlangıç değeri 1'dir ve döngünün dışında tanımlanmaktadır. *x*'in artım değeri ise döngünün içinde gerçekleşmektedir.



Görsel 3.8.1 *while* döngüsü akış diyagramı

Kod 3.8.1'de 1'den 7'ye kadar olan sayıların karelerinin toplamını alan işlemin *while* döngüsü ile gerçekleştirildiği kodlar gösterilmektedir. *while* ifadesinin yanına döngünün çalışma koşulu yazılır. Buraya *for* döngüsünde olduğu gibi herhangi bir sayaç tanımlanmaz. İhtiyaç olması halinde sayaç döngüden önce tanımlanır ve sayacın artırma veya azaltma işlemleri döngünün içerisinde gerçekleştirilir. *t* değişkeni toplam değeri tutacaktır ve başlangıç değeri olarak 0 atanır. *s* sayaç değişkenidir ve *while* döngüsünde *for* döngüsünün aksine döngünün dışında tanımlanır ve başlangıç değeri olarak 1 atanır. *while* ifadesinin yanına döngünün çalışma koşulu yazılır ve *s* değişkeninin değeri 8'den küçük olduğu sürece döngü çalışır. Döngü içerisinde sayaç değişkeninin karesi hesaplanır ve *t* değişkeninin üzerine eklenir. Yine for döngüsünün aksine *s* değişkeninin değeri döngü içerisinde 1 artırılma kodu yazılır. *s=8* olunca döngü sonlanır ve elde edilen toplam değer Ekran çıktısı 3.8.1' deki gibi ekrana yazdırılır.

Kod 3.8.1 1'den 7'ye kadar olan sayıların karelerinin toplamını alan program.

```
1. i=1
2. toplam=0
3. while(i<=8):
4.         toplam=toplam+i**2
5.         i=i+1
6. print(toplam)
```

Ekran çıktısı 3.8.1 1'den 7'ye kadar olan sayıların karelerinin toplamı

```
1. 204
```

while döngüsü sonsuz döngü olarak da kullanılır. Mesela tekrar tekrar çalıştırıp sonucunu gözlemlemek istediğimiz durumlar while(1), while(1==1) gibi sonsuz döngülerin içerisine aktarılarak da kullanılabilir. Böylelikle programı tekrar tekrar çalıştırmak zorunda kalmayız ve sonsuz döngü içerisinde sonuçlarımızı gözlemleyebiliriz. Kod 3.8.2'de bir probleme ait kodlar gösterilmektedir. Buradaki problemde bir işyerinde 10 farklı departman vardır ve her bir departmanda üç kişi çalışmaktadır. Her bir departman çalışanlarından yaşça en büyük olan kişiler belirlenerek genel müdür yardımcısı için aday kişiler seçilecektir. Yazılım birimine verilen görevde program tekrar tekrar başlatılmadan her bir departmandaki kişilerin yaşları girilerek en büyük kişiler seçilecektir. Yazılım birimi de Kod 3.8.2'deki gibi bir program geliştirir ve genel müdürlüğe sunar. Ekran çıktısı 3.8.2' de örnek olarak verilmiştir.

Kod 3.8.2 Üç kişiden yaşı en büyük olanı tekrar tekrar hesaplayabilen program.

```
1.  enbuyuk=-1
2.  while(1):
3.      y1=int(input("birinci kişinin yaşını giriniz: "))
4.      y2=int(input("ikinci kişinin yaşını giriniz: "))
5.      y3=int(input("üçüncü kişinin yaşını giriniz: "))
6.      if(y1>y2 and y1>y3):
7.          enbuyuk=y1
8.      if(y2>y1 and y2>y3):
9.          enbuyuk=y2
10. if(y3>y2 and y2>y1):
11.     enbuyuk=y3
12. print(enbuyuk)
```

Ekran çıktısı 3.8.2 Yaşı en büyük personeli tekrarlı olarak bulan örnek çıktılar.

```
1.  birinci kişinin yaşını giriniz: 50
2.  ikinci kişinin yaşını giriniz: 40
3.  üçüncü kişinin yaşını giriniz: 35
4.  50
5.  birinci kişinin yaşını giriniz: 25
6.  ikinci kişinin yaşını giriniz: 35
7.  üçüncü kişinin yaşını giriniz: 30
8.  35
```

Kod 3.8.2'ye bakıldığında *enbuyuk* en büyük kişinin yaş bilgisini tutacak değişkendir ve döngüye girmeden önce -1 olarak atanır. Tekrar tekrar karşılaştırma yapılabilmesi için işlemler *while(1)* sonsuz döngüsünün içine alınır. *y1*, *y2* ve *y3* değişkenlerinin içerisine her seferinde bir departmandaki üç çalışanın yaş değeri kullanıcı tarafından atanır. *if* blokları içerisinde en büyük elemanın yaş değeri belirlenir ve *enbuyuk* değişkenine atanır. Döngünün sonunda en büyük yaşa sahip olan kişinin yaş değeri ekrana yazdırılır ve tekrardan karşılaştırma yapılabilmesi için döngü başa döner.

while döngüsünün sonlanma durumu bir sayacın son değerine gelmesi durumuna bağlı değildir. Kod 3.8.3'deki programa ait kodlar, klavyeden atanan bir sayının basamakları çarpımını hesaplamaktadır. Programa göre klavyeden kullanıcı tarafından bir sayı atanır. Döngüye girmeden *sonuc* adında bir değişken tanımlanır ve basamakların çarpımı bu değişkende tutulacağı için başlangıç değeri 1 olarak atanır. *sayi* değeri 0' dan büyük olduğu sürece döngü çalışmaktadır. *sonuc* değişkeni her seferinde *sayi* değişkeninin birler basamağı belirlenerek ve kendi değeri ile çarpılarak belirlenir. *sayi* değişkeninin birler basamağındaki sayı değeri, değişkenin 10' a göre modu alınarak hesaplanır. Her seferinde aynı değer hesaplanmaması için *sayi* 10' a tam bölünür ve *sayi*'nin bir sonraki basamak değeri hesaplanır. Bu tam bölme işlemlerinin sonunda *sayi* değeri 0 olunca döngü sonlanır ve elde edilen *sonuc* değeri Ekran çıktısı 3.8.3'de verilen örnekteki gibi ekrana yazdırılır.

Kod 3.8.3 Klavyeden girilen bir sayının basamakları çarpımını bulan program

```
1.  sayi=int(input("klavyeden bir sayı giriniz: "))
2.  sonuc=1
3.  while(sayi>0):
4.          sonuc=sonuc*(sayi%10)
5.          sayi=sayi//10
6.  print("verilen sayının basamakları çarpımı",sonuc)
```

Ekran çıktısı 3.8.3 Örnek olarak verilen 235 sayısının basamakları çarpımı

```
1.  klavyeden bir sayı giriniz: 235
2.  verilen sayının basamakları çarpımı 30
```

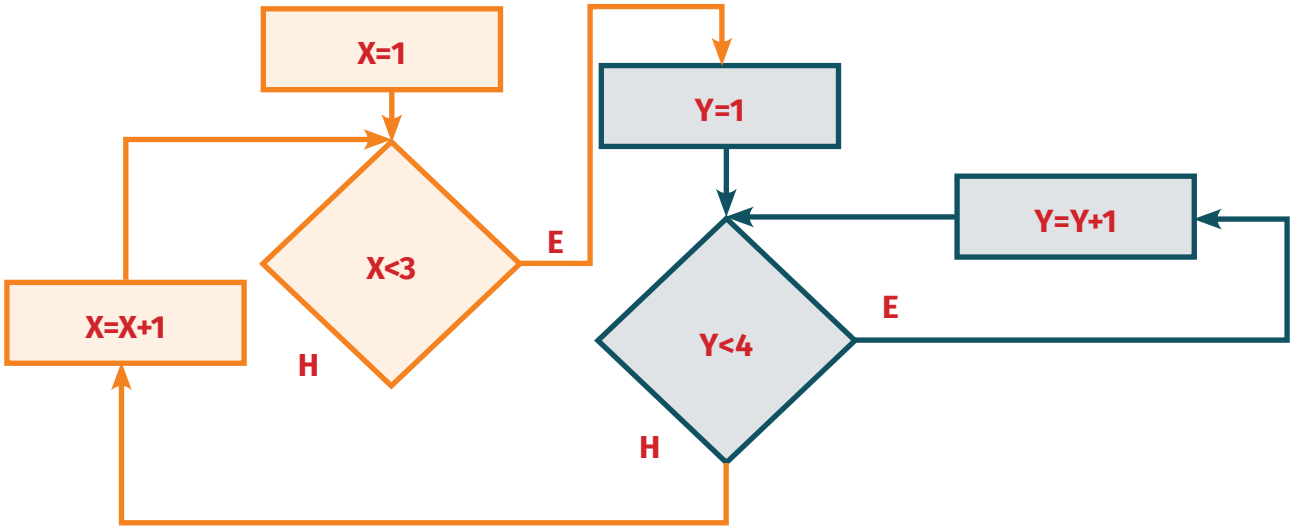
Etkinlik Kod 3.8.4'de verilen program ile devam ettirilir. Bu örnekte belirli bir aralıkta rastgele sayı üretilir. Döngü içerisinde üretilen sayının tahmini gerçekleştirilir. Üretilen sayı rastgele belirli bir aralıkta üretileceği için öncelikle *import random* komutu eklenir. Rastgele sayının üretileceği aralık *bas* ve *bit* değişkenleri içerisinde tutulur ve *tut* değişkenine üretilen rastgele sayı atanır (satır 4). Tahmin edilen değer *tahmin* değişkeninin içinde tutulacaktır ve döngüye girmeden önce değeri -1 olarak atanır. *tahmin* edilen değer rastgele tutulan sayıya eşit oluncaya kadar döngü tekrar edecektir. Döngünün başlangıcında başlangıç ve bitiş değeri arasında bir sayı tahmin edilmesi istenir ve bu sayı *tahmin* değişkenine atanır. Eğer tahmin edilen sayı *tut* değişkeninin değerinden büyükse *bit* değerine *tahmin* edilen değer atanır. Eğer *tahmin* edilen değer tutulan değerden küçük ise *tahmin* edilen değer *bas* değişkenine atanır. Böylelikle hatalı tahminler yapıldığında kullanıcı daha dar bir aralıkta tahminini gerçekleştirecektir. Tahmin edilen sayı tutulan sayıya eşitse ekrana tebrikler yazısı gelir ve oyun sonlanır.

Kod 3.8.4 Sayı tutmaca oyunu

```
1.  import random
2.  bas=1
3.  bit=100
4.  tut=random.randint(bas,bit)
5.  tahmin=-1
6.  while(tahmin!=tut):
7.          print(bas,"ve",bit,"arasında bir sayı tutunuz: ")
8.          tahmin=int(input(""))
9.          if(tahmin>tut):
10.                 bit=tahmin
11. elif(tahmin<tut):
12.                 bas=tahmin
13. else:
14.                 print("tebrikler")
```

for döngüsünde olduğu gibi *while* döngüleri de iç içe kullanılır. Görsel 3.8.2'de *while* döngüsünün iç içe

çalışma akış şeması gösterilmektedir. Görsele göre sarı ile gösterilen döngü dış döngü mavi ile gösterilen döngü iç döngüdür. *x* değeri 3 oluncaya kadar iç döngü çalışır ve *y* değeri her seferinde 1'den başlar ve 4 olunca iç döngü sonlanır *x* ve *y* değerlerinin değişimi Tablo 3.8.1' de gösterilmektedir.



3.8.2 İç içe *while* döngüsü akış diyagramı

Tablo 3.8.1 İç içe döngü içerisinde *x* ve *y* değişkenlerinin değişimi

x	1	1	1	2	2	2
y	1	2	3	1	2	3

Etkinlik yeni bir örnek ile devam ettirilir. aΔb işlemi $a+b^2$ şeklinde hesaplanmaktadır. 1' den 10' kadar olan tüm sayıların Δ işleminin sonucunu tablo halinde yazan program Kod 3.8.5'de verilmektedir. Kodlar öğrencilere yazdırılır ve çıktıları kontrol edilir.

Kod 3.8.5 aΔb işleminin kodları

```
1.  a=1
2.  while(a<=10):
3.      b=1
4.      while(b<=10):
5.          sonuc=a+b**2
6.          print(a,"Δ",b,"=",sonuc,end=" \t")
7.          b=b+1
8.      print("\n")
9.      a=a+1
```

Kodlar yazılıp çalıştırıldıktan sonra öğretmen: "*Bugün sizlerle while* döngüsü üzerine çalıştık. *while* döngüsü ve *while* döngüsünün iç içe kullanımı üzerine alıştırmalar yaptık. Bir sonraki dersimizde daha önce öğrendiğimiz listeler konusu ile tekrarlayan kod parçalarını ilişkilendirerek, bir arada kullanacağız." diyerek etkinliği sonlandırır.

DEĞERLENDİRME:

1. Kod 3.8.6'ya göre ekranda yazacak *i* değerlerini Tablo 3.8.2'ye işleyiniz.

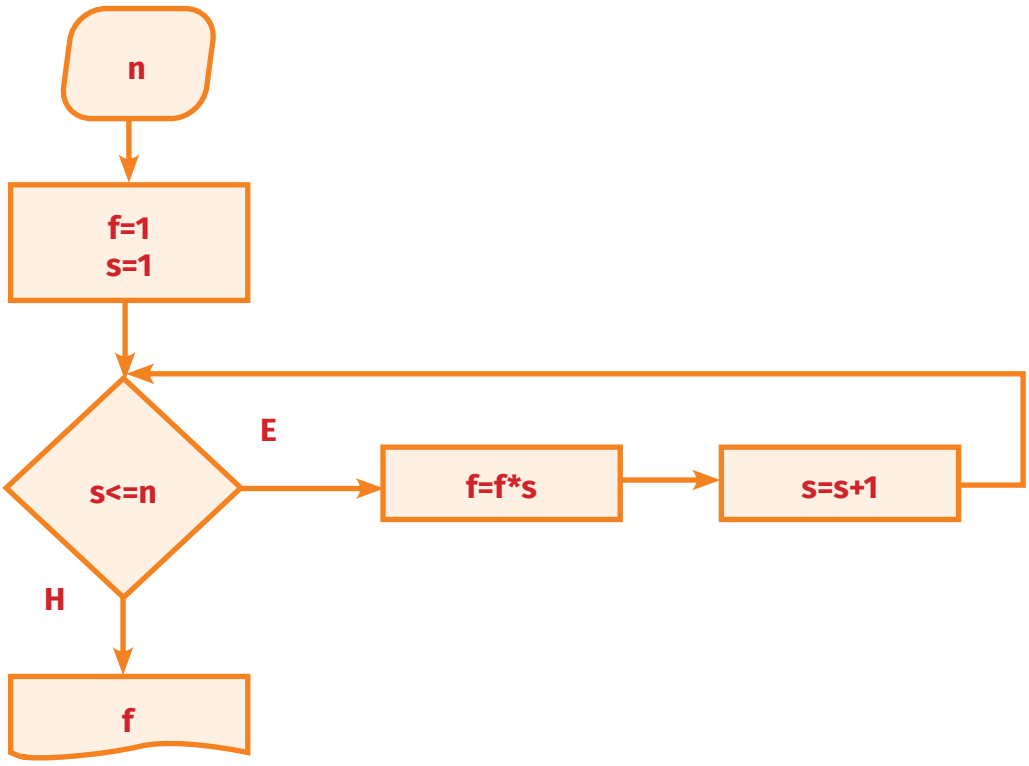
Kod 3.8.6 Örnek bir iç içe while döngüsü kodları

```
1.  i=2
2.  while(i<=20):
3.      sayac=0
4.      j=2
5.      while(j<=i):
6.          if(i%j==0):
7.              sayac=sayac+1
8.          j=j+1
9.      if(sayac==1):
10.     print(i)
11. i=i+1
```

Tablo 3.8.2 *i* değişkeni değişim tablosu

i								

2. Görsel 3.8.3'de akış diyagramı verilen programın kodlarını bilgisayar ortamında yazınız.



Görsel 3.8.3 Kodları yazılması istenilen programın akış diyagramı

ETKİNLİK | 9

ETKİNLİK NO	**3.9**
ETKİNLİK ADI	TEKRAR EDEN KOD PARÇALARI VE LİSTELER
SINIF/KADEME	Lise
SÜRE	40 + 40 =80 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Döngüler
KAZANIMLAR	**3.6.5.** Listelerle döngüleri bir arada kullanır.
TEMEL BECERİLER	Problem çözme, Analitik düşünme, Keşfetme
YÖNTEM VE TEKNİKLER	Doğrudan anlatım, Soru-cevap, Gösterip-yaptırma
ARAÇ-GEREÇLER	Bilgisayar, akıllı tahta veya projeksiyon, kodlama aracı
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında yüklü ya da çevrimiçi kullanabileceği bir uygulama geliştirme ortamı olması gerekir. ✓ Her öğrencinin bir bilgisayar karşına oturması sağlanır. ✓ Her öğrencinin kendi algoritmasını ve kodlamasını yapması gerekmektedir. ✓ **Döngü:** İşlemlerin belirli bir sayıda ya da sonsuz kere tekrar ettirilmesi durumudur. ✓ **Liste:** Aynı veya farklı değişken tiplerinin bir arada bulunduğu topluluktur.
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma. ✓ Temel düzeyde programlama bilgisi (değişken atamaları, girdi-çıktı işlemleri, operatör kavramı, aritmetiksel, mantıksal ve karşılaştırma operatörleri, karar kontrol yapıları, listeler)

SÜREÇ

Öğrencilere daha önce öğrendikleri listeler konusu hatırlatılır. Listelerin döngüler ile beraber kodlanabileceği belirtilerek öğrencilerin dikkatleri çekilir. Listelerin döngüler ile nasıl kullanılabileceği sorusu sorulur ve alınan cevaplar öğrenciler ile birlikte değerlendirilir.

x=[3,18,21,54,18,-1,27,13,9,10] şeklinde tam sayılardan oluşan bir liste oluşturulur. Oluşturulan listedeki sayılara 10 eklenerek liste güncellenmek istenmektedir. Burada istenilen tekrarlı bir durum olduğu için listeler döngüler ile kullanılabilir. Kod 3.9.1' de listenin döngü içerisinde nasıl değiştirildiği ve yine döngü kullanılarak nasıl yazdırılabileceği kodlar halinde gösterilmektedir. Oluşturulan *x* dizisinin eleman değerleri döngü üzerinde *i* isimli indeks değişkeninden faydalanılarak değiştirilir. İkinci *for* döngüsünde de elde edilen yeni değerler ekrana yazdırılmaktadır. Burada *x* dizisinin içinde tüm değerler *j* içerisine aktarılır ve herhangi bir indeks değeri kullanılmadan dizinin eleman değerlerini *j* değişkeni temsil eder.

Kod 3.9.1 Listenin döngü ile kullanımı

```
1.  x=[3,18,21,54,18,-1,27,13,9,10]
2.  for i in range(10):
3.      x[i]=x[i]+10
4.  for j in x:
5.      print(j,end="      ")
```

Ekran Çıktısı 3.9.1 Dizi elemanlarının yeni değerleri

```
1.  13  28  31  64  28  9  37  23  19  20
```

Metinsel (string) veriler de aslında karakterlerden oluşan listelerdir. "merhaba" kelimesini incelediğimizde 'm', 'e', 'r', 'h', 'e', 'b', 'a' karakterlerinden oluşmuş bir liste olduğu görülmektedir. Kod 3.9.2'de bu durumdan faydalanılarak oluşturulmuş bir örnek verilmektedir. Bir değişkenin içerisine metinsel bir veri aktarılmakta ve aktarılan veri döngü üzerinden tersten yazdırılmaktadır. Kodların çıktısı Ekran çıktısı 3.9.2'de gösterilmektedir.

Kod 3.9.2 Verilen metni döngü ile tersten yazdıran program

```
1.  metin="bilsem merhaba"
2.  for i in range(len(metin)-1,-1,-1):
3.      print(metin[i],end="")
```

Ekran çıktısı 3.9.2 Girilen metnin tersten yazılışı

```
1.  abahrem meslib
```

Etkinlik, kelime oyunu uygulaması ile devam ettirilir. Uygulamada öncelikle bir liste içerisine 10 adet kelime (istenirse daha fazla ya da bir veri tabanından kelimeler çektirilebilir.) eklenir. Rastgele bir kelime seçilir ve oyun oynanırken kullanıcıdan önce kelime ile ilgili 5 harf tahmini istenir. Harfler, kelimenin içerisinde geçiyor ise kullanıcıya gösterilir ve bulunamayan kısımlar __ ile gösterilir. Daha sonra kullanıcıya beş tahmin hakkı verilerek doğru kelimeyi bulması istenir. Uygulamanın kodları Kod 3.9.3'de verilmektedir. Kodlar incelendiğinde *kelimeler* adında 10 elemandan oluşan bir liste oluşturulduğu görülmektedir. İşlemler oyunun tekrar tekrar oynanabilmesi için sonsuz döngü içerisinde gerçekleştirilir. Listedeki elemanlardan bir tanesi rastgele seçilir ve *tut* değişkenine atanır. *tahmin* değişkeninin içerisinde tahmin edilecek kelime tutulacaktır ve başlangıç olarak herhangi bir şey atanmaz. *tutyedek* listesinde ise tutulan kelimenin gizli bir şekilde gösterimi yapılacaktır. Öncelikle kullanıcıdan 5 harf tahmin etmesi istenir ve bu harfler döngü kullanılarak harfler listesinde tutulur. Daha sonra iç içe döngü kullanılarak tahmin edilen harflerin *tut* değişkeninin içerisinde olup olmadığında bakılır. Eğer varsa harfler *tutyedek* içerisine aktarılır ve döngülerin bitiminde tahmin edilen durumun son hali kullanıcıya gösterilir. En son aşamada kullanıcıdan döngü kullanılarak beş tahmin yapması istenir. *tut* değişkeni *tahmin*' e eşit ise *break* komutu ile döngüden çıkılır ve oyun yeniden başlatılır. *tahmin* hatalı ise *continue* komutu ile döngü ve tahminler devam ettirilir.

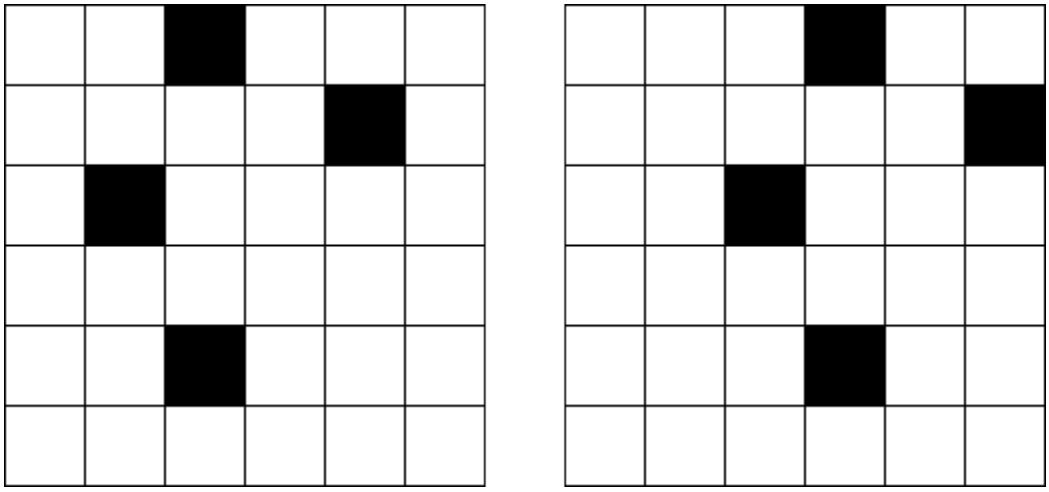
Kod 3.9.3 Kelime oyunu kodları

```
1.  import random
2. kelimeler=["patates","gaziantep","merhaba","bilgisayar", "çanakkale","istikrar","yazılım","fabrikatör","zararlı","nakliyatçı"]
3.  while(1):
4.      tut=kelimeler[random.randint(0,9)]
5.      tahmin=""
6.      tutyedek=["__" for i in range(len(tut))]
7.      print(tutyedek)
8.      harfler=["for i in range(5)]
9.      print("5 adet harf seçiniz:" )
10.     s=0
11.     while(s<5):
12.         print("tahmin ettiğniz ",s,end="")
13.         harfler[s]=input(".harfi giriniz: ")
14.         s=s+1
15. for i in range(0,len(harfler)):
16.     for j in range(0,len(tut)):
17.         if(harfler[i]==tut[j]):
18.         tutyedek[j]=harfler[i]
19. print(tutyedek)
20. print("5 tahmin hakkınız var.")
21. for i in range(1,6):
22.     print(i,end="")
23.     tahmin=input(". tahmininizi giriniz: ")
24.     if(tut==tahmin):
25.         print("tebrikler bildiniz.")
26.         break
27.     else:
28.         continue
```

Etkinlik yeni bir örnek ile devam ettirilir. *"Barış öğretmen bir yazılım aracılığı ile öğrencilerine 6x6 büyüklüğünde bir bulmaca hazırlar. Bulmacanın doldurulacak kısımları beyaz, doldurulmayacak kısımları ise siyah olarak gösterilmektedir (Kural: Kenarlara siyah kare konulmamaktadır). Barış öğretmen yazılımı kullanırken Tablo 3.9.1' deki gibi beyazlar için 1, siyahlar için 0 olan bir tablo hazırlar ve programa giriş olarak ekler. Yazılım Görsel 3.9.1'in sol tarafındaki bulmaca gibi bir çıktı verir. Fakat verilen çıktı tabloya uymamaktadır. Gerçekte bulmacanın görselin sağındaki bulmaca gibi olması gerekmektedir. Barış öğretmen yazılım konusunda deneyimli olan Murat öğretmenden yardım ister. Murat öğretmen yazılımdaki hatayı bulur. Hatalı ve doğru olan görsellere baktığında yazılımın gerçek değerleri bir sola kaydırarak çıktı aldığını fark eder. Murat öğretmen, Barış öğretmenin iki boyutlu listesini tanımlar ve daha sonra iç içe döngü kullanarak her değeri bir sağına kaydırır".* Murat öğretmenin iki boyutlu listeler ile ilgili yazdığı kodlar Kod 3.9.4'de gösterilmektedir. Verilen örnekteki çözüm öğrencilere yaptırılır ve Ekran çıktısı 3.9.4'ü elde etmeleri beklenir.

Tablo 3.9.1 Yazılımın girişine verilen tablo

1	1	1	0	1	1
1	**1**	**1**	**1**	**1**	**0**
1	**1**	**0**	**1**	**1**	**1**
1	**1**	**1**	**1**	**1**	**1**
1	**1**	**1**	**0**	**1**	**1**
1	**1**	**1**	**1**	**1**	**1**



Görsel 3.9.4. (Soldaki) Hatalı elde edilen bulmaca, (Sağdaki) doğru bulmaca

Kod 3.9.4 Doğru Bulmaca Şablonu'nun düzeltilmiş kodları

```
1.          bulmaca=[[1,1,0,1,1,1],[1,1,1,1,0,1],[1,0,1,1,1,1],[1,1,1,1, 1,1],[1,1,0,1,1,1],[1,1,1,1,1,1]]
2.  print(bulmaca)
3.  i=0
4.  while(i<6):
5.          j=0
6.          while(j<6):
7.                  if(bulmaca[i][j]==0):
8.                          bulmaca[i][j]=1
9.                          bulmaca[i][j+1]=0
10.                         j=j+1
11.                 j=j+1
12.         i=i+1
13. print(bulmaca)
```

Ekran çıktısı 3.9.3 Doğru Bulmaca Şablonu'nun düzeltilmiş kodlarının ekran çıktısı.

```
[[1, 1, 0, 1, 1, 1], [1, 1, 1, 1, 0, 1], [1, 0, 1, 1, 1, 1], [1, 1, 1, 1, 1, 1], [1, 1, 0, 1, 1, 1], [1, 1, 1, 1, 1, 1]] # hatalı bulmaca
```

```
[[1, 1, 1, 0, 1, 1], [1, 1, 1, 1, 1, 0], [1, 1, 0, 1, 1, 1], [1, 1, 1, 1, 1, 1], [1, 1, 1, 0, 1, 1], [1, 1, 1, 1, 1, 1]] # düzeltilmiş bulmaca
```

Etkinlik bir sonraki uygulama ile devam eder. Bu uygulamada bir sınıftaki öğrenci listesi oluşturulmaktadır. Oluşturulan listeye öğrenci adı, öğrenci soyadı, birinci yazılı notu, ikinci yazılı notu ve yazılı ortalama değerleri girilmektedir. Kod 3.9.5'te uygulamanın kodları verilmektedir. Öncelikle öğrenci adında boş bir liste oluşturulur. Listeye öğrenci ekleme işlemi *for* döngüsü içerisinde gerçekleştirilir. Öğrenciye ait bilgiler değişkenlere eklenir ve ogrenci.append komutu ile listeye ekleme işlemi tamamlanır. Döngü içerisinde iki öğrenci bilgisi eklenebilmektedir. Öğrenci isterse bu sayıyı artırabilir ya da sonsuz döngü ile kayıt işlemi sürekli hale de getirebilir. Diğer döngü içerisinde de listeye eklenen öğrenci bilgileri listelenir.

Kod 3.9.5 Öğrenci listesini oluşturan kodlar

```
ogrenci=list()
for i in range(0,2):
    ad=input("öğrenci adını giriniz: ")
    soyad=input("öğrenci soyadını giriniz: ")
    y1=int(input("birinci yazılı notunu giriniz: "))
    y2=int(input("ikinci yazılı notunu giriniz: "))
    ort=round(((y1+y2)/2),2)
    giris=(ad,soyad,y1,y2,ort)
    ogrenci.append(giris)
for i in range(0,2):
print(ogrenci[i])
```

Ekran çıktısı 3.9.4 Öğrenci listesini oluşturan kodların ekran çıktısı

```
öğrenci adını giriniz: Ahmet
öğrenci soyadını giriniz: Yenigün
birinci yazılı notunu giriniz: 100
ikinci yazılı notunu giriniz: 100
öğrenci adını giriniz: Ayşe
öğrenci soyadını giriniz: Yenigün
birinci yazılı notunu giriniz: 80
ikinci yazılı notunu giriniz: 90
('Ahmet', 'Yenigün', 100, 100, 100.0)
('Ayşe', 'Yenigün', 80, 90, 85.0)
```

Kodlar yazılıp çalıştırıldıktan sonra öğretmen: *"Bugün sizlerle tek boyutlu ve çok boyutlu listeleri döngü-*

ler ile bir arada kullanarak örnek alıştırmalar yaptık." diyerek etkinliği sonlandırır.

DEĞERLENDİRME:

Kod 3.9.6 Örnek liste kodları

```
1.  liste=[9,1,4,5,2,1,6]
2.  for i in range(0,len(liste)):
3.          liste[i]=liste[i]+(i**2)
4.  for x in liste:
5.          print(x,end="      ")
```

Kod 3.9.6' de verilen kodlara göre ekranda yazacak *x* değerlerini Tablo 3.9.2' ye işleyiniz.

Tablo 3.9.2 *x* değişkeni değişim tablosu

x							

2. Tablo 3.9.3' de verilen iki boyutlu listeyi oluşturup iç içe döngü kullanarak ekrana yazdırınız.

Tablo 3.9.3 iki boyutlu veriler tablosu

12	4	0	1
1	**1**	**7**	**3**
3	**6**	**1**	**0**
17	**1**	**2**	**8**

ETKİNLİK | 10

ETKİNLİK NO	**3.10**
ETKİNLİK ADI	FONKSİYONLARLA TANIŞMA
SINIF/KADEME	Lise
SÜRE	40 + 40 =80 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Fonksiyonlar
KAZANIMLAR	**3.7.1.** Fonksiyon kavramını açıklar. **3.7.2.** Kullandığı programlama dilinde var olan hazır fonksiyonları kullanır.
TEMEL BECERİLER	Problem çözme, Analitik düşünme, Karar verme
YÖNTEM VE TEKNİKLER	Doğrudan anlatım, Soru-cevap, Gösterip yaptırma
ARAÇ-GEREÇLER	Bilgisayar, Uygulama geliştirme ortamı, internet, Projeksiyon cihazı/ Etkileşimli tahta
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında yüklü ya da çevrimiçi kullanabileceği bir uygulama geliştirme ortamı olması gerekir. ✓ Ders içinde kullanılacak yerleşik fonksiyon, parametre, fonksiyon çağırma, geri değer döndürme gibi kavramlar öğretmen tarafından incelenir. ✓ Her öğrencinin bir bilgisayar karşına oturması sağlanır.
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma.

SÜREÇ

Öğretmen öğrencilere mail adresimi açmam için yapmam gereken işlemleri adım adım birlikte yazalım der.

Bilgisayarı başlat
Bilgisayar açıldı
İnternet sağlayıcıyı aç
Araç çubuğuna mail adresini yaz
Mail sayfası açıldı
Kullanıcı adınızı giriniz
İleri Tuşuna Bas
Şifrenizi Giriniz
İleri Tuşuna Bas
Mail Adresiniz açıldı

Bu işlemlerin tamamı bizim için mailini açma fonksiyonu olarak zihnimizde kayıtlıdır ve mailimizi açmak

istediğimizde bu adımları sıra ile takip ederiz.

"Mailimizi her açmak istediğimizde bu kodları tekrar tekrar yazmak yerine maili aç şeklide bir fonksiyon tanımlarsak işimiz çok daha kolaylaşır. Günlük hayatta çoğu zaman tekrar ettiğimiz davranış ve hareketleri karmaşık adımlar yerine tek kelime ile ifade ederiz. İletişim dili ile yaptığımız bu kısa ifadeler gibi yazılım dillerinde de çokça tekrar eden adımları bir isimle tanımlayıp gerektiğinde kullandığımız/çağırdığımız yapılara fonksiyon denir. Fonksiyonlar yapılan işlemleri kolaylaştırır ve zaman alıcı işlevleri kolay bir şekilde yerine getirilmesini sağlar." denilerek etkinliğe geçiş yapılır.

Etkinliğe geçmeden önce öğrencilere kavram olarak fonksiyon tanımı yapılır ve öğrencilerden bu kavram tanımını açıklamaları istenir. Bu adımdan sonra öğrencilere Python kurulumu ile hazır gelen fonksiyonlar olduğunu ve bunlara yerleşik (Built-in) fonksiyonlar dendiği anlatılarak Tablo 3.10.1 öğrencilere gösterilir. Bu tabloda bulunan fonksiyonların şu anda etkinlikte kullanacağımız yerleşik fonksiyonlar olduğu söylenir. Python programlama dilinde bulunan diğer yerleşik fonksiyonlar için programlama diline ait web sitesinden bilgi alabilecekleri öğrencilere belirtilir. Tablo 3.10.1'de bulunan fonksiyonlar öğrenciler tanıtıldıktan sonra tüm öğrencilere *fonksiyonlar1.py* adında bir dosya oluşturmaları ve Kod 3.10.1 'de verilen kodu yazıp çalıştırmaları istenir.

Tablo 3.10.1 Python programlama dilinde bulunan yerleşik fonksiyon örnekleri

Fonksiyon	Açıklaması	Fonksiyon	Açıklaması
len()	Bir listenin eleman sayısını verir.	min()	Bir grup içinden içinde en küçük olanı döndürür.
id()	Bir değişkenin kimlik bilgisini döndürür.	round()	Sayının yuvarlanmış halini döndürür.
max()	Bir grup içinden içinde en büyük olanı döndürür.	type()	Bir nesnenin tipini döndürür.
sorted()	Bir listenin sıralanmış halini döndürür.	sum()	Sayı değeri içeren bir listenin sayılarının toplamını verir.
abs()	Bir sayının mutlak değerini döndürür.		

Kod 3.10.1 Yerleşik fonksiyonların kullanımı

```
s1 = 1
s2 = 15.678
ad = "Eylül"
isimler= ["alparslan","ilke","atakan","oğuz","masal"]
agirliklar = [68,48,75,81,45]

en_agir=max(agirliklar)
en_hafif =min(agirliklar)
sirali = sorted(agirliklar)
ortalama_agirlik = sum(agirliklar)/len(agirliklar)
tip1 = type(s1)
tip2=type(isimler)
id1=id(ad)
id2=id(s2)
```

```
print(f"en ağır:{en_agir}, en hafif:{en_hafif}")
print("ağırlıklar sıralı:")
print(sirali)
print(f"ağırlıklar ortalaması:{ortalama_agirlik}")
print("s1 tipi:",tip1,"--""isimler tipi:",tip2)
print(f"ad id değeri:{id1}--s2 id değeri:{id2}")
print(f"{s2} sayısını bir basamağa yuvarlama", round(s2,1))
```

Öğrencilere Kod 3.10.1'deki kodun çıktısı Ekran çıktısı 3.10.1'deki gibi gösterilerek her fonksiyonun çalışması anlatılır.

Ekran çıktısı 3.10.1 Yerleşik fonksiyonlarının ekran çıktıları-1

```
en ağır:81, en hafif:45
ağırlıklar sıralı:
[45, 48, 68, 75, 81]
ağırlıklar ortalaması:63.4
s1 tipi: <class 'int'> -- isimler tipi: <class 'list'>
ad id değeri:1939862984944--s2 id değeri:1939862656112
15.678 sayısını bir basamağa yuvarlama 15.7
```

Öğrencilere Kod 3.10.1'de *"Kullandığınız fonksiyonları kendimizde tanımlayabiliriz."* denilerek örnek olarak *max* ve *min* komutlarının kullanıcı tanımlı fonksiyonları gösterilir. Kod 3.10.2'de yazılmış kullanıcı tanımlı fonksiyonlar üzerinde kavramlarda belirtilen parametre ve geriye değer döndürme kavramları açıklanır.

Öğrencilerden *max_min_benim.py* adından bir dosya açarak Kod 3.10.2'deki kodu yazmaları istenir.

Kod 3.10.2 Kullanıcı tanımlı min ve max fonksiyonları

```
def min_benim(listem):
    ek = listem[0]
    for deger in listem:
        if deger<ek:
            ek=deger
    return ek

def max_benim(listem):
    eb = listem[0]
    for deger in listem:
        if deger>eb:
            eb=deger
    return eb


agirliklar = [68,48,75,81,45]
en_agir = max_benim(agirliklar)
en_hafif = min_benim(agirliklar)
print(f"en ağır:{en_agir}, en hafif:{en_hafif}")
```

Kod 3.10.2'deki kodun çıktısı öğrencilere Ekran çıktısı 3.10.2.'deki gibi gösterilir.

Ekran çıktısı 3.10.2. Kullanıcı tanımlı min ve max fonksiyonları ekran çıktısı

```
en ağır:81, en hafif:45
```

Bu aşamada öğrencilere *def* anahtar kelimesi ile tanımlama yapmadan da bir listeye ait en küçük ve en büyük değerleri bulma işlemi yapılabileceği fakat bu tarz bir tanımlamanın tekrar kullanılabilir olduğu hatırlatılır.

Öğrencilere aşağıdaki problemi çözen Python kodunu yerleşik fonksiyonları da kullanarak yazmaları istenir:

"Kullanıcıdan 0 değeri girilene kadar girilen bütün sayıları bir listeye ekleyerek listenin ortancası ile ortalaması arasındaki farkın mutlak değerini bularak sonucu ekrana virgülden sonra iki basamak yuvarlayarak yazdırınız."

Problemin yapılabilecek çözümlerinden biri Kod 3.10.3' te verildiği gibidir.

Kod 3.10.3. Problem örnek çözümü

```
listem =[]
while True:
	girilen = float(input("(çıkmak için 0)Bir sayı giriniz:"))
	if girilen == 0.0:
		break
	else:
		listem.append(girilen)

es = len(listem)
toplam = sum(listem)
ortalama = toplam/es

sirali = sorted(listem)
ortanca=0.0
if es%2==1:
	ortanca = sirali[es//2]
else:
	ortanca = (sirali[es//2]+sirali[(es//2)-1])/2

mutlak_fark =round(abs(ortalama-ortanca),2)
print(f"{ortalama}-{ortanca}={mutlak_fark}")
```

Öğrencilerin test edebilmesi için örnek bir kodun çalışma ve ekran çıktısı Ekran çıktısı 3.10.3.'te verilmiştir.

Ekran Çıktısı 3.10.3. Problem çözümüne ait örnek ekran çıktısı

```
(çıkmak için 0)Bir sayı giriniz:10
(çıkmak için 0)Bir sayı giriniz:20
(çıkmak için 0)Bir sayı giriniz:30
(çıkmak için 0)Bir sayı giriniz:40
```

```
(çıkmak için 0)Bir sayı giriniz:50
(çıkmak için 0)Bir sayı giriniz:60
(çıkmak için 0)Bir sayı giriniz:70
(çıkmak için 0)Bir sayı giriniz:100
(çıkmak için 0)Bir sayı giriniz:0
ortalama:47.5
ortanca:45.0
 fark:2.5
```

Etkinliğin tamamlanmasının ardından öğretmen tarafından öğrencilere: *"Fonksiyon nedir? Fonksiyon kavramı size neyi anlatıyor"* soruları sorulur ve öğrencilerden alınan cevaplar üzerinde konuşulur. Gerekli dönütler sağlanır. Öğretmen daha sonra dersin hedeflerini vurgulayan, etkinliği özetleyen ve sonraki etkinlik hakkında kısa bilgi veren ders sonu konuşması yapılır:

"Sevgili öğrenciler bugün sizlerle temel olarak fonksiyon kavramını ve Python programlama dilinde var olan yerleşik fonksiyonların kullanımını ve neler olduklarını öğrendik. Bir sonraki dersimizde sizler ile kendi fonksiyonlarımızı yazmaya öğreneceğiz."

DEĞERLENDİRME:

Tablo 3.10.2: Kontrol listesi

Kontrol Listesi	Evet	Hayır
Fonksiyon kavramını açıklar		
Parametre kavramını açıklar.		
Yerleşik fonksiyon kavramını açıklar.		
Yerleşik fonksiyonları amacına uygun olarak kullanır.		

ETKİNLİK NO	**3.11**
ETKİNLİK ADI	FONKSİYON TANIMLIYORUM
SINIF/KADEME	Lise
SÜRE	40+40=80 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Fonksiyonlar
KAZANIMLAR	**3.7.3.** Parametre alan fonksiyonları yazar. **3.7.4.** Geriye değer döndüren ve döndürmeyen fonksiyonları yazar.
TEMEL BECERİLER	Problem çözme, Analitik düşünme, Karar verme
YÖNTEM VE TEKNİKLER	Doğrudan anlatım, Soru-cevap, Gösterip yaptırma
ARAÇ-GEREÇLER	Bilgisayar, Uygulama geliştirme ortamı, internet, Projeksiyon cihazı/ Etkileşimli tahta.
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında yüklü ya da çevrimiçi kullanabileceği bir uygulama geliştirme ortamı olması gerekir. ✓ Ders içinde kullanılacak yerleşik fonksiyon, parametre, fonksiyon çağırma, geri değer döndürme gibi kavramlar öğretmen tarafından incelenir. ✓ Ders içinde kullanılacak standart sapma ve aralık gibi istatistik kavramları öğretmen tarafından incelenir. ✓ Her öğrencinin bir bilgisayar karşına oturması sağlanır.
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma.

SÜREÇ

Öğretmen öğrencilere *"Bundan önce yaptığımız etkinlikte Python programlama dilinde yerleşik olarak bulunan fonksiyonlar ile işlemler yaptık. Bugün ise kullanıcı tanımlı fonksiyonlar yazacağız. Bu fonksiyonları yazmamızın sebebi daha önce belirttiğimiz gibi yeniden kullanılabilirliği artırmak ve kod yapısını sadeleştirmek. Fonksiyonları kullanmak için oluşturma adımına fonksiyon tanımlama, fonksiyonu kullanma adımına ise fonksiyon çağırma denir."*

Bu adımdan sonra öğrencilere parametre kavramı tanım olarak tekrar hatırlatılarak etkinliğe geçilir.

Öğrencilere Python programlama dilinde bir fonksiyon tanımlamanın temel yapısı Kod 3.11.1.'de gösterildiği gibi olduğu anlatılır. Köşeli parantez içinde belirtilen parametre ve geriye değer döndürme kısımlarının tanımlanacak fonksiyona göre seçimlik olduğu gerek duyulmadığı takdirde kullanılmayacağından bahsedilir.

Kod 3.11.1. Fonksiyon temel yapısı

```
def fonksiyon_adi([parametreler]):
    işlemler
    [return değer]
```

Öğrencilere *fonksiyon_parametre.py* isimli bir dosya oluşturmaları istenerek Kod 3.11.2'de bulunan kodu bu dosyaya yazmaları için bir süre verilir.

Öğrencilere kod yazma işlemi bittikten sonra kodu çalıştırmaları ve varsa hataları düzeltmeleri istenir. Sonuçları kontrol etmek için Ekran çıktısı 3.11.1 öğrenciler gösterilir.

Kod 3.11.2 Parametre alan ve almayan fonksiyonlar.

```
def kare_bas_4():
    print(4*"* ")
    for _ in range(3):
        print("*",5*" ","*",sep="")
    print(4*"* ")


def n_kare_bas(n:int):
    print(n*"* ")
    for _ in range(n-2):
        print("*",(2*n-3)*" ","*",sep="")
    print(n*"* ")

def n_kare_karakter_bas(n:int,k:str):
    print(n*(k+" "))
    for _ in range(n-2):
        print(k,(2*n-3)*" ",k,sep="")
    print(n*(k+" "))

kare_bas_4()
print(20*"-")
n_kare_bas(4)
print(20*"-")
n_kare_karakter_bas(5,"@")
```

Ekran çıktısı 3.11.1 Parametre alan ve almayan fonksiyonlar örneğinin ekran çıktısı

```
* * * *
*       *
*       *
* * * *
--------------------
* * *
*     *
* * *
--------------------
@ @ @ @ @
@           @
```

```
@          @
@          @
@ @ @ @ @
```

Burada öğrencilere *kare_bas_4* fonksiyonunun parametre almadığı diğer iki fonksiyonun ise tanımlanma amacına yönelik olarak parametre(ler) aldığı anlatılır.

Öğrencilerden *fonksiyon_parametre.py* dosyasına ek olarak bir karakter dizisi ve bir tam sayıyı parametre olarak alan ve karakter dizisini ekrana tam sayı değeri kadar alt alta ve bir sağa kaydırarak basan *n_kaydir* isimli bir fonksiyon tanımlamaları ve kullanmaları istenir.

Örneğin fonksiyona parametre olarak tam sayı değeri 5, karakter dizisi olarak "BİLSEM" değer gönderildiğinde oluşacak görüntü Ekran çıktısı 3.11.2. 'deki gibi oluşur. Bu çıktı öğrencilere gösterilerek uygun çözümü bulmaları istenir.

Ekran çıktısı 3.11.2. n_kaydir fonksiyonu örnek ekran çıktısı.

```
BİLSEM
 BİLSEM
 BİLSEM
      BİLSEM
      BİLSEM
```

Problemin çözüm yollarından birisi Kod 3.11.3' te gösterildiği gibidir.

Kod 3.11.3 n_kaydir fonksiyonu örnek çözümü

```
def n_kaydir(n:int,k:str):
      for i in range(n):
            print(i*" "+k)
```

Öğrencilere "Şimdiye kadar tanımladığımız fonksiyonlar bir görevi parametre alarak veya almayarak yerine getirip işleri bitiyordu. Buna göre bir program akışında fonksiyon çağrıldığında tanımı gereği olan işlemleri yapar ve görevi bitince program fonksiyonun çağrıldığı noktadan sonrasında çalışmaya devam eder. Buna fonksiyon işlemleri yapınca çağrıldığı noktaya geri döner diyebiliriz. Bu dönüşlerde çağrıldığı noktaya hesaplanan bir sonucu döndürmek ve onu bir değişkene aktarmak veya bu sonucu bir işlemde kullanmak isteyebiliriz." Denilerek geri değer döndürme kavramı hakkında öğrencilere bilgi verilir.

Öğrencilere *fonksiyon_deger_dondurme.py* isimli bir isimli bir dosya oluşturmaları istenerek Kod 3.11.4.'te bulunan kodu bu dosyaya yazmaları için bir süre verilir.

Burada öğrencilere tanımlanan *kare_farki*, *aralik_hesapla* ve *ortalama_hesapla* fonksiyonlarının çağrıldığı noktaya *return* anahtar kelimesi ile hesaplanan değerleri döndürdüğü ve bu değerlerin bir değişkene aktarılmasının iyi bir kullanım olacağı anlatılır.

Kod 3.11.4. Değer döndüren fonksiyon örneği

```
def kare_farki(a:float,b:float):
      kf = abs(a**2-b**2)
      return kf
```

```
def aralik_hesapla(listem):
    ek = min(listem)
    eb = max(listem)
    aralik = eb-ek
    return aralik

def ortalama_hesapla(listem):
    ortalama = sum(listem)/len(listem)
    return ortalama


veri = [10,20,30,40,50,60,70,100]
s1=10
s2=8

kf = kare_farki(s1,s2)
aralik = aralik_hesapla(veri)
ort = ortalama_hesapla(veri)

print(f"{s1} ve {s2} kare farkı :{kf}")
print(f"{veri} listesi aralık değeri:{aralik}")
print(f"{veri} listesi ortalama değeri:{ort}")
```

Kod 3.11.4. çalıştırıldığında oluşacak ekran çıktısı Ekran çıktısı 3.11.3. 'te verildiği gibidir.

Ekran çıktısı 3.11.3. Değer döndüren fonksiyon örneği

```
10 ve 8 kare farkı :36
[10, 20, 30, 40, 50, 60, 70, 100] listesi aralık değeri:90
[10, 20, 30, 40, 50, 60, 70, 100] listesi ortalama değeri:47.5
```

Bu işlemlerden sonra öğrencilerden aşağıda amacı belirtilen fonksiyonu tanımlamaları ve kullanmaları istenir.

fonksiyon_deger_dondurme.py dosyasının içine parametre değeri olarak sayılardan oluşan bir liste alan ve bu listedeki verilerin standart sapmasını hesaplayıp virgülden sonra iki basamak yuvarlayıp çağırıldığı yere döndüren fonksiyonu Python programlama dili ile yazınız.

Kod 3.11.5.'te problem çözümü verilmiştir. Kod çıktısı ise Ekran çıktısı 3.11.4.'te gösterildiği gibidir.

Kod 3.11.5. Standart sapma hesaplama örneği

```
def std_hesapla(listem):
    ort = sum(listem)/len(listem)
    toplam = 0.0
    for e in listem:
        toplam += (e-ort)**2
    std = (toplam/len(listem))**0.5
    return round(std,2)
```

```
veri = [10,20,30,40,50,60,70,100]
std_sapma = std_hesapla(veri)
print(f"{veri} listesi standart sapması:{std_sapma}")
```

Ekran çıktısı 3.11.4. Standart sapma hesaplama örneği

```
[10, 20, 30, 40, 50, 60, 70, 100] listesi standart sapması:27.27
```

Etkinliğin tamamlanmasının ardından öğretmen öğrencilere etkinliğin hedeflediği kazanım ve etkinliği özetleyen bir konuşma yapar:

"Sevgili öğrenciler bugün sizlerle temel olarak kullanıcı tanımlı fonksiyonlar tanımlamayı ve belli bir amaca yönelik fonksiyon tanımlamayı öğrendik. Bir sonraki dersimizde özyinelemeli fonksiyon kavramını öğrenip uygulayacağız."

DEĞERLENDİRME:

Tablo 3.11.1: Kontrol listesi

Kontrol Listesi	Evet	Hayır
Fonksiyon tanımlama kavramını açıklar		
Parametre kavramını açıklar.		
Geri değer döndürme kavramını açıklar.		
Amacına uygun fonksiyonun kodunu yazar.		

ETKİNLİK | 12

ETKİNLİK NO	**3.12**
ETKİNLİK ADI	FONKSİYONLARIM KENDİNİ ÇAĞIRIYOR
SINIF/KADEME	Lise
SÜRE	40 + 40 =80 Dakika
ÖĞRENME ALANI/KONU	Yazılım Geliştirme / Fonksiyonlar
KAZANIMLAR	**3.7.5.** Özyinelemeli fonksiyonların (recursive) çalışma mantığına uygun durumları tespit eder.
TEMEL BECERİLER	Problem çözme, Analitik düşünme, Karar verme
YÖNTEM VE TEKNİKLER	Doğrudan anlatım, Soru-cevap, Gösterip yaptırma
ARAÇ-GEREÇLER	Bilgisayar, Uygulama geliştirme ortamı, internet, Projeksiyon cihazı/ Etkileşimli tahta.
UYGULAYICI İÇİN ÖN HAZIRLIK	✓ Her öğrencinin bilgisayarında yüklü ya da çevrimiçi kullanabileceği bir uygulama geliştirme ortamı olması gerekir. ✓ Ders içinde kullanılacak özyinelemeli fonksiyon kavramı ve ardışık sayıların toplamı formülü öğretmen tarafından incelenir. ✓ Her öğrencinin bir bilgisayar karşına oturması sağlanır.
ÖĞRENCİNİN HAZIRBULUNUŞLUĞU	✓ Temel düzeyde bilgisayar okuryazarlığına sahip olma.

SÜREÇ

Öğretmen öğrencilere daha önce matruşka bebek görüp görmediklerini sorar. Bilmeyen öğrencilerin internetten görsellerde arama yaparak görmeleri sağlanır. Öğrencilere *"Matruşka, oyuncak bebek türüdür. Ahşap el yapımı olan bebekler ortasından açıldığında başka bir bebek çıkar, onu açtığınızda yine başka bir bebek çıkar. Tek anne figürünün içerisinde iç içe yerleştirilmiş beş veya yedi bebekten oluşur. Programlamada da bazen içe içe kendini tekrarlayan durumlar olabilir. Böyle durumlarda özyinelemeli fonksiyonları kullanırız."* diyerek etkinliğe geçilir.

Öğretmen öğrencilere *"Bugün öğreneceğimiz özyinelemeli fonksiyonlar, belli bir problemi daha küçük alt parçalara ayırarak çözme işlemidir. 1'den başlayarak 5'e kadar olan tüm tam sayıları toplamanın 3 farklı yolu vardır. Birinci yol ardışık sayıların toplamı formülü ile ikinci yol bir döngü yardımı ile üçüncü yol ise özyineleme yoludur."* diyerek öğrencilerden *tam_sayi_toplami.py* isimli bir dosya oluşturup dosyaya Kod 3.12.1.'deki kodu yazmaları istenir.

Kod 3.12.1. Ardışık tam sayıların toplamı

```
def topla_formul(n:int):
    toplam = (n*(n+1))//2
    return toplam
```

```
def topla_dongusel(n:int):
	toplam=0
	i=1
	while i<=n:
		toplam+=i
		i+=1
	return toplam

def topla_ozyineleme(n:int):
	if n==1:
		return 1
	else:
		return n+topla_ozyineleme(n-1)


deger = int(input("sayı giriniz:"))

sonuc1 = topla_formul(deger)
sonuc2 = topla_dongusel(deger)
sonuc3 = topla_ozyineleme(deger)

print("formül:",sonuc1)
print("döngüsel:",sonuc2)
print("özyinelemeli:",sonuc3)
```

Öğrencilere programı çalıştırıp sayı giriniz: kısmında 4 değeri girmeleri istenir. Bu işlem sonucunda oluşan ekran çıktısı Ekran çıktısı 3.12.1.'deki gibi olacaktır.

Ekran çıktısı 3.12.1. Ardışık tam sayılar toplamı 4 değeri ekran çıktısı

```
sayı giriniz:4
formül: 10
döngüsel: 10
özyinelemeli: 10
```

Ardından öğrenciler ile her üç çözüm yolu da incelenir.

"Birinci çözüm yolu incelendiğinde ilgili topla_formul fonksiyonu parametre olarak aldığı n tam sayısını ardışık tam sayıların toplamı formülüne (genellikle Gauss formülü denir) uygulayarak elde ettiği sonucu çağrıldığı yere return komutu ile döndürüyor."

"İkinci çözüm yolunda ise topla_dongusel fonksiyonu i döngü değişkenini parametre olarak aldığı n tam sayı değişkenine varıncaya kadar birer artırıyor ve her adımdaki i değerini başlangıç değeri 0 olan toplam değişkenine Tablo 3.12.1.'de gösterildiği gibi ekliyor. Döngü bittiğinde fonksiyon toplam değişkeninin son değerini fonksiyonun çağrıldığı yere return komutu ile döndürüyor."

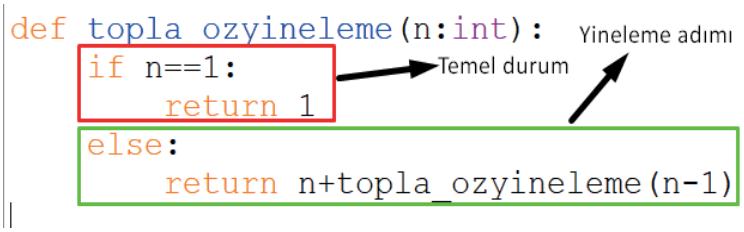
Tablo 3.12.1. *topla_dongusel* fonksiyonu 4 değeri için yerel değişkenleri değişimleri

i değeri	*toplam* ilk değeri	*toplam* son değeri
1	0	1
2	1	3
3	3	6
4	6	10

Üçüncü çözüm yolu özyineleme gereği *topla_ozyinelemeli* fonksiyonu 2 kısımdan oluşuyor temel/taban durum ve yineleme adımı. Kodlar incelendiğinde Görsel 3.12.1.'de gösterildiği gibi, kırmızı kutucuk içindeki kısım temel/taban durumu, yeşil kutucuk içinde kalan kısım ise yineleme adımını temsil etmekte. Tablo 3.12.2.' de ise fonksiyonun problemi temel/taban duruma varana kadar yineleme adımı ile parçalara ayırması ve temel durumdan geriye dönerek problemin son çözüm aşamasında son değeri *return* komutu ile fonksiyonun çağrıldığı yere döndürmesi görülmektedir. (Tablo 3.12.2.'de kısa kullanım için *topla_ozyinelemeli* fonksiyonun adı *fonk* olarak kısaltılmıştır.

Tablo 3.12.2. *topla_ozyinelemeli* fonksiyonu 4 değeri için yineleme ve temel durum adımları

Fonksiyon çağırımı	Adım sayısal durumlara	Adım açıklaması
fonk(4)	4+fonk(3)	Yineleme adımı
fonk(3)	4+(3+fonk(2))	Yineleme adımı
fonk(2)	4+(3+(2+fonk(1)))	1 değeri ile taban duruma varıldı. fonk(1) geriye 1 değeri döndürecek
fonk(1)	4+(3+(2+1))	En içteki (2+1) değeri bir önceki adıma döndürülecek
fonk(2)	4+(3+3)	(3+3) değeri bir önceki adıma döndürülecek
fonk(3)	4+6	4+6 değeri bir önceki adıma döndürülecek
fonk(4)	10	10 değeri fonksiyonun çağrıldığı yere döndürülecek.



Görsel 3.12.1 Özyinelemeli *topla_ozyineleme* fonksiyonun parçaları

Öğrencilere üç fonksiyonun çalışması anlatıldıktan sonra öğrencilerden aşağıdaki problemi çözen Python kodunu yazmaları istenir.

"Parametre olarak aldığı bir tam sayının faktöriyelini hesaplayıp sonucu geri döndüren biri döngüsel diğeri ise özyinelemeli çözümlerini kullanan iki farklı fonksiyon tanımlayarak kullanıcıdan aldığı değerler ile test kodu Python programlama dili ile yazınız. Dosyanıza faktöriyel.py adını veriniz."

Probleme ait örnek bir çözüm Kod 3.12.2'de verildiği gibi olabilir.

Kod 3.12.2 Faktöriyel hesaplama örnek çözümler.

```
def f_dongusel(n:int):
    if n<0:
        return "hata"
    f=1
    while n>1:
        f*=n
        n-=1
    return f

def f_ozyineleme(n:int):
    if n<0:
        return "hata"
    if n<1:
        return 1
    else:
        return n*f_ozyineleme(n-1)

deger = int(input("sayı giriniz:"))


sonuc1 = f_dongusel(deger)
sonuc2 = f_ozyineleme(deger)
print("döngüsel:",sonuc1)
print("özyinelemeli:",sonuc2)
```

Etkinliğin tamamlanmasının ardından öğretmen öğrencilere etkinliğin hedeflediği kazanım ve etkinliği özetleyen bir konuşma yapar:

"Sevgili öğrenciler bugün sizlerle yazılım geliştirmede sıkılıkla kullanılan çözüm yöntemlerinden olan özyinelemeli fonksiyonlar konusunu, özyineleme kavramını ve bu çözüm yöntemini nerde ve nasıl kullanacağımızı öğrendik."

DEĞERLENDİRME:

Tablo 3.12.3: Kontrol listesi

Kontrol Listesi	Evet	Hayır
Öz yinelemeli fonksiyon kavramını açıklar		
Öz yinelemeli fonksiyonun kullanabileceği durumları tespit eder.		
Öz yinelemeli fonksiyon yazar		