

2024-2025 EĞİTİM VE ÖĞRETİM YILI

PROGRAMLAMA DİLLERİ MODÜLÜ (PYTHON)

2. ÜNİTE

PYTHON PROGRAMLAMA DİLİ İLE YAZILIM GELİŞTİRME

1. Karar Yapıları Nedir?

- Karar yapıları, programların belirli koşullara göre farklı sonuçlar üretmesini sağlar.
- Gerçek hayat örneği: Eğer hava yağmurluysa şemsiye alırım, yoksa almam.



2. If-Else Yapısı

- If-else koşulu, bir durum doğruysa bir kod bloğu çalıştırır, aksi halde başka bir kod bloğunu çalıştırır.

- Örnek Kod:

```
hava_durumu = "yağmurlu"
```

```
if hava_durumu == "yağmurlu":
```

```
    print("Şemsiyeni al!")
```

```
else:
```

```
    print("Şemsiyeye gerek  
yok.")
```



3. If-Elif-Else Yapısı

- If-elif-else, birden fazla durumu kontrol etmek için kullanılır.

- Örnek Kod:

```
saat = 15
```

```
if saat < 12:
```

```
    print("Günaydın!")
```

```
elif saat < 18:
```

```
    print("İyi günler!")
```

```
else:
```

```
    print("İyi akşamlar!")
```

4. İç İçe If Yapıları

- Karar yapılarının içinde başka bir karar yapısı olabilir.

- Örnek Kod:

```
hava = "güneşli"
```

```
sıcaklık = 30
```

```
if hava == "güneşli":
```

```
    if sıcaklık > 25:
```

```
        print("Hadi denize gidelim!")
```

```
    else:
```

```
        print("Parkta yürüyüş  
yapabiliriz.")
```



Periodic Table of the Elements

- Örnek Kod:
yağmur_yağıyor = True
if yağmur_yağıyor:
 print("Şemsiyeni al.")

1																	13	14															
H																	IIIA	IVA															
Hydrogen																	5	6															
1.008																	B	C															
																	Boron	Carbon															
																	10.81	12.011															
																	13	14															
																	Al	Si															
																	Aluminum	Silicon															
																	26.9815385	28.085															
7	8	9	10	11	12																												
VIIB	VIIIB	VIIIB	VIIIB	IB	IIB																												
25	26	27	28	29	30	31	32																										
Mn	Fe	Co	Ni	Cu	Zn	Ga	Ge																										
Manganese	Iron	Cobalt	Nickel	Copper	Zinc	Gallium	Germanium																										
54.938044	55.845	58.933194	58.6934	63.546	65.38	69.723	72.630																										
43	44	45	46	47	48	49	50																										
Tc	Ru	Rh	Pd	Ag	Cd	In	Sn																										
Technetium	Ruthenium	Rhodium	Palladium	Silver	Cadmium	Indium	Tin																										
(98)	101.07	102.90550	106.42	107.8682	112.414	114.818	118.710																										
75	76	77	78	79	80	81	82																										
Re	Os	Ir	Pt	Au	Hg	Tl	Pb																										
Rhenium	Osmium	Iridium	Platinum	Gold	Mercury	Thallium	Lead																										
186.207	190.23	192.217	195.084	196.966569	200.592	204.38	207.2																										
107	108	109	110	111	112	113	114																										
Bh	Hs	Mt	Ds	Rg	Cn	Nh	Fl																										
Bohrium	Hassium	Meitnerium	Darmstadtium	Roentgenium	Copernicium	Nihonium	Flerovium																										
(270)	(269)	(278)	(281)	(282)	(285)	(286)	(289)																										
61	62	63	64	65	66	67	68																										
Pm	Sm	Eu	Gd	Tb	Dy	Ho	Er																										
Promethium	Samarium	Europium	Gadolinium	Terbium	Dysprosium	Holmium	Erbium																										
(145)	150.36	151.964	157.25	158.92535	162.500	164.93033	167.259																										
93	94	95	96	97	98	99	100																										
Np	Pu	Am	Cm	Bk	Cf	Es	Fm																										
Neptunium	Plutonium	Americium	Curium	Berkelium	Californium	Einsteinium	Fermium																										
(237)	(244)	(243)	(247)	(247)	(251)	(252)	(257)																										

6. If-Else ve If-Elif-Else Karşılaştırması

If-else: İki durum kontrol edilir.

If-elif-else: Birden fazla durumu kontrol etme imkânı sağlar.

7. Farklı Problemler

- Problem: Bir öğrencinin notuna göre başarı durumunu belirleyelim.

- Örnek Kod:

```
notu = 85
```

```
if notu >= 90:
```

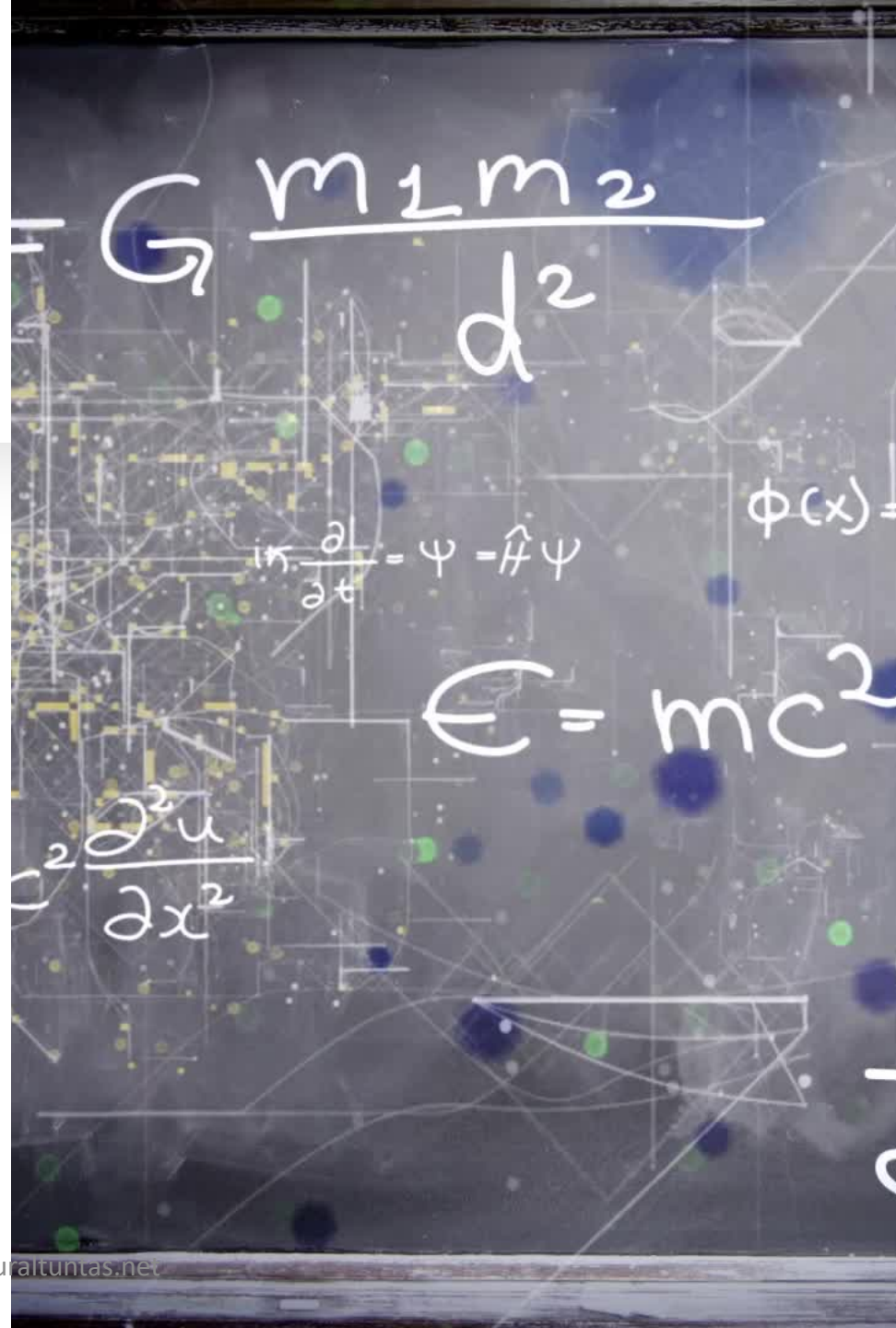
```
    print("Pekiyi")
```

```
elif notu >= 70:
```

```
    print("İyi")
```

```
else:
```

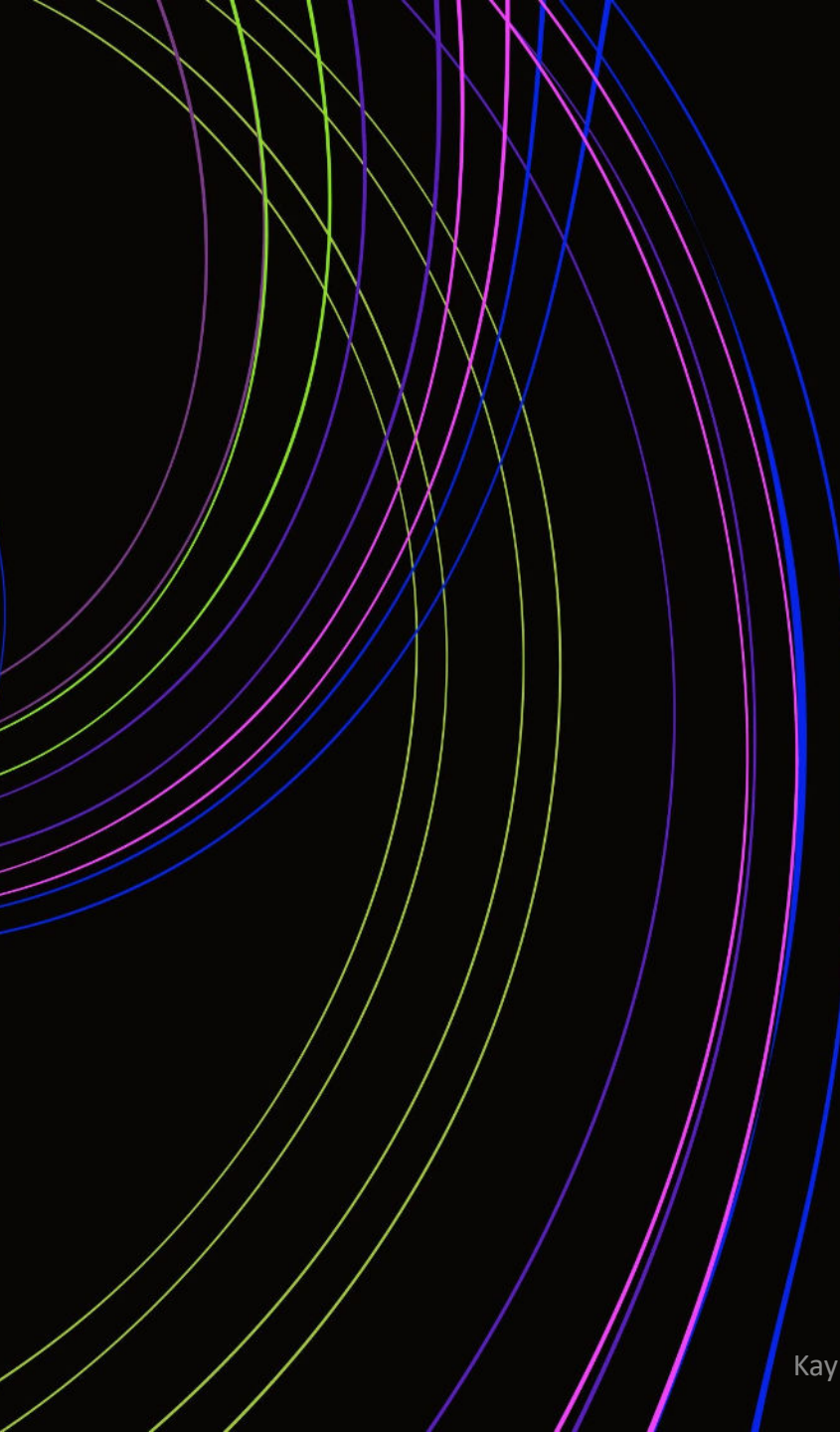
```
    print("Daha fazla çalışmalısın.")
```



1. Döngü Yapıları Nedir?

- Döngü yapıları, belirli koşullar sağlandığı sürece aynı işlemi tekrar tekrar gerçekleştiren yapılar.
- Gerçek hayat örneği: Bir arkadaşına 5 kez 'Merhaba' demek istediğinde aynı kodu tekrar yazmak yerine bir döngü kullanabilirsin.





2. For Döngüsü

- For döngüsü, bir veri kümesi veya belirli bir aralık üzerinden tekrar eder.
- Örnek Kod:

```
for i in range(5):  
    print("Merhaba")
```

3. While Döngüsü

- While döngüsü, koşul doğru olduğu sürece çalışmaya devam eder.

- Örnek Kod:

```
sayı = 0
```

```
while sayı < 5:
```

```
    print("Merhaba")
```

```
    sayı += 1
```

4. İç İçe Döngüler

- Bir döngünün içinde başka bir döngü çalıştırılabilir. Genellikle çok boyutlu veri yapılarında kullanılır.
- Örnek Kod:
- `for i in range(3):`
- `for j in range(2):`
- `print(f"Dış döngü: {i}, İç döngü: {j}")`

5. For ve While Döngüsü Arasındaki Fark

- For döngüsü genellikle bilinen bir aralıkta kullanılır.
- While döngüsü ise koşul sağlandığı sürece çalışır.

- Örnek Kod:

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

While: `i = 0`

`while i < 3:`

`print(i)`

`i += 1`

6. Range Kavramı

- Range, belirli bir aralıkta döngünün çalışmasını sağlar.

- Örnek Kod:

```
for i in range(1, 10, 2):
```

```
    print(i) # 1'den 9'a kadar 2 artırarak ilerler
```


7. Break ve Continue

- Break, döngüyü tamamen sonlandırır.
- Continue, döngünün o adımını atlayarak bir sonrakine geçer.

- Örnek Kod:

```
for i in range(5):  
    if i == 3:  
        break  
    print(i)
```

```
for i in range(5):  
    if i == 3:  
        continue  
    print(i)
```

1. Çoklu Koşullar ve Mantıksal Operatörler

- Çoklu koşullar, birden fazla durumun değerlendirilmesi gereken yerlerde kullanılır.
- Mantıksal operatörler: 'and', 'or', 'not'.
- Pozitif Mantık: Tüm koşullar doğru olmalı (and).
- Negatif Mantık: Tek bir koşul doğruysa yeterli (or).

2. Pozitif ve Negatif Mantık

- Pozitif mantıkta tüm koşulların doğru olması gerekir (and).
- Negatif mantıkta tek bir koşulun doğru olması yeterlidir (or).

- Örnek Kod:

```
x = 5
```

```
y = 10
```

```
if x > 0 and y > 0:
```

```
    print("Her iki sayı da pozitif")
```

```
if x < 0 or y > 0:
```

```
    print("Bir koşul doğru")
```

3. İç İçe Döngüler

- Bir döngü içinde başka bir döngü olabilir. Özellikle çok boyutlu yapılar ve tekrar eden hesaplamalar için kullanılır.
- Örnek: Çarpım tablosu
- Örnek Kod:

```
for i in range(1, 11):  
    for j in range(1, 11):  
        print(f"{i} x {j} = {i*j}")
```

4. Kod Tekrarını Azaltma

- Döngüler sayesinde aynı işlemi tekrar tekrar yazmak zorunda kalmazsın.
- Öğrencilere içinde tekrar eden komutların bulunduğu kod parçaları verilerek, bunların döngülerle nasıl sadeleştirileceği tartışılır.

5. For ve While Döngüleri

- For döngüsü: Belirli bir aralık ya da veri kümesi üzerinde döner.
- While döngüsü: Koşul sağlandığı sürece döner.

- Örnek Kod:

For: `for i in range(5): print(i)`

While: `i = 0`

`while i < 5:`

`print(i)`

`i += 1`

6. Döngülerin Programı Sadeleştirmesi

Döngü kullanarak aynı işlemi tekrar tekrar yazmaktan kaçınabiliriz.



Bu, kodu daha okunabilir ve bakımı daha kolay hale getirir.

1. Fonksiyon Nedir?

- Fonksiyonlar, belirli bir işi gerçekleştiren kod bloklarıdır.
- Bir kez tanımlanır ve ihtiyaç duyuldukça çağrılır.
- Örnek: Bir işlemi tekrar tekrar yapmak yerine, bir fonksiyon yazarak tekrar kullanılabilir hale getirebiliriz.



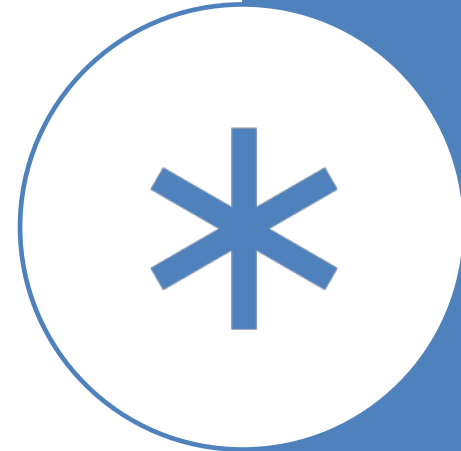
2. Fonksiyon Tanımlama ve Çağırma

- Fonksiyon tanımlaması 'def' anahtar kelimesi ile yapılır.

- Örnek Kod:

```
def merhaba():  
    print("Merhaba!")
```

- merhaba() # Fonksiyonu çağırma



3. Parametrelili ve Parametresiz Fonksiyonlar

- Parametresiz fonksiyonlar: Herhangi bir değer almaz, sabit işlemleri gerçekleştirir.
- Parametrelili fonksiyonlar: Girdi alarak işlemlerini bu girdilere göre yapar.

- Örnek Kod:

```
def selam_ver(isim):
```

```
    print(f"Merhaba, {isim}!")
```

```
selam_ver("Ali")
```



4. Fonksiyon Kullanımının Avantajları



- Fonksiyonlar, kod tekrarını azaltır ve programı daha düzenli hale getirir.
- Aynı işlemi birçok yerde yapmak gerektiğinde fonksiyonları kullanarak kodları tekrar yazmak yerine fonksiyon çağırısı yapabiliriz.

5. Öğrencilerden İsim Yazdıran Fonksiyon

- Bir fonksiyon yazın ve bu fonksiyon, kullanıcının girdiği ismi ekrana yazdırsın.

- Örnek Kod:

```
def isim_yazdir(isim):  
    print(f"İsim: {isim}")
```

```
isim = input("Adınızı girin: ")  
isim_yazdir(isim)
```

6. Parametrelili ve Parametresiz Fonksiyonlar Arasındaki Fark

- Parametrelili fonksiyonlar dışarıdan veri alır, parametresiz fonksiyonlar sabit işlemler yapar.

- Örnek Kod:

- Parametresiz:

```
def sabit_mesaj():  
    print("Bu bir sabit mesaj.")
```

- Parametrelili:

```
def mesaj_yazdir(mesaj):  
    print(mesaj)
```

1. Ön Tanımlı Fonksiyonlar

- Python'da birçok fonksiyon önceden tanımlanmıştır ve kullanıma hazırdır.
- Örnek: `len()`, `count()`, `insert()` gibi fonksiyonlar.
- Örnek Kod:

```
liste = [1, 2, 3, 4]  
print(len(liste)) # 4  
print(liste.count(2)) # 1  
liste.insert(2, 5)  
print(liste) # [1, 2, 5, 3, 4]
```



2. Math Kütüphanesi

- Math kütüphanesi, matematiksel işlemler için kullanılır.

- Örnek: `math.factorial(8)`

- Örnek Kod:

```
import math
```

```
print(math.factorial(8)) # 40320
```



3. Hazır Kütüphaneler



Python'da yerleşik birçok kütüphane vardır, bu kütüphaneler belirli işlevleri hızlıca gerçekleştirmeye yarar.



Örnek: math, random, datetime, os

4. Gömülü Fonksiyonlar ve Kütüphaneler

Gömülü fonksiyonlar: Python'da önceden tanımlanmış ve hemen kullanılabilir fonksiyonlardır.

Kütüphaneler: Belirli bir işlevi gerçekleştiren fonksiyonları barındıran koleksiyonlardır.

Örnek Kütüphaneler:
NumPy, Pandas,
Matplotlib

5. Üst Düzey Fonksiyonlar

Üst düzey fonksiyonlar, fonksiyonları argüman olarak alabilen ve/veya fonksiyonları döndüren fonksiyonlardır.

Örnek: `map()`, `filter()`, `reduce()`

6. Kendini Çağırان Fonksiyonlar (Recursion)

- Bir fonksiyon, kendi içinde kendini çağırabilir. Bu teknik 'recursion' olarak bilinir.

- Örnek Kod:

```
def faktoriyel(n):
```

```
    if n == 1:
```

```
        return 1
```

```
    else:
```

```
        return n * faktoriyel(n-1)
```

- `print(faktoriyel(5)) # 120`

7. Kütüphanelerin Önemi

- Python kütüphaneleri, kodu daha verimli hale getirir ve belirli işlemleri kolaylaştırır.
- Örnek:
- NumPy ile bilimsel hesaplamalar, Pandas ile veri analizi.

