Python Ders Notları - 1

Yazılım geliştirme süreci şu şekilde işler;

- Programcı programlama dili kullanarak kodları oluşturur,
- Yazılan kod bütünü, **hata ayıklayıcı (debugger)** kullanılarak hatalara karşı denetlenir,
- Hataları giderilmiş kodlar, **derleyici (compiler)** kullanılarak bilgisayarın yorumlayabileceği elektriksel sinyallere dönüştürülür. Bu süreç sonunda bilgisayar, elektriksel sinyalleri yorumlayarak komutların gereğini yapar.

Yazılım, bilgisayarın donanımını anlamlı hale getiren, bilgisayarları kullanıcıların amaçları doğrultusunda kullanmasını sağlayan kod ve komutlardır.

Yapılacak bir işlemi ya da hesaplamayı gerçekleştirmek için birbirini izleyen komut veya yönergelerden oluşan yapılardır.

Yazılım Geliştirme Süreci:

- Kullanıcı programı çalıştırdığında, yorumlayıcı programlar, Python kodunu makine koduna çevirir.
- Üst düzey program kodu **kaynak kod (source code)** olarak adlandırılır. Bu koda karşılık gelen makine diline ise **hedef kod (target code)** adı verilir. **Yorumlayıcı**, kaynak kodu hedef koda dönüştürür.
- Üst düzey programların güzelliği, kodlamanın donanımdan bağımsız olarak yapılabilmesidir. Üstünde çalışılan platform ne olursa olsun, Python yorumlayıcısı kurulu ise tüm programlar tüm platformlarda çalıştırılabilir.

Editörler, programcının kaynak kodu yazmasını ve dosyaya kaydetmesini sağlar. Dili oluşturan parçaların kurallara uygun bir şekilde düzenlenmesi **söz dizimi (syntax)** olarak ifade edilir. Derleyiciler, kaynak kod içeriğini dönüştürerek hedef kod içeren bir dosya oluşturur. Derlenerek oluşturulan popüler dillere örnek olarak C, C++, Java ve C# verilebilir.

Yorumlayıcılar da derleyiciler gibi üst düzey kaynak kodu hedef koda (genellikle makine kodu) çevirir ancak derleyicilerden farklı çalışır. Derleyiciler herhangi farklı bir dönüşüm gerekmeden defalarca çalışılabilir bir program kodu üretirken yorumlayıcılar kullanıcı kaynak kodu her çalıştırdığında satır satır makine diline çevirir. Derlenmiş bir program, değişiklik yapılmadığı sürece tekrar derlenmez ancak yorumlayıcı ile çalışan program için yorumlama işlemi değişiklik yapılmamış olsa bile tekrarlanmalıdır. Bu nedenle yorumlanan diller daha çok senaryo dili (scripting language) olarak ifade edilir. Python, yorumlanan bir dil olmakla birlikte, bunun derleyicileri de vardır. Popüler senaryo dillerine örnek olarak Python, Ruby, Perl ve web ortamı için Javascript verilebilir.

Bir kaynak kodu hedef koda çevirdikten sonra çalıştıran ve dolayısıyla koddaki hataları yakalama işlemini ve kodun iyileştirilmesini daha kod çalıştırmadan yapan çeviricilere **derleyici,** kodu satır satır veya bloklar halinde çalıştırıp sırası gelmeyen satırları hiç çalıştırmayan bu satırlardaki hataları hiçbir zaman göremeyen ve kodun bütününe ait iyileştirmeleri yapamayan çeviricilere de **yorumlayıcı** (interpreter) adı verilmektedir.

Hata ayıklayıcılar, programcının bir programdaki olası hataları bulmasına ve düzeltmesine olanak sağlayarak programın doğru çalışması için yardımcı olur.

Geliştirici bütünleştirilmiş geliştirme ortamları (integrated development environment-**IDE**) editörleri, hata ayıklayıcıları ve diğer programlama yardımcılarını kapsar.

Python Nedir?

- Python **özgür ve ücretsiz** bir programlama dilidir.
- **Guido Van Rossum** adlı Hollandalı bir programcı tarafından 90'lı yılların başında geliştirilmeye başlanmıştır. Guido Van Rossum 2005 ile 2012 yılları arasında Google'da çalışmıştır.
- Adı **"The Monty Python"** adlı bir İngiliz komedi grubundan esinlenerek konmuştur.
- Python, öğrenmesi kolay, tamamen özgür ve ücretsiz bir programlama dilidir.
- Python'un dili başka programlama dilleri ile kıyaslandığında, bunun daha az kod ile işlemleri yapmasının mümkün olduğu görülecektir.
- Python, bütün işletim sistemleri ile uyum içerisinde çalışmaktadır.
  Programlama yapısı içerisinde birçok kütüphaneyi barındırmaktadır. Bu kaynaklarla daha az kod yazmak mümkündür.
- Pythton ile masaüstünde çalışan uygulamalar geliştirilebileceği gibi, web üzerinde çalışan uygulamalar geliştirmek hatta Rasperry-Pi gibi donanımları da programlamak mümkündür.

Python Sürümleri:

1. Piyasada iki çeşit Python sürümü vardır. Eğer bir Python sürümü 2 ile başlıyorsa (2.7.11 gibi) **Python 2.x,** şayet 3 ile başlıyorsa (3.6.2 gibi) Python 3.x serisine aittir.
2. Python3, Python2'ye göre daha güçlüdür ve hatalardan arınmıştır.
3. Python2 ile yazılmış bir program Python3'te çalışmaz. Aynı şekilde Python3 ile yazdığınız bir program Python2'de çalışmaz.

Python'u Kurmak:

1) Python'ı kurmak için **http://www.python.org/downloads** adresine tıklayın.
2) **Downloads** linkini tıkladığınızda **'Download Python 3.x.x'** ve **'Download 2.x.x'** yazan yan yana iki düğme göreceksiniz.

IDLE: Python.org web sitesinde yer alan ücretsiz program geliştirme ortamıdır.

Python Ders Notları - 2

VERİ TÜRLERİ:

string: Karakter dizileri. Tırnak içindeki her türlü karakterlerdir.

>>>“Merhaba Dünya”
‘Merhaba Dünya’

Not: string fonksiyonunda çift tırnak kullanmak şart değildir. Tek tırnak veya üç tırnak da kullanabiliriz. Üç tırnak için, üç tek tırnak veya üç çift tırnak karakterleri kullanılabilir. Örnek;

>>>‘Merhaba Dünya’ **→tek tırnak**
‘Merhaba Dünya’

>>>“””Merhaba Dünya””” **→üç çift tırnak**
‘Merhaba Dünya’

Not: Python’da üç tırnak kullanmamızın sebebi alt satıra geçebilmektir. **Ör:**

```
print('''Bilgisayar
Bilimi
Dersi''')
```
Bilgisayar
Bilimi
Dersi

→ Peki, neden bazen tek tırnak veya çift tırnak kullanmamız gerekiyor? Örnek olarak şöyle bir çıktı elde etmek isteyelim.
İstanbul'un havası

Yukarıdaki ifadede bir kesme işareti var. İçinde kesme işareti olan bir string’i tek tırnak içine alamayız. Bu yüzden diğer bir alternatif olan çift tırnağı ya da üç tırnağı kullanmalıyız.

```
>>>print("İstanbul'un havası")          İstanbul'un havası
>>>print("""İstanbul'un havası""")      İstanbul'un havası
```

integer: Tam sayılar. Pozitif, negatif ya da sıfır değeri alabilir. Kesirli değer içermez.
Ör: 10, 0, -10

float: Reel Sayılar. Kayan noktalı sayılar da denir. Ancak burada virgül yerine nokta kullanmamız gerekir. **Ör:** 12.6, 12.0, -12.0, 0.0

complex: Karmaşık sayılardır.
Ör: 10+2j

bool: Herhangi bir ifadenin doğruluğunu veya yanlışlığını sorgular. True, False

Python Ders Notları - 3

FONKSİYONLAR

print(): Ekrana yazdırma fonksiyonudur. Örnekler;

print("Merhaba Dünya") →**stringleri tırnak içerisine almamız gerekir. Burada çift tırnak kullanıldı**
Merhaba Dünya
print(Merhaba Dünya) →**tırnak işareti olmadığı için hata verir**
hata
print(“Merhaba Dünya) →**tırnak işaretini kapatmadığım için hata verir**
hata
print(‘Merhaba Dünya’) → **stringlerde tek tırnak kullanabiliriz.**
Merhaba Dünya
print(“’Merhaba Dünya’”) → **stringlerde üç adet tek tırnak kullanabiliriz.**
Merhaba Dünya
print(“””Merhaba Dünya”””) → **stringlerde üç adet çift tırnak kullanabiliriz.**
Merhaba Dünya

print(“””””Ali”””””)
“”Ali
Açıklaması: İlk üç tırnak ile string açıldı, sonra kalan iki tırnak ve Ali ekrana yazıldı, Ali’den sonra ki üç tırnak ile string kapandı, daha sonra tırnak ile yeni bir string açılıp yine tırnakla kapanarak içi boş bir string yazıldı.

print(“Bilgisayar”,”Bilimi”,”Kodlama”) → **virgüller stringleri ayırır ve yan yana yazdırır.**
Bilgisayar Bilimi Kodlama

print("bilgi"+”sayar")
bilgisayar

print(“Bilgisayar"+" "+"Bilimi”) →**ortadaki çift tırnağın içinde bir boşluk karakteri var.**

print("Bilgisayar" " " "Bilimi”) →**birleştirmelerde + işaretini kullanmak zorunda değiliz**.
Bilgisayar Bilimi

print(5) **5**
print(“5”) **‘5’**
print(-5) **-5**
print(5+4) **9**
print(“5+4”) **‘5+4’**
print(“ ”) →**bir boşluk karakteri yazdırır. Boşluk karakteri de bir string’dir.**
print(999, “9”) **999 9**
print(999+ “9”) **hata**

Python Ders Notları - 4

type(): Verilerin tipini sorgular.

type("Merhaba") **<class 'str'>**
type(Merhaba) → **tırnak işareti olmadığı için hata verir.**
type("") **<class 'str'>**
type("4") **<class "str">**
type(4) **<class "int">**
type(-4) **<class "int">**
type(4 + 7) **<class "int">**
type("4 + 7") **<class "str">**
type(4.2) **<class 'float'>**
type(4.0) **<class 'float'>**
type(0.0) **<class 'float'>**
type(0) **<class "int">**
type(2.5+2.5) **<class 'float'>**

str(): Verileri string'e (karakter dizisi) çevirir.

str(4) **'4'**
str(4.0) **'4.0'**
str(10+2) **'12'**
str(10/2) **'5'**
str(10*2) **'20'**

int(): Verileri integer'a (tamsayı) çevirir.

int("5") **5**
int("-5") **-5**
int(28.9) **28**
int("28.9") **28**
int(5+4) **9**
int("5+4") →**hata**
int(5/3) **1**
int(5*0.4) **2**
int(0.8+0.9) **1**

float(): Verileri float'a (reel sayı) çevirir.

float(5) **5.0**
float(-5) **-5.0**
float("5") **5.0**
float(0) **0.0**
float(3+4) **7.0**
float("3+4") **hata**
float(3.5+4) **7.5**

len(): stringlerin uzunluğunu ölçer. Örnekler;

len("Türkiye") **7**

len("Bilgisayar Bilimi") **17**
#"Bilgisayar Bilimi" stringinin uzunluğu 17 karakterdir. Burada boşluk karakterinin de sayıldığına dikkat edelim.

len("Bilgisayar Bilimi")+ len("Dersi") **22**
 # "Bilgisayar Bilimi" stringi ile "Dersi" stringinin uzunluğu sayısal olarak toplandı.

len("Bilgisayar Bilimi")- len("Dersi") **12**
 # "Bilgisayar Bilimi" stringi ile "Dersi" stringinin uzunluğu sayısal olarak toplandı.

Örnekler:

len("len") **3**
len("0") **1**
len("4") **1**
len("-4") **2**
len("4.5") **3**
len("-0.5") **4**
len(563) →**hata (string değil)**
len("5+4") **3**
len("100-2") **4**
len(str(999)) **3**
len(str(10-9)) **1**
len(int("10") **hata (parantez içinde integer olduğu için len komutu çalışmaz)**

Python Ders Notları - 5

OPERATÖRLER (İŞLEÇLER):

Aritmetik Operatörler:

Toplama : +
Çıkarma : -
Çarpma : *
Bölme : /
Tam Bölme: //
Üs Alma : **
Mod : %

+ operatörü: Toplama işlemi ve string birleştirme için kullanılır.

```
>>>10+20        30
>>>5.5+3.8      9.3
>>>9+4.0        13.0
```

→ Sayıların çift tırnak içine alınmadığına dikkat edin. Eğer çift tırnak içine alınsaydı veri türü bir integer değil string olacaktı. Bu durumda matematiksel işlem yapılamayacaktı. Örnekler;

```
>>>"10"+"20"      '1020'
>>>"10" + 20       →hata
>>>"5" + str(10)  '510'
>>>5 + int("10")   15
```

- operatörü:

```
>>>50-30          20
>>>-7- - 9        2
>>>1.5-0.5        1.0
>>>4.0-1          3.0
```

*** operatörü:**

```
>>>10*5           50
>>>-6*-8          -48
>>>1.5*1.5        2.25
```

Ör:

```
>>>x=4
>>>y=3
>>>3*x+2*y-5
13
```

Ör:

```
>>>"w" * 3                    www
>>>"aheste " * 2              aheste aheste
>>>"-" * 10                   ----------
>>> "uzaksın bana" + " " * 5 + "çook uzak...")
uzaksın bana     çoook uzak...
```

/ operatörü: Bölme işlemi gerçekleştirir.

```
>>>21/3          7.0
>>>int(21/3)     7
>>>21/0          →hata
```

// operatörü: Bölme işleminde kalan sayı göz ardı edilir.(Taban Bölme)

```
>>>25//6        4
>>>6//25        0
>>>4.5//1.2     3.0
>>>2.1//1       2.0
```

Not: - eksi ve / bölü işleçlerini karakter dizileri ile birlikte kullanamayız.

Üs Alma:

Birinci Yol:

```
5**2        25
5**-1       0.2
5**0        1
```

İkinci Yol:

```
pow(5,2)   25
pow(5,-1)  0.2
pow(5,0)   1
pow(11,3,4)  3      →11'in 3. kuvveti olan 1331'in  4'e bölümünden kalan sayı
```

$5*10^2$ gibi bir ifade şöyle yazılabilir:

```
5e2         5000          →simge olarak "e" yerine "E" de kullanılabilir.
```

Mod İşlemi: Bölme sonucunda kalan sayıyı verir.

```
25%7      4
22%11     0
6%25      6
0%25      0
25%0      →hata
```

Karekök Alma: Bir sayının 0.5. kuvveti o sayının kareköküdür.

```
>>>144**0.5       12
```

Yuvarlama:

```
round(28.71)      29
round(28.47)      28
round(29.5)       30
round(30.5)       30          →en yakın çift sayıya yuvarlıyor
```

```
x = 8793.748
round(x)          8794
round(x, 1)       8793.7
round(x, 2)       8793.75
round(x, 0)       8794.0
round(x, -1)      8790.0
round(x, -2)      8800.0
```

_ (alt çizgi işareti): Son verinin değerini hafızada tutar.

Ör:

```
>>>10+5
15
>>>_
15     →son işlem olan 15 değerini hafızada tutup bize gösterdi.
 >>>_+5                  →altçizgi ile 5'i topladık.
20                       →artık  son öğe 15 değil 20 oldu.
>>>_/10
2.0
```

Ör:

```
>>> "www"
'www'
>>> _ + ".google.com"
'www.google.com'
```

Python Ders Notları - 6

Aritmetik Alıştırmalar:

>>>5+4*3/3-9
0.0
>>>15-2**4/2+(-2-2)
3.0
>>>0-9**0
-1
>>>100**0.5/10*2
2.0
>>>10%6-1
3
>>>39//12+12/2
9.0
>>> “10+20+30”
10+20+30
>>> “55”+”55”
5555
>>> “110”,”110”
111 110
>>>5+5+”5”
hata
>>>str(12+8)+"0"
200
>>>int("56")+int("12")
68
>>>int("10"),int("10")
10 10
>>>str(1),int("666")
1 666
>>> “k”*3+str(5)
kkk5
>>>2*"6",66)
66 66
>>>5+4*10/(4+5-9)
hata

Sabitler, Değişkenler ve Atama: Değişmeyen değerlere sabit, onlara atanan ifadelere ise değişken denir. Böylece daha sade, işlevsel ve zaman kazandırıcı işlemler yapılır. Örnekler;

>>>a = "Trabzon" → **a burada değişkendir. “Trabzon” ise sabit değerdir. = ise atama işlemi yapar.**
>>>print(a)
Trabzon
>>>len(a)
7
>>>type(a)
<class 'str'>

Ör:
>>>a= 5
>>>b=-4
>>>c=30
>>>print(a*b+c)
10

Not: Burada atama (=) sembolünün anlamı matematikte kullanıldığı şeklinden farklıdır. Matematikte bu sembol eşitlik sağlar fakat Python ’da simetri olmadığı için 5 = x gibi bir ifade hatalı olacaktır.
>>>5=x
>>>print(x) →**hata**

Not: Python operatör kullanırken kısaltmalar yapabilir. Örneğin x= x + 5 deyimi x + = 5 olarak kısaltılabilir. Bu ifade “ x’i 5 arttır.” anlamına gelir.

Ör:
```
>>> x=3
>>> x=x+2
>>> print(x)
5
>>> x+=7        → x=x+7 ile aynı anlama gelir.
>>> print(x)
12
>>> x -= 4
>>> print(x)
8
>>> x //= 3
>>> print(x)
2
>>> x *= 8
>>> print(x)
16
>>> x %= 6
>>> print(x)
4
>>> x**= 3
>>> print(x)
64
```

Python Ders Notları - 7

Değişken Kuralları:

1-Değişken adları sayı ile başlamaz.

3_kilo_elma = “10 tl” ✗

kilo_elma_3 = "10 tl" ✔

2-Değişken adları özel sembol içermez (_ altçizgi hariç).

gelir?= “500 TL” ✗

kullanici_adi= “admin” ✔

3- Değişken adlarında boşluk olmaz.

kullanici adi = “admin” ✗

kulllanici_adi = “admin” ✔

Not: Değişken adlarında Türkçe karakter kullanabiliriz. Ancak uyum sorunu ihtimaline karşı bundan kaçınınız.

4- Değişken adlarında bazı özel anlam ifade eden kelimeler kullanılmaz.

True=5 ✗

true=5 ✔ →Küçük büyük harf duyarlılığından hata oluşmaz.

and=8 ✗

And=8 ✔ →Küçük büyük harf duyarlılığından hata oluşmaz.

Not: Python’da özel anlam ifade eden kelimeleri görmek için aşağıdaki kodları yazın.

```
>>>import keyword
>>>keyword.kwlist
```

['False', 'None', 'True', 'and', 'as', 'assert', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']

Ör:

```
>>> import keyword
>>> a=keyword.kwlist
>>> len(a)   → Python’da özel anlam ifade eden kaç adet kelime var?
33
>>> type(a)  → Özel anlam ifade eden kelime grubu hangi veri türüne girer?
<class 'list'>
```

Not: Bir değere iki veya daha fazla değişken atayabiliriz.

```
>>>a=b=c=4
>>>print(a*b+c)   → 4*4+4
20
```

Not: Bir değişkene defalarca farklı değerler atayabiliriz:

```
>>>x = 10
>>>x = 20
>>>x = 30        → Değişkenin en son değeri
>>>print(x)      → Değişkenin en son değeri geçerlidir.
30
```

Çoklu atama:

```
>>> x, y, z = 4,3,2
>>> print(x*y-z)   → 4*3-2
10
>>> x, y, z = 4,3,2
>>> print("x =", x, " y =", y, " z =", z)
x = 4  y = 3  z = 2
```

Değişken Takası:

```
>>> x,y,z=4,3,2
>>> x,y,z=y,z,x        → x artık y oldu, y artık z oldu, z artık x oldu.
>>> print(z**y*x)   → 4**2*3
48
```

Değişken İptali: Değişken iptali için komut penceresini kapatıp açabiliriz ya da del komutunu kullanabiliriz.

```
>>>a=2
>>>print(a)
2
>>>del a        → a değişkeni artık yok yani iptal edildi.
>>>print(a)     → a değişkenini iptal ettiğimiz için kod hata verir. ✗
```

Hata Çeşitleri:

1-Söz Dizimi Hataları (SyntaxError) :

Ör1: Hatalı atama işlemi;

```
>>>y = 5
>>>x + 2 = y
>>>print(x)
```

doğrusu:

```
y = 5
x=y-2
print(x)
3
```

Ör2: Eşleşmeyen parantez;

```
print(5+(3*4)   hata
print(5+(3*4))  doğrusu
```

Ör3: Eşleşmeyen tırnak işareti;

```
print(‘hello”)   hata
print(“hello”)   doğrusu
print(‘hello’)   doğrusu
```

Ör4: Hatalı girinti;

```
a=4
b=6
if a<b:
print("a, b'den küçüktür")
```

Doğrusu:

```
a=4
b=6
if a<b:
    print("a, b'den küçüktür")
```

2-Çalışma Zamanı Hataları:

Ör1: Atanmayan değişken;

```
x=5
print(x+y)
```

→**y diye bir değişken yok, yani atanmamış.**

doğrusu:

```
x=5
y=8
print(x+y)
13
```

Ör2: Sıfıra bölme işlemi;

```
32/0      # Bir sayı 0’a bölünemez.
```

Ör3: Sayıları string’e bölmek;

```
print(5/”a”)
```

Ör4: Hiç gerçekleşmeyecek koşullar;

```
if 2>4:
    print("Merhaba Uzaylı")
```

3-Mantık Hataları (Anlam Bilimsel Hatalar) : Program, genellikle hata mesajı vermeden çalışır ancak istenilen işlemi gerçekleştirmez. En zor hata ayıklama türüdür.

Python'da Program Yazma

1. Başlat menüsünden **IDLE** (Python GUI)'ı çalıştırınız
2. "File" menüsünden "New Window" yazarak yeni pencere açınız
3. **Kodları yazınız.**
4. "File" menüsünden "Save" veya Ctrl S ile oluşturduğunuz dosyayı kaydediniz.
5. "Run" menüsünden "Run module" veya F5 ile programı çalıştırınız.
6. Sonucu "Python Shell" de gözleyiniz.

input() Fonksiyonu

Python'da kullanıcıdan herhangi bir veri alıp, yazdığımız programları tek taraflı olmaktan kurtarmak için input() adlı bir fonksiyondan faydalanıyoruz.

Kullanımı:

```
sayı1 = input("Toplama işlemi için ilk sayıyı girin: ")
sayı2 = input("Toplama işlemi için ikinci sayıyı girin: ")
print(sayı1, "+", sayı2, "=", sayı1 + sayı2)
```

---

format()

'karakter dizisi biçimlendirme' işlemleri için kullanılır. Programlama yaparken bazı yerlerde bir stringin içinde daha önceden tanımlı string,float, int vs. değerleri yerleştirmek isteyebiliriz. Böyle durumlar için Pythonda *format()* fonksiyonu bulunmaktadır.

Kullanımı:

```
>>> print("{} {} yaşında bir {}dur".format("Ahmet", "18", "futbolcu"))
'Ahmet 18 yaşında bir futbolcudur'


>>>a = 3
>>>b = 4
>>>print("{} + {} 'nin toplamı {} 'dir".format(a,b,a+b))
'3 + 4 'nin toplamı 7'dir'
```

Python'da Yorum (Açıklama) Satırı Oluşturma

Yorum satırları yazdığımız bir şeyi kod olarak algılattırmaz ve ekrana yazdırmaz. Böylece kendimize göre küçük notlar yazabiliriz. Mesela bir kod yazdırdık bu kodun ne işe yaradığını veya ne için yazdığımızı yorum satırında yazarız ve ekrana yazdırmaz. Böylece geri dönüp kodlara baktığımızda neyi ne için yazdığımızı kolayca hatırlamamıza yardımcı olur

İşareti = Bu işaret sadece tek satırlık yorum satırı oluşturmak için kullanılır.

Kullanımı:

>>>print ("Pythonda Yorum Satırı Oluşturma") # Bu bir yorum satırıdır

Python yukarıdaki satırda # işaretinden sonraki yazıyı yorum satırı olarak algılar ve ekrana yazdırmaz.

İlk Program Örnekleri

2 sayıyı toplama

```
print(" ***** İlk Program ***** ")
print("-"*25)
sayi1=int(input("1.Sayıyı giriniz:"))
sayi2=int(input("2.Sayıyı giriniz:"))
toplam=sayi1+sayi2
print("-"*25)
print("İşlem Sonucu:",sayi1,"+",sayi2,"=",toplam)
```

Fahrenheit-Derece Çevirme

```
f=float(input("Sıcaklığı F cinsinden giriniz: "))
c=(f-32)/1.8
print(f, "F = ", c,"C'dir")
```

Kare Alan ve Çevre Hesaplama

```
kenar=int(input("Kenar uzunluğunu giriniz:"))
cevre=kenar*4
alan=kenar**2
print("Karenin Çevresi:", cevre)
print("Karenin Alanı:", alan)
```

Girilen Saniye Değerini Saate Dönüştürme

```
saniye=int(input("Saniye Giriniz:"))
saat=saniye//3600   #1saat 3600 saniyeye eşittir.
saniye=saniye%3600
dakika=saniye//60
saniye=saniye%60
print("Girdiğiniz Saniyenin Saat Dönüşümü:",saat,"sa",dakika,"dk",saniye,"sn")
```

Beden Kitle İndeksi Hesaplama Programı

```
boy=float(input("Boyunuzu giriniz(m): "))
kilo=float(input("Kilonuzu giriniz(kg): "))
bki=kilo/(boy**2)
print("Beden Kitle İndeksi Değeriniz:",round(bki,2))
```

Kişi Bilgi Alma Programı(format kullanma)

```
print("--Kişi Bilgi Alma Programı--")
ad=input("Adınızı Giriniz: ")
soyad=input("Soyadınızı Giriniz: ")
sinif=input("Sınıfınızı Giriniz: ")
print("")
print("Bilgiler kaydediliyor...")
print(" Ad:{}\n Soyad:{}\n Sınıf:{}".format(ad,soyad,sinif))   #  "\n" alt satıra yazdırmayı sağlar.
print("Kaydedildi...")
```

Karşılaştırma Operatörleri:

==	**Eşittir**
!=	**Eşit değildir**
>	**büyüktür**
<	**küçüktür**
>=	**büyük eşittir**
<=	**küçük eşittir**

Bool Kavramı: Bool herhangi bir ifadenin doğruluğunu veya yanlışlığını sorgular. Eğer bir sorgulamanın sonucu doğru ise True, yanlış ise False çıktısı alıyoruz.

a=1

a==1	**True**
a==2	**False**
a!=5	**True**
a!=1	**False**
a>2	**False**
8>a	**True**
a>=1	**True**
a>=2	**False**
“Bilim”<”Kodlama”	**True** →**alfabetik sıraya bakar**

Bool işleçleri sadece doğruluk-yanlışlık sorgulamaya yarayan araçlar değildir. Bilgisayar biliminde her şeyin bir bool değeri vardır. 0 değeri ve boş veri tipleri False ‘tur. Bunlar dışında kalan her şey ise True ‘dur.

bool(5)	**True**
bool(5.8)	**True**
bool(-5)	**True**
bool(“Steve Jobs”)	**True**
bool(“0”)	**True**
bool(“ “)	**True**
bool()	**False**
bool(““)	**False**
bool(0)	**False**
bool(0.0)	**False**

and, or, not operatörleri

```
>>>a = 23
>>>b = 10
>>>a == 23 and b == 10
True
>>>a == 23 and b == 56
False
>>>a == 23 or b == 56
True
```

not: Değil anlamı taşır. Kullanıcı tarafından bir değişkene veri girilip girilmediğini denetlemek için kullanılabilir. Örneğin:

```
>>>a = 23
>>>not a
False
>>>a = ""

>>>not a
True
>>>a=0
>>>not a
True
```

Aitlik Operatörü: Aitlik işleçleri, bir karakter dizisi ya da sayının, herhangi bir veri içinde bulunup bulunmadığını sorgular. Python ’da bir tane aitlik işleci bulunur. Bu işleç de **in** işlecidir.

```
>>>a = "abcd"
>>>"b" in a          →“b” ifadesi a değişkeninin içinde mi?
True
>>>"f" in a          →“f” ifadesi a değişkeninin içinde mi?
False
```

Kimlik Operatörü: Python ‘da her şeyin bir kimlik numarası vardır. id() fonksiyonu ile bu kimlik numarasını bulabiliriz.

```
>>>a = 100
>>>id(a)
137990748          →a değişkeninin temsil ettiği 100 sayısının kimlik numarası
```

Koşullu Durumlar (if-else koşullu durumları)

if Bloğu

if bloğu programımızın içinde herhangi bir yerde belli bir koşulu kontrol edecek isek kullanılan bloklardır. Yazımı şu şekildedir;

```
if (koşul):
        # if bloğu - Koşul sağlanınca (True) çalışır. Bu hizadaki her işlem bu if bloğuna ait.. if bloğu girintiyle oluşturulur.
        Yapılacak İşlemler
```

if bloğu *eğer koşul sağlanırsa anlamı taşır*. Eğer if kalıbındaki koşul sağlanırsa (True) if bloğu çalıştırılır, koşul sağlanmazsa (False) if bloğu çalıştırılmaz. Hemen bir örnek ile koşullu durumları anlamaya çalışalım.

```
# 18 yaş kontrolü
yaş = int(input("Yaşınızı giriniz:"))

if (yaş < 18):
    # if bloğu -  Girinti ile sağlanıyor.
    print("Bu mekana giremezsiniz.")
```

```
Yaşınızı giriniz:17
Bu mekana giremezsiniz.
```

else Bloğu

else blokları if koşulu sağlanmadığı zaman (False) çalışan bloklardır. Kullanımı şu şekildedir;

```
else:
        # else bloğu - Yukarısındaki herhangi bir if bloğu (veya ilerde göreceğimiz elif bloğu) çalışmadığı zaman çalışır.
        # else bloğu - Girintiyle oluşturulur.
        Yapılacak İşlemler
```

Dikkat ederseniz burada else koşulunun yanına herhangi bir koşul yazmadık. Çünkü zaten else bloğunun çalışması ondan önce gelen diğer koşulların sağlanmamasına bağlı oluyor. İsterseniz, 18 yaş kontrolü örneğini else bloğunu dahil ederek tekrar yazalım.

```
# 18 yaş kontrolü
yaş = int(input("Yaşınızı giriniz:"))

if (yaş < 18):
    # if bloğu -  Girinti ile sağlanıyor.
    print("Bu mekana giremezsiniz.")
else:
    # else bloğu -  if koşulu sağlanmazsa çalışacak.
    print("Mekana hoşgeldiniz.")
```

```
Yaşınızı giriniz:17
Bu mekana giremezsiniz.
```

if-elif-else Blokları

Programlamada bir durum bir çok koşula bağlı olabilir. Örneğin bir hesap makinesi programı yazdığımızda kullanıcının girdiği işlemlere göre koşullarımızı belirleyebiliriz. Bu tür durumlar için if-elif-else kalıplarını kullanırız. Programlarımızda, kaç tane koşulumuz var ise o kadar elif bloğu oluşturabiliriz. Ayrıca else in yazılması zorunlu değildir. if - elif - else kalıplarında, hangi koşul sağlanırsa program o bloğu çalıştırır ve if-elif blokları sona erer. Şimdi isterseniz kullanıcıya işlem seçtirdiğimiz bir programla , bu kalıbı öğrenmeye çalışalım.

```
işlem =  int(input("İşlem seçiniz:")) # 3 tane işlemimiz olsun.

if işlem == 1:
    print("1. işlem seçildi.")
elif işlem == 2:
    print("2. işlem seçildi.")
elif işlem == 3:
    print("3. işlem seçildi.")
else:
    print("Geçersiz İşlem!")
```

```
İşlem seçiniz:1
1. işlem seçildi.
```

Koşullu Durum Örnekleri

1- İki sayıyı karşılaştırma

```
a=int(input("1. Sayıyı Giriniz: "))
b=int(input("2. Sayıyı Giriniz: "))
if a>b:
   print("Birinci sayı büyük!")
elif a<b:
   print("İkinci sayı büyük!")
else:
   print("Sayılar eşit!")
```

2- Girilen sayınının 10'la karşılaştırması

```
sayi=int(input("Sayı Giriniz: "))
if sayi>10:
   print("Girdiğiniz Sayı 10'dan büyüktür!")
if sayi<10:
   print("Girdiğiniz Sayı 10'dan küçüktür!")
if sayi==10:
   print("Girdiğiniz Sayı 10'dur!")
```

3- 1..10 Arası Değer Alma

```
deger=int(input("0...10 aralığında sayı giriniz: "))
if deger>=0 and deger<=10:
   print("Doğru bir değer girdiniz...")
else:
   print("Aralık dışında sayı girdiniz...")
```

4- Sıfıra Bölme

```
bolum=int(input("Bölünecek sayıyı giriniz: "))
bolen=int(input("Bölen sayıyı giriniz "))
if bolen!=0:
   print("Sonuç: ", bolum/bolen)
else:
   print("Sıfıra bölme işlemi yapılamaz!")
```

5- Şifre

```
parola= input("Parola gir:")
if parola=="python":
   print ("hoşgeldiniz...")
   print ("yönlendiriliyorsunuz...")
else:
   print ("geçersiz parola")
   print ("tekrar dene")
print ("---------------------------------")
```

6- Takdir&Teşekkür Durumu

```
ortalama=float(input("Not ortalamanızı giriniz: "))
if ortalama>=85:
   print("Takdir Belgesi Alamaya Hak Kazandınız!")
elif ortalama>=70:
   print("Teşekkür Belgesi Alamaya Hak Kazandınız!")
elif ortalama>=50:
   print("Sınıfı Geçtiniz")
else:
   print("Sınıf tekrarı...")
```

7- Girilen rakamı yazıya çevirme (1-5)

```
deger=int(input("1...5 arasında değer giriniz: "))
if deger<1:
   print("Değer çok küçük...")
elif deger==1:
   print("Bir")
elif deger==2:
   print("İki")
elif deger==3:
   print("Üç")
elif deger==4:
   print("Dört")
elif deger==5:
   print("Beş")
else:
   print("Değer çok büyük...")
print("Tamamlandı...")
```

8- Girilen Sayının Tek veya Çift Olduğunu Bulma

```
sayi=int(input("Sayı giriniz: "))
if sayi%2==0:
   print("Girdiğiniz sayı çifttir")
else:
   print("Girdiğiniz sayı tektir")
```

9- Sayının 2 ve 3'e Bölünme Durumu

```
print("-----Girilen Sayının 2 ve 3'e Bölünme Durumu----")
sayi=int(input("Sayı Giriniz: "))
if sayi%2==0:
   if sayi%3==0:
      print("Girdiğiniz sayı 2 ve 3'e tam bölünüyor.")
   else:
      print("Girdiğiniz sayı sadece 2'ye tam bölünür.")
elif sayi%3==0:
   print("Girdiğiniz sayı sadece 3'e tam bölünür.")
else:
   print("Girdiğiniz sayı 2 ve 3'e tam bölünmez.")
```

Python Ders Notları - 13

Döngü Yapıları

Döngüler, sıralı bir kod bloğunun istenilen sayıda tekrarlanmasıdır. Döngü ve karar yapıları, algoritma oluşturma ve programlamada birçok problemin çözümünde kullanılır.

FOR Döngüsü

For döngüleri belirli sayıda işlemlerin tekrarlanması için kullanılan döngülerdir. For döngüleri başlangıç ve bitiş değerleri arasında artım miktarına göre istenilen sayıda tekrar yapar. Genellikle range() fonksiyonu ile birlikte kullanılmakla beraber string ve liste veri türleri içerisindeki elemanlar üzerinde gezinmekte için de kullanılabilmektedir. Örnek Kullanımı:

```
for i in range(1,6):
  print(i)
```

FOR Döngüsü Söz Dizimi

range (başlangıç değeri, son değer, arttırma/azaltma değeri) :

Başlangıç değeri: Döngü değişkeninin alacağı ilk değerdir. Eğer boş bırakılırsa 0 olarak belirlenir.

Son değer: Döngü değişkeninin bitiş değeridir. Boş bırakılmamalıdır. Artırma/azaltma değeri: Döngü değişkeninin artırma veya azaltma miktarını belirler. Eğer boş bırakılırsa, 1 olarak belirlenir.

Ör: 1'den 5' kadar olan sayıları yazdıralım.

```
print(1)
print(2)
print(3)
print(4)
print(5)
```

Çıktı:
1
2
3
4
5

→Ancak, 1'den 100'e kadar yazmak gerekirse böyle bir çözüm yolu doğru olmayacaktır! Bu durumda döngü yapıları tercih edilmelidir. Python dilinde döngü için while ve for yapıları kullanılır.

Ör: 1'den 100'e kadar olan sayıları yazdıralım. Değişkenimiz n olsun:

```
for n in range(1,100):
    print(n)
```

Çıktı
1
2
.
.
98
99

Ör: Şimdi de 1'den 100'e kadar olan tek sayıları yazdıralım. Değişkenimiz yine n olsun:

```
for n in range(1,100,2):
    print(n)
```

Çıktı:
1
3
5
.
.
97
99

→Açıklama: range(1,100,2) ifadesindeki 1 başlangıç sayısıdır. Eğer burası boş bırakılırsa sayı otomatik olarak sıfırdan başlar. 100 ise yazılacak sayıların sınırıdır. 100 çıktıya dâhil değildir. 2 ise artış miktarını gösterir. Yani sayıyı 2'şer arttırır.

Örnekler:

range(10) → **0,1,2,3,4,5,6,7,8,9** **→başlangıç ve artış değeri yok. Sadece bitiş değeri var.**

range(1, 10) → **1,2,3,4,5,6,7,8,9** **→ artış değeri yok. Sadece başlangıç ve bitiş değeri var.**

range(1, 10, 2) → **1,3,5,7,9** **→ başlangıç, bitiş ve artış değeri var.**

range(10, 0, -1) → **10,9,8,7,6,5,4,3,2,1** **→ buradaki artış değeri eksiye doğru gitmektedir.**

range(10, 0, -2) → **10,8,6,4,2**

range(2, 11, 2) → **2,4,6,8,10**

range(-5, 5) → **−5,−4,−3,−2,−1,0,1,2,3,4**

range(1, 2) → **1**

range(1, 1) → **()**

range(1, -1) → **()**

range(1, -1, -1) → **1,0**

range(0) → **()**

Ör: 21'den 0'a kadar olan sayıları 3'er 3'er azaltarak yazdıralım.

```
for n in range(21,0,-3):
    print(n)
```

Çıktı:
21
18
15
12
9
6
3

→ Yukarıdaki çıktıyı bir de yan yana gelecek şekilde yazdıralım.

```
for n in range(21,0,-3):
    print(n, end=" ")
```

Çıktı:
21 18 15 12 9 6 3

Python Ders Notları - 13

Ör: 1'den 100' kadar olan sayıların toplamı:

```
top= 0
for i in range(1,100):
     top+= i
print(top)
```

Çıktı: 4950

Ör: 10 ve 10'un üstleri yazdıran program:

```
for i in range(7):
  print("{}".format(10**i))
```

Çıktı:

1
10
100
1000
10000
100000
1000000

Ör: Bir string değişkeni oluşturarak, string'teki her bir karakteri ayrı ayrı işleme ve yazdırma:

```
sayılar = "12345"
for sayı in sayılar:
     print(int(sayı) * 2, end=" ")   #her karakteri tamsayıya çevirip 2 ile çarp
```

Çıktı: 2 4 6 8 10

Ör: Parola girilirken Türkçe karakter uyarısı veren program:

```
tr_harfler = "şçöğüİıÖÜ"
parola = input("Parolanız: ")

for karakter in parola:
    if karakter in tr_harfler:
        print("Parolada Türkçe karakter kullanılamaz")
```

Ör: Amacımız ilk_metin'de olan, ama ikinci_metin'de olmayan öğeleri yazdırmak olsun.

```
ilk_metin = "Bilgisayar"
ikinci_metin = "Bilişim"

for s in ilk_metin: #ilk_metin'deki her öğeye s diyoruz
    if not s in ikinci_metin: #eğer bu öğeler ikinci_metinde yoksa
        print(s, end=" ") #bu olmayan olmayan s'leri yazdır.
```

Çıktı:
g s a y a r

Ör: Klavyeden girilen sayının faktöriyelini hesaplayan yazılımın for döngüsü ile yapımı.

```
sayi = int(input("Bir sayı giriniz: "))
faktoriyel = 1
for i in range(1,sayi+1):
  faktoriyel = faktoriyel * i
print(faktoriyel)
```

Ör: Klavyeden girilen metnin for döngüsü ile alt alta ekrana yazımı.

```
metin = input("Bir sayı giriniz: ")
for i in metin:
  print(i)
```

Ör: Girilen değere kadar toplama işlemi

```
a=int(input("Sayı Gir:"))
toplam=0
for i in range(1,a+1):
  toplam=toplam+i
print(toplam)
```

Ör: Çarpım tablosu örneği

```
sayi = int(input("Lütfen tablo ölçüsünü giriniz: "))
for satir in range(1, sayi + 1):
  for sutun in range(1, sayi + 1):
    deger = satir*sutun
    print("{0:4}".format(deger), end="")
  print()
```