

BİLGİSAYAR BİLİMİ DERSİ DERS NOTLARI

DEĞİŞKENLER

Değişkenler veri tutan ve saklayan birimlerdir

1- Değişkenlere Tek Tek değer atayabiliriz

Ad="İsmail"

Soyad="YALÇIN"

2-Değişkenlere Sıra ile değer atayabiliriz

Ad,soyad,sinif,yas="Ahmet","SOYLU","9K","15"

3-Birden fazla değişkene aynı değeri atayabiliriz

Elma=kiraz=erik=3

VERİ TIPLERİ

Şimdiye kadar 3 veri tipi gördük

VERİ TİPİ	AÇIKLAMA	ÖRNEK
String veri tipi	Metin , yazı ve karakter dizileri string veri tipine örnektir	"Ahmet" "Anibal" "54kljlışlı87" "5056464656"
Integer veri tipi	Tam sayılar bu veri tipine aittir	23 5 656556456
Float veri tipi	Noktalı sayılar bu veri tipine aittir	3.14 6.545465 0.235

type() fonksiyonu

Veri tiplerini öğrenmek için kullanılır

Örnek:

parola="kljals@uhd'1+?"

sayi1=65

pi=3.14

type(parola)----→ string değerini verir

type(sayı1)----→ int değerini verir

type(pi)-----→ float değerini verir

print() Fonksiyonu

Program çalıştığı anda içeriklerin gösterilmesini yazda yazardırmasını sağlar

Örnek

İsim="Ahmet"

print("hoş geldin ", isim)

yukarıdaki örnek " hoş geldin Ahmet " yazar

len() fonksiyonu

String bir verideki karakterlerin sayısını verir

ÖRNEK

İsim="Kadir"

karakterSayısı=len(isim)

print(karakterSayısı)

sonuç olarak "merhaba" string bir veridir ve karakter sayısı=7 dir

del() Fonksiyonu

Değişkenleri silmek için kullanılır

ad="Ahmet"

Soyad="DİNÇER"

del(ad)

del(soyad)

Yukarıdaki örnekte ad ve soyad değişkenleri, silinmiştir

Matematiksel operatörler

Matematiksel işlemleri yaparken kullanılan operatörler aşağıdaki tabloda gösterilmiştir

Toplama	+	Toplam=a+b
Çıkarma	-	Cikarma=a-b
Çarpma	*	Çarpma=a*b
Bölme	/	Bolme=a/b
Us Alama	**	usAl=a**b
Mod alma	%	modAl=a%b

Input() Fonksiyonu

Bu fonksiyon ile program çalıştığında veri girişi yapılmasını sağlar. Girilen veriler string veri tipindedir. Girilen veriler üzerinde sayısal işlem yapmak için tam sayıya =integer(int) veya noktalı sayı=float veri tipine dönüştürülmelidir

ÖRNEK1: Girilen isme göre işlem yapan kodlar aşağıdaki gibidir.

```
isim=input("İsminizi giriniz")
print("Hoş geldin ",isim)
```

Bu örnekte Program çalıştırıldığında " isminizi giriniz " mesajı çıkacak ve imleç yanıp sönecek . İsim girildikten sonra mesala "Ayşe") print fonksiyonu da isim değişkeninin tuttuğu veriyi "Hoş geldin Ayşe" gibi yazdıracaktır

ÖRNEK2 Girilen a ve b sayısını toplayan kodlar aşağıdaki gibidir. Bu örnekte girilen değerler üzerinde sayısal işlem yapılacağından int() fonksiyonu ile tam sayıya dönüşüm yapılmıştır.

```
A=int ( input ( "A sayısını giriniz:" ) )
B=int ( input ( "B sayısını giriniz:" ) )
Topla=A+B
print(" toplama işleminin sonucu =",Topla)
```

Algoritmalar

Program yazmaya başlamadan önce yapılacak işlemlerin adım adım yazılmasına «algoritma» denir

ÖRNEK: Dikdörtgenin çevresini bulan programın algoritmasını yazınız

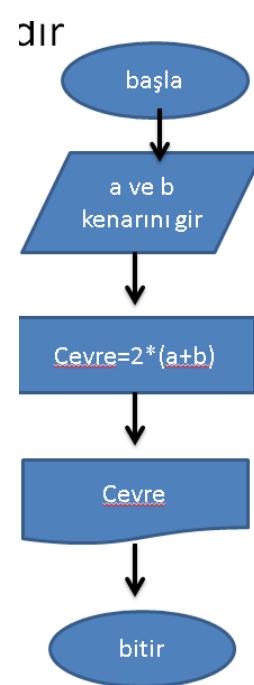
- 1.adım:başla
- 2.adım:a ve b kenarlarının değerini gir
- 3.adım:çevreyi hesapla>> $cevre=2*(a+b)$
- 4.adım:çevreyi göster-yazdır
- 5.adım:bitir

İş Akış Şeması

Akış şeması ise algoritmanın görsel gösterimidir. Akış şeması ile programın çalışma yapısı daha anlaşılır hale gelecektir.

Simgə	İşlev
	Başla/Bitir
	Giriş
	Atama/İşlem
	Denetim (Karar)
	Çıkış
	Döngü
	Akış Yönü
	Bağlaç
	Önceden Tanımlı İşlem/Fonksiyon

Şekil 1.10: Akış şeması sembollerı



KARŞILAŞTIRMA OPERATÖRLERİ

Python'da verileri karşılaştırmak için karşılaştırma operatörleri kullanılır.

Operatör	Anlamı	Örnek	Açıklama
==	Eşittir	x==y	X değeri y değerine eşitse Sonuç doğru(true) değilse Yanlıştır(False)
<	Küçüktür	x<y	X değeri y değerinden küçükse sonuç doğru (true) değilse Yanlıştır (false)
>	Büyükütür	x>y	X değeri y değerinden büyükse sonuç doğru (true) değilse Yanlıştır (false)
<=	Küçük veya eşittir	x<=y	X değeri y değerinden küçük veya eşitse sonuç doğru(true) değilse yanlışdır(false)
>=	Büyük veya eşittir	x>=y	X değeri y değerinden büyük veya eşitse sonuç doğru(true) değilse yanlışdır(false)
!=	Eşit değildir?	x!=y	X değeri y değeri ile eşit değilse Sonuç Doğru(true) eşitse yanlışdır(false)

ÖRNEK

Operatör	Anlamı	Örnek	sonuç
==	Eşittir	6==6	Doğru-True
<	Küçüktür	3<2	Yanlış-false
>	Büyükütür	8>5	Doğru-true
<=	Küçük veya eşittir	3<=3	Doğru-true
>=	Büyük veya eşittir	6>=7	Yanlış-False
!=	Eşit değil	4!=3	Doğru-True

ÖRNEK

Girilen a ve b sayılarını karşılaştıralım ve önermelerin doğru mu, yanlış mı olduğunu yazdıralım

```
karsilastirmaopereatorleri.py - C:\Users\ogr\Desktop...
File Edit Format Run Options Window Help
a=int(input("a sayısını giriniz:")) #a değişkeni
b=int(input("b sayısını giriniz:")) #b değişkeni

print("a küçüktür b-----",a<b)
print("a büyüğütür b-----",a>b)
print("a eşittir b-----",a==b)
print("a küçük veya eşittir b----",a<=b)
print("a büyük veya eşittr b----",a>=b)
print("a eşit değil b-----",a!=b)
```

```
*Python 3.6.2 Shell*
File Edit Shell Debug Options Window Help
Python 3.6.2 (v3.6.2:5fd33b5, Jul  8 201
7, 04:14:34) [MSC v.1900 32 bit (Intel)]
on win32
Type "copyright", "credits" or "license(
)" for more information.
>>>
==== RESTART: C:\Users\ogr\Desktop\ornek
ler\karsilastirmaopereatorleri.py ====
a sayısını giriniz:6
b sayısını giriniz:3
a küçüktür b----- False
a büyüğütür b----- True
a eşittir b----- False
a küçük veya eşittir b---- False
a büyük veya eşittr b---- True
a eşit değil b----- True
```

If,else Koşul ifadelerinin kullanılması

Türkçede EĞER anlamına gelen ifadesi, adından da anlaşılacağı üzere, koşula bağlı durumları kontrol etmek amacıyla kullanılır.

Koşul doğru(true) ise if altındaki satırlar çalışır

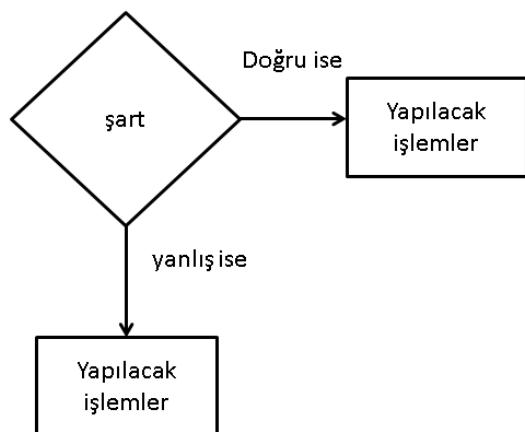
Koşul yanlış(false) ise else altındaki satırlar çalışır

if Koşul:

koşul doğru(true) ise yapılacak işlemler

else:

Koşul yanlış(false) ise yapılacak işlemler



Girilen sayının 5 den küçük olup olmama durumunu gösteren programın algoritma, akış şeması ve python kodlarını yazalım

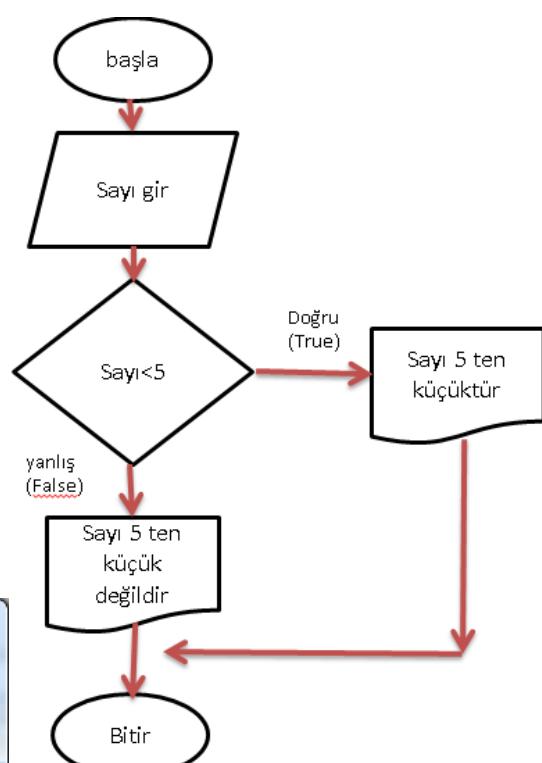
1.adım: Başla

2.adım: Sayı gir

3.adım: Eğer sayı 5 ten küçükse « girilen sayı 5ten küçüktür yaz»

değilse «girilen sayı 5 ten küçük değildir» yaz

4.adım:Bitir



```
*if5tenkucukmu.py - C:\Users\ogr\Desktop\ornekler\if5tenkucuk... X
File Edit Format Run Options Window Help
sayi=int(input("Lütfen sayı giriniz"))

if sayi<5:
    print("girilen sayı 5 ten küçüktür")
else:
    print("girilen sayı 5 ten küçük değildir")
```

A screenshot of a Python code editor window titled 'if5tenkucukmu.py'. The code defines a variable 'sayi' and prints a message based on whether it is less than 5 or not.

```
Python 3.6.2 (v3.6.2:5fd33b5, Jul  8 2017, 04:14:34) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\ogr\Desktop\ornekler\if5tenkucukmu.py =====
Lütfen sayı giriniz:9
girilen sayı 5 ten küçük değildir
>>>
```

A screenshot of a Python shell window showing the execution of the script. It prompts the user for a number, receives '9' as input, and prints the corresponding message 'girilen sayı 5 ten küçük değildir'.

İf, elif, else Koşul İfadelerinin Kullanımı

if ve else kullanımını örnekler ile gördük. Eğer veri üzerinde birden fazla şart kullanılcaksa elif fonksiyonu kullanılmalıdır.

if şart1:

şart 1 doğru ise yapılacaklar

elif şart2:

şart 2 doğru ise yapılacaklar

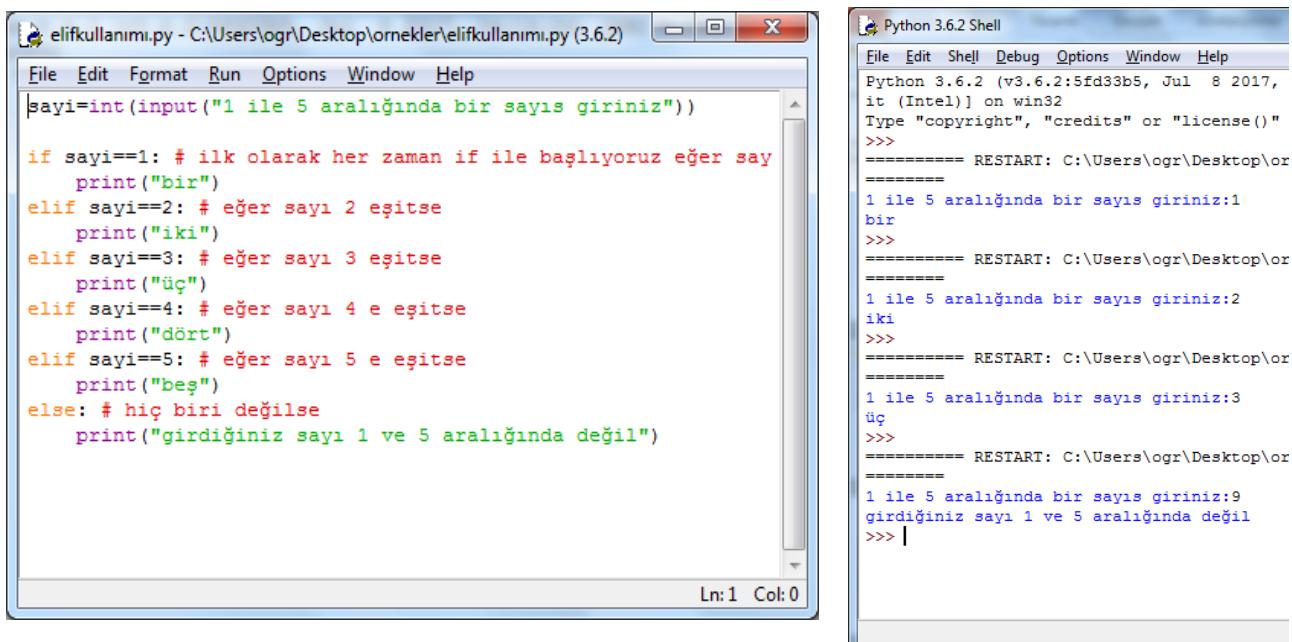
elif şart3:

şart 3 doğru ise yapılacaklar

else:

Hiçbir şart uygun değilse yapılacaklar

ÖRNEK: 1 den 5 e kadar girilen sayıların yazılışlarını gösteren programı yazınız



The screenshot shows a Windows desktop with two windows open. The left window is a Notepad file named 'elifikullanimi.py' located at 'C:\Users\ogr\Desktop\ornekler\elifikullanimi.py'. It contains the following Python code:

```
sayi=int(input("1 ile 5 aralığında bir sayı giriniz"))

if sayi==1: # ilk olarak her zaman if ile başlıyoruz eğer sayı
    print("bir")
elif sayi==2: # eğer sayı 2 eşitse
    print("iki")
elif sayi==3: # eğer sayı 3 eşitse
    print("üç")
elif sayi==4: # eğer sayı 4 e eşitse
    print("dört")
elif sayi==5: # eğer sayı 5 e eşitse
    print("beş")
else: # hiç biri değilse
    print("girdiğiniz sayı 1 ve 5 aralığında değil")
```

The right window is a Python 3.6.2 Shell window. It shows the command 'python elifikullanimi.py' being run and the resulting output:

```
Python 3.6.2 (v3.6.2:5fd33b5, Jul 8 2017, 16:18:59) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information
>>>
=====
1 ile 5 aralığında bir sayı giriniz:1
bir
>>>
=====
1 ile 5 aralığında bir sayı giriniz:2
iki
>>>
=====
1 ile 5 aralığında bir sayı giriniz:3
üç
>>>
=====
1 ile 5 aralığında bir sayı giriniz:4
dört
>>>
=====
1 ile 5 aralığında bir sayı giriniz:5
beş
>>>
=====
1 ile 5 aralığında bir sayı giriniz:9
girdiğiniz sayı 1 ve 5 aralığında değil
>>> |
```

KOŞULLU İFADELERDE AND, OR, NOT ve İN KULLANIMI

AND KULLANIMI

«And» ifadesi «ve» anlamına gelir. Koşullu ifadelerde birden fazla şartın doğru(true) olması durumunda işlem yapılması isteniyor ise and işlevi kullanılmalıdır.

if şart1 **and** şart2:

Tüm şartların doğru(true) olması durumunda yapılacak işlemler

else:

şartların birinin veya tümünün yanlış(false) olma durumunda yapılacak işlemler

ÖRNEK:Bir hesaba kullanıcı adı ve parola ile giriş yapılacaktır. Eğer Kullanıcı adı «admin» ve parola «abcdefg» ise sisteme giriş yapılacaktır. değilse, «kullanıcı adınız veya parolanız yanlış» çıktısını veren kodları yazınız.

```
ifandkullanimi.py - C:\Users\ogr\Desktop\ornekler\ifandkullanimi.py (3.4.4)
File Edit Format Run Options Window Help
kAdi=input("Kullanıcı adını giriniz:")
parola=input("Parolanızı giriniz:")

if kAdi=="admin" and parola=="abcdefg":
    print("Sisteme hoşgeldiniz")
else:
    print("Kullanıcı adı veya parolayı hatalı girdiniz")
```

«And» işlevinin mantıksal gösterimi

if şart1 and şart2:

Şart1	Şart2	Sonuç	İşletilecek yer
True	True	True	if altındaki satırlar
True	False	False	Else nin altındaki satırlar
False	True	False	Else nin altındaki satırlar
False	False	False	Else nin altındaki satırlar

OR(VEYA) KULLANIMI

«or» ifadesi «veya» anlamına gelir. Koşullu ifadelerde birden fazla şarttan sadece birinin doğru(true) olması durumunda işlem yapılması isteniyor ise or işlevi kullanılmalıdır.

```
if şart1 or şart2:  
    şartlardan biri doğru(true) olması durumunda yapılacak işlemler  
else:  
    şartların tümünün yanlış(false) olma durumunda yapılacak işlemler
```

ÖRNEK: Bir sisteme giriş yapılacaktır. Girilen sayı 15 veya 55 ise «sisteme kabul edildiniz» değilse «giriş yetkiniz yok» yazan programı oluşturunuz

```
File Edit Format Run Options Window Help  
sayi=int(input("Sisteme girmek için bir sayı giriniz"))  
  
if sayi==15 or sayi==55:  
    print("sisteme hoş geldiniz")  
else:  
    print("sitem için gerekli sayılarından birini bile doğru giremediiniz")
```

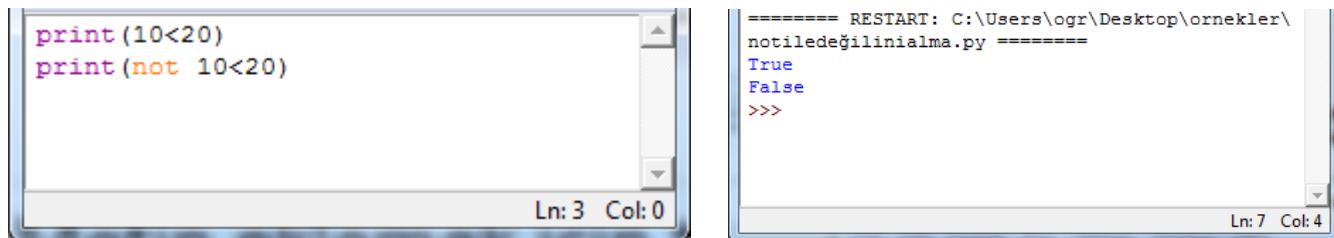
«or» işlevinin mantıksal gösterimi

if şart1 or şart2:

Şart1	Şart2	Sonuç	İşletilecek yer
True	True	True	if altındaki satırlar
True	False	True	if altındaki satırlar
False	True	true	if altındaki satırlar
False	False	false	Else nin altındaki satırlar

NOT(DEĞİL) KULLANIMI

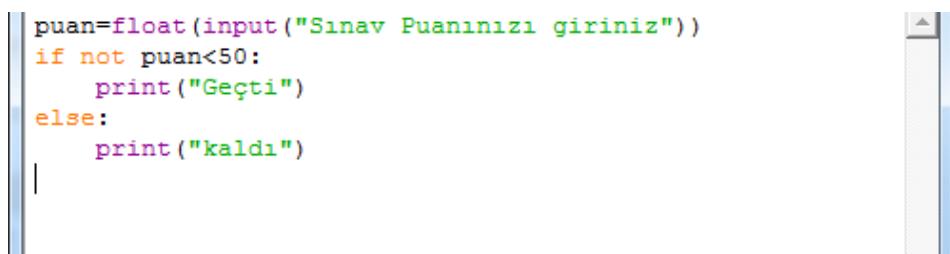
- Not Türkçede değil anlamına gelir. Bu işaret, **true** değeri **False** yaparken **False** bir durumu **true** yapar
- Aynı zamanda özellikle kullanıcı tarafından bir değişkene veri girilip girilmediğini denetlemek için kullanılabilir.



```
print(10<20)
print(not 10<20)
```

```
===== RESTART: C:\Users\ogr\Desktop\ornekler\notiledeğilinialma.py =====
True
False
>>>
```

ÖRNEK: Girilen puan 50 den küçük **değilse** geçti yazan programı oluşturalım



```
puan=float(input("Sınav Puanınızı giriniz"))
if not puan<50:
    print("Geçti")
else:
    print("kaldı")
```

not (değil) işaretinin mantıksal göstesimi

Şart	not şart
True	False
False	True

İN (İÇİNDE) AİTLİK İŞLECİNİN KULLANILMASI

- Aitlik işlemleri, bir karakter dizisi ya da sayının, herhangi bir verinin içinde bulunup bulunmadığını sorgulamamızı sağlayan işlemlerdir. Varsa **True** yoksa **False** sonucunu verir.



```
File Edit Format Run Options Window Help
print("h" in "ahmet")
print("s" in "ahmet")
```

```
=====
True
False
>>>
```

ÖRNEK: Girilen bir email adresi içerisinde «@» işaretini varsa «email adresiniz geçerlidir» yoksa «Email adresiniz hatalıdır» yazan Python kodlarını yazınız

```
email=input("Lütfen email adresinizi giriniz:")  
  
if "@" in email:  
    print("Email adresiniz geçerlidir")  
else:  
    print("email adresinizde @ işaretini olmalı")
```

ÖRNEK: Girilen harf sesli mi, değil mi? kontrolünü yapan programı yazalım

```
sesliHarfler="aüöüeiö"  
harf=input("Bir karakter giriniz")  
if harf in sesliHarfler:  
    print("Girdiğiniz karakter sesli harftir")  
else:  
    print("Giediğiniz karakter sesli harf değildir")  
  
#Açıklama  
#1. satırda SesliHarfler değişkenine sesli harflerimiz atandı  
#2.satırda Kullanıcının karakter girişi yapması isteniyor ve harf değişkenine atanıyor  
#3.satırta eğer alınan harf sesli harflerin içinde var ise ...  
#      Gidiğiniz karakter sesli harfti cevabı yazdırılıyor  
#değilse  
#      Girdiğiniz karakter seli harf değildir yazdırılıyor
```

FOR DÖNGÜSÜNÜN KULLANILMASI

Döngü nedir?

Döngüler, sıralı bir kod bloğunun istenilen sayıda tekrarlanmasıdır.

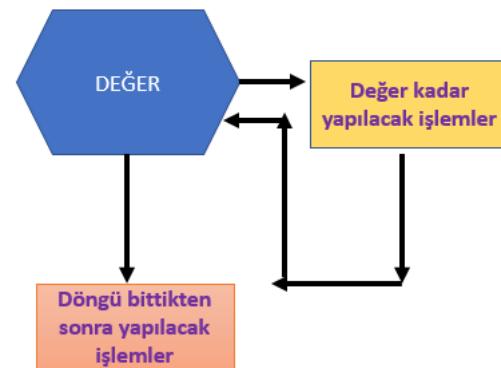
For döngüleri belirlitilen sayıdaki işlemlerin tekrarlanması için kullanılan döngülerdir. For döngüleri başlangıç ve bitiş değerleri arasında artım miktarına göre istenilen sayıda tekrar yapar.

For değişken **in** değer-aralık:

Yapılacak işlem

Yapılacak işlem

Yapılacak işlem



range () fonksiyonunun kullanılması

Range aralık demektir. Range fonksiyonunu kullanarak belirtlen aralıktaki sayıları for döngüsünde kullanabiliriz.

RANGE FONKSİYONUNUN ÜÇ KULLANIMI VARDIR.

range(değer)

range(başlangıç, bitiş)

range(başlangıç, bitiş, artış değeri)

Range Kullanımı	sonuç	Tanımlama
<code>for sayı in range(5): print(sayı)</code>	0,1,2,3,4	range(değer) 0 dan başlar, değerin 1 eksigine kadar sayı üretir. Bu döngü 5 kez- defa döner
<code>for i in range(1,11): print(i)</code>	1,2,3,4,5,6,7,8,9,10	range(başlangıç, bitiş) başlangıç değeri ile başlar, bitiş değerinin 1 eksigine kadar döngü devam eder. 1 den 10 'a kadar sayıları yazar. 11 dahil değildir.
<code>for sayı in range(5,20,3): print(sayı)</code>	5,8,11,14,17	range(başlangıç, bitiş, artış değeri) Kullanımı başlangıç değerinden ,bitiş değerinin 1 eksigine kadar, artış değerince döngü devam eder

FOR DÖNGÜSÜ İLE İLGİLİ ÖRNEKLER

10 defa "Anibal" yazan programı oluşturalım	<code>for i in range(10): print("anibal")</code>
girilen ismi , girilen sayı kadar (defa) yazdırın program	<code>isim=input(" isim giriniz") kez=int(input("kaç defa yazılsın")) for x in range(kez): print(isim)</code>
50 den 100'e kadar sayıları yazdırılın	<code>for sayı in range(50,101): print(sayı)</code>
0,100 e kadar 5 er artarak yazdırın programın kodlarını yazalım	<code>for sayı in range(0,100,5): print(sayı)</code>

YENİ ATAMA İŞLEÇLERİ

Atama İşleçleri	isim	Sayı=10	Sayı değişkeninin yeni değeri
=	Ata	Sayı=10	10
+=	Topla ve Ata	Sayı+=5	15
-=	Çıkar ve Ata	Sayı-=5	5
=	Çarp ve Ata	Sayı=3	30
/=	Böl ve Ata	Sayı/=2	5

```

sayi=10
print(sayi) # sayı değeri 10 oldu
sayi+=5
print(sayi) #sayı değeri 15 oldu
sayi+=20
print(sayi) #sayı değeri 35 oldu
sayi-=10
print(sayi) #sayı değeri 25 oldu
sayi*=2
print(sayi) #sayı değeri 50 oldu
sayi/=10
print(sayi) #sayı değeri 5 oldu

```

Örnek : For döngüsünü kullanarak 1 den 100'e kadar(100 dahil) sayıların toplamını bulalım

```

toplam=0 # döngüden gelen sayılar toplam değişkenin üzerine eklenecek

for sayı in range(1,101):
    toplam+=sayı # bu satırda sayılar toplam değişkeninin üzerine ekleniyor

print("Toplama işleminin sonucu:",toplam)

```

FOR DÖNGÜSÜNDE STRING VERİLERİN KULLANILMASI

For döngüsü ile string veri içerisindeki her bir karakteri alabiliriz

Aşağıdaki örnek ile belirtilen string ifade döngü içerisinde adındı ve her harf tek tek alınarak yazdırıldı

```

forString.py - C:/Users/ogr/Desktop/...
File Edit Format Run Options Window Help
for harf in "Bu gün hava çok güzel":
    print(harf)

Ln: 2 Col: 35

```

```

forString.py - C:/Users/ogr/Desktop/...
File Edit Format Run Options Window Help
cumle="Bu gün hava çok güzel"
for harf in cümle:
    print(harf)

Ln: 4 Col: 0

```

B
u

g
ü
n

h
a

ç
o
k

g
ü
z
e
l
>>>

Örnek: Girilen cümle içerisindeki sesli harflerin sayısını bulalım

```

sesliHarfler="aüöüeiö"
cumle=input("lütfen istediğiniz bir cümleyi giriniz")
sayac=0

for harf in cümle:
    if harf in sesliHarfler:
        sayac+=1
print("Girdiğiniz cümledeki sesli harf sayısı:",sayac)

#Açıklama
#1. satırda SesliHarfler değişkenine sesli harflerimiz atandı
#2. satırda Kullanıcının cümle girişi yapması isteniyor ve cümle değişkenine atanıyor
#3. satırda sesli harflerin sayısını tutmak için sayaç oluşturuluyor ve ilk değer olarak 0 atanıyor
#4. satırda döngü başlıyor. For döngüsü cümle içerisindeki karakter sayısı kadar devam edecektir
#her turda cümle içerisindeki karakterler sıra ile harf değişkenine alınacaktır
#5. satırta eğer alınan harf sesli harflerin içinde var ise ...
#6. Satırda sayaç değerini 1 arttırılacaktır
#değilse döngü devam edecek ve cümle içindeki sıradaki harf alınacaktır
#7. Döngü tamamlandıktan sonra print ile sayaçtaki değer yazdırılır

```

DÖNGÜYÜ BREAK KOMUTU İLE BİTİRME-KESME

Döngü devam ederken(tamamlanmadan) belli bir kriterde göre döngüyü kesebilir ve bitirebiliriz.

ÖRNEK 1) 1 den 10 a kadar devam eden döngüyü . 5.Sayıya gelince kesen programı oluşturalım.

```
for sayı in range(1,11):
    print(sayı)
    if sayı==5:
        break
    1
    2
    3
    4
    5
```

Döngü normalde 1,2,3,4,5,6,7,8,9,10 sayılarını yazacakken if sayı==5: koşu ifadesi ile döngü sayı değişkenine 5 değerini atadıktan sonra break komutu çalışıyor ve döngü sonlandırılıyor

Örnek2) 1000 defa dönen döngü içerisinde girilen kelime «çıkış» ise döngüyü sonlandıran kodları yazalım

```
for i in range(1000):
    kelime=input("Bir kelime giriniz")
    if kelime=="çıkış":
        break
```

Yandaki örnek ile 1000 defa kelime girişi yapılabilir. Girilen kelime «çıkış »ise döngü ye son verilecektir.

ÖRNEK 3)1000 defa işlem yapmamıza izin veren döngüyü klavyeden girilen seçenek ile istediğimiz zaman bitirmek

```
for i in range(1000): # for döngüsü 1000 defa çalışacak
    print("""
-----İşlemler-----
1-toplama
2-çıkarma
3-ÇIKIŞ
""")
    secim=int(input("yapmak istediğiniz işlemi seçiniz:"))
    if secim==1:
        sayı1=int(input("1.sayıyı giriniz"))
        sayı2=int(input("2.sayıyı giriniz"))
        toplam=sayı1+sayı2
        print("Toplama işleminin sonucu=",toplam)
    elif secim==2:
        sayı1=int(input("1.sayıyı giriniz"))
        sayı2=int(input("2.sayıyı giriniz"))
        fark=sayı1-sayı2
        print("Çıkarma işleminin sonucu=",fark)
    elif secim==3: #secim 3 ise döngü sona erecek
        break
    else:
        print("yaptığınız seçimde işlem yoktur")
```

Daha önce menü içerisindeki seçeneklerden biri seçildiğinde döngü 1000 defa işlem yapabiliyor. Yandaki programda for döngüsü ile bu hakkımızı 1000 defa yapabiliyoruz. Aynı zamanda break komutu ile döngüyü durdurabiliyoruz

FONKSİYONLAR

Fonksiyon Nedir?

Fonksiyonlar, belirli işlemleri gerçekleştiren kod bloğudur.

Python içerisinde yer alan çok fazla sayıda hazır fonksiyon vardır. print(), input(), len(), del() gibi fonksiyonlar bunlardan bir kaçıdır.

Bu dersimizde Python'da kendi fonksiyonlarımızı oluşturup kullanacağız.

Fonksiyon Tanımlama

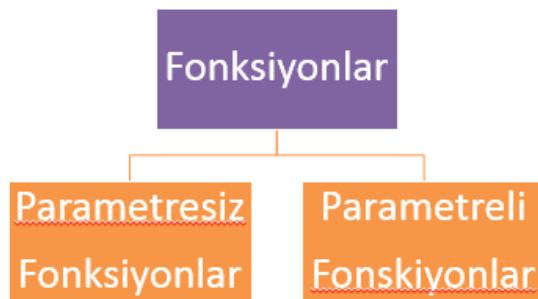
Fonksiyonları program içerisinde kullanabilmek için iki aşama vardır.

1-Fonksiyon tanımlanmalı

2-Fonksiyon çağrılmalı

Fonksiyonlar ikiye ayrılır

FONKSİYONLAR İKİYE AYRILIR

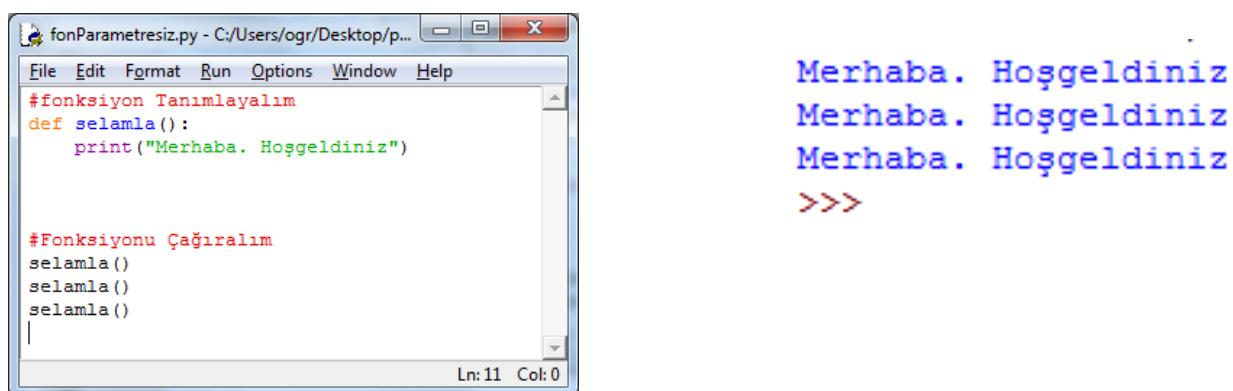


PARAMETRESİZ FONKSİYONLARIN KULLANIMI

Bu tür fonksiyonları tanımlarken fonksiyon ismi yanındaki parantez içerisinde parametre girilmez. Fonksiyon içeresinde yer alan komutları icra eder

```
def fonksiyon_ismi():
    Fonksiyona ait kodlar
    Fonksiyona ait kodlar
    Fonksiyona ait kodlar
```

Örnek1: Selamla isminde parametresiz fonksiyon tanımlandı. Selamla() fonksiyonu her çağrıldığında “Merhaba Hoşgeldiniz” Yazısı görülecektir

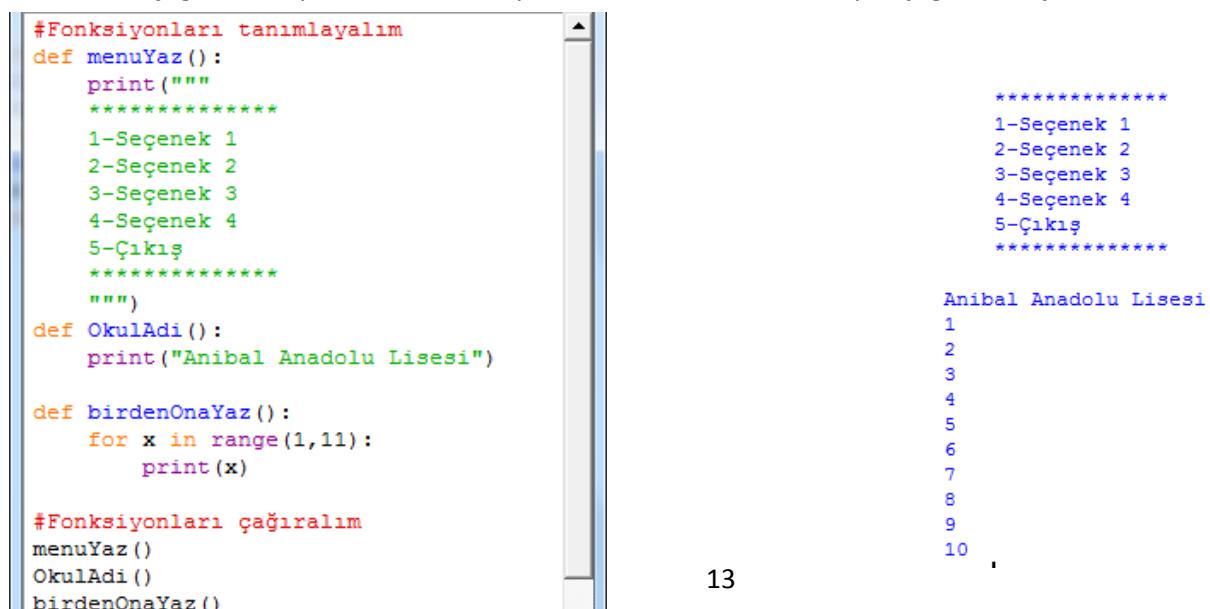


```
#fonksiyon Tanımlayalım
def selamla():
    print("Merhaba. Hoşgeldiniz")

#Fonksiyonu Çağıralım
selamla()
selamla()
selamla()
|
```

Merhaba. Hoşgeldiniz
Merhaba. Hoşgeldiniz
Merhaba. Hoşgeldiniz
>>>

ÖRNEK2: Aşağıdaki dosyada 3 adet fonksiyon tanımlandı ve har fonksiyon çağrılarak içindeki kodlar icra edildi



```
#Fonksiyonları tanımlayalım
def menuYaz():
    print("""
    ****
    1-Seçenek 1
    2-Seçenek 2
    3-Seçenek 3
    4-Seçenek 4
    5-Çıkış
    ****
    """)
def OkulAdi():
    print("Anibal Anadolu Lisesi")

def birdenOnaYaz():
    for x in range(1,11):
        print(x)

#Fonksiyonları çağıralım
menuYaz()
OkulAdi()
birdenOnaYaz()
```

1-Seçenek 1
2-Seçenek 2
3-Seçenek 3
4-Seçenek 4
5-Çıkış

Anibal Anadolu Lisesi
1
2
3
4
5
6
7
8
9
10

ÖRNEK3: İki sayıyı toplayan parametresiz fonksiyonu oluşturalım. Sayılar Fonksiyon içerisindeki input ile alınacaktır. topla() fonksiyonu 3 defa çağrılmıştır.

```
*FonTopla.py - C:/Users/ogr/Desktop/p...
File Edit Format Run Options Window Help
def topla():
    a=int(input("a sayısını giriniz:"))
    b=int(input("b sayısını giriniz:"))
    sonuc=a+b
    print(sonuc)
topla()
topla()
topla()

Ln: 6 Col: 0
```

a sayısını giriniz:65
b sayısını giriniz:32
97
a sayısını giriniz:23
b sayısını giriniz:21
44
a sayısını giriniz:54
b sayısını giriniz:12
66
>>> |

PARAMETRELİ FONKSİYONLARIN KULLANILMASI

Parametre alan fonksiyonlar tanımlanırken, Fonksiyon isminin yanındaki parantez içine , kullanılacak parametreler virgülle ayıralarak yazılmalıdır. Parametreler fonksiyon içerisindeki kodlarda kullanılır.

ÖRNEK1: Toplama işlemini parametre alarak yapan fonksiyonun tanımlanması ve çağrılmaması aşağıdaki gibidir

```
toplaFonksiyonu.py - C:/Us...
File Edit Format Run Options Window Help
#Fonksiyon tanımlamasını yapalım
def topla(a,b):
    sonuc=a+b
    print(a,"+",b,"=",sonuc)

#fonksiyonu çağıralım
topla(20,30)
topla(50,100)
topla(80,90)

Ln: 2 Col: 0
```

20 + 30 = 50
50 + 100 = 150
80 + 90 = 170
>>> |

ÖRNEK2: Selamla adındaki fonksiyona gönderilen isim parametresine göre sonuç üretmenktedir.

```
selamla.py - C:/Us...
File Edit Format Run Options Window
def selamla(isim):
    print("Hoş geldin ",isim)

selamla("Ahmet")
selamla("Mehmet")
selamla("Fatma")
|
```

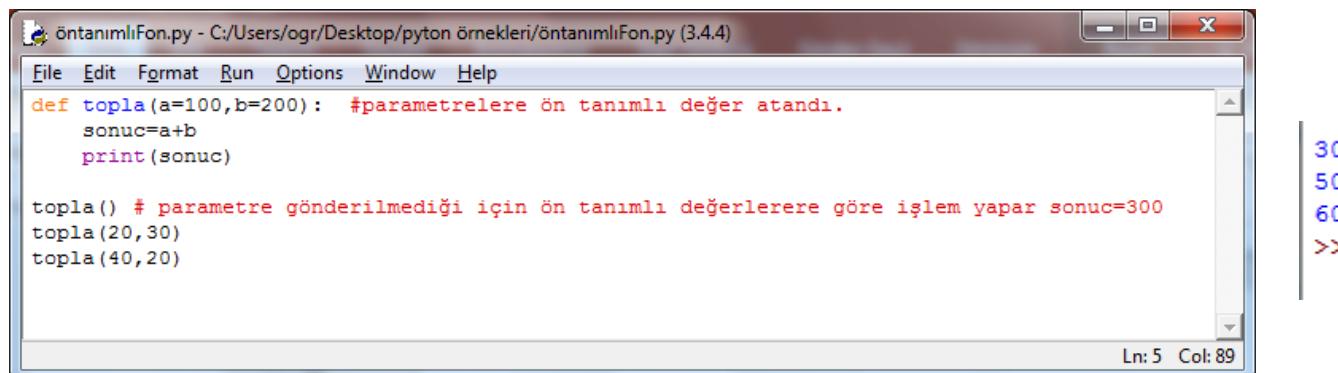
Hoş geldin Ahmet
Hoş geldin Mehmet
Hoş geldin Fatma
>>> |

ÖRNEK3: Belirtilen sayılar arasındaki sayıları toplayan parametreli fonksiyon aşağıdaki gibidir

```
def arasiTopla(x,y):  
    toplam=0;  
    for sayı in range(x,y+1):  
        toplam+=sayı  
    print(x,"ile", y, "arasındaki sayıların toplamı=",toplam)  
  
arasiTopla(1,10)  
arasiTopla(1,5)  
|  
  
1 ile 10  arasındaki sayıların toplamı= 55  
1 ile 5  arasındaki sayıların toplamı= 15  
...
```

FONKSİYON OLUŞTURURKEN PARAMETREYE VARSAYILAN DEĞER ATAMA

Fonksiyon içerisindeki parametrelere ön tanımlı değer atanabilir. Aşağıdaki Örneği İnceleyiniz



```
öntanimliFon.py - C:/Users/ogr/Desktop/python örnekleri/öntanimliFon.py (3.4.4)  
File Edit Format Run Options Window Help  
def topla(a=100,b=200): #parametrelere ön tanımlı değer atandı.  
    sonuc=a+b  
    print(sonuc)  
  
topla() # parametre gönderilmediği için ön tanımlı değerlerere göre işlem yapar sonuc=300  
topla(20,30)  
topla(40,20)  
  
300  
50  
60  
>>>  
Ln: 5 Col: 89
```

RETURN İLE FONKSİYONDAN DEĞER DÖNDÜRME

Fonksiyon içerisinde işlenip, üretilen değerler, programın herhangi bir yerinde kullanılmak istenirse «return» komutu ile gönderilmelidir.

```
def kok(sayı):  
    sonuc=sayı**0.5  
    return sonuc  
  
def usal(sayı,us):  
    sonuc=sayı**us  
    return sonuc  
  
#denklem için gerekli kodları yazalım  
  
c=kok(usal(3,2)+usal(4,2))  
  
print("bulunan değer=",c)
```

Yandaki örnekte fonksiyon içerisinde üretilen değerler return ile fonksiyonun çağrııldığı yere geri döndürülmüştür

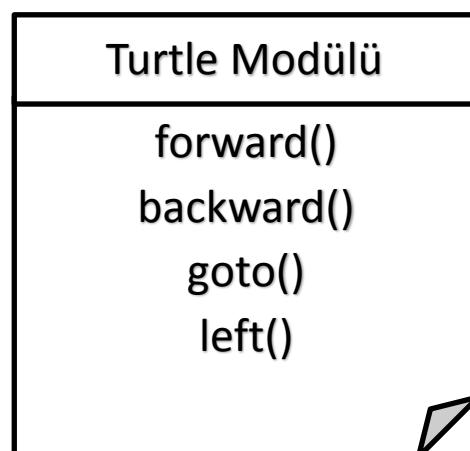
MODÜLLER

Modüllerin, bazı işlevleri kolaylıkla yerine getirmemizi sağlayan birtakım fonksiyonları ve nitelikleri içinde barındıran araçlar olduğunu söyleyebiliriz. Modülleri python dosyasına import ederek içerisindeki fonksiyonları kullanabiliriz.

İki çeşit modül vardır.

1-Hazır Modüller

2-Kendi tanımladığımız modüller

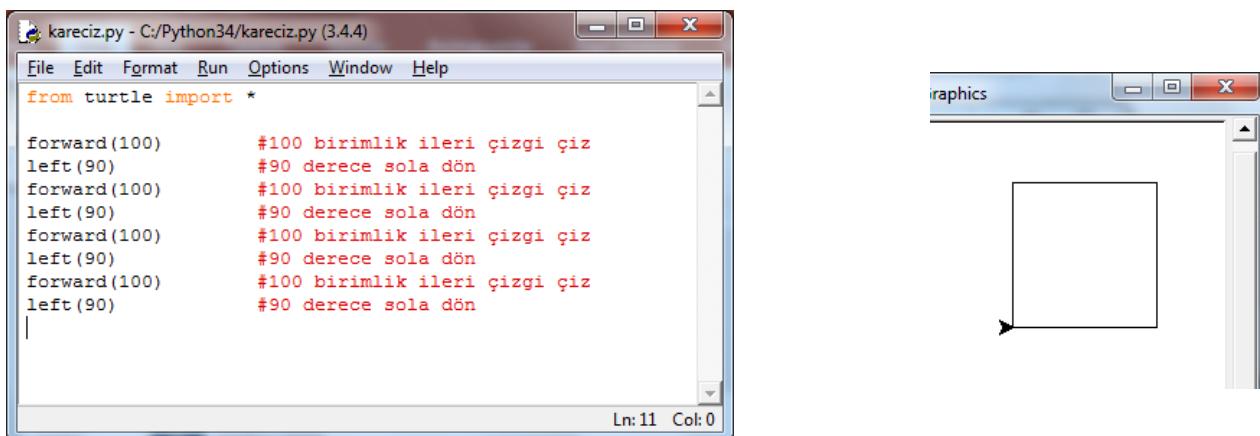


TURTLE MODÜLÜ

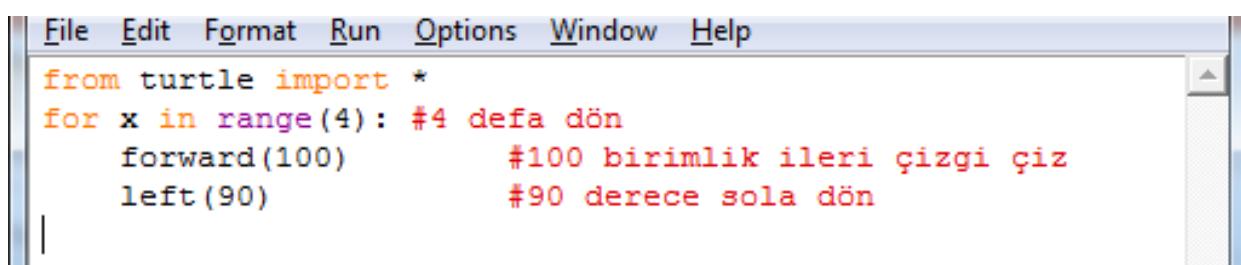
Turtle modülünü Python sayfalarımıza import edebilir ve içerisinde yer alan çizim fonksiyonları ile çizimler yapabiliriz. Bu modülü import etmek için en önce aşağıdaki kod satırı yazılmalıdır.

```
from turtle import *
```

ÖRNEK1:Turtle modülünü kullanarak 100 birimlik kare çizelim



Örnek2 : Turtle modülünü kullanarak kareyi for döngüsü ile çizen program



ÖRNEK3: Turtle modülünü kullanarak kareyi fonksiyon oluşturarak ile çizen program(aşağıdaki program aynı yere 3 defa kare çizer. Fonksiyon oluşturulmuş ve 3 defa çağırılmıştır)

```
*kareciz.py - C:/Python34/kareciz.py (3.4.4)*
File Edit Format Run Options Window Help
from turtle import *      #turtle modülü import et
def kareciz():           #kareciz parametresiz fonksiyonunu tanımla
    for x in range(4):    #4 defa dön
        forward(100)      #100 birimlik ileri çizgi çiz
        left(90)           #90 derece sola dön

kareciz()                #kareciz fonksiyonunu çağır
kareciz()                #kareciz fonksiyonunu çağır
kareciz()                #kareciz fonksiyonunu çağır
```

ÖRNEK4: Girilen kenar sayısına göre 100 ve X birimlik ilgili şekli çizen programı yazalım

```
*sekilciz.py - C:/Python34/sekilciz.py (3.4.4)*
File Edit Format Run Options Window Help
from turtle import *      #turtle modülü import et
def sekilciz(kenarsayisi,birim): #sekilciz parametreli fonksiyonunu ta
    disAci=360/kenarsayisi #dönme açısını hesapla
    for x in range(kenarsayisi): #kenar sayısı kadar dön
        forward(birim)          #100birim kadar ileri çizgi çiz
        right(disAci)           #hesaplanan derece kadar sağa dön

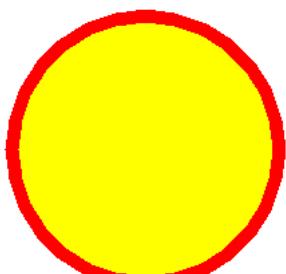
sekilciz(3,50)  #50 birimlik Üçgen çiz
clear()         #Pencereyi temizler
sekilciz(4,100) #100 birimlik Kare çiz
clear()         #Pencereyi temizler
sekilciz(5,150) #150 birimlik beşgen çiz
clear()         #Pencereyi temizler
sekilciz(6,100) #100 birimlik 6 gen çizer
clear()         #Pencereyi temizler
sekilciz(7,100) #100 birimlik 7 gen çizer
clear()         #Pencereyi temizler
```

Ln: 16 Col: 30

```
*dairerenklendir.py - C:/Python34/dairerenklendir.py (3.4.4)*
File Edit Format Run Options Window Help
from turtle import *      #turtle modülünü import et

pensize(10)               #kalem kalınlığını 10 birim yap
color("red","yellow")     #color(cizgirenk,dolgurenk) renk tanımla
begin_fill()              #renklendirme başla
circle(100)                #100 birimlik daire çiz

end_fill()                #renklendirme bitir
```



```

merdiven.py - C:\Python34\merdiven.py (3.4.4)
File Edit Format Run Options Window Help
from turtle import *      #turtle modülü import et

def merdiven(basamaksayisi,uzunluk):  #merdiven isminde parametreli fonksiyon tanımla
    speed(1)      #turtle hızını 1 yap( 1 yavaş-10 hızlı)
    penup()       #kalemi kaldır
    goto(-300,-300) #penceredeki -300-, -300 konumuna git
    pendown()     #kalemi bastır
    for x in range(basamaksayisi): #basamak sayısı kadar tekrarla
        forward(uzunluk) #uzunluk kadar ileri çizgi çiz
        left(90)          #90 derece sola dön
        forward(uzunluk) # uzunluk kadar ileri çizgi çiz
        right(90) #90 derece sağa dön

merdiven(5,100) # merdiven fonksiyonunu çağır basamak için 5 uzunluk için 100 değerini gönder
clear() #ekrani temizle
merdiven(10,50) # merdiven fonksiyonunu çağır basamak için 10 uzunluk için 50 değerini gönder
clear() #ekrani temizle
merdiven(10,20) # merdiven fonksiyonunu çağır basamak için 30 uzunluk için 20 değerini gönder

Ln: 19 Col: 11

```

TUTLE MODÜLÜNDE ÖĞRENDİĞİMİZ FONKSİYONLAR

Fonksiyon adı-kullanımı	Açıklaması
forward(100)	100 birim ileri çizgi çiz
backward(50)	50 birim geri çizgi çiz
left(60)	60 derece sola dön
right(90)	90 derece sağa dön
pensize(10)	Kalem ucu kalınlığını 10 birim yap
color("red","yellow")	Çizgi rengini kırmızı, dolgu rengini sarı yap
begin_fill()	Boyamayı başlat
end_fill()	Boyamayı bitir
circle(50)	50 birimlik daire çiz
speed(1)	turtle hızını ayarla(1 yavaş-10 hızlı)
penup()	kalemi kaldır
pendown()	kalemi bastır
goto(100,200)	pencere de x =100 ,y =200 koordinatına git
clear()	ekrani temizle

<https://docs.python.org/3.3/library/turtle.html?highlight=turtle> Bu link ile turtle modülü hakkında daha fazla bilgiye ulaşabilirsiniz.

TİME MODÜLÜ

Zamanla ilgili bilgi ve işlemlerin yer aldığı modül, “time” modülüdür. Time modülünü import etmek için aşağıdaki kod satırı yazılmalıdır

```
from time import *
```

clock() fonksiyonu

clock() fonksiyonu ile programın belli bölümlerinin çalışma süresini saniye cinsinden ölçebiliriz .iki clock() fonksiyonu arasında geçen süre sn cinsinden hesaplanmaktadır.

```
*gencenzaman.py - C:\Python34\gencenzaman.py (3.4.4)*
File Edit Format Run Options Window Help
from time import * #time modülünü import et

print("""
Ceza ve Güvenlik Tedbirlerinin İnfazı Hakkında Kanun gereğince, çocuk eğ
evlerinde ve çocuk kapalı ceza infaz kurumlarında kalmakta olan hükümlü
tutukluların örgün ve yaygın eğitimlerine devam etmelerine ait usul ve e
""")

baslangic=clock() #başlangıç süresi
yazi=input("Yukarıdaki yazılıyı yazınız. Süreniz başlamıştır=")
bitis=clock() #bitiş süresi
sure=bitis-baslangic

print("yazılıyı ",sure,"saniyede yazdınız")

Ln: 15 Col: 0
```

sleep() Fonksiyonu

Sleep() fonksiyonu ise programın çalışması sırasında belirtilen sure kadar durmasını sağlar.

Örneğin geriye sayımda her sayıdan sonra 1 saniye beklemek için aşağıda görülen kod kullanılır.

```
*gerisaytime.py - C:/Python34/gerisaytime.py (3.4.4)*
File Edit Format Run Options Window Help
from time import* #time modülünü import et
for x in range(10,0,-1): # for döngüsünü başlat
    print(x) #x değerini yazdır
    sleep(1) #1 saniye bekle(uyu)

Ln: 5 Col: 0
10
9
8
7
6
5
4
3
2
1
>>> |
```

strftime() fonksiyonu

Bu fonksiyon içerisine yazılan özel parametrelere göre zaman hakkında bilgi vermektedir.

```
*tarihgoterimleri.py - C:/Python34/tarihgoterimleri.py (3.4.4)*
File Edit Format Run Options Window Help
from time import *
print(strftime("%c")) #tarih ve saati gösterir
print(strftime("%x")) #sadece tarihi gösterir
print(strftime("%j")) #yılın kaçinci gününde olduğumuzu gösterir
print(strftime("%U")) #yılın kaçinci haftasında olduğumuzu gösterir

Ln: 2 Col: 20
04/27/18 14:33:40
04/27/18
117
16
>>>
```

RANDOM MODÜLÜ

Bu modülü import ederek bilgisayarın rastgele sayılar üretmesini sağlayabiliriz. Random modülünü import etmek için aşağıdaki satır yazılmalıdır.

```
from random import *
```

The screenshot shows a Python IDLE window with the following code:

```
*randomFonksiyonları.py - C:/Users/ogr/Desktop/python örnekleri/randomFonksiyonları.py (3.4.4)*
File Edit Format Run Options Window Help
from random import*
print(random()) #0 ile 1 arasında ondalıklı sayı üretir
print(randint(1,10)) #1 ile 10 arasında ( 1 ve 10 dahil) arasında tam sayı üretir
print(uniform(0.5,2.5)) #belirtilen ondalıklı sayılar arasında sayı üretir
```

The output pane shows the results of the print statements:

```
0.5865020954771984
3
1.173638135720657
>>>
```

Ln: 4 Col: 4

ÖRNEK1: Rastgele 1 ile 100 arasında 10 adet tam sayı üreten kodları yazalım.

The screenshot shows a Python IDLE window with the following code:

```
rastgele10sayı.py - C:/Python34/rastgele10sayı....
File Edit Format Run Options Window Help
from random import*
for i in range(10):
    print(randint(1,100))|
```

The output pane shows 10 random integers between 1 and 100:

```
52
90
63
56
40
90
97
93
13
24
>>>
```

Ln: 4 Col: 25

ÖRNEK2: 1-10 sayıları arasında bilgisayarın üreteceği sayıyı bulmaya çalışalım. For döngüsü ile 3 hakkımız olduğunu unutmayalım.

The screenshot shows a Python IDLE window with the following code:

```
sayı tahminrandom.py - C:\Python34\sayı tahminrandom.py (3.4.4)
File Edit Format Run Options Window Help
from random import*
sayı=randint(1,10)
for i in range(3):
    tahmin=int(input("1-10 arası bir sayı tahmin ediniz:"))
    if tahmin==sayı:
        print("tebrikler bildiniz")
        break
    else:
        print("bilemediniz")
print("bilgisayarın ürettiği sayı=",sayı)
```

The output pane shows the interaction with the user:

```
1-10 arası bir sayı tahmin ediniz:6
bilemediniz
1-10 arası bir sayı tahmin ediniz:3
tebrikler bildiniz
bilgisayarın ürettiği sayı= 3
>>> |
```

Ln: 1 Col: 0

KENDİ MODÜLÜMÜZÜ OLUŞTURALIM

Kendi oluşturduğumuz fonksiyonları modül olarak kaydedebilir ve sayfalarımıza import edebiliriz.

- Bunun için modül içerisinde sadece fonksiyon tanımlaması yapılmalıdır.
- Modül Python>lib klasörüne kaydedilerek tüm çalışmalara import edilebilir.

UGULAMA:

"dortislem" adındaki modülümüz ve içerisinde yer alan fonksiyonlar oluşturuldu ve Bu dosya "lib" klasörüne kaydedildi. Aşağıda "dortislem" adındaki modül dosyamızı görüyorsunuz.

```
dortislem.py - C:\Python34\Lib\dortislem.py (3.4.4)
File Edit Format Run Options Window Help
#Modül içerisinde sadece fonksiyon tanımlaması yapılır

def topla(a,b):
    sonuc=a+b
    print("toplama işleminin sonucu=",sonuc)

def fark(a,b):
    sonuc=a-b
    print("Çıkarma işleminin sonucu=",sonuc)

def carp(a,b):
    sonuc=a*b
    print("çarpma işleminin sonucu=",sonuc)

def bol(a,b):
    sonuc=a/b
    print("Çarpma işleminin sonucu=",sonuc)
```

Daha sonra modül import edilerek içerisindeki fonksiyonlar çağrılarak kullanılır. Aşağıda bunun ile ilgili kodları görebilirsiniz

```
dortislemcagir.py - C:/Python34/dortislem...
File Edit Format Run Options Window Help
from dortislem import *

topla(20,30)
fark(10,2)
carp(5,8)
bol(20,10)
|
```

```
toplama işleminin sonucu= 50
Çıkarma işleminin sonucu= 8
çarpma işleminin sonucu= 40
Çarpma işleminin sonucu= 2.0
>>>
```