# PYTHON


**İÇERİK**

- ✓ **Python'da Koşullu Durumları**
- ✓ **While Döngüsü**
- ✓ **For Döngüsü**

KAZANIMLAR

- ✓ İf deyimini ve hangi alanlarda nasıl kullanıldığını öğrenir.
- ✓ Else ve elif deyimlerini ve kullanım alanlarını öğrenir.
- ✓ While döngüsünü ve söz dizimini kavrar.
- ✓ For döngüsünü, yapısını ve kullanım alanlarını öğrenir.
- ✓ While döngüsü ile for döngüsü arasındaki farkı kavrar.


**Python'da Koşullu Durumlar**

Python'da veri tiplerini inceledik fakat şu ana kadar yaptığımız bütün uygulamalarda bir olasılık üzerinden gittik. Bir durum söz konusu olduğunda ne yapmamız gerektiğini söylemedik. Örnek durum ver

Python programlama dilinde koşullu durumları belirtmek için 3 adet deyim bulunmaktadır. Bunlar:

- ✓ if
- ✓ else
- ✓ elif

Bu deyimlere geçmeden önce dikkat etmemiz gereken mantıksal koşul ifadeleri var. Bunlar:

- Eşittir: `a == b`
- Eşit Değil: `a! = B`
- Küçüktür: `a <b`
- Küçüktür veya eşittir: `a <= b`
- Şundan büyük: `a> b`
- Büyük veya eşit: `a> = b`

Bu koşullar, en yaygın olarak "if ifadeleri" ve döngülerde olmak üzere çeşitli şekillerde kullanılabilir.

if deyimi İngilizce bir kelime olan if Türkçede eğer anlamındadır. Anlamından da çıkartabileceğimiz gibi koşul (şart) ifadesi barındırmaktadır. if deyimini kullanım dizilimi

```
if koşul :
```

Şimdi if deyimini kullanarak kullanıcıya yaşını soran ve yaşına göre dönüt veren koşumuzu yazalım. Öncelikle kullanıcının yaşını soralım ve yaşı eğer 18 den büyük ise ekranda giriş yapabilirsiniz yazıdrsın:

```
yaş= int(input("yaşınızı girin: "))
if yaş > 18:
 print("Giriş yapabilirsin")
```

Eğer 18 den küçük olursa ne yapabiliriz burada else deyimimiz işe koşuluyor:

```
yaş= int(input("yaşınızı girin: "))
if yaş > 18:
 print("Giriş yapabilirsin")
else:
   print("Giriş yapamazsınız")
```

Yaş uygunluğunu kontrol ettikten sonra şimdi siteme kullanıcı adı ve şifre ile giriş yapmasını sağlayan bir kod yazalım. Bunun için öncelikle

```
print("Robotik kodlama oyununa hoşgelidiniz")
kullanıcıadı=input("Kullanıcı adı girniz: ")
şifre=input("şifre giriniz:")

if kullanıcıadı=="bahar" and şifre=="12345":
   print("hesaba giriş yapıldı")
else:
   print("kullanıcı adı veya şifre hatalı")
```

Bu kısımda bağlaç olarak and ifadesini kullandık birden fazla koşulu gerçekleştirmesini istiyorsak programımızın and ifadesini kullanırız.

Eğer iki koşuldan birinin sağlanması yeterli ise bunun için or ifadesini kullanırız. Şifre uygulamasında bulunan or uygulamasını siz deneyiz..

Şimdi kullanıcıdan iki farklı değer alalım ve bunların birbirlerine göre durumlarını inceleyim.

```
ilksayı= int(input("ilk sayıyı giriniz"))
ikincisayı= int(input("ikinci sayıyı giriniz"))

if ilksayı > ikincisayı:
 print("ilksayı büyüktür ikincisayı")

else:
   print("ilksayı büyük değildir ikincisayı")
```

Burada kullanıcıdan iki sayı alarak bunların büyüklük veya küçüklük durumlarını inceledik.


Bir durumun sağlanması birden fazla koşula bağlıysa elif deyimi kullanılır.

```
ilksayı= int(input("ilk sayıyı giriniz:"))
ikincisayı= int(input("ikinci sayıyı giriniz:"))

if ilksayı > ikincisayı:
  print("ilksayı büyüktür ikincisayı")

elif ilksayı==ikincisayı:
    print("ilksayı eşittir ikincisayı")

else:
     print("ilksayı büyük değildir ikincisayı")
```

Not: Koşul bloğu elif deyimi ile başlayamaz, koşullar her zaman if deyimi ile başlamalıdır.

## Sıra Sizde

Eğer ilk girilen sayı ikinci girilen sayıdan büyük ise ekrana “Merhaba Dünya” yazması için boşluğu uygun şekilde doldurun.

```
ilksayı=input()
ikincisayı=input()

      ilksayı      ikincisayı
  print ("Merhaba Dünya")
```


## Dögüler

Şimdiye kadar öğrendiklerimiz sayesinde Python'la ufak tefek programlar yazabilecek düzeye geldik. Bu bölümde, programlarımızın sürekli olarak çalışmasını nasıl sağlayabileceğimizi, yani programlarımızı bir döngü içine nasıl sokabileceğimizi öğreneceğiz.

Python'da programlarımızı tekrar tekrar çalıştırabilmek için döngü adı verilen bazı ifadelerden yararlanacağız. Python'da iki tane döngü bulunur: while ve for. while döngüsü ile başlayalım

**while Döngüsü**

İngilizce bir kelime olan *while*, Türkçede '... iken, ... olduğu sürece' gibi anlamlara gelir. Python'da while bir döngüdür. Bir önceki bölümde söylediğimiz gibi, döngüler sayesinde programlarımızın sürekli olarak çalışmasını sağlayabiliriz.

Bu bölümde Python'da while döngüsünün ne olduğunu ve ne işe yaradığını anlamaya çalışacağız. Öncelikle while döngüsünün temellerini kavrayarak işe başlayalım.

Basit bir while döngüsü örneği:

```
i = 1

while i < 6:

 print(i)

 i += 1
```

while döngüsü ile, bir koşul doğru olduğu sürece bir dizi ifadeyi çalıştırabiliriz.

Bu örnekte i 6'dan küçük olduğu sürece a'yı yazdır. Yani koşul sağlandığı sürece döngü devam eder. Bu örnekten de anlaşılacağı gibi while döngüsünün söz dizimi

```
koşul

while koşul:

Döngü içi
```


While döngüüsnü daha iyi anlamak adına bir örnek daha yapalım:

Kullanıcıdan iki sayı alalım ve bu sayılardan bir tanesi için 50'den küçük olma koşulu ve diğeri içinde 100'den küçük olma koşulu sağlayalım.

```
x = int(input("50'den küçük bir sayı giriniz: "))

y = int(input("100'den küçük bir sayı giriniz: "))

while x < 50 and y < 100:

  x = x + 1

  y = y + 1

  print(x, y)
```

yani bir while döngüsünde takip edilmesi gereken adımlar şunlardır:

1)Koşulu kontrol et.

2)Bloktaki kodu çalıştır.

3)Tekrar ettir.

## Sıra Sizde

Kod üzerinde bulunan eksik kısımları tamamlayın . 10'dan küçük olan sayıları ekrana yazdırın

```
i = 1

      i <10

   print (i)

   i + = 1
```

Cevap: while i<10:


**for Döngüsü**

for döngüsü tıpkı while döngüsü gibi bir döngüdür. Yani tıpkı while döngüsünde olduğu gibi, programlarımızın birden fazla sayıda çalışmasını sağlar. Ancak for döngüsü while döngüsüne göre biraz daha yeteneklidir. while döngüsü ile yapamayacağınız veya yaparken çok zorlanacağınız şeyleri for döngüsü yardımıyla çok kolay bir şekilde halledebilirsiniz.

Fakat for döngüsü while döngüsünün bir alternatifi değildir. Evet, while ile yapabildiğiniz bir işlemi for ile de yapabilirsiniz çoğu zaman, ama bu döngülerin, belli koşullar için tek seçenek olduğu durumlar da vardır. Fakat bu iki döngünün çalışma mantığı birbirinden farklıdır.

Şimdi for döngüsünü inceleyelim ve örnekler ile pekiştirelim. Oluşturduğumuz bir liste veya demet içerisindeki bütün elemanları for döngüsü ile yazdırabiliriz:

```
meyveler = ["elma", "muz", "kiraz","armut", "çilek"]

for meyve in meyveler:

  print(meyve)
```

Bu örnekte listeler isminde bir liste oluşturduk daha sonra listenin her bir elemanını yazdırmak için for döngüsü kullandık. Burada dikkat edilmesi gereken nokta döngüyü oluştururken

```
for değişken adı  in değişken:

   yapılacak işlemler
```

Bizim döngümüze baktığımızda meyveler listesi içerisindeki her bir ögeyi meyve olarak adlandırdık ve meyve olarak adlandırdığımız her bir ögeyi ekrana yazdırdık.


Diziyi break ifadesiniz kullanarak durdurabiliriz. Örneğin

```
meyveler = ["elma", "muz", "kiraz","armut", "çilek"]

for meyve in meyveler:

print(meyve)

 if meyve=="kiraz":

   break
```

Bu örnekte meyve adında oluşturduğumuz değişkenin içerisine eğer kiraz gelirse döngüyü durdur geri kalanı yazdırma dedik.

Tıpkı bunun gibi continue ifadesi ile döngü içerisinde yazdırmak istemediğimiz ögeyi atlayıp devam edebiliriz

```
meyveler = ["elma", "muz", "kiraz","armut", "çilek"]

for meyve in meyveler:

 if meyve=="kiraz":

   continue

 print(meyve)
```

Bu örnekte olduğu gibi continue ifadesi ile liste içerisindeki kiraz ögesini yazdırmadan döngüyü çalıştırmaya devam ettik.

Python'da iç içe döngüler kullanılabileceğiniz biliyor musunuz? Bu durumu bir örnek ile inceleyelim.


```
sıfatlar = ["kırmızı", "büyük", "lezzetli"]

meyveler = ["elma", "muz", "kiraz"]

for sıfat in sıfatlar:

  for meyve in meyveler:

    print(sıfat, meyve)
```

Bu örnekte de gördüğümüz gibi oluşturduğumuz iki tane döngü var. Bunlardan biri iç biri dış döngüdür. Dış döngü(sıfatlar) iç döngünün(meyveler) tamamını çalıştırır. Daha sonra tekrardan dış döngü tamamlanıncaya kadar devam eder.

Döngülerle ilgili örneklerimiz şimdilik bu kadar ilerleyen zamanlarda programlarımız yazarken döngüleri sık sık kullanacağız.

## Neler Öğrendik

Bu bölümde, if, else ve elif deyimlerinin nasıl kullanılacağını öğrendik. Tekrarlayan görevleri gerçekleştirmek için döngüler kullandık. Python'a tekrar etmesini istediğimiz şeyi söyledik görevleri, döngülerin içine koyduğumuz kod bloklarının içine yazdık. İki tür döngü kullandık: while ve for bu iki döngü birbirine benzer ancak farklı şekillerde kullanılabilir. Break ifadesi ile döngüyü nasıl durdurmamız gerektiğini öğrendik. İç içe döngü kullanmayı öğrendik.


python