

EĞİTİM PLANI

Hafta	Konu Başlıkları
1. Hafta: Python'a Giriş	<ul style="list-style-type: none">- Python Nedir? Kullanım Alanları- Python Kurulumu ve IDE'ler- Python Çalışma Mantığı: İfadeler, Yorumlayıcı- <code>print()</code> ve <code>input()</code> Fonksiyonları- Veri Türleri (<code>int</code>, <code>float</code>, <code>string</code>, <code>bool</code>)- Yorum Satırları- Basit Matematiksel İşlemler- Casting (Tür Dönüşümü)- Koşullu İfadeler (<code>if</code> , <code>else</code> , <code>elif</code>)
2. Hafta: Döngüler ve Veri Yapıları	<ul style="list-style-type: none">- Döngüler: <code>for</code> , <code>while</code>- Döngülerde <code>break</code> , <code>continue</code> , <code>range()</code>- Listeler (<code>list</code>)- Demetler (<code>tuple</code>)- Sözlükler (<code>dictionary</code>)
3. Hafta: Fonksiyonlar ve Değişkenler	<ul style="list-style-type: none">- Fonksiyonlar ve Fonksiyon Tanımlama- <code>return</code> İfadesi- Global ve Yerel Değişkenler- Fonksiyonlarda Parametreler
4. Hafta: Dosya İşlemleri	<ul style="list-style-type: none">- Dosya Açma, Okuma, Yazma- Dosya Kapatma- Dosya Modları (<code>r</code> , <code>w</code> , <code>a</code>)- Hata Yönetimi ve Dosya İşlemleri

FONKSİYONLAR VE FONKSİYON TANIMLAMA

Fonksiyonlar, bir programda belirli bir işlemi tekrar tekrar yapmak için kullanılan kod bloklarıdır.

Programlama dillerinde fonksiyonlar, bir işlemi veya işlemler dizisini tek bir isim altında toplar ve bu işlevi çağırarak kullanmamızı sağlar.

- **Python'da fonksiyonlar def anahtar kelimesi ile tanımlanır.**
- **Fonksiyon tanımlarken ona bir isim vermeniz ve parantez içine isteğe bağlı parametreler yazmanız gerekir.**
- **Fonksiyonun gövdesindeki kodlar ise, fonksiyon çağırıldığında çalıştırılır.**



```
def hello_world(): # hello_world adında bir fonksiyon tanımlandı
    print("Hello World!") # Fonksiyonun gövdesinde çalışacak kod

hello_world() # Fonksiyonu çağırdık
```

**hello_world() fonksiyonu çalıştırıldığında
"Merhaba!" yazdırır.**

**Bu fonksiyon parametre almadığı için her
çağrıldığında aynı çıktıyı üretir.**

RETURN İFADESİ

return ifadesi, bir fonksiyonun sonucunu dışarıya döndürmesini sağlar. Bu sayede, fonksiyonun ürettiği sonuç başka bir değişkende saklanabilir veya başka işlemlerde kullanılabilir.

return ile döndürülen değer, fonksiyon çağrıldığı yere geri gönderilir ve orada kullanıma hazır hale gelir.



```
def topla(a, b): # İki parametre alan bir fonksiyon
    return a + b # Toplamı döndürür

sonuc = topla(3, 4) # Fonksiyonun döndürdüğü değeri sonuc
değişkenine atıyoruz
print(sonuc) # Çıktı: 7
```

Bu örnekte, topla fonksiyonu a ve b adında iki parametre alır ve bu iki sayının toplamını return ile döndürür.

Fonksiyonu çağırdığımızda elde edilen sonucu `sonuc` değişkenine atıyoruz ve ekrana yazdırıyoruz.

FONKSİYONLAR İLE BİRDEN FAZLA RETURN KULLANIMI

```
def sayi_kontrol(sayi):  
    if sayi > 0:  
        return "Pozitif"  
    elif sayi < 0:  
        return "Negatif"  
    else:  
        return "Sıfır"  
  
# Kullanıcıdan bir sayı alın ve sonucu ekrana yazdırın  
sayi = int(input("Bir sayı girin: "))  
sonuc = sayi_kontrol(sayi)  
print("Girilen sayı:", sonuc)
```

UYGULAMA: Basit Bir Sağlık Kontrol Uygulaması (BMI Hesaplayıcı)

Proje Detayları:

- **bmi_hesapla** fonksiyonu ile boy (metre cinsinden) ve kilo bilgisi alacak.
- BMI formülü kullanılarak sonuç hesaplanacak ve duruma göre yorum yapılacaktır.

```
def bmi_hesapla(kilo, boy):  
  
    bmi = kilo / (boy ** 2)  
    if bmi < 18.5:  
        return "Zayıf"  
    elif 18.5 <= bmi < 24.9:  
        return "Normal"  
    elif 25 <= bmi < 29.9:  
        return "Fazla Kilolu"  
    else:  
        return "Obez"  
  
# Kullanıcıdan boy ve kilo bilgisi alın  
kilo = float(input("Kilonuzu girin (kg): "))  
boy = float(input("Boyunuzu girin (m): "))  
  
# Fonksiyonu çağır ve sonucu ekrana yazdır  
sonuc = bmi_hesapla(kilo, boy)  
print("BMI Sonucu:", sonuc)
```

FONKSİYON İÇİ FONKSİYON KULLANIMI (İÇ İÇE FONKSİYONLAR)

```
def islem_yap(a, b):  
  
    def toplama(x, y):  
        return x + y  
  
    def carpma(x, y):  
        return x * y  
  
    toplam = toplama(a, b)  
    carpim = carpma(a, b)  
    return toplam, carpim  
  
# Sonuçları al ve yazdır  
a = 3  
b = 4  
  
sonuc_toplam, sonuc_carpim = islem_yap(a, b)  
print("Toplam:", sonuc_toplam)  
print("Çarpım:", sonuc_carpim)
```


GLOBAL VE YEREL DEĞİŞKENLER

Global değişkenler, programın herhangi bir yerinde tanımlanmış ve program boyunca erişilebilir olan değişkenlerdir.

Fonksiyonların dışında tanımlanan değişkenler global değişken olarak kabul edilir.

Yerel değişkenler ise bir fonksiyon içinde tanımlanır ve yalnızca o fonksiyon içinde kullanılabilir.

Fonksiyonun dışında yerel değişkene erişmeye çalışırsanız hata alırsınız.

```
x = 5 # Global değişken

def ornek():
    y = 3 # Yerel değişken
    print(x) # Fonksiyon içinde global değişkeni kullanabiliriz
    print(y) # Bu fonksiyon içinde yerel değişkene erişebiliriz

ornek()
print(x) # Fonksiyon dışında global değişkene erişebiliriz
print(y) # Hata verir çünkü y sadece ornek fonksiyonunda tanımlı
```

- Bu örnekte, x global bir değişken olarak tanımlanmış ve ornek() fonksiyonu dışında da erişilebilir.
- y değişkeni ise ornek() fonksiyonunda yerel olarak tanımlanmıştır ve yalnızca o fonksiyon içinde kullanılabilir.

GLOBAL ANAHTAR KELİMESİ KULLANIMI

**global anahtar kelimesi
kullanılarak, bir fonksiyon içinde
global değişkenler değiştirilebilir**

```
x = 10 # Global değişken


def degistir():
    global x # x'in global olduğunu belirtiriz
    x = 20 # x'in global değerini değiştiririz

degistir()
print(x) # Çıktı: 20
```

FONKSİYONLARDA PARAMETRELER

Parametreler, fonksiyona dışarıdan veri aktarımı yapmamızı sağlar.

Fonksiyon tanımlanırken parantez içine yazılan değişkenlere parametre denir.



```
def carp(a, b): # İki parametre alan bir fonksiyon
    return a * b

sonuc = carp(4, 5) # 4 ve 5 sayıları fonksiyona parametre olarak
verildi
print(sonuc) # Çıktı: 20
```

Bu örnekte `carp` fonksiyonu a ve b adında iki parametre alır ve bu parametrelerin çarpımını döndürür.

VARSAYILAN PARAMETRE KULLANIMI



```
def selamla(isim="Misafir"):  
    print("Merhaba,", isim)
```

```
# Varsayılan değer ile fonksiyonu çağırma  
selamla()
```

```
# Yeni bir değer ile fonksiyonu çağırma  
selamla("Çağrı")
```

UYGULAMA: Fibonacci Dizisi Hesaplama

Proje Detayları:

Bu projede, döngüleri kullanarak Fibonacci dizisinin belirli bir terime kadar olan sayılarını yazdıracağız.

Fibonacci dizisi, ilk iki terimin 0 ve 1 olduğu ve sonraki her terimin, önceki iki terimin toplamı olduğu bir dizidir.



```
def fibonacci(n):  
  
    dizi = [0, 1] # İlk iki terim  
    for i in range(2, n):  
        yeni_terim = dizi[i - 1] + dizi[i - 2]  
        dizi.append(yeni_terim)  
    return dizi[:n]  
  
# Kullanıcıdan terim sayısını al  
terim_sayisi = int(input("Fibonacci dizisinin kaç terimini görmek istersiniz? "))  
  
# Diziyi hesapla ve yazdır  
dizi = fibonacci(terim_sayisi)  
print("Fibonacci Dizisi:", dizi)
```