Temel Programlama 1

Öğr. Gör. Furkan DURMUŞ

Samsun Üniversitesi
Teknik Bilimler
Meslek Yüksekokulu

Bu Haftanın Ders Kazanımları

Bu haftayı bitirdiğinizde,

{ 1 } Önceki Hafta Tekrarı

{ 2 } Fonksiyon Nedir?

{ 3 } Fonksiyonların Kullanımı

{ 4 } Gömülü Fonksiyonların ve Modüllerin Kullanımı

{ 5 } Fonksiyon Tanımlama

öğrenmiş olacaksınız.

Öğr. Gör. Furkan DURMUŞ

Önceki Hafta Tekrarı

{01} Önceki Hafta Tekrarı

Girilen sayı 0 (sıfır) olana kadar girilen tüm sayıları toplayan ve ekranda gösteren programı yazınız.

```
toplam=0
sayi=1
while (sayi!=0):
    sayi=int(input(“Bir sayı giriniz: “))
    toplam=toplam+sayi
print(“Sonuc=”,toplam)
```


Öğr. Gör. Furkan DURMUŞ

{01} Önceki Hafta Tekrarı

Girilen şifre “Python” olana kadar “Tekrar deneyiniz” uyarısı veren, “Python” girildiğinde “Giriş başarılı” uyarısı veren kodu yazınız.

```
sifre = ""

while sifre != "Python":
    sifre = input("Şifreyi giriniz: ")
    if sifre != "Python":
        print("Tekrar deneyiniz.")

print("Giriş başarılı")
```

{01} Önceki Hafta Tekrarı

Örnek Çıktı:```
1-100 arası sayı tahmin edin: 50
Daha büyük bir sayı deneyin!
1-100 arası sayı tahmin edin: 75
Daha küçük bir sayı deneyin!
1-100 arası sayı tahmin edin: 62
Tebrikler! Doğru tahmin!

```python
import random
gizli_sayi = random.randint(1, 100)
tahmin = 0

while tahmin != gizli_sayi:
    tahmin = int(input("1-100 arası sayı tahmin edin: "))

    if tahmin < gizli_sayi:
        print("Daha büyük bir sayı deneyin!")

    elif tahmin > gizli_sayi:
        print("Daha küçük bir sayı deneyin!")

    else:
        print("Tebrikler! Doğru tahmin!")
```

{01} Önceki Hafta Tekrarı

Alışveriş örneği:

```
butce = int(input("Bütçenizi giriniz: "))
while butce<=0:
    butce = int(input("Bütçenizi sıfırdan büyük giriniz: "))

while butce > 0:
    fiyat = int(input("Ürün fiyatını giriniz : "))
    if fiyat>butce:
        print("Ürün fiyatı bütçeden büyük olamaz...")
    elif fiyat <=0:
        print("Program Sonlandırılıyor.")
        print(f"Kalan Bütçe : {butce}")
        butce = 0
    else:
        butce -= fiyat
        print(f"Kalan Bütçe : {butce}")
```

{01} Önceki Hafta Tekrarı

```
i=1
while True:
    if (i==5):
        print(“Döngüden çıkıldı”)
        break
    print(i)
    i=i+1
```

Çıktı :

1
2
3
4
Döngüden çıkıldı

Öğr. Gör. Furkan DURMUŞ

{01} Önceki Hafta Tekrarı

Kullanıcıdan 1 ile 5 arasında bir sayı girmesini isteyiniz. Kullanıcı 3 sayısını girdiğinde break komutu ile döngüden çıkılarak “3 sayısı girildi ve döngü sona erdi” çıktısı veren kodu yazınız.

```
while True:
    sayi = int(input("1 ile 5 arasında bir sayı girin: "))
    if sayi == 3:
        print("3 sayısı girildi ve döngü sona erdi")
        break
```


Öğr. Gör. Furkan DURMUŞ

{01} Önceki Hafta Tekrarı

Kullanıcıdan 8 karakterlik bir şifre girmesini isteyiniz. Kullanıcı 8'den az ya da daha fazla karakter içeren bir şifre girdiğinde "Şifreniz 8 karakter olmalıdır." şeklinde uyarı verdiriniz. Kullanıcı şartlara uygun bir şifre girdiğinde de "Şifreniz kaydedildi." uyarısı verdiriniz.

```
while True:
    sifre = input("8 karakterlik bir şifre girin: ")
    if len(sifre) != 8:
        print("Şifreniz 8 karakter olmalıdır.")
    else:
        print("Şifreniz kaydedildi.")
        break
```

{01} Önceki Hafta Tekrarı

Kullanıcı çift sayı girene kadar girdi alan Python kodunu yazınız.

```
while True:
    sayi = int(input("Sayı girin: "))
    if sayi % 2 == 0:
        print("Teşekkürler, çift sayı!")
        break
```

{01} Önceki Hafta Tekrarı

Kullanıcıdan sayılar alan ve toplam 100'ü geçtiğinde döngüyü bitiren Python kodunu yazınız.

```
toplam = 0
while True:
    sayi = int(input("Sayı girin: "))
    toplam += sayi
    if toplam > 100:
        print("Toplam 100'ü geçti!")
        break
```

{01} Önceki Hafta Tekrarı

```
i = 0
while i < 10:
   i=i+1
   if i == 5:
      continue
   print(i)
```

Çıktı :

```
1
2
3
4
6
7
8
9
10
```

{01} Önceki Hafta Tekrarı

Örnekteki kodun yorumlayınız.

```
i = 0
while i < 50:
    i=i+1
    if i>10 and i<45:
        continue
    print(i)
```


Öğr. Gör. Furkan DURMUŞ


Fonksiyon
Nedir

{02} Fonksiyon Nedir?

Bir bilgisayar programı yazılırken bazı işlemlerin programın farklı yerlerinde sürekli tekrarlanması gerekebilir.

Örneğin arazi hesapları ile ilgili bir program yazılıyorsa sık sık geometrik şekillerin alanı hesaplanmak zorunda kalınabilir. Her gerektiğinde alan hesabı işlemini yerine getiren kodları yazmak hem programcının iş yükünü hem de hata yapma olasılığını arttırır.

Öğr. Gör. Furkan DURMUŞ

{02} Fonksiyon Nedir?

Bu nedenle programcılar, sık tekrarlanan işler için aynı kodu defalarca yazmak yerine, işlemi yerine getiren kod bloğunu yazıp adlandırarak ihtiyaç hâlinde bu adla basit bir şekilde çağırıp kullanmayı tercih ederler.

{02} Fonksiyon Nedir?

İhtiyaç duyulduğunda çağrılıp çalıştırılabilen bu kod paketlerine **fonksiyon** adı verilir. Fonksiyon farklı programlama dillerinde prosedür ,metot veya yordam olarak da adlandırılabilir.

Sık tekrarlanan işlemleri gerektiğinde kullanılacak küçük kod parçalarına bölüp yazma yani “fonksiyon” yaklaşımı bütün programlama dillerinde kullanılabilecek bir yöntemdir.

{02} Fonksiyon Nedir?

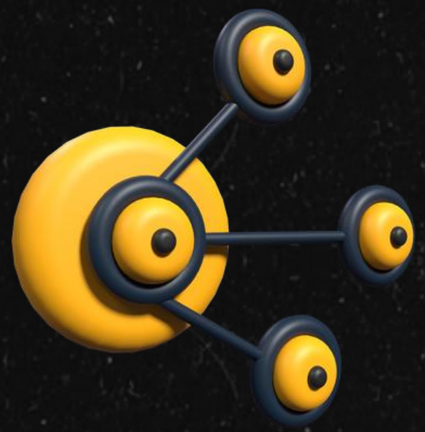
Fonksiyonların ne gibi artıları olabilir ?

{02} Fonksiyon Nedir?

Fonksiyonlar sayesinde;

- Programcı aynı kodları defalarca yazma yükünden kurtulur.
- Daha az kod yazılacağı için hata yapma olasılığı azalır.
- Fonksiyon sadece çağrıldığında kullanılacağı için bilgisayarın bellek kullanımından tasarruf edilir.
- Kod okunabilirliğini arttırır ve kod analizini daha kolay hâle getirir.
- Karmaşık problemlerin daha basit küçük parçalara ayrılarak çözülmesini kolaylaştırır.

{02} Fonksiyon Nedir?



Fonksiyonlar çağrıldıklarında, barındırdıkları kod kümelerini işleyerek oluşan sonuçları döndürebilir. Ayrıca istenirse kendilerine parametre olarak gönderilen verileri işleyip ürettikleri sonucu da döndürebilir.

Fonksiyonların Kullanımı

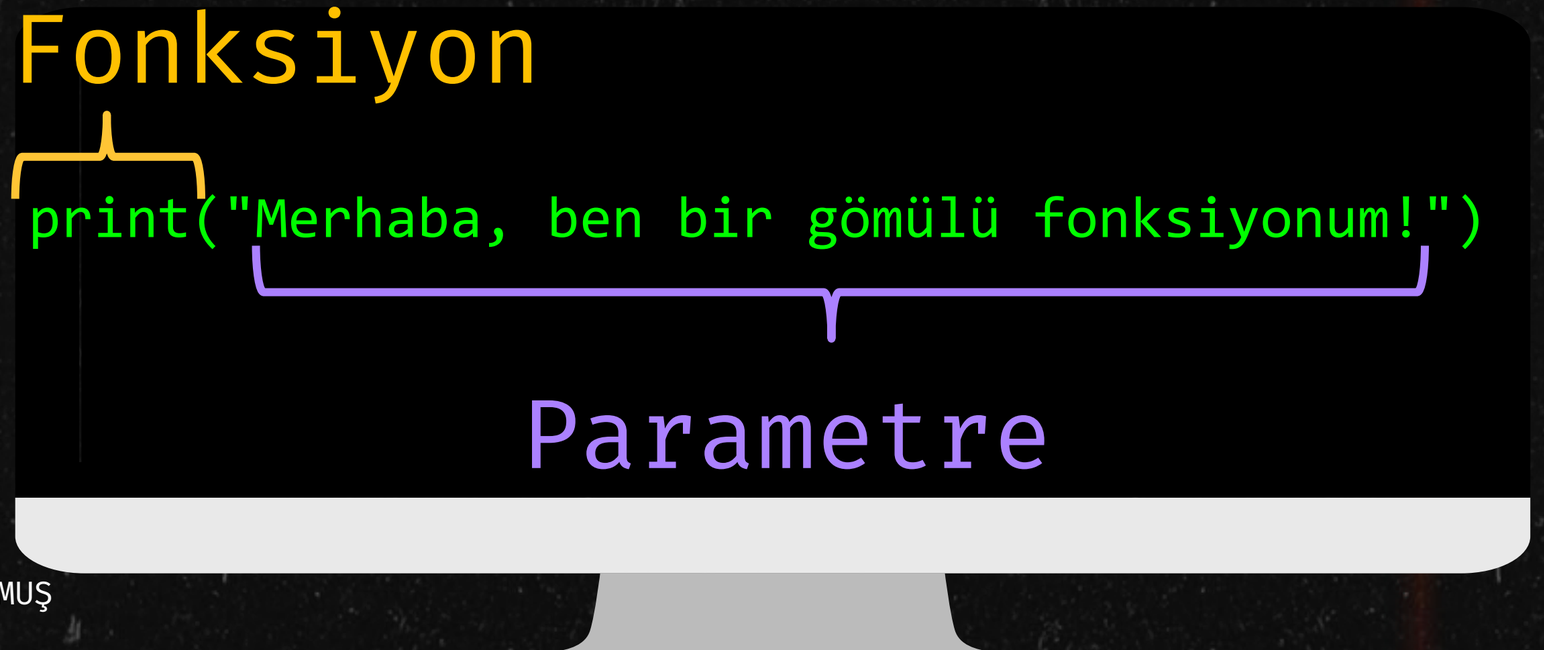
{03} Fonksiyonların Kullanımı

Bu bölüme kadar yazılan örnek programlarda programlama dilinin bazı hazır fonksiyonları kullanıldı. Örneğin ekrana veri yazdırmak için kullandığınız print() bir fonksiyondur.

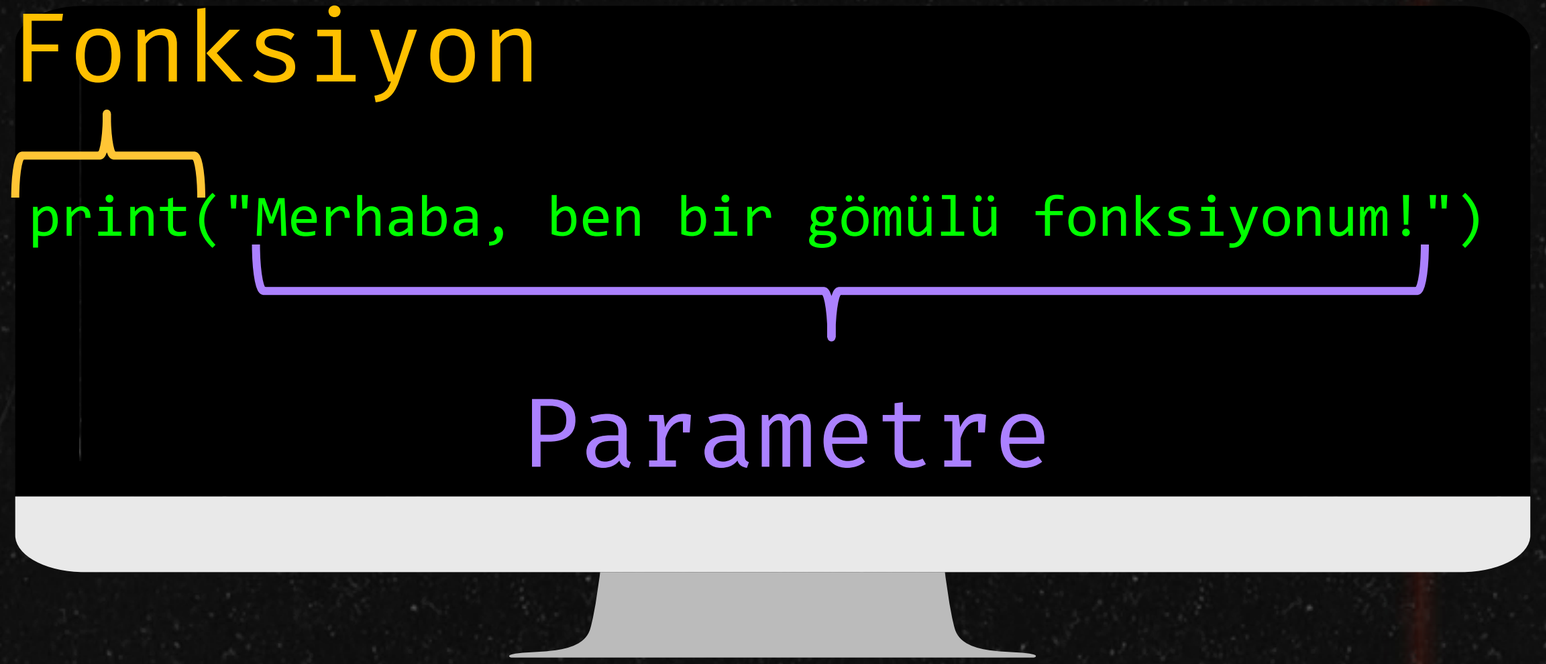
Programlama dilleri yazılımcının gerektiğinde kullanabileceği birçok hazır fonksiyonla beraber gelir. Bunlara **built-in** (**gömülü fonksiyonlar**) denir.

{03} Fonksiyonların Kullanımı

Bir fonksiyonu çağırıp çalıştırmak için fonksiyona verilen ismi yazmak gerekir. Eğer fonksiyon parametre alıyorsa isminin yanına parantez içinde fonksiyona gönderilecek parametreleri de yazmak gerekir.



{03} Fonksiyonların Kullanımı



Yukarıdaki komut çalıştırıldığında programlama dili ile beraber gelen print isimli fonksiyon, ekrana getireceği metin fonksiyona parantez içinde parametre olarak gönderilerek çağrılmış olur.

{03} Fonksiyonların Kullanımı

Tanımladığımız veya programlama dili ile hazır gelen fonksiyonlar çağrılmadıkça o fonksiyon bloğu içinde yer alan kodlar çalıştırılmaz. Fonksiyonu başka bir fonksiyondan ya da doğrudan programdan ismi ile birlikte parantez içinde parametre bilgilerini yazarak çağırabilirsiniz.

Gömülü Fonksiyonların ve Modüllerin Kullanımı

{04} Gömülü Fonksiyonların ve Modüllerin Kullanımı

Programlama dili ile temel işlemleri yerine getiren birçok fonksiyon hazır ve tanımlanmış olarak gelir. Şu ana kadar kullanılmış olan print, input, type, int, float, str gibi fonksiyonlar programlama dili içinde gömülüdür.

Gömülü fonksiyonlar, geliştiricileri tarafından programlama dili içine gömülmüş ve tanımlamaya gerek kalmadan kullanılabilen fonksiyonlardır. Gömülü fonksiyonlarda tek yapılması gereken fonksiyonu çağırmak ve kullanmaktır.

Öğr. Gör. Furkan DURMUŞ

{04} Gömülü Fonksiyonların ve Modüllerin Kullanımı

docs.python.org/3/library/functions.html

Previous topic
Introduction

Next topic
Built-in Constants

This page
Report a bug
Show source

Built-in Functions

A	E	L	R
abs()	enumerate()	len()	range()
aiter()	eval()	list()	repr()
all()	exec()	locals()	reversed()
anext()			round()
any()	F	M	
ascii()	filter()	map()	S
	float()	max()	set()
B	format()	memoryview()	setattr()
bin()	frozenset()	min()	slice()
bool()			sorted()
breakpoint()	G	N	staticmethod()
bytearray()	getattr()	next()	str()
bytes()	globals()		sum()
		O	super()
C	H	object()	
callable()	hasattr()	oct()	T
chr()	hash()	open()	tuple()
classmethod()	help()	ord()	type()
compile()	hex()		
complex()		P	V
	I	pow()	vars()
D	id()	print()	
delattr()	input()	property()	Z
dict()	int()		zip()
dir()	isinstance()		
divmod()	issubclass()		_
	iter()		__import__()

abs(number, /)

{04} Gömülü Fonksiyonların ve Modüllerin Kullanımı

Bu gömülü fonksiyonlar haricinde farklı işlevler için geliştirilmiş fonksiyon kütüphaneleri vardır. Örneğin matematik işlemlerinde ihtiyaç duyabileceğiniz tüm fonksiyonlar, hazır olarak programlama dili ve “Math” isimli bir kütüphane ile gelir.

İhtiyaç duyduğunuzda makine öğrenmesi, oyun geliştirme, ağ işlemleri gibi alanlarda size gerekli işlevleri sağlayacak kütüphaneler programlama diline eklenip kullanılabilir.

Öğr. Gör. Furkan DURMUŞ

{04} Gömülü Fonksiyonların ve Modüllerin Kullanımı

Kendiniz de projenizde kullanmak için yazdığınız fonksiyonları bir kütüphane hâlinde toplayarak ihtiyacı olan programcılara dağıtabilirsiniz. Bu şekilde bir konuda belirli işlevleri yerine getiren fonksiyonların bir araya getirildiği Python dosyalarına **modül** denir.

Hâlihazırda programlama dili kurulumu ile birlikte birçok modül bilgisayarınıza yüklenir. Bu modüller haricinde ihtiyaç duyabileceğiniz modüller de üçüncü parti sağlayıcılardan bulunabilir.

{04} Gömülü Fonksiyonların ve Modüllerin Kullanımı

Ders kapsamında hiç modül kullandık mı?

{04} Gömülü Fonksiyonların ve Modüllerin Kullanımı

Programlama diline eklenmiş olan modülü ve içerdiği fonksiyonları kullanabilmek için önce yazılan kodun başına “**import**” komutu eklenerek modüle erişim sağlanır.

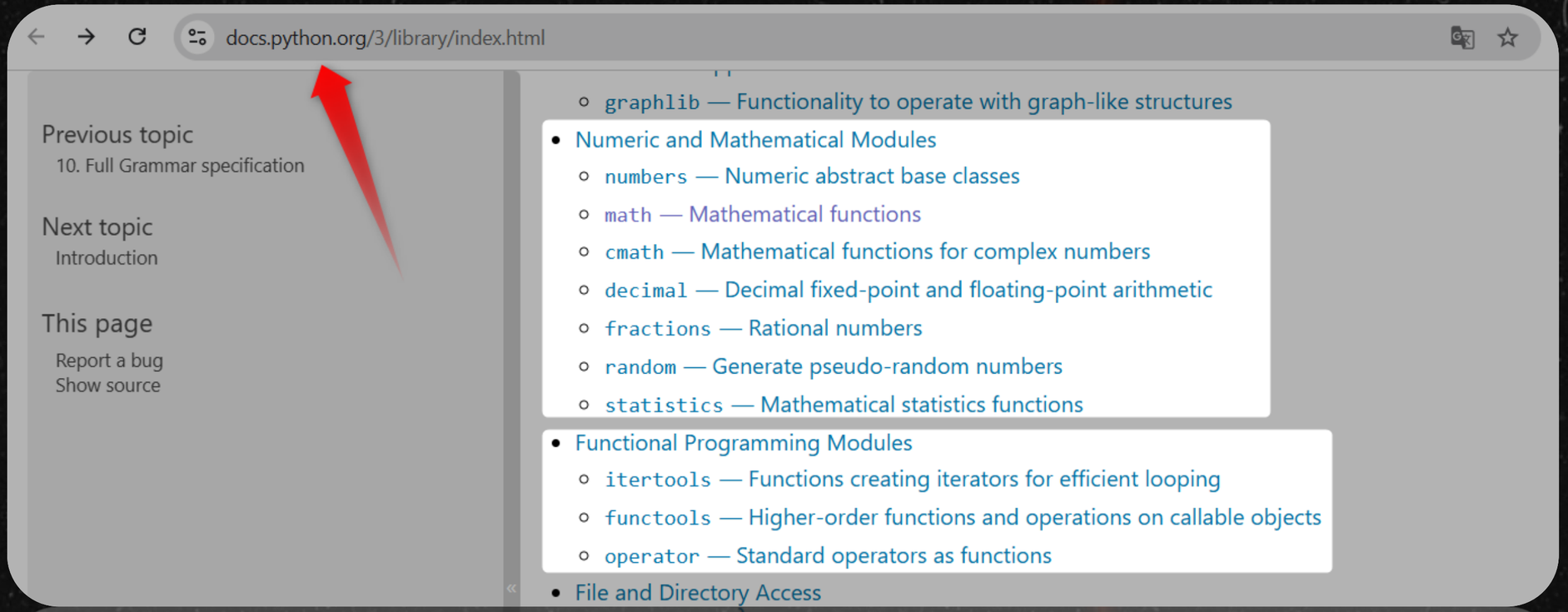
```
from modül_adi import fonksiyon_adi
```

Programlama dili kurulumu ile gelen, matematik fonksiyonlarını içeren “math.py” dosyasına yani Math modülüne erişmek için programın başına aşağıdaki gibi erişim ifadesi eklenmesi gerekir.

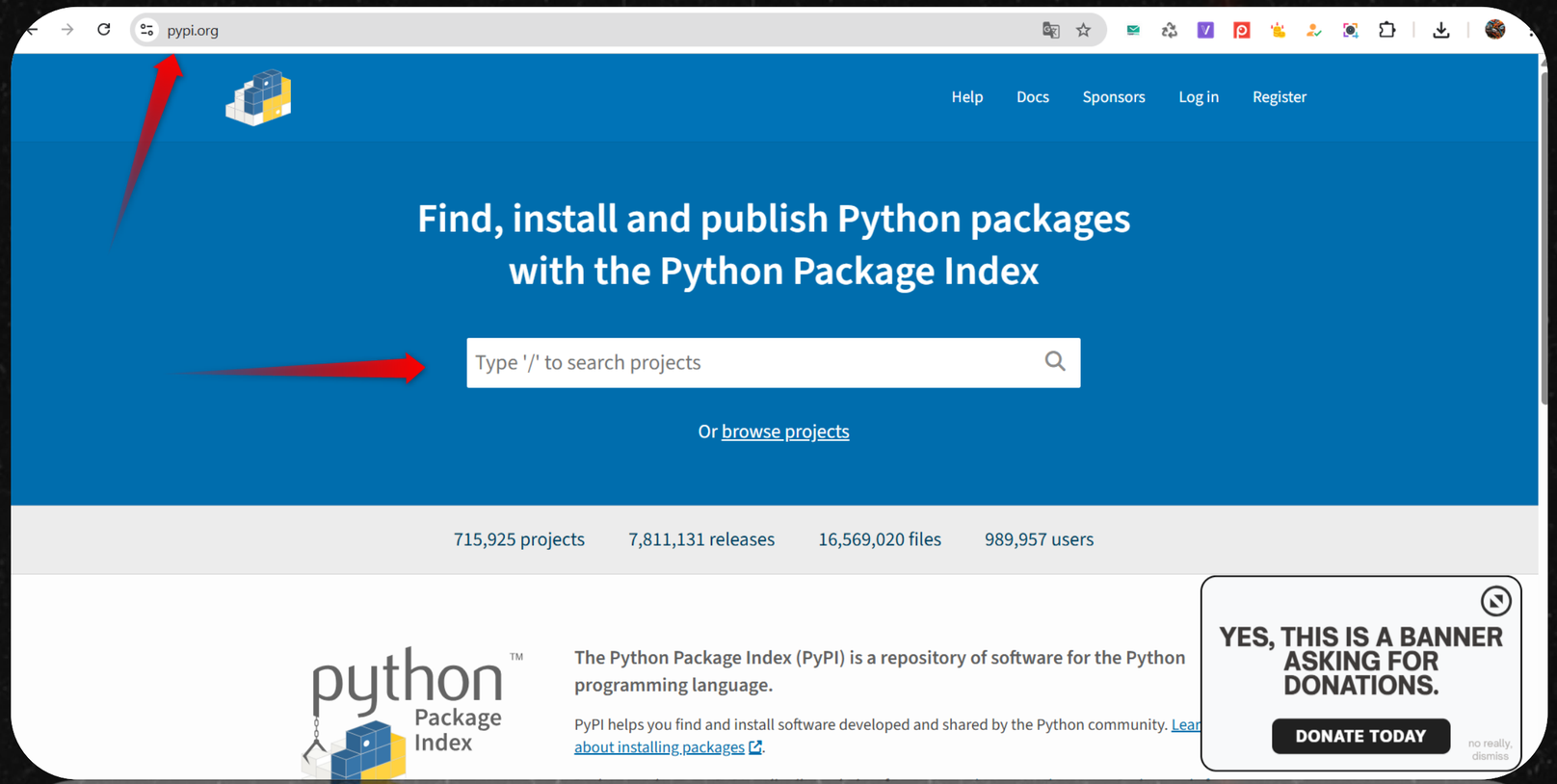
```
from math import sin
```


Öğr. Gör. Furkan DURMUŞ

{04} Gömülü Fonksiyonların ve Modüllerin Kullanımı



{04} Gömülü Fonksiyonların ve Modüllerin Kullanımı



{04} Gömülü Fonksiyonların ve Modüllerin Kullanımı

```
from math import sin
```

satırı eklendiğinde Math modülünden parametre olarak verilen sayının sinüs değerini veren fonksiyona erişim sağlanmış olur ve sin() tanımlı fonksiyon programda istediğiniz yerde kullanılabilir.

{04} Gömülü Fonksiyonların ve Modüllerin Kullanımı

Birden fazla fonksiyona erişim sağlamak isteniyorsa fonksiyonları aşağıdaki gibi beraber belirtmek yeterli olacaktır.

```
from math import sin, sqrt, cos, pow
```

{04} Gömülü Fonksiyonların ve Modüllerin Kullanımı

```
from math import sin, sqrt, cos, pow
print(  sqrt(4) )
print(  sin(30) )
print(  cos(45) )
print( pow(3,2) )
```

Çıktı :

```
2.0
0.9974949866040544
0.23523757330298942
9.0
```

{04} Gömülü Fonksiyonların ve Modüllerin Kullanımı

```
import math
print( math.pow(3,12) )
print( math.sqrt(9) )
print( math.sin(math.pi/2) )
```

Eğer aynı modülden çok sayıda fonksiyon kullanılması gerekiyorsa fonksiyon isimlerini spesifik olarak yazmak yerine programa import modül_adı satırı eklenerek modüldeki tüm fonksiyonlara erişim sağlanabilir.

Fonksiyonlar isimleri ile erişime açılmadığı için programdan çağrılırken isimlerinin başlarına modül adları da eklendi. Bu sayede program hangi modülden hangi fonksiyona erişilmek istendiğini anlayarak fonksiyonun işlevini yerine getirmiştir.

Öğr. Gör. Furkan DURMUŞ

Fonksiyon Tanımlama

{05} Fonksiyon Tanımlama

Fonksiyonlar def komutu kullanılarak tanımlanabilir. Fonksiyon tanımlarken izlenecek yol aşağıdaki gibidir:

1. def komutu yazılarak yeni bir fonksiyon tanımlanacağı programlama diline bildirilir.

2. Anahtar sözcükten sonra fonksiyon çağrılırken kullanılacak olan isim Python'un isimlendirme kurallarına uygun olarak belirlenmelidir. Burada fonksiyonun işlevi ile ilintili bir isim vermek kod okunabilirliği açısından mantıklı olacaktır.

{05} Fonksiyon Tanımlama

Fonksiyonlar def komutu kullanılarak tanımlanabilir. Fonksiyon tanımlarken izlenecek yol aşağıdaki gibidir:

3. Parantezler arasına fonksiyona gönderilecek parametreler yazılır, eğer fonksiyonumuz parametre almıyorsa parantez araları boş bırakılır. Tanım sonuna iki nokta üst üste konarak alt satırdan itibaren kod bloğunun başladığı belirtilir.

4. Tanım ve isim satırının altında bir sekme (tab) boşluk bırakılarak fonksiyon çağrıldığında çalışacak kodlar yazılır.

{05} Fonksiyon Tanımlama

```
def fonksiyon_adi  (varsa parametre listesi) :
        kod_bloğu
        .
        .
        .
```

{05} Fonksiyon Tanımlama

```
def selamla():
        print("Merhaba Arkadaşlar!")
```

Yandaki kod bloğu çalıştırıldığında ekrana hiçbir şey gelmeyecektir. Çünkü fonksiyon tanımlanmasına rağmen henüz çağrılmadı. Tanımlanan fonksiyonlar sadece çağrıldıklarında çalışır.

{05} Fonksiyon Tanımlama

```
def selamla():
            print("Merhaba Arkadaşlar!")


selamla()
```

Program çalıştırıldığında fonksiyonun çağrılıp yürütüldüğü ve ekrana “Merhaba Arkadaşlar!” yazısının geldiği görülür. İstenirse aynı fonksiyon birden fazla çağrılabilir.

{05} Fonksiyon Tanımlama

```
def selamla():
            print("Merhaba Arkadaşlar!")

def isimAl():
    isim = input("İsminiz :")

isimAl()
selamla()
```

Yandaki kodu yorumlayınız.

{05} Fonksiyon Tanımlama

Bir programda fonksiyon kullanılmadan önce tanımlanmalıdır, aksi hâlde programınız hata verecektir.

{05} Fonksiyon Tanımlama

```
def selamla():
     ad   =    str(input("Adınızı giriniz:  "))
     if ad == ‘Furkan’:
          print ("Merhaba " + str(ad))
     else:
          print("Merhaba yabancı!")

selamla()
```

Yandaki kodu yorumlayınız.

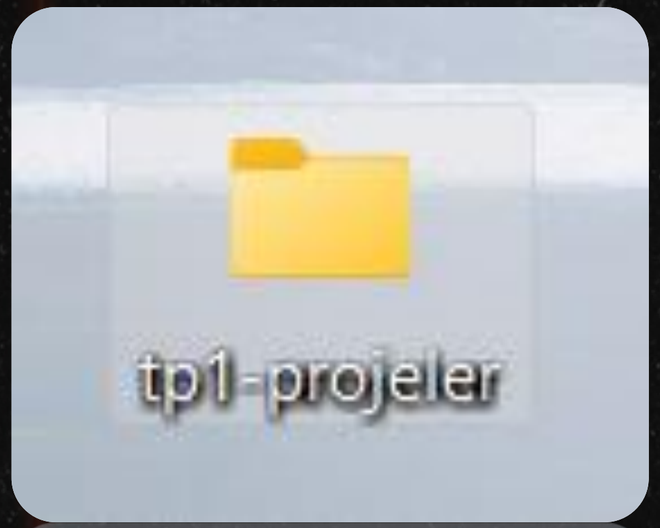
{05} Fonksiyon Tanımlama

```
def onakadar_say():
    a = 0
    while a<10:
        a += 1
        print(a)

print("program başlıyor...")
tercih = input("saymaya başlansın mı?")

if tercih == "E":
    onakadar_say()
```

Yandaki kodu yorumlayınız.

{05} Fonksiyon Tanımlama

Masaüstünde "**tp1-projeler**" adında bir klasör oluşturunuz.

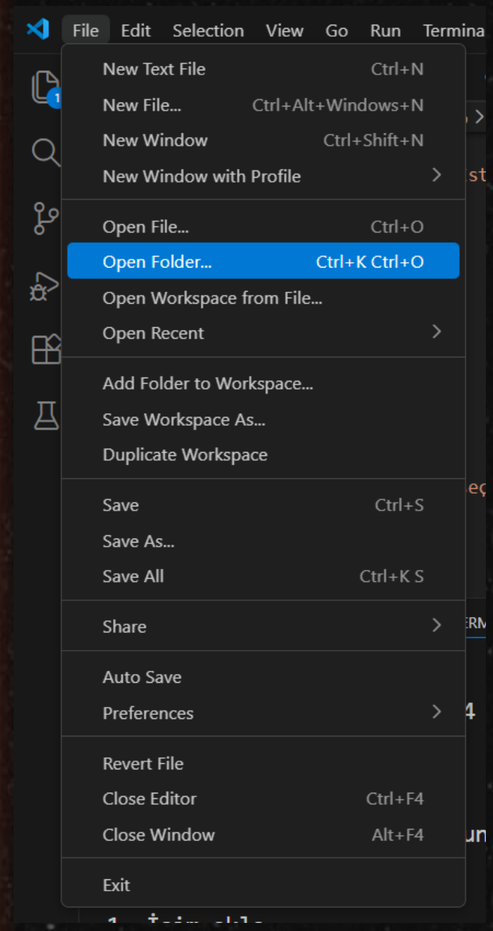


{05} Fonksiyon Tanımlama

Vs Code editöründe

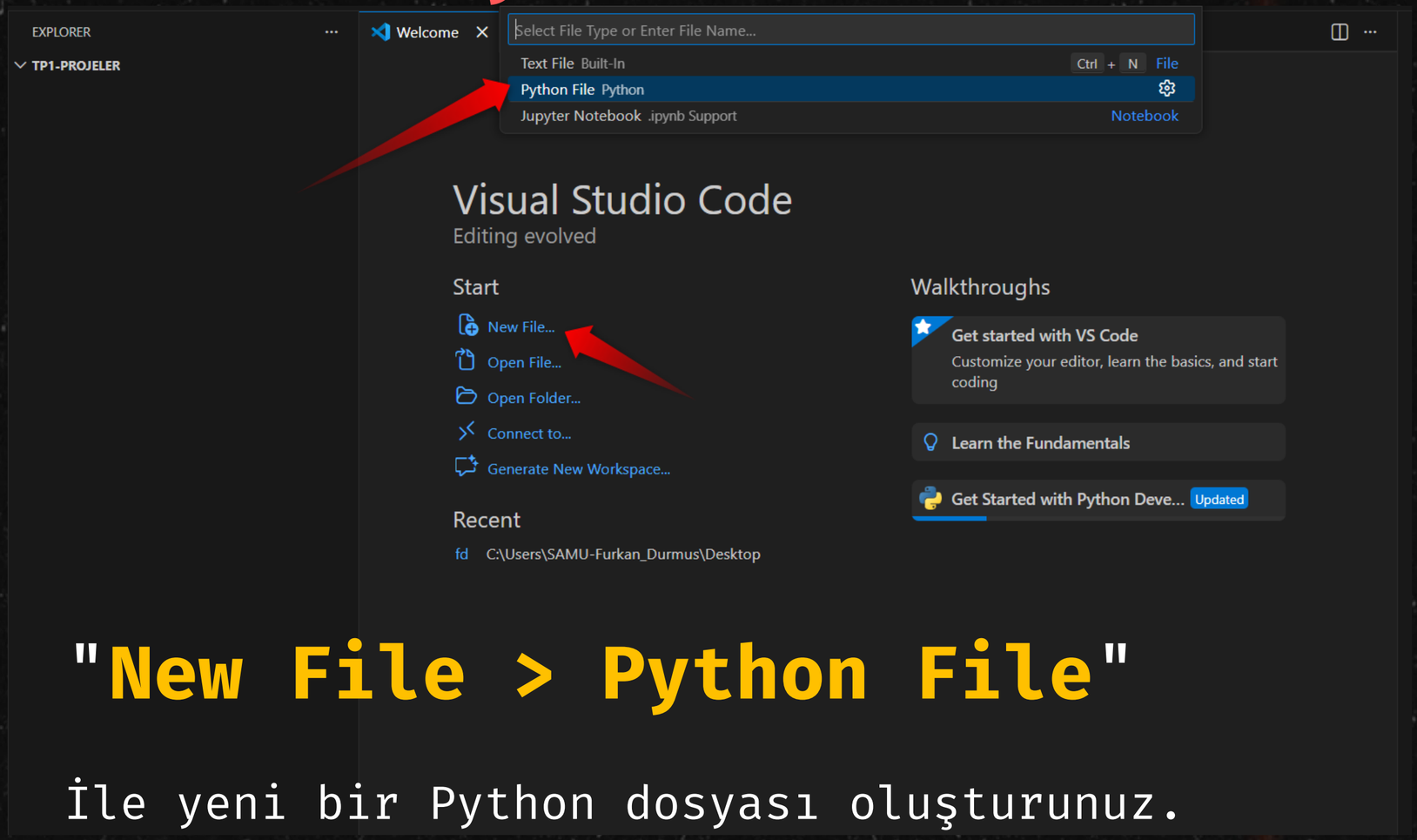
"**File > Open Folder…**"

İle oluşturduğunuz klasörü seçiniz.



Öğr. Gör. Furkan DURMUŞ

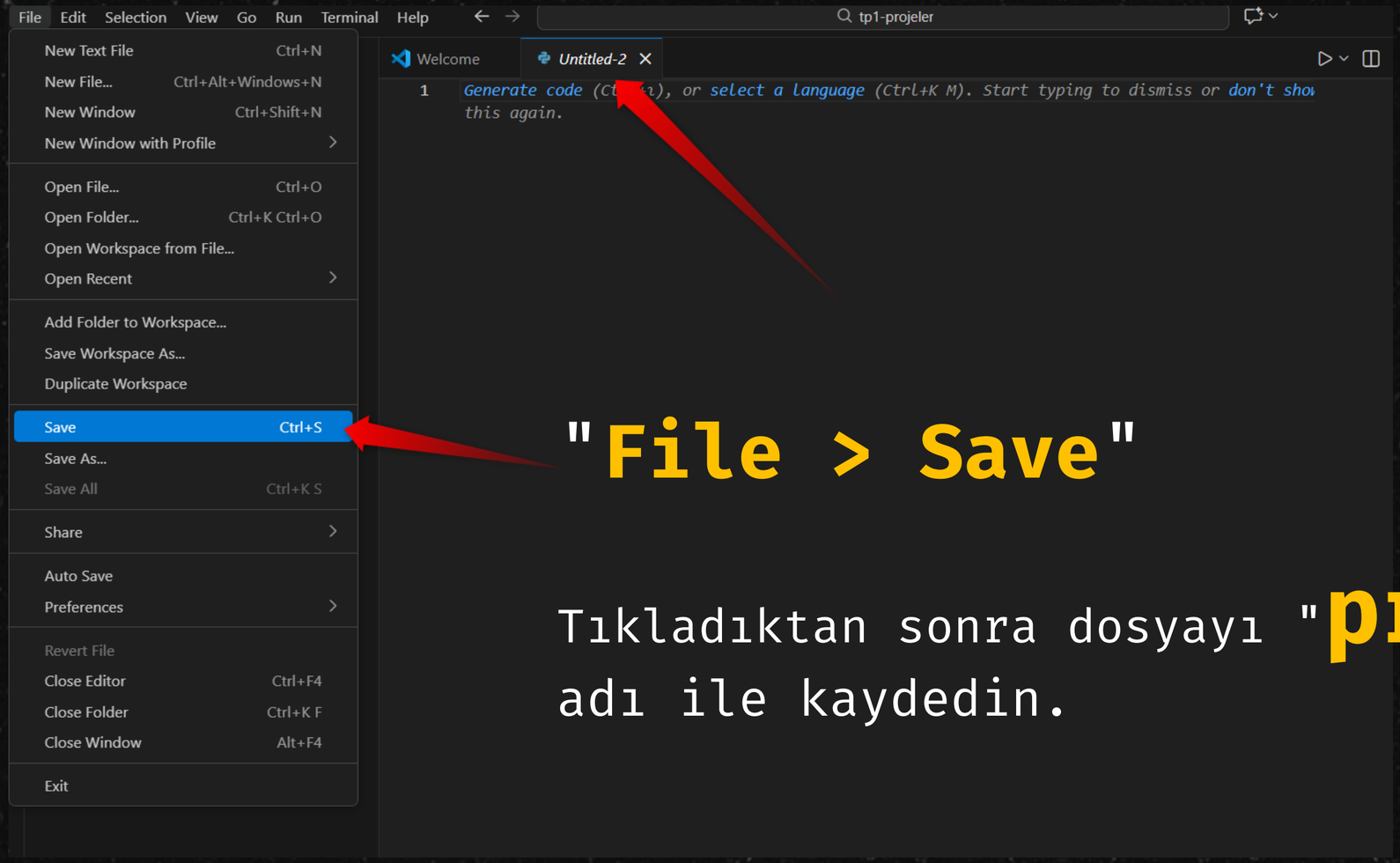
{05} Fonksiyon Tanımlama



"New File > Python File"

İle yeni bir Python dosyası oluşturunuz.

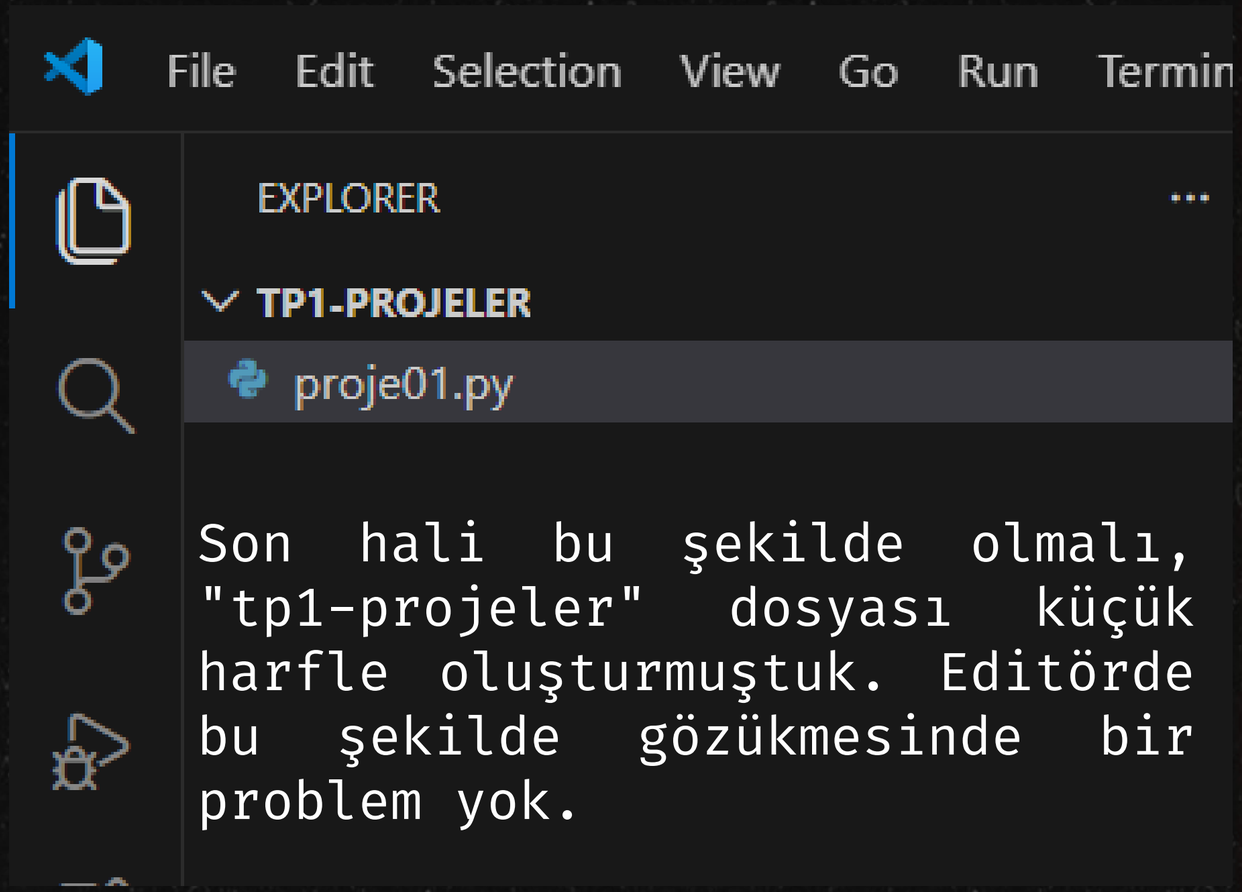
{05} Fonksiyon Tanımlama



"**File > Save**"

Tıkladıktan sonra dosyayı "**proje01**" adı ile kaydedin.

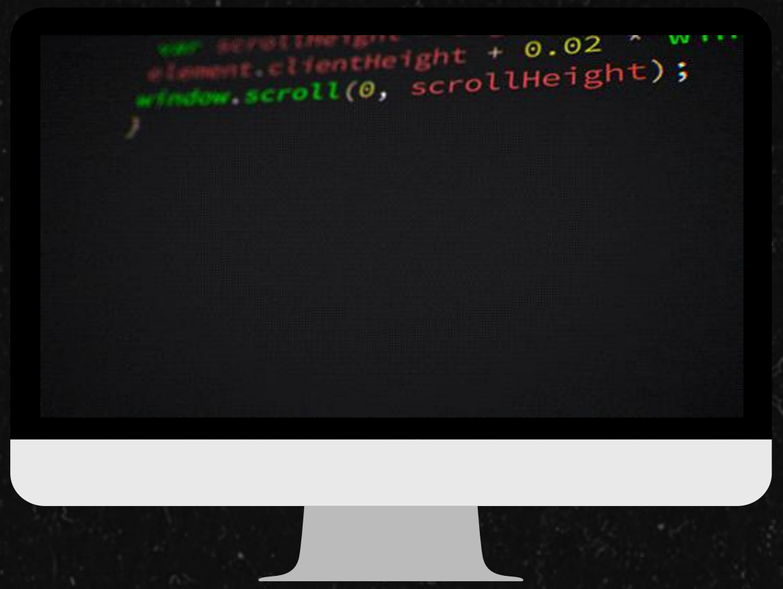
{05} Fonksiyon Tanımlama



Son hali bu şekilde olmalı, "tp1-projeler" dosyası küçük harfle oluşturmuştuk. Editörde bu şekilde gözükmesinde bir problem yok.

{05} Fonksiyon Tanımlama

proje01.py dosyasında yandaki çıktıyı verecek Python kodunu, fonksiyon, döngüler ve koşullu ifadeler kullanarak yazınız.




```
--- HESAP MAKİNESİ ---
1 - Toplama
2 - Çıkarma
3 - Çarpma
4 - Bölme
0 – Çıkış

Seçiminiz: 1
1. sayıyı giriniz: 10
2. sayıyı giriniz: 5
Sonuç: 15

--- HESAP MAKİNESİ ---
1 - Toplama
2 - Çıkarma
3 - Çarpma
4 - Bölme
0 – Çıkış

Seçiminiz: 4
1. sayıyı giriniz: 10
2. sayıyı giriniz: 0
Hata: Bir sayı 0'a bölünemez!
```

```
--- HESAP MAKİNESİ ---
1 - Toplama
2 - Çıkarma
3 - Çarpma
4 - Bölme
0 - Çıkış

Seçiminiz: 0
Programdan çıkılıyor...
```

{05} Fonksiyon Tanımlama

```
48
49      elif secim == "2":
```

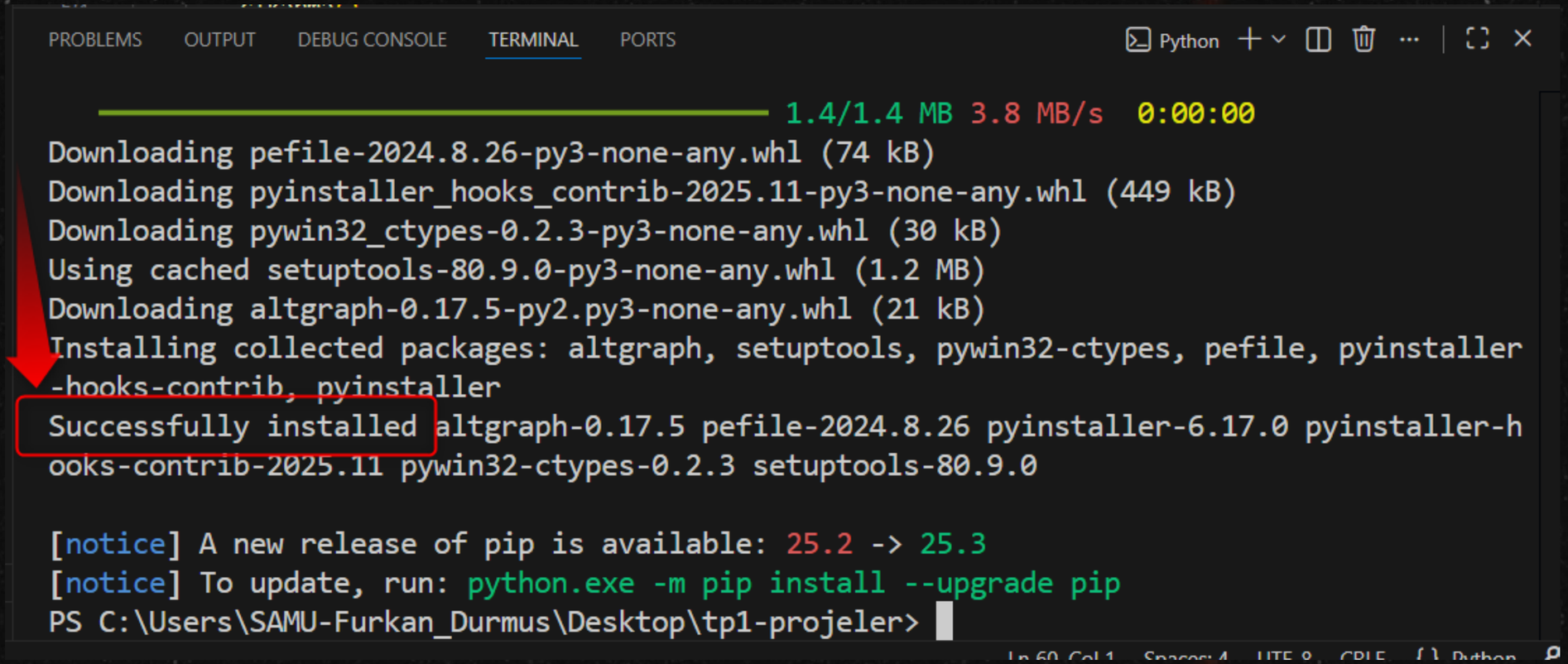
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Python

```
2. sayıyı giriniz: 46
Sonuç: 920.0

--- HESAP MAKİNESİ ---
1 - Toplama
2 - Çıkarma
3 - Çarpma
4 - Bölme
0 - Çıkış

Seçiminiz: 0
Programdan çıkılıyor...
PS C:\Users\SAMU-Furkan_Durmus\Desktop\tp1-projeler> pip install pyinstaller
>>
```

Ln 60, Col 1 Spaces: 4 UTF-8 CRLF {} Python

Kodunuz çalışmadığı esnada terminal ekranına "**pip install pyinstaller**" yazıp enter'a basın.

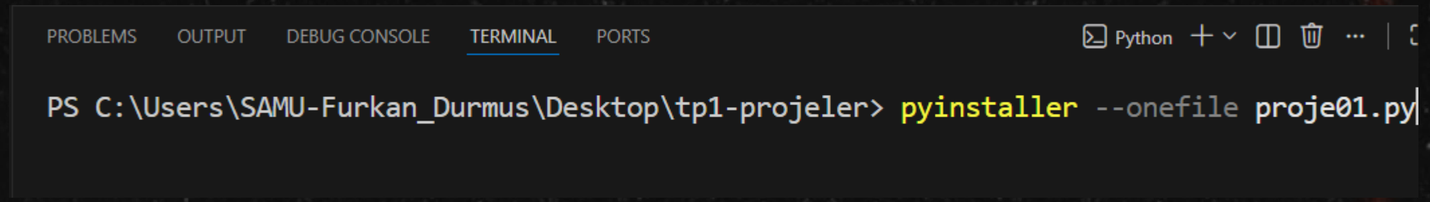
{05} Fonksiyon Tanımlama



Hata almadan successfully installed yazısını görmeniz gerekmektedir.

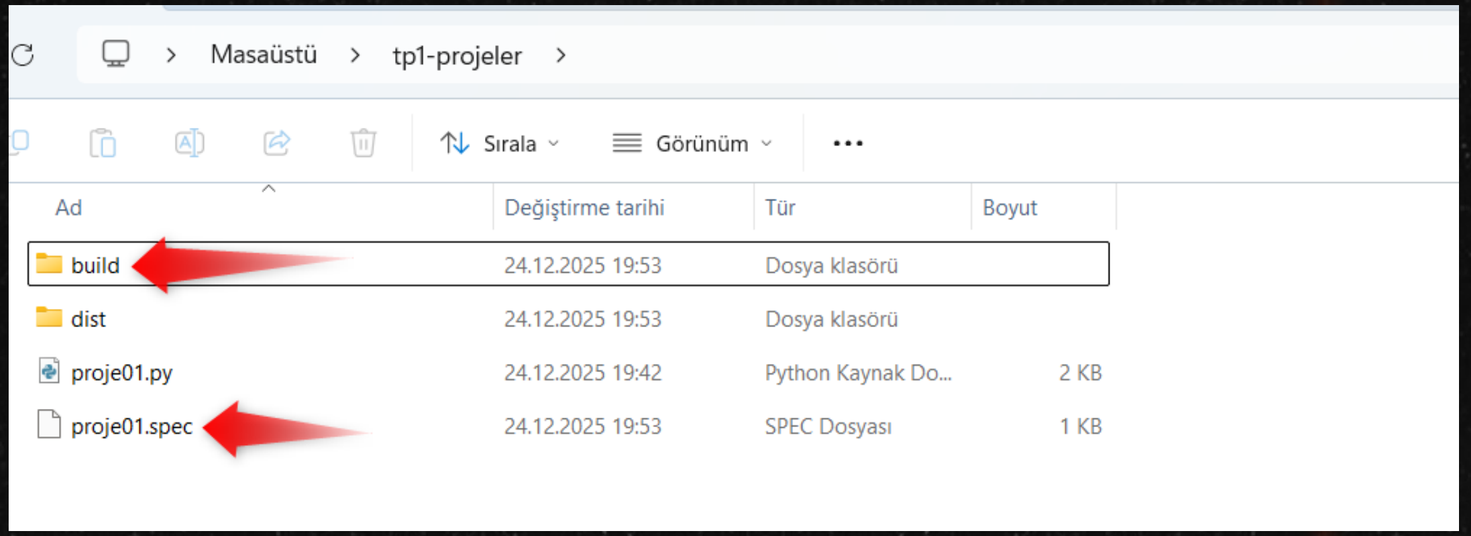
Öğr. Gör. Furkan DURMUŞ

{05} Fonksiyon Tanımlama



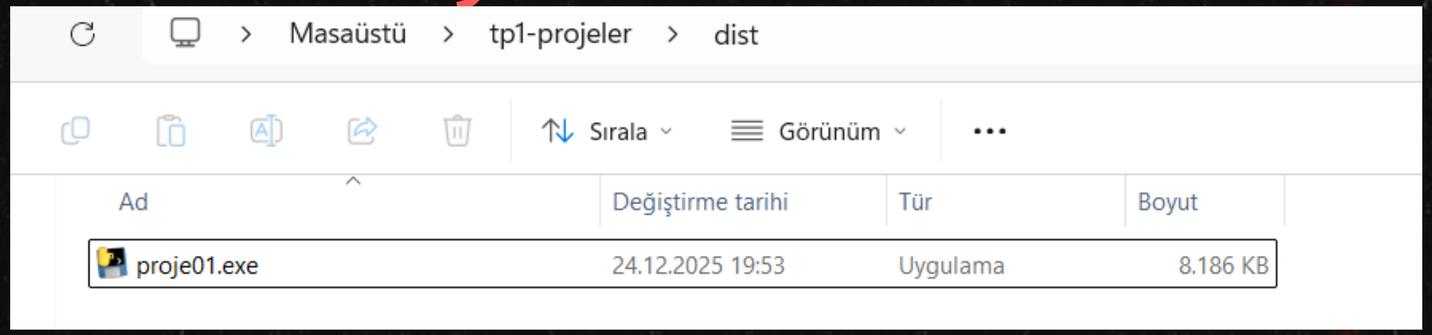
Aynı terminale "**pyinstaller --onefile proje01.py**" yazıp enter'a basınız.

{05} Fonksiyon Tanımlama



Bu 2 dosyayı silebilirsiniz.

{05} Fonksiyon Tanımlama



Tebrikler ilk exe'nizi oluşturdunuz.

Dist klasörünün içindeki **exe** tüm Windows'larda çalışabilen hesap makineniz.

{05} Fonksiyon Tanımlama

```
bakiye = 0

def para_yükle():
    bakiye += 10
    print("bakiye:", bakiye)

para_yükle()
para_yükle()
para_yükle()
para_yükle()
```

Yandaki kodu bilgisayarınızda çalıştırınız.

{05} Fonksiyon Tanımlama

```
bakiye = 0

def para_yükle():
    bakiye += 10
    print("bakiye:", bakiye)

para_yükle()
para_yükle()
para_yükle()
para_yükle()
```

Yandaki kodu bilgisayarınızda çalıştırınız.

Neden hata vermiş olabilir ?

{05} Fonksiyon Tanımlama

```
bakiye = 0

def para_yükle():
    bakiye = 0
    bakiye += 10
    print("bakiye:", bakiye)

para_yükle()
para_yükle()
para_yükle()
para_yükle()
```

Yandaki kodu bilgisayarınızda çalıştırınız.

{05} Fonksiyon Tanımlama

```
bakiye = 0

def para_yükle():
    bakiye = 0
    bakiye += 10
    print("bakiye:", bakiye)

para_yükle()
para_yükle()
para_yükle()
para_yükle()
```

Yandaki kodu bilgisayarınızda çalıştırınız.

Fonksiyon içindeki ‘bakiye’ **local** değişkendir. Local değişkenler sadece bulundukları fonksiyonda çalışır ve dışarıdaki değişkenleri etkileyemezler.

{05} Fonksiyon Tanımlama

```
bakiye = 0

def para_yükle():
    bakiye = 20
    print("bakiye:", bakiye)

para_yükle()
print("bakiye:", bakiye)
```

Yandaki kodun çıktısını ve sebebini söyleyiniz.

{05} Fonksiyon Tanımlama

```
bakiye = 0

def para_yükle():
    global bakiye
    bakiye += 10
    print("bakiye:", bakiye)

para_yükle()
para_yükle()
para_yükle()
para_yükle()
```

Yandaki kodu bilgisayarınızda çalıştırınız.

{05} Fonksiyon Tanımlama

```
bakiye = 0

def para_yükle():
    global bakiye
    bakiye += 10
    print("bakiye:", bakiye)

para_yükle()
para_yükle()
para_yükle()
para_yükle()
```

global anahtar kelimesi, fonksiyon içinde, fonksiyon dışında tanımlanmış bir değişkeni güncellememizi sağlar.

{05} Fonksiyon Tanımlama

```
sepet_adet = 0

def sepete_ekle():
    ...
def sepet_bosalt():
    ...

sepete_ekle()
sepete_ekle()
sepet_bosalt()
sepete_ekle()
```

Yandaki kodu sepete ekleme ve çıkarma yapacak şekilde ... Olan yerleri doldurunuz.

{05} Fonksiyon Tanımlama

```
sepet_adet = 0

def sepete_ekle():
    global sepet_adet
    sepet_adet += 1
    print("Sepet adet:", sepet_adet)

def sepet_bosalt():
    global sepet_adet
    sepet_adet = 0
    print("Sepet boşaltıldı.")

sepete_ekle()
sepete_ekle()
sepet_bosalt()
sepete_ekle()
```

Yandaki kodu sepete ekleme ve çıkarma yapacak şekilde ... Olan yerleri doldurunuz.

Son