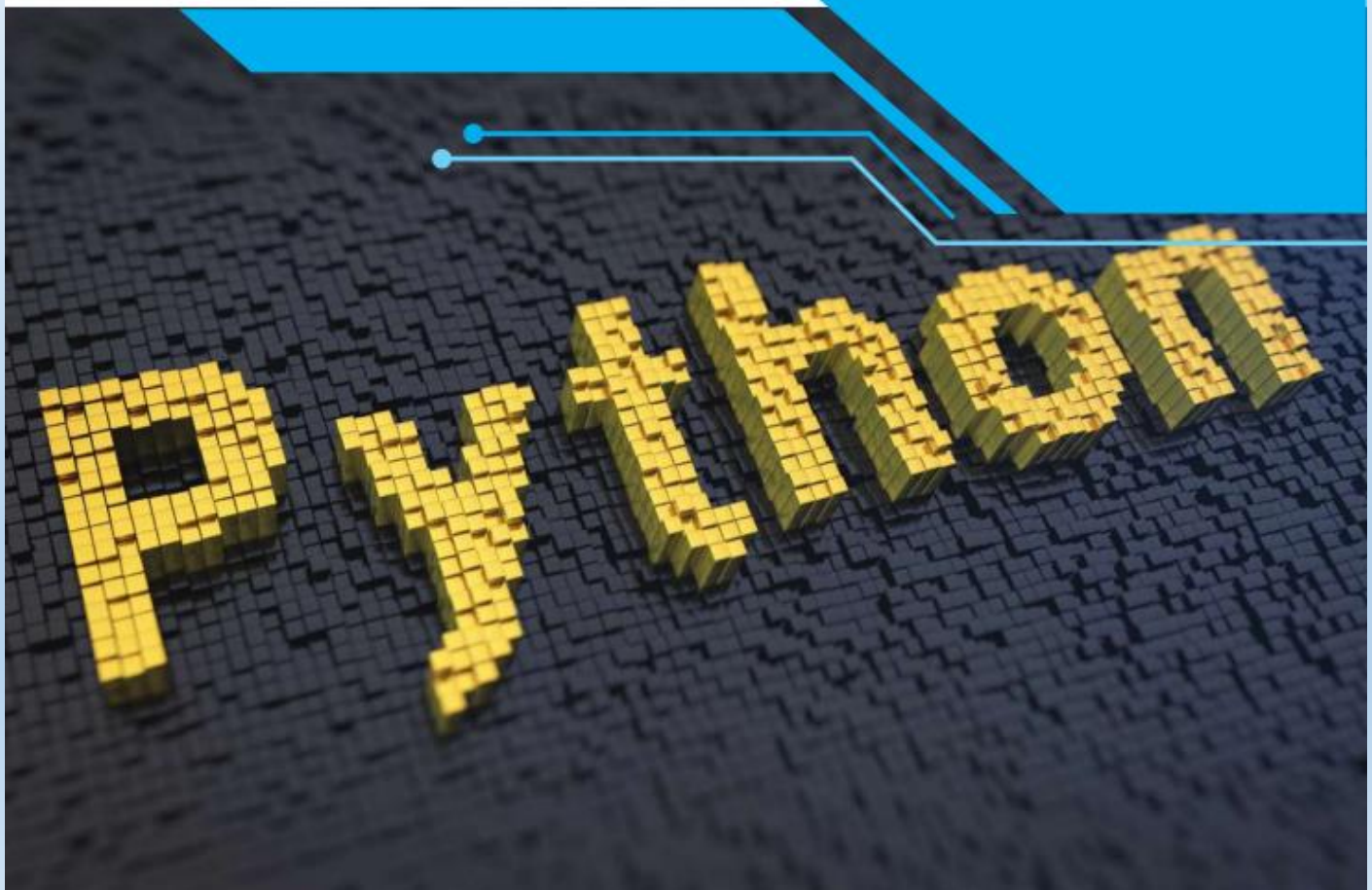
PYTHON İLE PROGRAMLAMANIN TEMELLERİ



Bilgisayar Programları Nasıl Çalışır?

Bilgisayarlar yazılım ve donanım olmak üzere iki bileşenden oluşur. Donanımlar, bilgisayarların fiziksel parçalarıdır. Klavye, fare, ekran gibi bileşenler bilgisayarın donanım birimleridir.

Yazılım ise bilgisayarda donanıma hayat veren ve bilgi işlemde kullanılan programlar, yordamlar, programlama dilleri ve belgelemelerin tümüdür.

Bilgisayarlarda bulunan oyunlar, müzik programları, ofis programları hatta işletim sistemleri dâhil tüm yazılımlar bir programlama dili ile yazılmıştır.

Doğal Diller ve Programlama Dilleri

Dil, düşünceleri ifade etmek ve kaydetmek için bir araçtır. Her toplumun kullandığı dil, birbirinden farklı olduğu gibi toplum içinde de dil bölgelere göre farklı biçimlerde kullanılabilir.

Bilgisayarın anladığı gerçek dil, 0 ve 1'lerden ibarettir. Bilgisayar bizim kendisine verdiğimiz komutları makine diline çevirerek icra eder.

Doğal Diller ve Programlama Dilleri

Gerçekte bilgisayarları insanlar programlamaktadır. Programcıların direkt makine dilinde program yazmaları çok zahmetli ve uzun bir süreçtir. Ayrıca bilgisayar donanımındaki hızlı gelişim, yazılım geliştirme sürecinde daha pratik yöntemlerin gelişmesini gerekli kılmıştır. Bu gelişmeler yazılım geliştiren bireyler için programlama dillerinin ortaya çıkmasına neden olmuştur. Böylece yazılım geliştiricilere, kendi dillerine daha yakın ve hızlı geliştirme yapmaları için imkân tanınmıştır. Programlama dillerinde yazılan komutlar, direkt olarak bilgisayarın anlayabileceği komut yazma işlemlerini derleyici veya yorumlayıcılara bırakmışlardır

Doğal Diller ve Programlama Dilleri

C, Java, Python ve Pascal gibi programlama dilleri çıkıp, insan diline benzer yapıda yazılım geliştirme olanağı sağlayınca, yazılım geliştirme işi kolaylaşmış ve yazılımcı sayısında da artış görülmüştür.

Python dili de İngilizce düşünen ve konuşan insanlar için neredeyse biriyle konuşuyormuşçasına yazılım geliştirme imkânı tanımıştır.

Doğal Diller ve Programlama Dilleri

Makine dilinde 0 ve 1'lerin herhangi bir işleme tabi tutulmadan işlem yapılması, muhtemelen çok zor bir süreç olurdu.

Makine dili; merkezi işlem birimi (CPU) tarafından anlaşılıp çalıştırılabilen komutlardan oluşan program yazma aracıdır. Makine dili üzerinden herhangi bir işlem ya da dönüştürmeye gerek kalmadan bilgisayar tarafından doğrudan anlaşılır. Bu dil, sadece 0 ve 1 sayılarından oluşan, ikili kodların anlamlı kombinasyonlarından meydana gelmektedir.

Makine dili insanlar tarafından oluşturulmuştur. Ancak 0 ve 1'lere bakarak bir çıkarımda bulunmak zordur. Bilgisayara verilen komutların bilgisayar tarafından anlaşılması için bir dizi işlemden geçmesi gerekmektedir

Derleyici ve Yorumlayıcılar

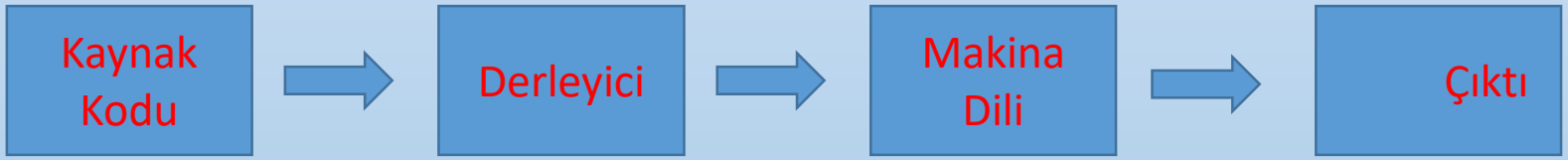
Makine dili, düşük seviyeli bir programlama dilidir ve bu programlama dili ile bilgisayara komutlar vermek, işlem yaptırmak zordur. Kullanıcıların kolaylıkla program yazmaları için yüksek seviyeli programlama dillerini kullanmaları gerekir.

Yüksek seviyeli dil ile yazılmış olan programa kaynak kodu denir. Kaynak kodunun bilgisayar tarafından anlamlandırılabilmesi için makine koduna dönüştürülmesi gerekir. İşte, bu iş ya derleyici (compiler) ya da yorumlayıcı (interpreter) tarafından gerçekleştirilir.

Derleyici ve Yorumlayıcılar

Derleyici (compiler), belirli bir programlama dilinde yazılmış ifadeleri işleyen ve bunları bir bilgisayar işlemcisinin kullandığı makine diline veya “koda” dönüştüren özel bir programdır.

Derleyici, kaynak kodu derlemeden önce kodları kontrol eder. Olası yazım hataları veya dilin yapısına uygun olmayan kalıplar varsa kullanıcıya çıktı olarak bildirir. Bu sayede kullanıcı, compile işlemi esnasında hataları ayıklar.



Derleyici işlem adımları

Derleyici ve Yorumlayıcılar

Yorumlayıcılar (interpreter), programı okuma ve yürütme yeteneğine sahip sistem yazılımı olarak adlandırılabilir.

Yorumlayıcı programı satır satır yorumlar, bu işlem yukarıdan aşağı doğru yapılır ve kullanıcıya bildirilir. Daha alt satırlarda hata varsa bu hatalar bulunamaz çünkü satır satır işlem yapılmaktadır.

Derleyici – Yorumlayıcı Farkı

Derleyici	Yorumlayıcı
❑ Tüm programı bir bütün olarak makine diline çevirir. ❑ Makine diline çevrilmiş bir programın çalışması daha hızlıdır. ❑ Daha fazla bellek gerektirir. ❑ Tüm hatalar belirlenerek bir seferde çıktı verir. ❑ Hata ayıklamak zordur. ❑ C, C++ gibi diller derleyici kullanır. ❑ Exe dosyaları derlenmiş uygulamaya örnek olarak verilebilir	❑ Programın bazen tek satırını bazen tek ifadesini alıp işler, satır satır işlem görür. ❑ Daha yavaştır. ❑ Hafızada fazla yer kaplamaz. ❑ Gördüğü ilk hatada çalışmayı durdurur. Hata ayıklamak daha kolaydır. ❑ Python, Ruby, Java dilleri yorumlayıcı kullanır.

Python Nedir?

Python, 90'lı yılların başında Amsterdam'da Guido Van Rossum tarafından geliştirilmeye başlanan bir programlama dilidir.

Python yazılım geliştirme ve veri analizinde ön plana çıkmıştır.

Python'ın standart kütüphanesi; geliştirme araçları diğer birçok kütüphanesi açık kaynak kod olarak ücretsiz şekilde indirilebilmektedir.

Python nesne yönelimli, yorumlanabilen ve yüksek seviyeli bir programlama dilidir.

Neden Python?

Python, popülerliğini hızlı bir şekilde yükseltmektedir. İlk defa programlamaya başlangıç yapanlar ya da farklı dillerde uzmanlaşanlar “Python öğrenmeli miyim?” sorusunu sormaktadır.

Uzun yıllar önce programlama öğrenmek isteyenler için bir programlama dilini öğrenmek veya aşina olmak şimdikinden çok daha zordu. Ancak zamanla insan diline yakın denilebilecek yüksek seviyeli programlama dilleri ortaya çıkmıştır

Neden Python?

Python kodlarını yorumlamak ve öğrenmek diğer dillere göre daha kolaydır. Diğer dillerde bulunan noktalama işareti zorunlulukları, parantezler veya kurallar programlamaya yeni başlayan kullanıcı için zaman zaman zorluklar çıkarmaktadır. Ancak Python'da bu tür zorunluluklar olmadığı gibi yapısı itibariyle diğer dillere göre daha sadedir.

Ayrıca Python, yıllar içerisinde belirli bir olgunluğa, geliştirici topluluğuna ve öğretici dokümana sahip olmuştur.

Neden Python?

Python dili ile program geliştirirken karşılaşmanızın muhtemel olduğu birçok sorunun cevabı stackoverflow gibi sitelerde bulunmaktadır. Bu da Python'u öğrenirken hızlı ilerlemenize olanak sağlamaktadır.

Python, web geliştirme uygulamalarında kullanılabileceği gibi, kullanıcı ara yüzüne sahip (GUI) PyQt gibi kütüphanelere de sahiptir.

Python'un veri bilimi, veri analizi ve yapay zekâ ile ilgili gelişmiş kütüphaneleri olduğu için bu alanda ilerlemek isteyen kişilerin en çok kullandığı programlama dillerinin başında Python gelmektedir.

Neden Python?

- Büyük kuruluşlar (Google, YouTube ve Yahoo! gibi) her zaman Python programcılarına ihtiyaç duymaktadır.
- Python ile masaüstü, oyun, mobil, web ve ağ alanlarında programlar yazılabilmektedir.
- Python kodları sade, basit ve hızlıdır. Derlenmeye ihtiyaç duymaz.
- Python farklı işletim sistemleri üzerinde çalışabilir. Linux, Windows, Mac OS X, MS-DOS, iOS ve Android vbs…

EDİTÖRLER

Editör, Kod yazmak, saklamak, düzenlemek ve geliştirmek için kullandığımız programlardır.

IDE (Integrated Development Environment), Türkçe ifadesi ile Tümleşik Geliştirme Ortamı, bilgisayar yazılımcılarının daha kolay şekilde yazılım geliştirebilmesi için tasarlanan ve yazılım geliştirme aşamasında geliştiriciye birçok kullanışlı araç sunarak daha kolay ve etkili şekilde yazılım geliştirmesine yardımcı olan yazılımlardır.

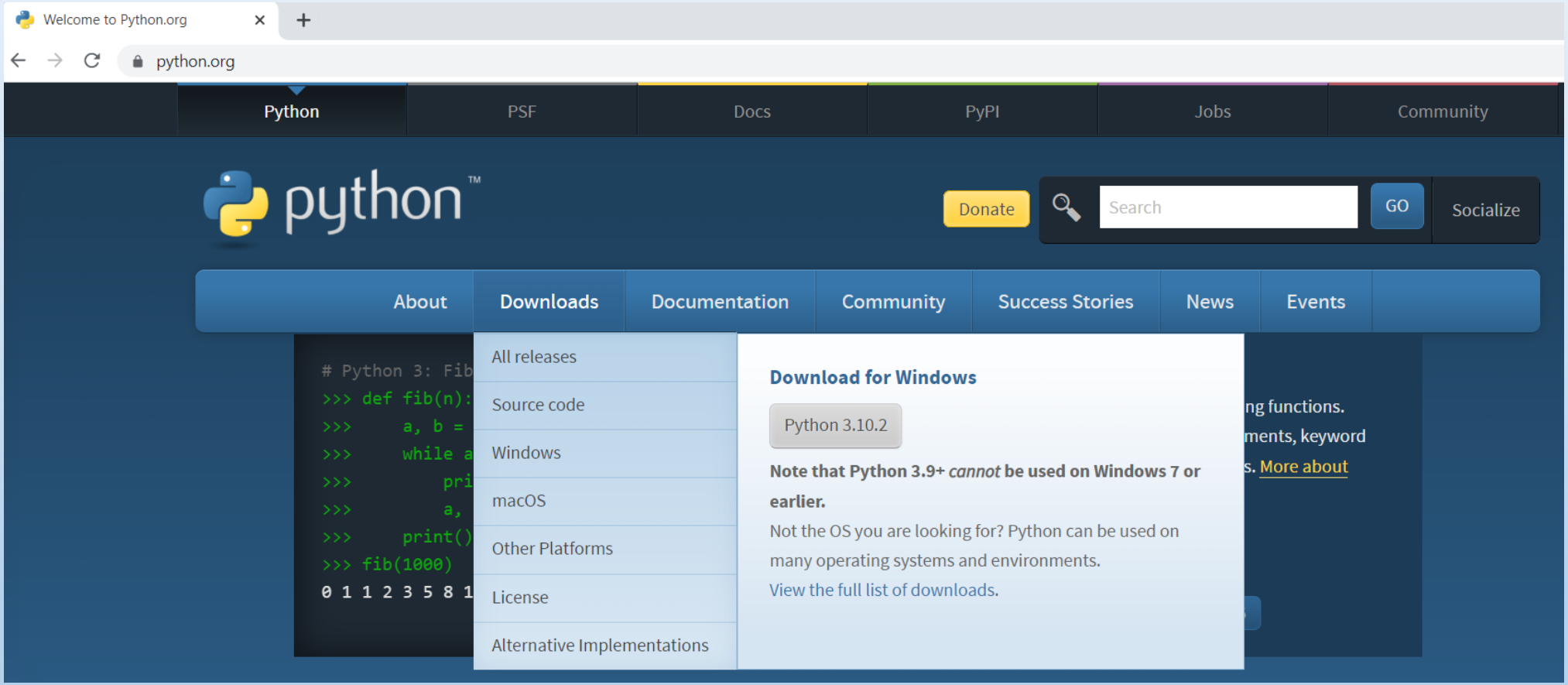
Python İçin Hangi Editörleri Kullanabiliriz?

- Python IDLE
- Wing IDE:
- Visual Studio Code
- Pycharm
- anaconda
- jupyter notebook
- ipython
- orange
- spyder
- Pydev
- komodo

…

GEREKLİ YAZILIMLARI İNDİRMEK

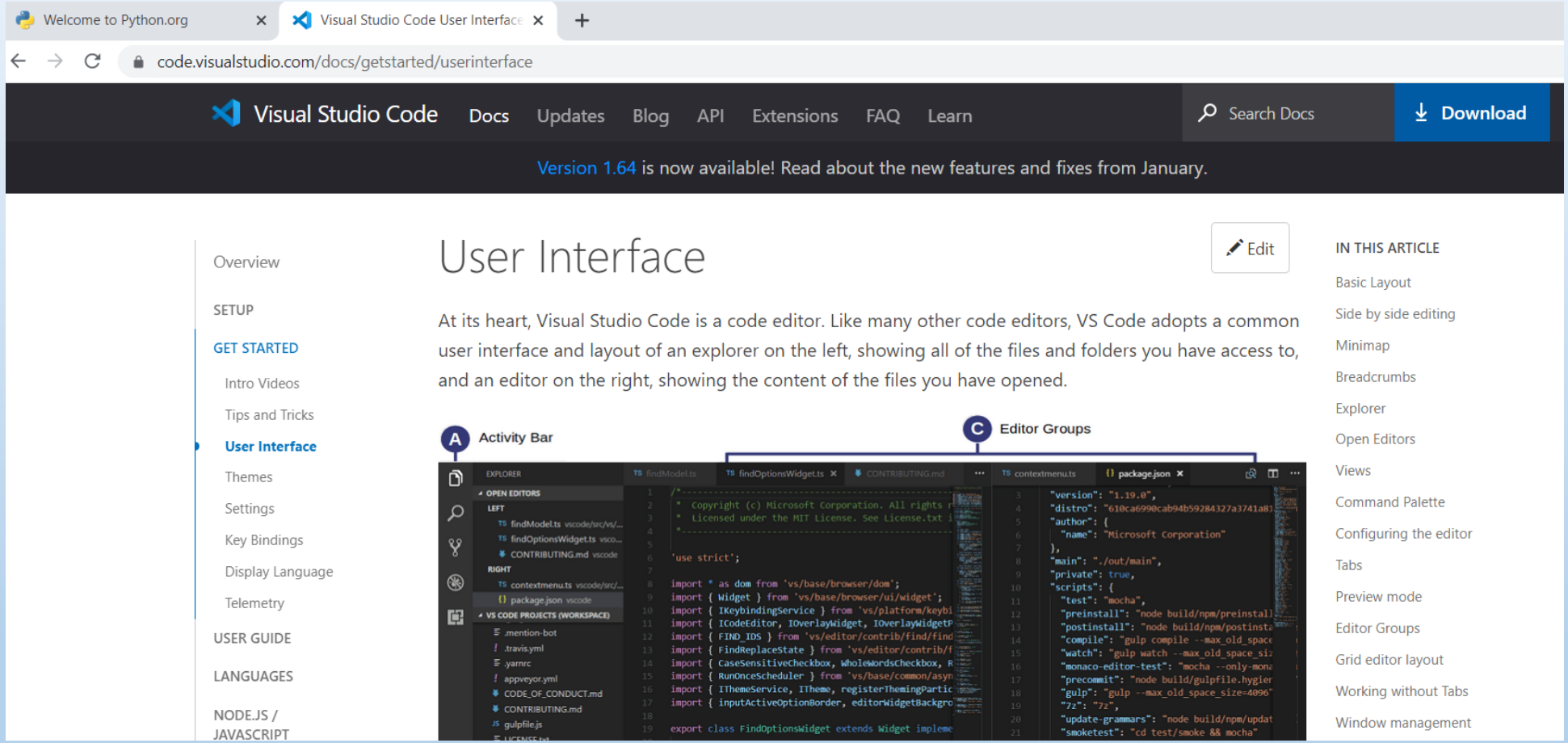
PYTHON PROGRAMININ İNDİRİLMESİ:



https://www.python.org/ sayfasından Downloads menüsünün üzerine gelerek bilgisayarınızda kurulu işletim sistemine uygun Python sürümünü görüntülenir ve butona tıklayarak indirmeyi başlatabilirsiniz.

GEREKLİ YAZILIMLARI İNDİRMEK

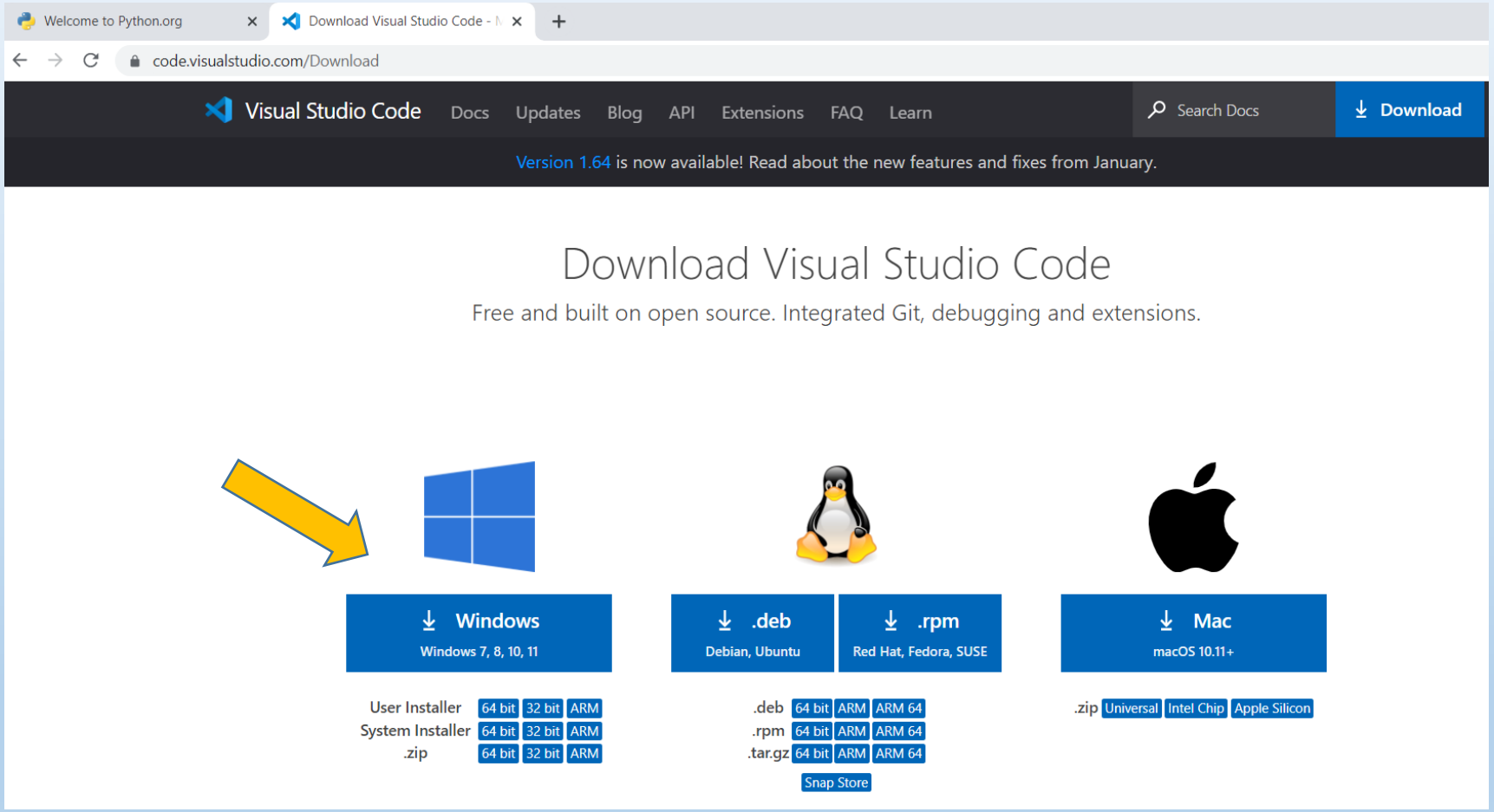
VISUAL STUDIO CODE PROGRAMININ İNDİRİLMESİ:



https://code.visualstudio.com/docs/getstarted/userinterface linkinden Download butonuna tıklayınız.

GEREKLİ YAZILIMLARI İNDİRMEK

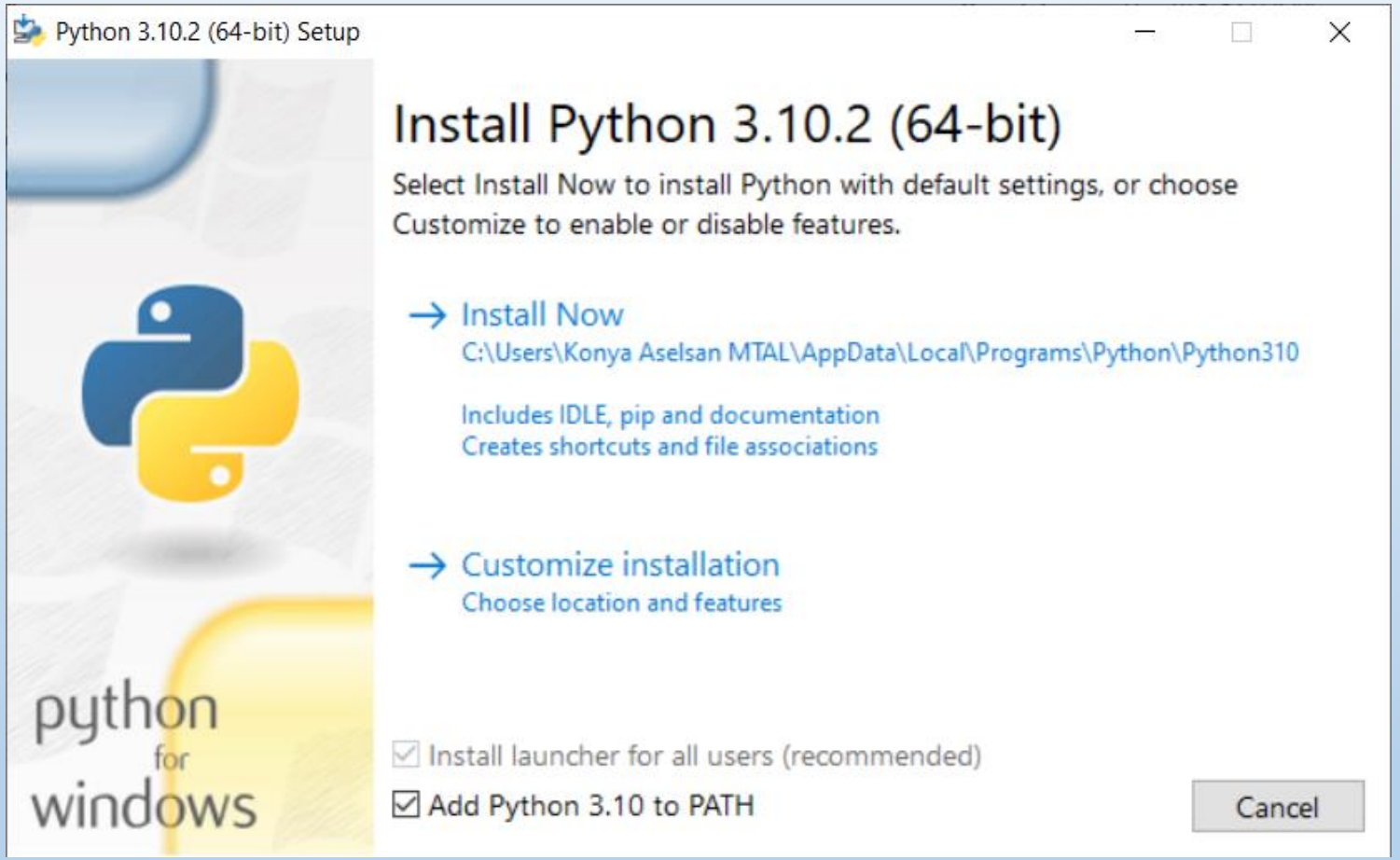
VISUAL STUDIO CODE PROGRAMININ İNDİRİLMESİ:



İşletim sisteminize uygun olan programı seçiniz. Örneğin Windows İşletim Sistemi kullananlar Windows butonuna basarak programın indirilmesini başlatabilir...

PROGRAMLARIN KURULMASI

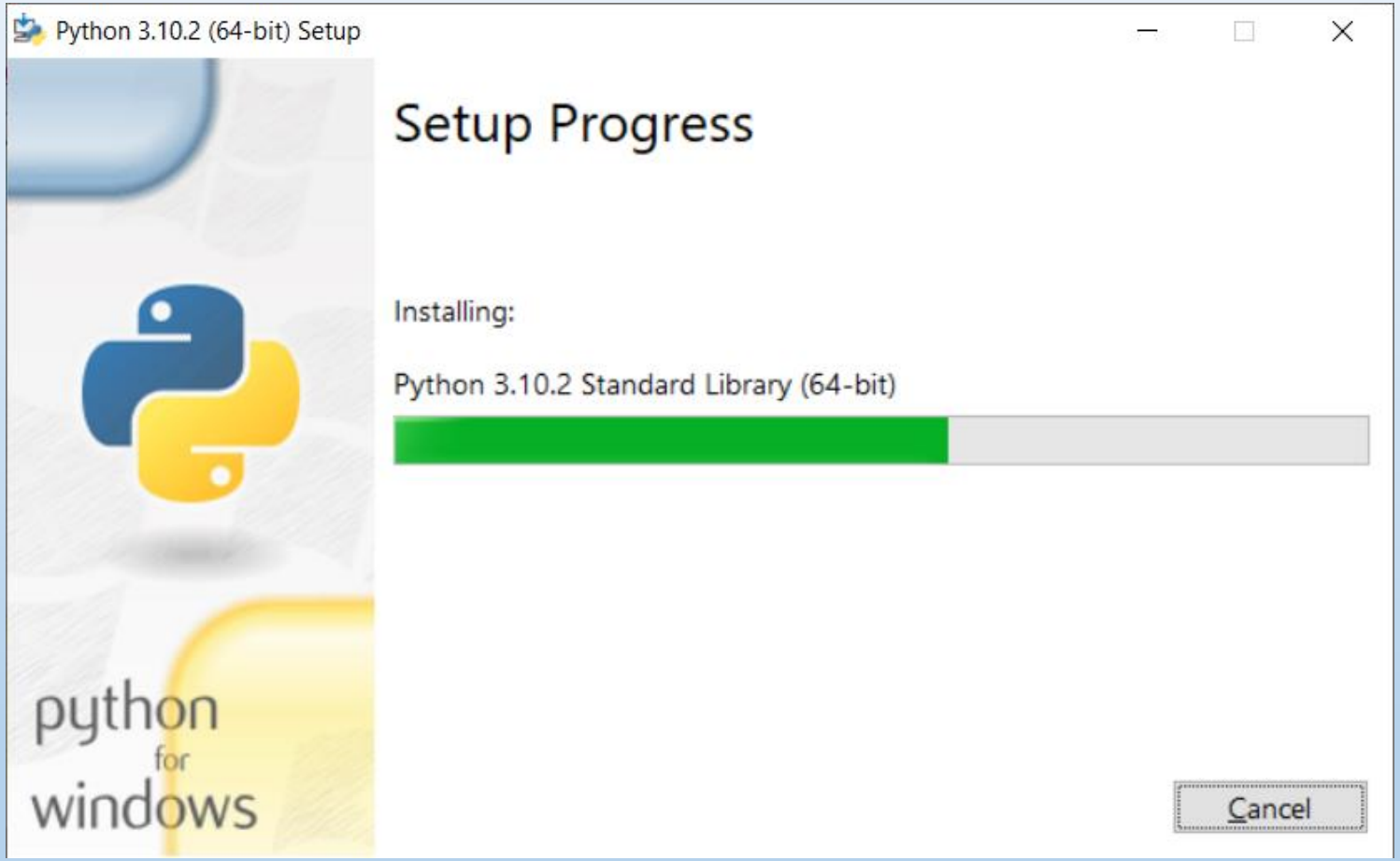
PYTHON PROGRAMININ KURULMASI:



Pencerede görüldüğü gibi Add Python 3.10 PATH kutucuğunu da işaretledikten sonra Install Now butonuna basınız.

PROGRAMLARIN KURULMASI

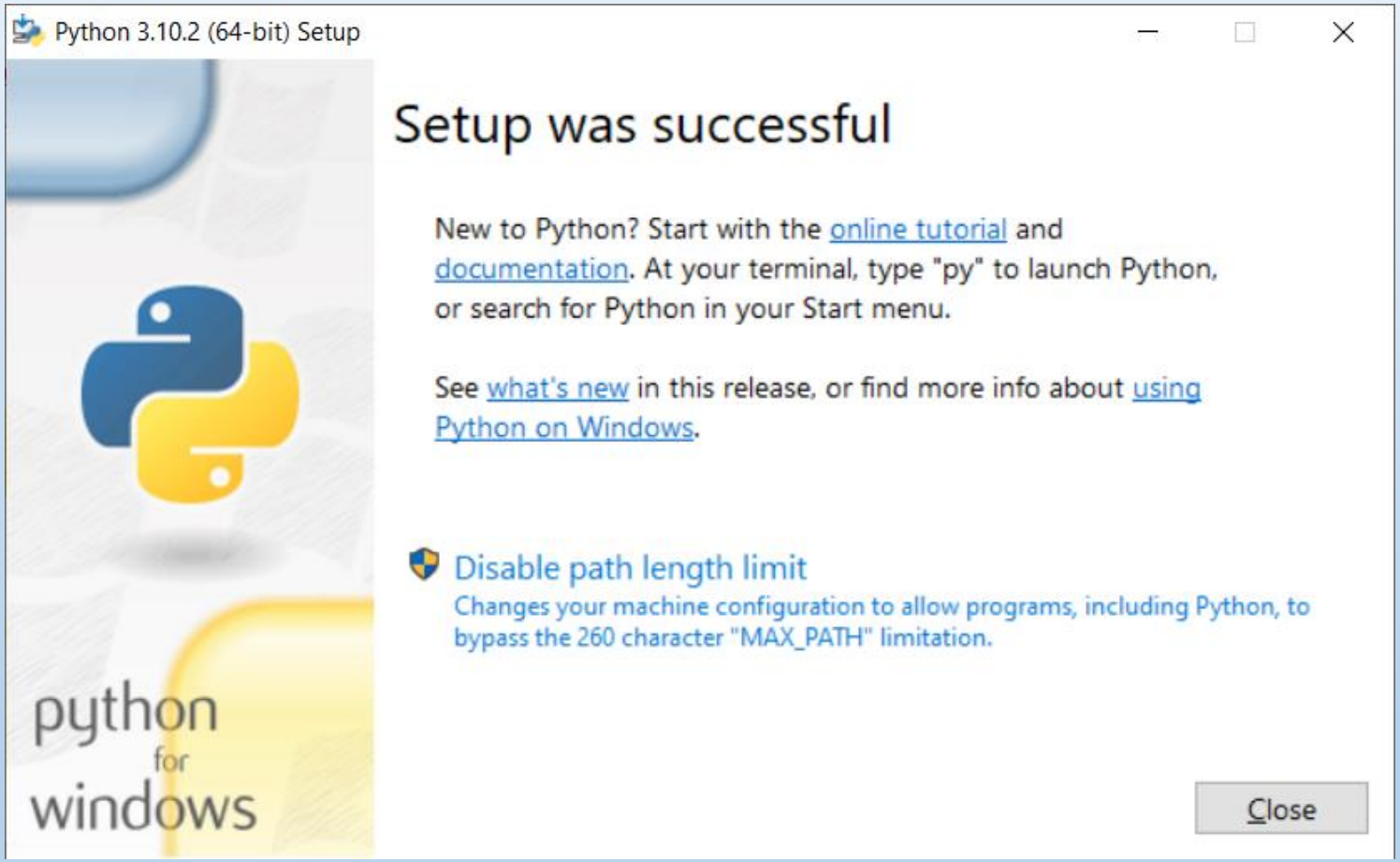
PYTHON PROGRAMININ KURULMASI:



Kurulum işlemine devam edecek..

PROGRAMLARIN KURULMASI

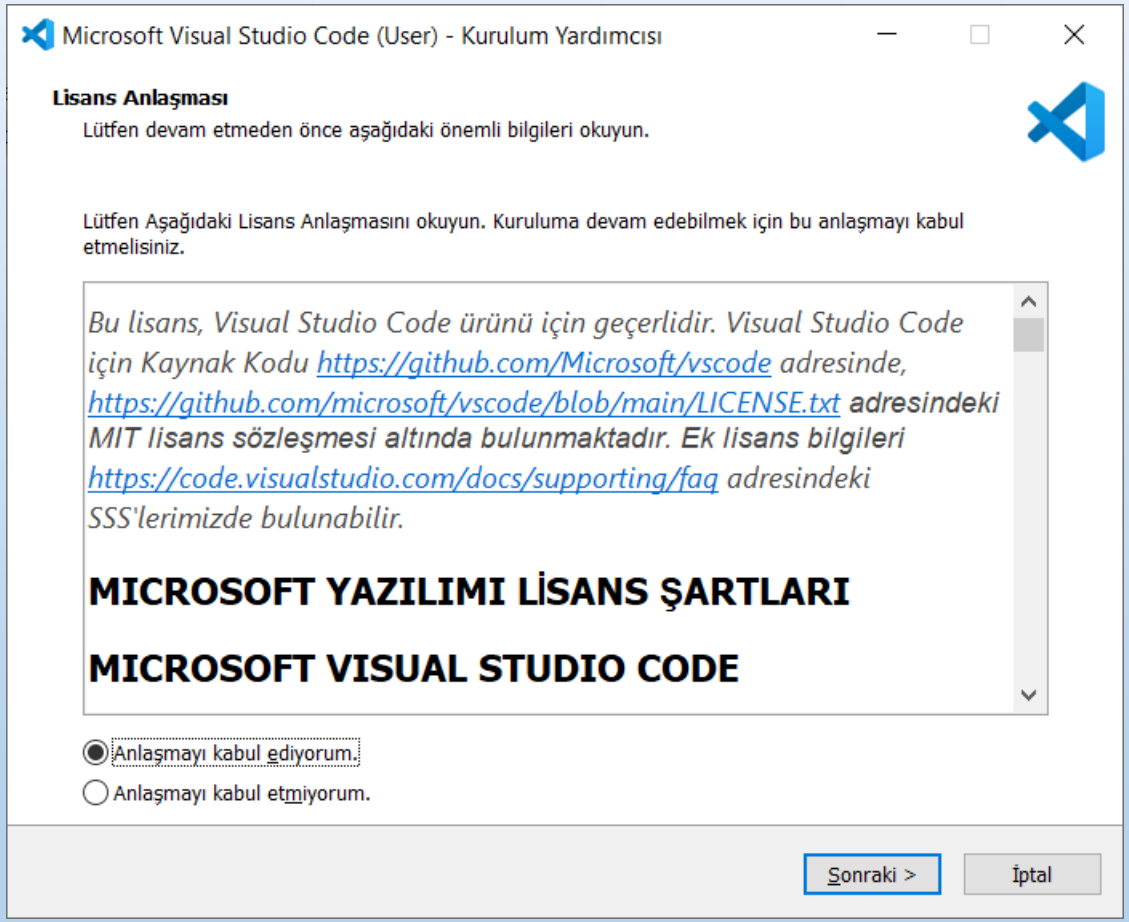
PYTHON PROGRAMININ KURULMASI:



Setup was successful mesajını aldığımızda kurulum işlemi tamamlanmış olacaktır. Close butonuna basarak pencereyi kapatınız.

PROGRAMLARIN KURULMAI

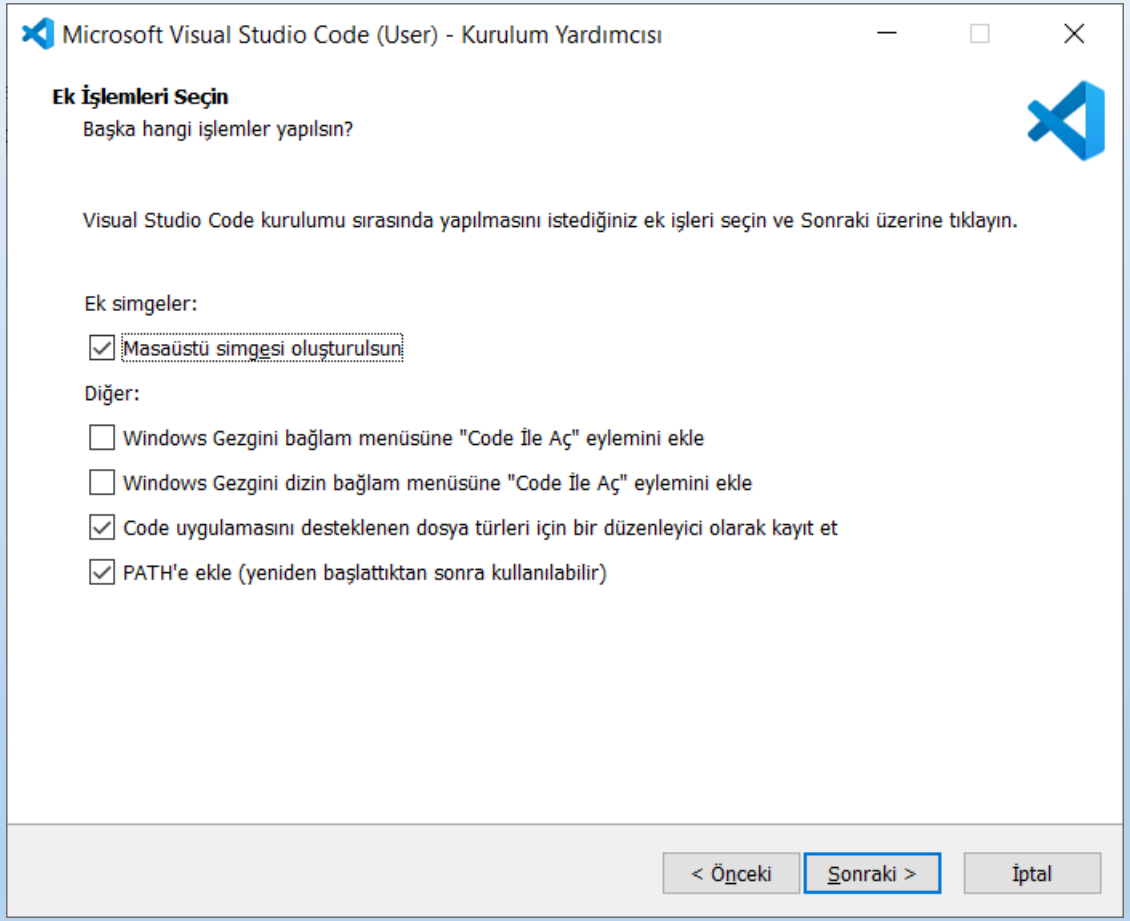
VISUAL STUDIO CODE PROGRAMININ KURULMASI:



Daha önce indirdiğimiz VSCodeUserSetup-x64-1.64.0 kurulum dosyasına tıklayınız ve Anlaşmayı kabul ediyorum seçeneğini işaretleyiniz. Ardından Sonraki butonuna basınız.

PROGRAMLARIN KURULMASI

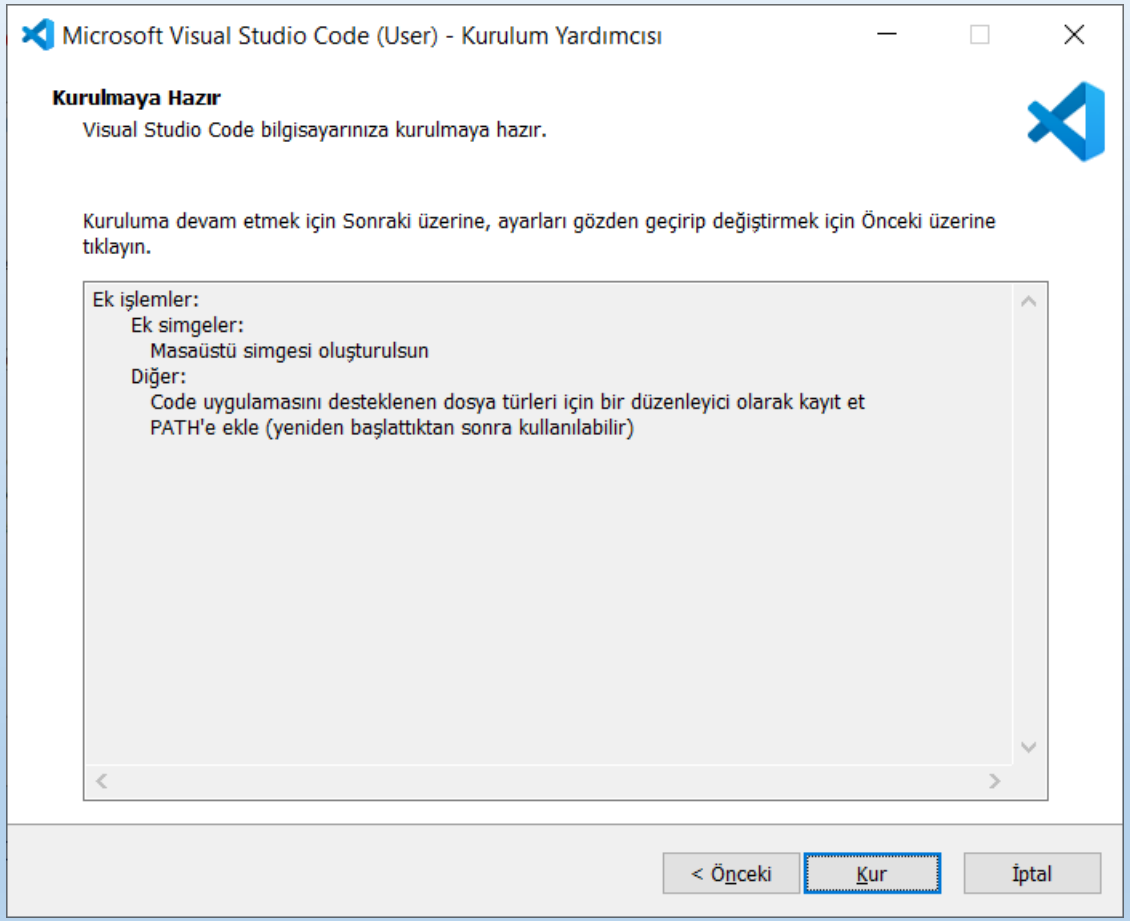
VISUAL STUDIO CODE PROGRAMININ KURULMASI:



Penceredeki kutucukları işaretleyip Sonraki butonuna basınız.

PROGRAMLARIN KURULMASI

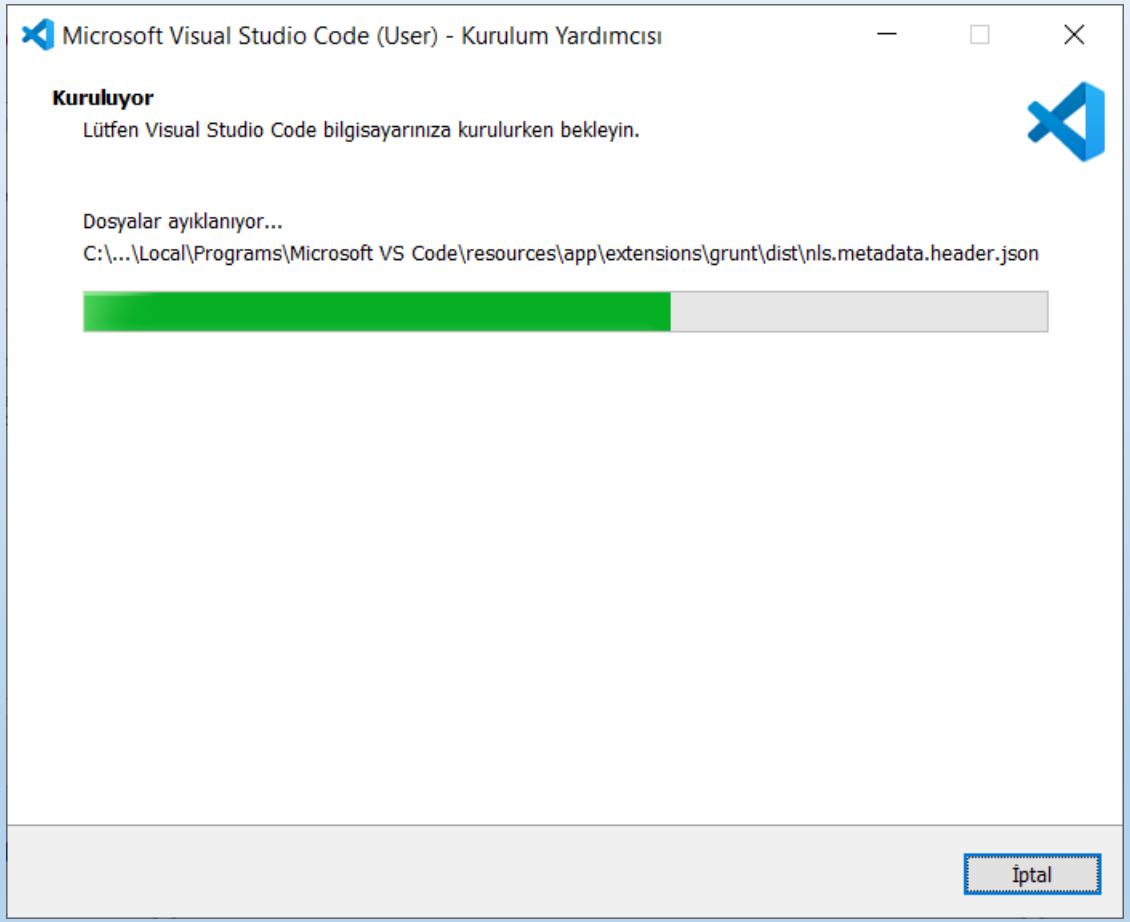
VISUAL STUDIO CODE PROGRAMININ KURULMASI:



Kur butonuna basıp kurulum işlemini başlatınız...

PROGRAMLARIN KURULMASI

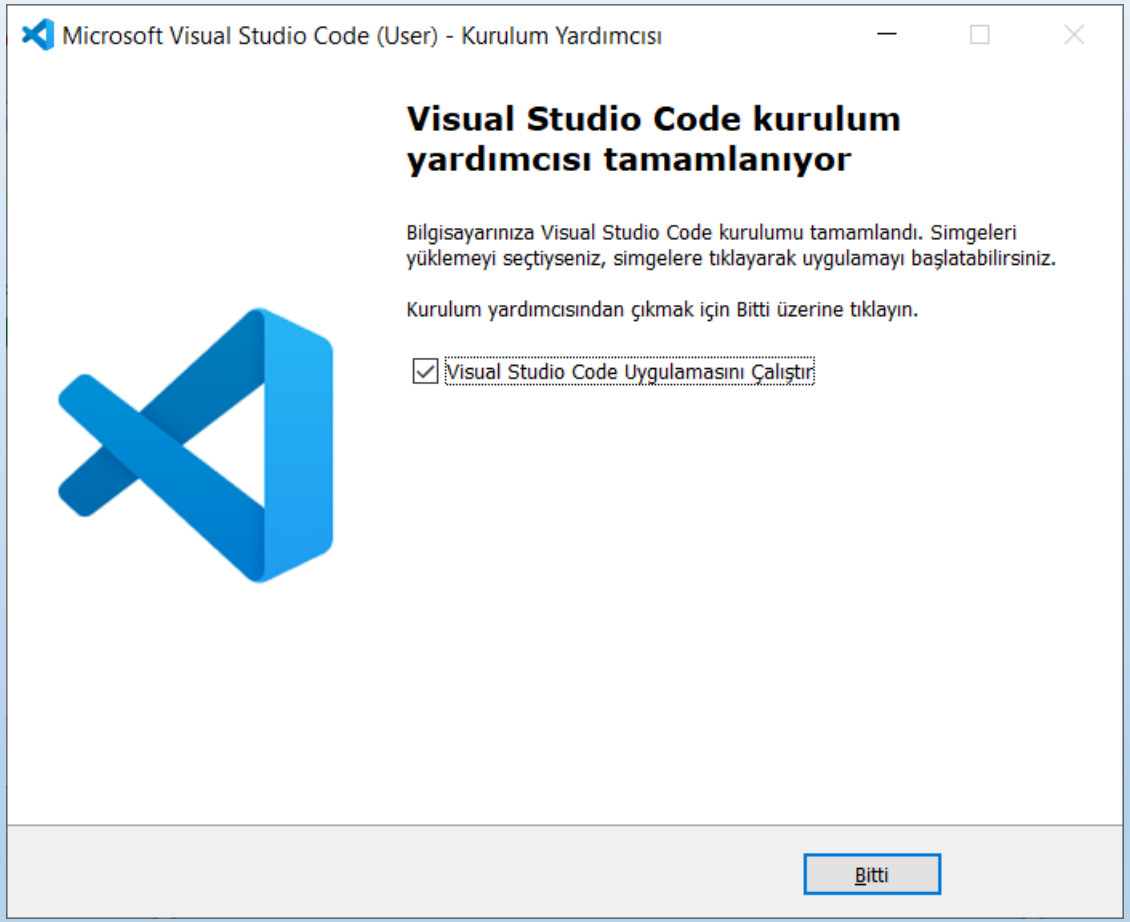
VISUAL STUDIO CODE PROGRAMININ KURULMASI:



Kurulum işlemi devam ediyor…

PROGRAMLARIN KURULMASI

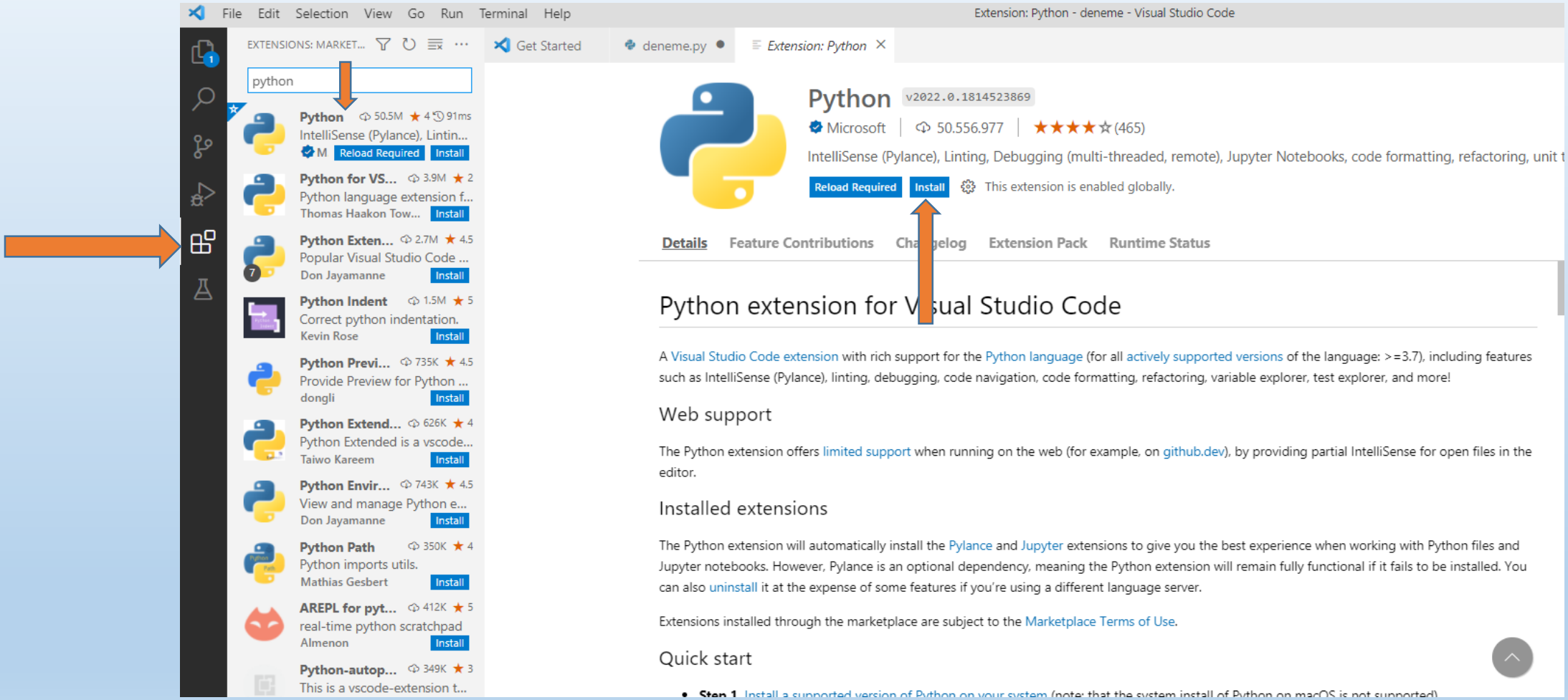
VISUAL STUDIO CODE PROGRAMININ KURULMASI:



Bitti butonuna basarak kurulum işlemini tamamlayınız…

PROGRAMLARIN KURULMASI

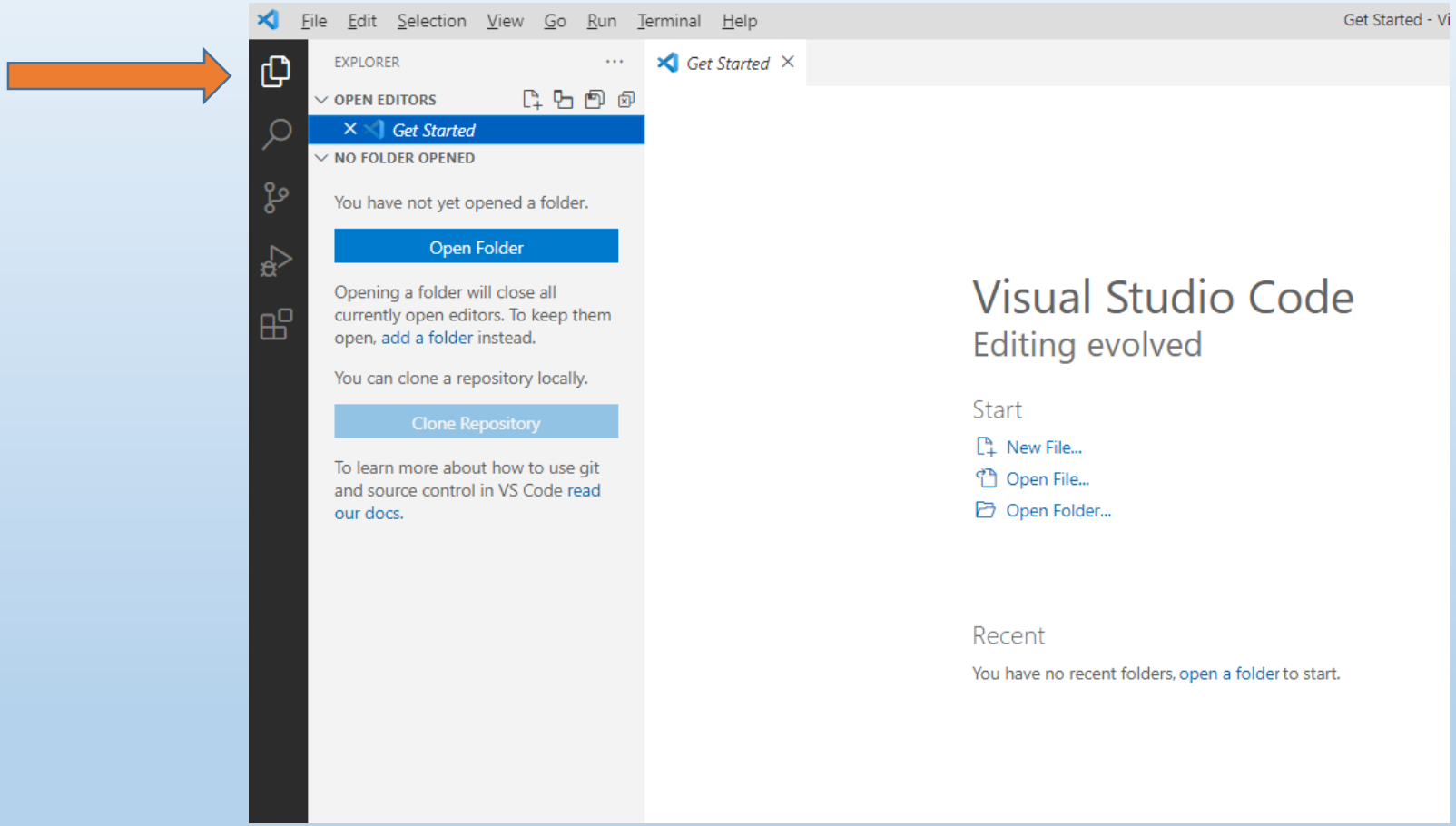
VISUAL STUDIO CODE PROGRAMININ YAPILANDIRILMASI:



Ok ile gösterilen Extension butonuna basarak arama bölümüne Python yazarak, gelen programı tıklayarak Install butonuna basınız ve Visual Studio Code programında Python Kodlama ayarını tamamlayınız.

PROGRAMLARIN KURULMASI

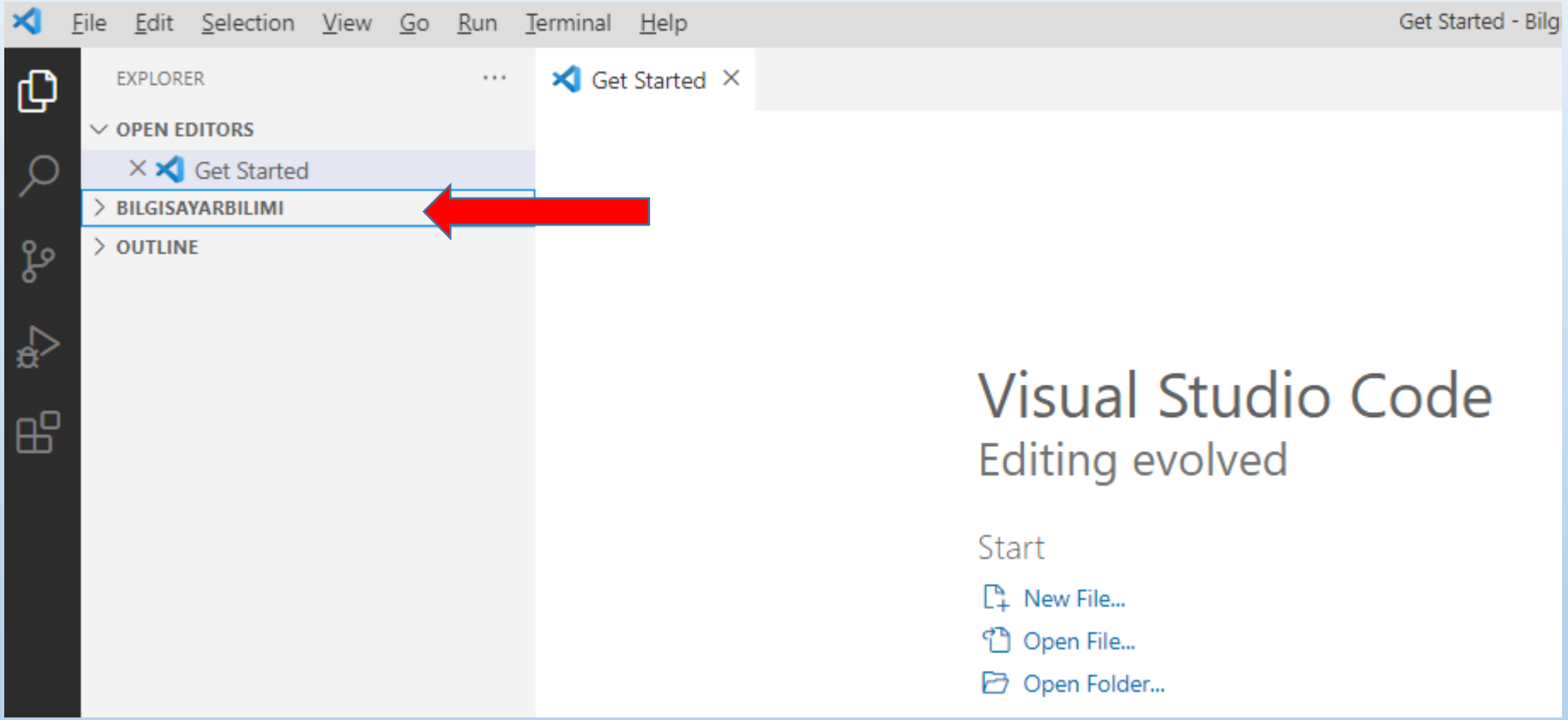
VISUAL STUDIO CODE PROGRAMINDA ÇALIŞMA BAŞLATMAK :



Sol Üst Menüde yer alan Explorer butonuna basarak çalışmalarınızı kaydedeceğiniz daha önce oluşturduğunuz klasörü Open Folder butonu ile yolunu işaretleyerek seçiniz..

PROGRAMLARIN KURULMASI

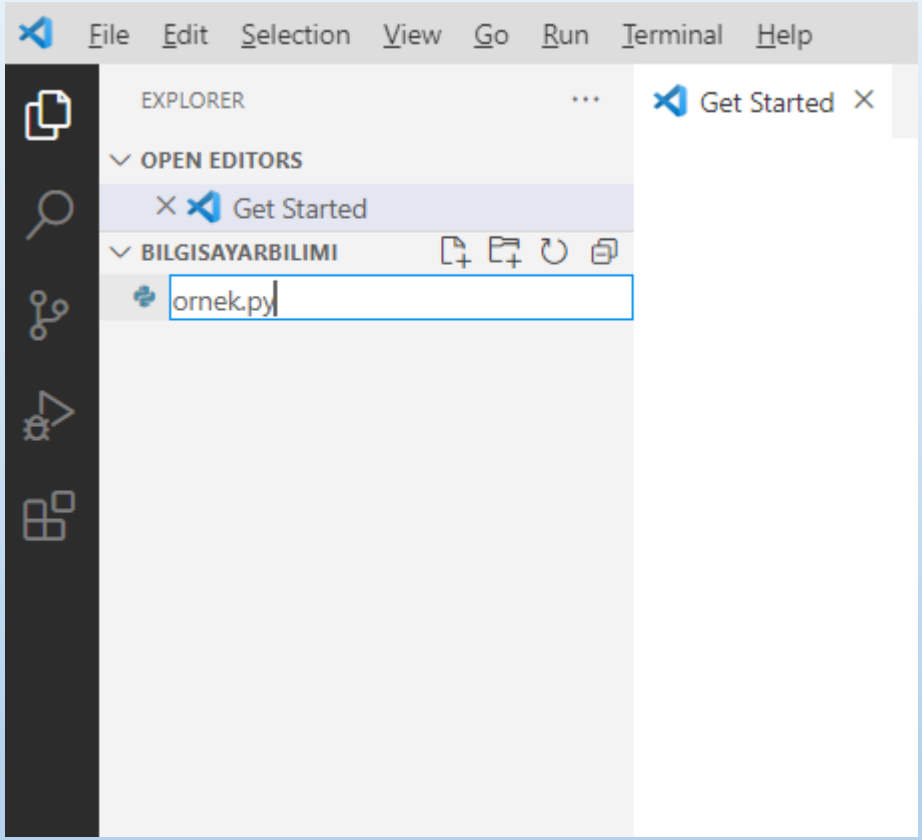
VISUAL STUDIO CODE PROGRAMINDA ÇALIŞMA BAŞLATMAK :



Seçilen klasör görselde ok işaretinin gösterdiği şekli ile görüntülenecektir.

PROGRAMLARIN KURULMASI

VISUAL STUDIO CODE PROGRAMINDA YENİ BİR PYTHON DOSYASI OLUŞTURMAK:



Daha önce oluşturduğumuz BILGISAYARBILIMI klasörüne tıkladığımızda hemen sol tarafında yer alan bölümdeki New File ikonuna tıklayarak açılan pencerede oluşturacağımız python dosyasının ismini veriniz.
Örnek: ornek.py
.py python dosyasının uzantısıdır.

print() Fonksiyonu

print() fonksiyonu ekrana ekrana çıktı almak için kullanılan bir fonksiyondur.

KULLANIMI:

Print() fonksiyon'nun dört farklı kullanımı var;

- Tek tırnak (‘ ‘)
- Çift tırnak (” “)
- Üç çift tırnak (“”” “””)
- Üç tek tırnak (‘’’ ‘’’)

print() Fonksiyonu

ÖRNEK:

```
print("Aselsan Konya MTAL")
print('Aselsan Konya MTAL')
print("""Aselsan Konya MTAL""")
print(‘‘‘Aselsan Konya MTAL’’’)
```

EKRAN ÇIKTISI:

Aselsan Konya MTAL

Aselsan Konya MTAL

Aselsan Konya MTAL

Aselsan Konya MTAL

print() Fonksiyonu

print() fonksiyonunun bu üç farklı kullanımının olması, ekrana yazdırılmak istenilen bilgiler içerisinde tek tırnak ('') veya çift tırnak ("") sembollerinin kullanılmasına ihtiyaç duyulması gerektiğinde diğer yöntemlerin kullanılması ile problemin amaçlanmıştır.

ÖRNEK:

```
#Tek tırnak arasına yazıldığında kesme işareti ile karışacaktır
print('Etliekmek Konya'nın meşhur yemeğidir. ') #Yanlış
print("Etliekmek Konya'nın meşhur yemeğidir.") #Doğru
```

ÖRNEK:

```
#Çift tırnak arasına yazıldığında alıntı anlamına gelen çift tırnak ile karışacaktır
print("Sabah annemin "Uyan okula geç kalacaksın" sesi ile uyandım") #Yanlış
print("""Sabah annemin sabah "Uyan okula geç kalacaksın" sesi ile uyandım""")#Doğru
print('Sabah annemin sabah "Uyan okula geç kalacaksın" sesi ile uyandım') #Doğru
```

print() Fonksiyonu

print() fonksiyonu, kendisine parametre olarak verilen aritmetiksel ve mantıksal işlemlerin sonuçlarını ekrana yazdırır.

ÖRNEK:

```
print(2+3)
print(6*7)
print(12/4)
print(2+15/3*4-5)
print(5>2)
print(7<5)
```

EKRAN ÇIKTISI:

```
5
42
3.0
17.0
True
False
```

print() Fonksiyonu

print() fonksiyonunun kullanımında + sembolü metin birleştirme işlemi yapar.

ÖRNEK:

```
print("Salih"+"Kul")
```

EKRAN ÇIKTISI:

SalihKul

print() Fonksiyonu

print() fonksiyonu argümansız olarak kullanıldığında boş bir satırın açılmasını sağlar.

ÖRNEK:

```
print("Salih")
print()
print("Kul")
```

EKRAN ÇIKTISI:

Salih

Kul

print() Fonksiyonu

print() fonksiyonunun kullanımında + sembolü metin birleştirme işlemi yapar.

ÖRNEK:

```
adi="Salih"
soyadi="Kul"
dogum_yeri="Konya"
print(adi+" "+soyadi+"'un doğum yeri "+dogum_yeri+"'dır...")
```

EKRAN ÇIKTISI:

Salih Kul'un doğum yeri Konya'dır...

print() Fonksiyonu

print() fonksiyonunun kullanımında virgül (,) sembolü metin birleştirme işlemi yapar.

ÖRNEK:

```
print("Salih","Kul")
```

EKRAN ÇIKTISI:

Salih Kul

print() Fonksiyonu

print() fonksiyonunun kullanımında virgül (,) sembolü metin birleştirme işlemi yapar.

ÖRNEK:

```
adi="Salih"
soyadi="Kul"
yasi=42
print(adi," ",soyadi," ",yasi," yaşında")
```

EKRAN ÇIKTISI:

Salih Kul 42 yaşında

print() Fonksiyonu

print() fonksiyonunun kullanımında virgül (,) ve (+) sembolü kullanımında, string bir bilgi ile sayısal bir bilgi birleştirilerek yazılmak istenildiğinde virgül (,) kullanılmalıdır. (+) birleştirme sembolü kullanılmak istenildiğinde program sayısal bir bilgi ile string bir bilginin aritmetiksel bir işleme tabi tutulmaya çalışıldığını yorumlar ve hata verir.

ÖRNEK:

```
adi="Salih"
soyadi="Kul"
yasi=42
print(adi+" "+soyadi+" "+yasi+" yaşında")
#string bir veri ile sayısal bir veri
(+) sembolü ile birleştirilemez.
```

EKRAN ÇIKTISI:

TypeError: can only concatenate str (not "int") to str

print() Fonksiyonu

print() fonksiyonunun kullanımında virgül (,) ve (+) sembolü kullanımında, string bir bilgi ile sayısal bir bilgi birleştirilerek yazılmak istenildiğinde virgül (,) kullanılmalıdır. (+) birleştirme sembolü kullanılmak istenildiğinde program sayısal bir bilgi ile string bir bilginin aritmetiksel bir işleme tabi tutulmaya çalışıldığını yorumlar ve hata verir.

ÖRNEK:

```
adi="Salih"
soyadi="Kul"
yasi=42
print(adi," ",soyadi," ",yasi," yaşında")
#string bir veri ile sayısal bir veri
(+) sembolü ile birleştirilemez.
```

EKRAN ÇIKTISI:

Salih Kul 42 yaşında

print() Fonksiyonu

String (metinsel) bir bilgi string (metinsel) bir bilgi ile (+) veya (,) sembolü ile birleştirilebilir. Fakat string (metinsel) bir bilgi sayısal bir bilgi ile veya sayısal bir bilgi string (mestinsel) bir bilgi ile sadece (,) sembolü ile birleştirilebilir.

ÖRNEK:

```
adi="Salih"
soyadi="Kul"
yasi=42
print(adi+" "+soyadi+" ",yasi," yaşında")
```

EKRAN ÇIKTISI:

Salih Kul 42 yaşında

print() Fonksiyonu

\n parametresi:Bu parametreye newline adı verilir. print() fonksiyonu içerisinde kullanıldığında ilgili yerden bir alt satıra geçiş yapılmasını sağlar.

ÖRNEK:

```
adi="Salih"
soyadi="Kul"
print("Öğrencinin:","\nAdı:",adi,"\nSoyadi:",soyadi)
```

EKRAN ÇIKTISI:

Öğrencinin:

Adı: Salih

Soyadi: Kul

print() Fonksiyonu

\t parametresi: print() fonksiyonu içerisinde kullanıldığında ilgili yerden bir tab kadar boşluk bırakır.

ÖRNEK:

```
print("Numara\tAdi\tSoyadi")
print("42\tSalih\tKul")
```

EKRAN ÇIKTISI:

```
Numara  Adi     Soyadi
42      Salih   Kul
```

YORUM SATIRLARI KULLANMA

Yorum satırları, Python yorumlayıcısı tarafından dikkate alınmayan ve yorumlanmayan ifadelerdir.

Python'da yorum satırları genel olarak aşağıdaki işlemler için kullanılır:

- Bir hatırlatma eklemek Program veya kodlarla ilgili bir açıklama yapmak
- Kullanılmayan bir kod satırını pasif hale getirmek
- Süsleme yapmak

Bu tür açıklama satırları kodun başkaları tarafından daha iyi anlaşılmasını sağlar.

YORUM SATIRLARI KULLANMA

Python'da tek satırlık açıklama için "#" işareti kullanılır. "#" işareti kullandığınızda o satırdaki metin kod olarak işlenmez.

Birden fazla yorum satırı kullanılacaksa yorumlar üçlü tek tırnak(''' ''')veya **üçlü çift tırnak** (""" """)blokları arasına yazılır.

ÖRNEK:

```
#bu satır yorum satırıdır.
print("Yorum satırı kullanımı")
"""
Üç çift tırnak arası yorum satırıdır
"""
print ("Yorum satırı kullanımı")
'''
Üç tek tırnak arası da yorum satırıdır
'''
print("Yorum satırı kullanımı")
```

EKRAN ÇIKTISI:

```
Yorum satırı kullanımı
Yorum satırı kullanımı
Yorum satırı kullanımı
```

DEĞİŞKEN İSİMLENDİRME KURALLARI:

- Değişkenler bir harf (a - z, A - Z) veya alt çizgi (_) ile başlamalıdır. Bunların dışında sayı veya başka bir karakter ile de başlayamaz.
- Değişken isminde Değişken isminde rakam, alt çizgi(_), büyük veya küçük harf kullanılabilir, özel karakterler kullanılamaz (+,-*/?^!=)({&% vb)
- Python Türkçe karakter setini kullanmaya izin verse de İngilizce karakter setinin kullanılması tavsiye olunur.
- Programlama dilindeki özel komut ve deyimler değişken ismi olarak verilemez.
- Küçük Büyük harf duyarlılığı olduğu göz önünde bulundurulmalıdır.

Sayi<>sayi ----→Sayi değişkeni sayi değişkeni ile aynı değildir.

DEĞİŞKEN İSİMLENDİRMESİ İÇİN STANDARTLAR

- ✓Değişken adlandırma için bazı standartlar vardır. Bu standartlar değişken adının ve içeriğinin anlaşılmasına yardımcı olarak programcıların daha kolay çalışmasını sağlar. Değişken adı, onun içeriği hakkında bilgi verirse kodun anlaşılması kolaylaşır.
- ✓Değişken adlarının yazımında bazı standart kullanımlar vardır. Birden fazla kelimenin kullanılacağı değişken adlarında kelimelerin ilk harfi büyük olabilir. Camel standardında (başka standartlar da bulunmaktadır) değişken adlarının görünüşü deve hörgücüne benzetilmektedir. Değişkenin adına küçük harfle başlanır ve sonraki her kelime büyük harfle başlar

ÖRNEK

```
sayi1=5
print('Değişkenin içindeki sayı: ', sayi1)
sayi1=10
print('Değişkenin içindeki sayı: ', sayi1, 'oldu')
sayi1=10.5
print ('Değişkenin içindeki sayı: ', sayi1, 'oldu')
```

EKRAN ÇIKTISI

```
Değişkenin içindeki sayı:  5
Değişkenin içindeki sayı:  10 oldu
Değişkenin içindeki sayı:  10.5 oldu
```

ÖRNEK: 3 değişkene de tek satırda 1 değeri atanmıştır.

```
a = b = c = 1
print ('1. sayı=', a)
print ('2. sayı=', b)
print ('3. sayı=', c)
```

EKRAN ÇIKTISI

1. sayı= 1

2. sayı= 1

3. sayı= 1

ÖRNEK: Değişkenler aralarına virgül eklenerek yan yana yazılır. Değerleri de aynı sıralama ile karşılarına yazılır

```
adi, soyadi, yasi='Salih', 'Kul', 42
print ("Adı=", adi)
print ("Soyadı=", soyadi,)
print ("Yaşı=", yasi)
```

EKRAN ÇIKTISI

Adı= Salih

Soyadı= Kul

Yaşı= 42

ÖRNEK: Değişkenlere değer atamak için başka bir yöntem aralarına noktalı virgül “;” ekleyerek değişken - değer ikilileri şeklinde yazmaktır.

```
adi='Salih'; soyadi='Kul'; yasi=42
print ("Adı=", adi)
print ("Soyadı=", soyadi,)
print ("Yaşı=", yasi)
```

EKRAN ÇIKTISI

Adı= Salih

Soyadı= Kul

Yaşı= 42

ÖRNEK: Değer atanmayan ve/veya tanımlanmamış bir değişken kullanılırsa Python hata verir.

```
print(sayi)
```

EKRAN ÇIKTISI

NameError: name 'sayi' is not defined

```
sayi
```

EKRAN ÇIKTISI

NameError: name 'sayi' is not defined

Değer atamadan tanımlamak için degisken=veritipiAdi ( ) kodu kullanılabilir. veritipiAdi yerine int, float, str, bool gelebilir.

Bu durumda int veri türünde varsayılan değer 0, float veri tipinde 0.0, str veri tipinde null (boş), bool veri tipinde False atanır.

ÖRNEK:

```
puan=int()          #varsayılan değer 0
ortalama=float()    #varsayılan deger 0.0
adi=str()           #varsayılan deger null yani boş, degersiz
cevap=bool()        #varsayılan deger False
print(puan)
print(ortalama)
print(adi)
print(cevap)
```

EKRAN ÇIKTISI:

```
0
0.0


False
```

ÖRNEK: Değişkenler veri tiplerine göre kullanılmazsa Python hata verir.

```
#Bir sayı ile bir metin,
kelime toplanamaz.
sayi=5
adi="Salih"
print(sayi+adi)
```

EKRAN ÇIKTISI

TypeError: unsupported operand type(s) for +: 'int' and 'str'

VERİ TİPLERİ

VERİ TİPİ	SINIFI	AÇIKLAMA
Integer	int	Tam sayıların tutulduğu veri tipidir. (Örnek: 3, 5, 369963)
Float	float	Ondalıklı sayıların tutulduğu veri tipidir. (Örnek: 10.45, 3.14)
Karakter dizisi (String)	str	Karakter dizilerini (metinleri) göstermek için kullanılır. Çift tırnak veya tek tırnak içinde gösterilir. Örnek: “Merhaba Dünya”
Boolean	bool	Sadece True veya False değeri alır. int(True)=1 iken int(False)=0 dır.
Complex	complex	karmaşık sayıların tutulduğu veri tipidir. A+Bj tipinde veriler tutulur. (Örnek: 4+5j)
Liste	list	Farklı veri türleri içerebilir. listem=[‘Salih’, 42, ‘Mühendis’, True]
Demet (tüple)	list	Farklı veri türleri içerebilir. demet1=(‘Salih’, 42, ‘Mühendis’, True)
Sözlük (dictionary)	dict	Farklı veri türleri içerebilir. sozluk={‘adi’: ‘Salih’,’yasi’=24, ‘meslekUnvani’:’Mühendis’, ‘askerlikDurumu’: True}

VERİ TİPLERİ

Python’da bir değişkene değer atandığında veri tipleri atanan değere göre otomatik olarak belirlenir.

- `#int tipinde bir veri`
- `int yas=42`
- `#float tipinde bir veri`
- `piSayisi=3.14`
- `#string veri tipinde bir veri`
- `adiSoyadi="Salih Kul"`

VERİ TİPLERİ

Değer atamadan tanımlamak için ***degiskenAdi=VeriTipi( )*** Formatında tanımlama yapılabilir.

- `#Değer ataması yapılmamış integer tipinde bir değişken`
- `yas=int()`
- `#Değer ataması yapılmamış float tipinde bir değişken`
- `piSayisi=float()`
- `#Değer ataması yapılmamış string tipinde bir değişken`
- `adiSoyadi=str()`

Bu durumda değişkene sayısal veri tipi için ilk değer olarak “0” sıfır stringi veri tipi için null (boş) değer atanır.

VERİ TİPLERİ

ÖRNEK:

```
puan=int()          #varsayılan değer 0
ortalama=float()    #varsayılan deger 0.0
adi=str()           #varsayılan deger null yani boş, degersiz
cevap=bool()        #varsayılan deger False
print(puan)
print(ortalama)
print(adi)
print(cevap)
```

EKRAN ÇIKTISI:

```
0

0.0


False
```

Karakter Dizisi (string) Veri Tipi

Karakter dizisi, kullanıcıdan alınan değerlerin text(metin) formatında tutulduğu veri tipleridir. Karakter dizisi şeklinde tanımlanan değişkenlere atanan değerler çift tırnak (" "), veya (‘ ‘) arasında yazılmalıdır. Python karakter dizisi oldukça kullanışlı işlevlere sahiptir. Bir karakter dizisi ekrana yazdırılabilir, başka bir karakter dizisiyle birleştirilebilir.

ÖRNEK:

```
metin1="Aselsan"
metin2="Konya"
metin3="MTAL"
#metin1 değişkeninin değerini yazar
print(metin1)
#metin1 ile metin2 değişkenini birleştirirek yazar
print(metin1+metin2)
#metin1 metin2 ve metin3 değişkenini birleştirerek yazar
print(metin1+metin2+metin3)
```

EKRAN ÇIKTISI:

```
Aselsan
AselsanKonya
AselsanKonyaMTAL
```

Karakter Dizisi (string) Veri Tipi

Bir önceki örnekte metin birleştirme işlemi yapılmakta ve kelimeler arasında boşluk bulunmamaktadır. Metin birleştirme yaparken kelimeler arasına boşluk bırakmak için çift tırnak (" ") veya tek tırnak (' ') karakterleri arasına boşluk konularak yapılabilir.

ÖRNEK:

```
metin1="Aselsan"
metin2="Konya"
metin3="MTAL"
#metin1 metin2 ve metin3 değişkenini aralarına boşluk
ekleyerek birleştirir
print(metin1+" "+metin2+" "+metin3)
#metin1 metin2 ve metin3 değişkenini aralarına boşluk
ekleyerek birleştirir
print(metin1+' '+metin2+' '+metin3)
```

EKRAN ÇIKTISI:

```
Aselsan Konya MTAL
Aselsan Konya MTAL
```

Karakter Dizisi (string) Veri Tipi

Bir değişkenin sakladığı string tipindeki bir karakter dizisi istenilen sayıda ekrana yazdırılabilir.

ÖRNEK:

```
metin="Maşallah "
print(metin*3)
```

EKRAN ÇIKTISI:

Maşallah Maşallah Maşallah

Karakter Dizisi (string) Veri Tipi

len() metodu: Kendisine parametre olarak verilen bir karakter dizisinin karakter sayısını döndürür.

ÖRNEK:

```
metin="Aselsan Konya MTAL"
uzunluk=len(metin)
print(uzunluk)
```

EKRAN ÇIKTISI:

18

NOT: Boşluk da bir karakterdir.

Karakter Dizilerinde (string) Dilimleme İşlemleri

- Bir karakter dizisinin içindeki karakterlere tek tek veya belirli bir aralıkta erişilebilir.
- Köşeli parantez içinde [ ] tek bir sayı verildiğinde bu karakter dizisinin indisini ifade eder.
- İndis numarası “0” dan başlayarak karakterin metindeki kaçıncı sırada yer aldığını gösterir.
- metin[0] ifadesi metin değişkenindeki 1. karakteri verir.

Karakter Dizilerinde (string) Dilimleme İşlemleri

- Belirli bir aralıktaki karakteri alırken “[başlangıç indisi:bitiş indisi]” şeklinde ifade edilir.
  - metin[0:5] metin değişkeninde indisi 0, 1, 2, 3 ve 4 olan karakterleri dilimler. Bitiş indisi dilimlemeye dahil edilmez.
- İndislerin “0” dan başladığı unutulmamalıdır.
- Başlangıç indisi verilmeyen karakter dizisinde örnek: metin[:7] başlangıç indisi otomatik olarak sıfır (0) olur.
  - Örnek 0, 1, 2, 3, 4, 5 ve 6. karakterlerden oluşan bir metin verir.
- Bitiş indisi değeri verilmezse başlangıç indisinden başlanarak son karakter dâhil dilimleme işlemi yapılır.

Karakter Dizilerinde (string) Dilimleme İşlemleri

ÖRNEK:

```
metin="Aselsan Konya MTAL"
#metin değişkeninin karakter sayısı yazdırılıyor
print(len(metin))
#metin değişkeninin 0. indisi yazdırılıyor
print(metin[0])
#metin değişkeninin 5.indisi yazdırılıyor
print(metin[5])
#metin değişkeninin 17.indisi yazdırılıyor
print(metin[17])
```

EKRAN ÇIKTISI:

```
18
A
a
L
```

NOT: Boşluk da bir karakterdir.

Karakter Dizilerinde (string) Dilimleme İşlemleri

ÖRNEK:

```
metin="Aselsan Konya MTAL"
#metin değişkeninin karakter sayısı yazdırılıyor
print(len(metin))
#metin değişkeninin 0.indisinden 7.karaktere kadar yazdırılıyor
print(metin[0:7])
#metin değişkeninin 2.indisinden 7.karaktere kadar yazdırılıyor
print(metin[2:7])
#metin değişkeninin 8.indisinden 13.karaktere kadaryazdırılıyor
print(metin[8:13])
```

EKRAN ÇIKTISI:

18

Aselsan

elsan

Konya

Karakter Dizilerinde (string) Dilimleme İşlemleri

ÖRNEK:

```
metin="Aselsan Konya MTAL"

#metin değişkeninin 0.indisinden 7.karaktere kadar yazdırılıyor

print(metin[:7])

#metin değişkeninin verilen indis numarasından itibaren tüm karakterler yazdırılıyor

print(metin[8:])

#metin değişkeninin verilen indis numarasından itibaren tüm karakterler yazdırılıyor

print(metin[14::])
```

EKRAN ÇIKTISI:

Aselsan

Konya MTAL

MTAL

Karakter Dizilerinde (string) Dilimleme İşlemleri

- Negatif İndis Sayıları: Karakter dizisine sondan başlandığını ifade eder.
- metin[-1] karakter dizisinin en sonundaki karakteri verir.
- metin [-2] ise sondan ikinci karakteri verir.
- Karakter dizilerinde indisler ritmik atlanarak dilimleme yapılabilir.

Bunun için;

- [başlangıç indisi: bitiş indisi: ritmik artış miktarı] şeklinde kullanılır.

metin[0:8:2] : 0'dan başlayarak 0, 2, 4 ve 6 indis numaralı karakterleri dilimler

Karakter Dizilerinde (string) Dilimleme İşlemleri

ÖRNEK:

```
metin="Aselsan Konya MTAL"
#metin değişkeninin son karakteri yazdırılıyor
print(metin[-1])
#metin değişkeninin sondan ikinci karakteri yazdırılıyor
print(metin[-2  ])
#metin değişkeninin 0. indisten itibaren ritmik 3 karakter
dilimler
print(metin[0:10:3])
#metin değişkeninin 8. indisten itibaren 13.karaktere kadar
ritmik 2 karakter dilimleniyor
print(metin[8:13:2])
```

EKRAN ÇIKTISI:

Aselsan

Konya MTAL

MTAL

Alno

type( ) Metodu Kullanımı

Python, her ne kadar veri tiplerini otomatik olarak verse de bir değişkenin veri tipini kontrol etmek ve kullanım amacına göre değiştirmek gerekebilir. Bir değişkenin veri tipini öğrenmek için ***"type( )"*** komutu kullanılır.

Kullanımı:

```
type(degiskenAdi)
```

ÖRNEK:

```
adiSoyadi="Salih Kul"
pi=3.14
sinavPuanı=90
gectiMi=True
print("adiSoyadi değişkeninin veri tipi=",type(adiSoyadi))
print("pi değişkeninin veri tipi=",type(pi))
print("sinavPuanı değişkeninin veri tipi=",type(sinavPuanı))
print("gectiMi değişkeninin veri tipi=",type(gectiMi))
```

EKRAN ÇIKTISI:

```
adiSoyadi değişkeninin veri tipi= <class 'str'>
pi değişkeninin veri tipi= <class 'float'>
sinavPuanı değişkeninin veri tipi= <class 'int'>
gectiMi değişkeninin veri tipi= <class 'bool'>
```

Veri Tiplerini Dönüştürmek

Tam sayılarla işlem yapıldığında “int”, kesirli sayılarla işlem yapıldığında “float” veri tipi kullanılmaktadır.

Değerler üzerinde işlem yaparken (örneğin “input” ile kullanıcıdan veri alırken) içinde sadece rakamlar bulunan “string” ifadeyi sayısal veri tipine dönüştürmek, bazen de bunun tersini yapmak gerekebilir.

Veri tipini dönüştürmek için kullanılan temel fonksiyonlar şunlardır:

int( ) : Veri tipini integer’a çevirir.

float() : Veri tipini float’a çevirir.

str() : Veri tipini karakter dizisine çevirir.

“int” tipinde bir sayıyla “float” tipinde bir sayı çarpıldığında sonuç “float” tipinde olacağı için Python bu veri tipini otomatik olarak belirler.

Veri Tiplerini Dönüştürmek

ÖRNEK:

```
bilgi1="4"
bilgi2="2"
#metin birleştirme işlemi yapılır
print(bilgi1+bilgi2)
"""bilgi1 ve bilgi2 değişkenleri integer veri tipine dönüştürülür
ve toplama işlemi yapılır"""
print(int(bilgi1)+int(bilgi2))
```

EKRAN ÇIKTISI:

42

6

NOT:Karakter dizisi olan bir değer sayısal bir ifadeye dönüştürüldüğünde bu ifadenin sadece sayısal karakterler içermesi gerekir. “A” harfi veya metinsel bir ifade int(“A”) kullanılarak sayıya çevrilemez.

Veri Tiplerini Dönüştürmek

ÖRNEK:

```
#float tipinde pi değişkeni tamınlanıyor
pi=3.14
#integer tipinde puan değişkeni tanımlanıyor
puan=85
#pi değişkeni tamsayıya dönüştürülüp yazdırılıyor
print(int(pi))
#puan değişkeni float (ondalıklı) veri tipine dönüştürülüp yazdırılıyor
print(float(puan))
```

EKRAN ÇIKTISI:

```
3
85.0
```

Veri Tiplerini Dönüştürmek

Python, her ne kadar veri tiplerini otomatik olarak verse de bir değişkenin veri tipini kontrol etmek ve kullanım amacına göre değiştirmek gerekebilir. Bir değişkenin veri tipini öğrenmek için ***"type( )"*** komutu kullanılır.

Kullanımı:

```
type(degiskenAdi)
```

ÖRNEK:

```
adiSoyadi="Salih Kul"
pi=3.14
sinavPuanı=90
gectiMi=True
print("adiSoyadi değişkeninin veri tipi=",type(adiSoyadi))
print("pi değişkeninin veri tipi=",type(pi))
print("sinavPuanı değişkeninin veri tipi=",type(sinavPuanı))
print("gectiMi değişkeninin veri tipi=",type(gectiMi))
```

EKRAN ÇIKTISI:

```
adiSoyadi değişkeninin veri tipi= <class 'str'>
pi değişkeninin veri tipi= <class 'float'>
sinavPuanı değişkeninin veri tipi= <class 'int'>
gectiMi değişkeninin veri tipi= <class 'bool'>
```

input() Fonksiyonu

input( ) fonksiyonu kullanıcıya bilgi vererek, kullanıcıdan bilgi alarak, alınan değerin bir değişkene atanmasını

ÖRNEK:

```
adiSoyadi=input("Adınızı giriniz:")
print("Merhaba",adiSoyadi)
```

EKRAN ÇIKTISI:

Adınızı giriniz:Salih Kul

Merhaba Salih Kul

input() Fonksiyonu

Örnek: Klavyeden girilen iki sayının toplamını bulup ekrana yazdıran programı yazınız.

ÖRNEK:

```
sayi1=int(input("Birinci sayıyı giriniz:"))
sayi2=int(input("İkinci sayıyı giriniz:"))
toplam=sayi1+sayi2
print("Toplam=",toplam)
```

EKRAN ÇIKTISI:

Birinci sayıyı giriniz:5

İkinci sayıyı giriniz:10

Toplam= 15

input() Fonksiyonu

Örnek: Klavyeden girilen iki sayının toplamını bulup ekrana yazdıran programı yazınız.

ÖRNEK:

```
sayi1=int(input("Birinci sayıyı giriniz:"))
sayi2=int(input("İkinci sayıyı giriniz:"))
toplam=sayi1+sayi2
print("Toplam=",toplam)
```

EKRAN ÇIKTISI:

Birinci sayıyı giriniz:5

İkinci sayıyı giriniz:10

Toplam= 15