**BASİC DATA TYPES**

**Introduction to Data Types**

Each data has a type, whether constant or variable. This type of data defines how you store it in memory and it also describes which process can be applied to it.

In fact, data types are nothing but variables you use to reserve some space in memory. Python variables do not need an explicit declaration to reserve memory space. The declaration happens automatically when you assign a value to a variable.

**💡Tips:**

* Note that, we assign value to a variable using 👉 **=**

You can think of **types** in the real world, bees are bee type and palm trees are palm type. That is, they have certain formats and common features.

We will now discuss some simple data types commonly used in Python:

* String,
* Signed Integer,
* Floating Point,
* Complex,
* Boolean.

**Veri Türlerine Giriş**

Her verinin bir türü vardır, ister sabit ister değişken olsun. Bu tür veriler, onu bellekte nasıl saklayacağınızı tanımlar ve ayrıca ona hangi işlemin uygulanabileceğini tanımlar.  
  
Aslında veri türleri, bellekte biraz yer ayırmak için kullandığınız değişkenlerden başka bir şey değildir. Python değişkenleri, bellek alanı ayırmak için açık bir bildirime ihtiyaç duymaz. Bir değişkene değer atadığınızda bildirim otomatik olarak gerçekleşir.  
  
💡İpuçları:  
Unutmayın, bir değişkene 👉 = kullanarak değer atarız.  
Gerçek dünyadaki türleri düşünebilirsiniz, arılar arı türü ve palmiye ağaçları palmiye türüdür. Yani belirli biçimleri ve ortak özellikleri vardır.  
  
Şimdi Python'da yaygın olarak kullanılan bazı basit veri türlerini tartışacağız:  
  
Sicim,  
  
Imzalı Tamsayı,  
  
Kayan nokta,  
  
Karmaşık  
  
Boole.

**Strings**

Strings are identiﬁed as a contiguous set of characters represented in the quotation marks. Python allows for either pair of single or double (or even triple) quotes.

Strings are immutable sequence data type, i.e each time one makes any changes to a string, a completely new string object is created. You will be able to better understand **immutability** with examples that will continue in the next lessons.

If you want to work with any textual characters in your code, you have to work with strings. The string type is called str. Strings are the most common and useful data type in Python.

Carefully examine the following examples that will help you understand the type of string.

input :

text1 = "I have learned strings" # surrounded with double quotes

print(text1)

output :

I have learned strings

input :

e\_mail = 'joseph@clarusway.com' # surrounded with single quotes

print(e\_mail)

output :

joseph@clarusway.com

input :

print('632') # this is also a string type

output :

632

**String**

Dizeler, tırnak içinde gösterilen bitişik bir karakter kümesi olarak tanımlanır. Python, tek veya çift (hatta üçlü) tırnak çiftine izin verir.  
  
Dizeler değişmez dizi veri türüdür, yani bir dizede her değişiklik yapıldığında, tamamen yeni bir dize nesnesi oluşturulur. Sonraki derslerde devam edecek örneklerle değişmezliği daha iyi anlayacaksınız.  
  
Kodunuzdaki herhangi bir metin karakteriyle çalışmak istiyorsanız, dizelerle çalışmanız gerekir. Dize türü str olarak adlandırılır. Dizeler Python'daki en yaygın ve kullanışlı veri türüdür.  
  
Dize türünü anlamanıza yardımcı olacak aşağıdaki örnekleri dikkatlice inceleyin.  
  
Giriş:  
  
Text1 = 'Dizeleri öğrendim' # çift tırnak içine alınır  
Yazdır(metin1)  
Çıktı :  
  
Dizeleri öğrendim  
Giriş:  
  
E\_mail = 'joseph@clarusway.com' # tek tırnakla çevrili  
Yazdır(e\_mail)  
Çıktı :  
  
Joseph@clarusway.com  
Giriş:  
  
Print('632') # bu da bir dize türüdür  
Çıktı :  
  
632

**Numeric Types**

For any programmer, using numbers is the most important issue. You can hardly write a serious program without using numbers, so let's talk about some basic numeric types. There are three distinct numeric types: **signed integer numbers**, **floating point numbers** and **complex numbers**.

* **Signed Integer** type is called int, they are whole numbers (positive, negative or zero), including no decimal point. For example: 71, -122, 0
* **Floating point** type is called float and they stand for real numbers with a decimal point. For example: 71.0, -33.03
* **Complex** type is called complex and they are written in the form, **x + yj** , where x is the real part and y is the imaginary part. For example: 3.14j. Imaginary numbers, also called complex numbers, are used in real-life applications, such as electricity, as well as quadratic equations. In quadratic planes, imaginary numbers show up in equations that don't touch the x-axis. Imaginary numbers become particularly useful in advanced calculus. We will not use this type much.

**💡Tips:**

* 71 and 71.0 have the same numerical value. But they differ in terms of numeric type. The types of these numbers are int and float respectively.

Q: What are the numerical data types in Python and their properties?  
A:

* **Integers :** they are whole numbers (positive, negative or zero), including no decimal point.
* **Floats :** they stand for real numbers with a decimal point.
* **Complexes :** they are written in the form, **x + yj** , where x is the real part and y is the imaginary part.

**Sayısal Türler**

Herhangi bir programcı için sayıları kullanmak en önemli konudur. Sayıları kullanmadan ciddi bir program yazamazsınız, bu yüzden bazı temel sayısal türlerden bahsedelim. Üç farklı sayısal tür vardır: işaretli tamsayılar, kayan noktalı sayılar ve karmaşık sayılar.  
  
İşaretli Tamsayı türüne int denir, bunlar ondalık nokta içermeyen tam sayılardır (pozitif, negatif veya sıfır). Örneğin: 71, -122, 0  
  
Kayan nokta türüne kayan nokta denir ve ondalık noktalı gerçek sayıları temsil ederler. Örneğin: 71.0, -33.03  
  
Karmaşık türe karmaşık denir ve x + yj biçiminde yazılırlar, burada x gerçek kısım ve y sanal kısımdır. Örneğin: 3.14j. Karmaşık sayılar olarak da adlandırılan hayali sayılar, elektrik ve ikinci dereceden denklemler gibi gerçek hayattaki uygulamalarda kullanılır. İkinci dereceden düzlemlerde, x eksenine dokunmayan denklemlerde hayali sayılar görünür. Hayali sayılar özellikle ileri matematikte kullanışlı hale gelir. Bu türü pek kullanmayacağız.  
  
💡İpuçları:  
71 ve 71.0 aynı sayısal değere sahiptir. Ancak sayısal tür açısından farklılık gösterirler. Bu sayıların türleri sırasıyla int ve float şeklindedir.  
S: Python'daki sayısal veri türleri ve özellikleri nelerdir?  
A:  
  
Tamsayılar : ondalık nokta içermeyen tam sayılardır (pozitif, negatif veya sıfır).  
Floats : Ondalık noktalı gerçek sayıları temsil ederler.  
Kompleksler : x + yj şeklinde yazılırlar, burada x reel kısım ve y sanal kısımdır.

**Boolean**

Boolean types are called bool and their values are the two constant objects **False** and **True**. They are used to represent truth values (other values can also be considered false or true). In numeric contexts (for example, when used as the argument to an arithmetic operator), they behave like the integers 0 and 1, respectively.

Bools are important data types that are widely used in Python as they can find use in every aspect of our daily lives. For example, imagine, whether the TV is turned on or off in your home or if the weather is rainy can be explained easily with bools.

Bools are mostly used in **conditional operations** which we will discuss in the next lessons.

tv\_open = True # it seems TV is on now

is\_rainy = False # I love sunny weather

Q: Describe the Boolean types in detail.  
A: **Boolean** types are called bool and their values are the two constant objects **True** and **False**. They are used to represent truth values (other values can also be considered false or true).  
  
In numeric contexts (for example, when used as the argument to an arithmetic operator), they behave like the integers 0 and 1, respectively.  
  
**Bools** are important data types that are widely used in Python as they can find use in every aspect of our daily lives. For example, imagine, whether the TV is turned on or off in your home or if the weather is rainy can be explained easily with bools.

**Boole**

Boole türleri bool olarak adlandırılır ve değerleri False ve True iki sabit nesnesidir. Doğruluk değerlerini temsil etmek için kullanılırlar (diğer değerler de yanlış veya doğru olarak kabul edilebilir). Sayısal bağlamlarda (örneğin, bir aritmetik operatörün argümanı olarak kullanıldığında), sırasıyla 0 ve 1 tam sayıları gibi davranırlar.  
  
Booller, günlük hayatımızın her alanında kullanım bulabildikleri için Python'da yaygın olarak kullanılan önemli veri türleridir. Örneğin, evinizde televizyonun açılıp kapanmadığını veya havanın yağmurlu olup olmadığını boollerle kolayca açıklanabileceğini hayal edin.  
  
Bool'lar daha çok sonraki derslerde ele alacağımız koşullu işlemlerde kullanılır.  
  
Tv\_open = Doğru # şu anda TV açık görünüyor  
Is\_rainy = Yanlış # Güneşli havayı severim  
S: Boolean türlerini ayrıntılı olarak açıklayın.  
C: Boole türleri bool olarak adlandırılır ve değerleri True ve False olmak üzere iki sabit nesnedir. Doğruluk değerlerini temsil etmek için kullanılırlar (diğer değerler de yanlış veya doğru olarak kabul edilebilir).  
  
Sayısal bağlamlarda (örneğin, bir aritmetik operatörün argümanı olarak kullanıldığında), sırasıyla 0 ve 1 tam sayıları gibi davranırlar.  
  
Booller, günlük hayatımızın her alanında kullanım bulabildikleri için Python'da yaygın olarak kullanılan önemli veri türleridir. Örneğin, evinizde televizyonun açılıp kapanmadığını veya havanın yağmurlu olup olmadığını boollerle kolayca açıklanabileceğini hayal edin.

**Type Conversion**

We can convert the types of data to each other if the type allows to be converted. There are some functions to convert the types:

* str() converts to **string** type
* int() converts to **signed integer** type
* float() converts to **floating point** type

**💡Tips:**

* We can print the types of data using type() function.

Look at the examples below to how we learn the types of data.

input :

example1 = 'sometimes what you say is less important than how you say it'

print(type(example1))

output :

<class 'str'>

input :

example2 = '71'

print(type(example2))

output :

<class 'str'>

input :

example3 = 71

print(type(example3))

output :

<class 'int'>

input :

example4 = 71.0

print(type(example4))

output :

<class 'float'>

input :

example5 = 3.14j

print(type(example5))

output :

<class 'complex'>

input :

example6 = True

print(type(example6))

output :

<class 'bool'>

**💡Tips:**

* Note that we write the first letter of True in **uppercase**. This is the rule of Python that we must write like this : True, False.

Here are some examples on converting between different types:

input :

f = 3.14 # the type is float

print(type(f))

output :

<class 'float'>

input :

f = 3.14 # the type is float

s = str(f) # converting float to string

print(type(s))

output :

<class 'str'>

input :

f = 3.14 # the type is float

i = int(f) # while converting a float value to an integer its decimal part is disregarded

print(i, '\n')

print(type(i))

output :

3

<class 'int'>

input :

i = 3

f = float(i)

print(f, '\n')

print(type(f))

output :

3.0

<class 'float'>

input :

x = 39

v = "11"

y = "2.5"

z = "I am at\_"

print(x-int(v))

print(x-float(y))

print(z+str(x))

output :

28

36.5

I am at\_39

**Tür Dönüştürme**

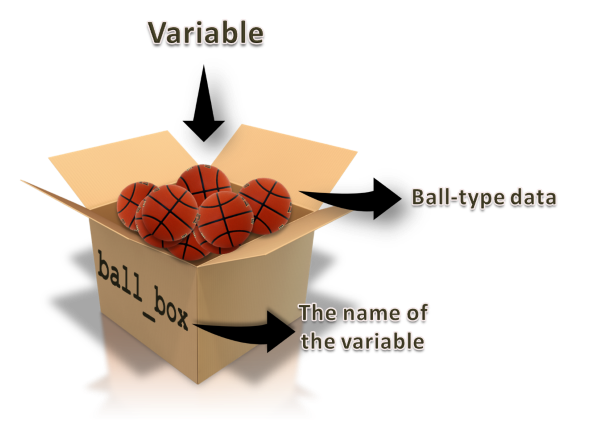
Tür dönüştürülmeye izin veriyorsa, veri türlerini birbirine dönüştürebiliriz. Türleri dönüştürmek için bazı işlevler vardır:  
  
Str(), dize türüne dönüştürür  
Int(), işaretli tamsayı türüne dönüştürür  
Float() kayan nokta tipine dönüştürür  
💡İpuçları:  
Type() fonksiyonunu kullanarak veri tiplerini yazdırabiliriz.  
Veri türlerini nasıl öğrendiğimize ilişkin aşağıdaki örneklere bakın.  
  
Giriş:  
  
Example1 = 'bazen ne söylediğiniz, nasıl söylediğinizden daha az önemlidir'  
Yazdır(tür(örnek1))  
Çıktı :  
  
Sınıf 'str'  
Giriş:  
  
Örnek2 = '71'  
Yazdır(tür(örnek2))  
Çıktı :  
  
Sınıf 'str'  
Giriş:  
  
Örnek3 = 71  
Yazdır(tür(örnek3))  
Çıktı :  
  
'int' sınıfı  
Giriş:  
  
Örnek4 = 71.0  
Yazdır(tür(örnek4))  
Çıktı :  
  
'yüzer' sınıfı  
Giriş:  
  
Örnek5 = 3.14j  
Yazdır(tür(örnek5))  
Çıktı :  
  
Sınıf 'karmaşık'  
Giriş:  
  
Örnek6 = Doğru  
Yazdır(tür(örnek6))  
Çıktı :  
  
'bool' sınıfı  
💡İpuçları:  
True'nun ilk harfini büyük harfle yazdığımızı unutmayın. Bu, Python'un şöyle yazmamız gereken kuralıdır: Doğru, Yanlış.  
Farklı türler arasında dönüştürmeye ilişkin bazı örnekler:  
  
Giriş:  
  
F = 3.14 # tip yüzer  
Yazdır(tür(f))  
Çıktı :  
  
'yüzer' sınıfı  
Giriş:  
  
F = 3.14 # tip yüzer  
  
S = str(f) # kayan noktayı dizgeye dönüştürme  
Yazdır(tür(ler))  
Çıktı :  
  
Sınıf 'str'  
Giriş:  
  
F = 3.14 # tip yüzer  
  
I = int(f) # bir kayan nokta değerini tam sayıya dönüştürürken ondalık kısmı dikkate alınmaz  
Yazdır(i, '  
')  
Yazdır(tür(i))  
Çıktı :  
  
3  
  
'int' sınıfı  
Giriş:  
  
Ben = 3  
  
F = kayan nokta(i)  
Yazdır(f, '  
')  
Yazdır(tür(f))  
Çıktı :  
  
3.0  
  
'yüzer' sınıfı  
Giriş:  
  
X = 39  
V = '11'  
Y = '2,5'  
Z = 'Ben\_'deyim'  
  
Yazdır(x-int(v))  
Print(x-float(y))  
Yazdır(z+str(x))  
Çıktı :  
  
28  
36.5  
39 yaşındayım

**Variables**

**Variable** is a location designated where a value can be stored and accessed later. Imagine a box where you store something. That's a variable.

Let's create a box (variable) in which contains basketball balls. Let's name it ball\_box. It is also the name of the variable.

Creating, naming the variable and assigning a value to it happen simultaneously by this syntax : ball\_box = 20 basketball balls



Python variables do not need an explicit declaration to reserve memory space. The declaration happens automatically when you assign a value to a variable.

To create a variable in Python, all you need to do is specify the variable name and then assign a value to it using 👉🏻**=**

The formula syntax of creating a variable and assigning a value to it is :

**variable name = value**

Remember, according to the PEP8, we had to give the variables a meaningful name for the data they kept inside.

Let's define variables and assign values to them :

input :

color = 'red' # str type variable

season = 'summer'

price = 250 # int type variable

pi = 3.14 # float type variable

color = 'blue' # You can always assign a new value to a created variable

price = 100 # value of 'price' is changed

season = 'winter'

print(color, price, season, sep=', ')

output :

blue, 100, winter

**Scratch Time ! :**Solve this example with **[scratch](https://scratch.mit.edu/projects/341546766/editor/)**.

**💡Tips:**

* Note that, the last value assigned to a variable is valid.

In Python, it is possible to assign the value of one variable to another variable:

input :

a = 5

b = 55

c = 555

c = a

b = c

a = b

print(a, b, c, sep=', ')

output :

5, 5, 5

**⚠️Avoid ! :**

* Note that, If you use undefined name of a variable in the code you write, you will get an 'NameError' message.

Q: What is the 'variable' and how do you assign a value to it?  
A: **Variable** is a location designated where a value can be stored and accessed later. Imagine a box where you store something. That's a variable.  
  
Python variables do not need an explicit declaration to reserve memory space. The declaration happens automatically when you assign a value to a variable.  
  
To create a variable in Python, all you need to do is specify the variable name and then assign a value to it.

**Değişkenler**

Değişken, bir değerin saklanabileceği ve daha sonra erişilebileceği belirlenmiş bir konumdur. Bir şeyler sakladığınız bir kutu hayal edin. Bu bir değişken.  
  
İçinde basketbol toplarının olduğu bir kutu (değişken) oluşturalım. Adını ball\_box koyalım. Aynı zamanda değişkenin adıdır.  
  
Değişkeni yaratmak, adlandırmak ve ona bir değer atamak aynı anda şu sözdizimi ile gerçekleşir: ball\_box = 20 basketbol topu  
  
  
  
Python değişkenleri, bellek alanı ayırmak için açık bir bildirime ihtiyaç duymaz. Bir değişkene değer atadığınızda bildirim otomatik olarak gerçekleşir.  
  
Python'da bir değişken oluşturmak için tek yapmanız gereken değişken adını belirtmek ve ardından 👉🏻= kullanarak ona bir değer atamak.  
  
Bir değişken oluşturmanın ve ona bir değer atamanın formül sözdizimi şöyledir:  
  
Değişken adı = değer  
  
Unutmayın, PEP8'e göre değişkenlere içinde tuttukları veriler için anlamlı bir isim vermemiz gerekiyordu.  
  
Değişkenleri tanımlayalım ve onlara değer atayalım:  
  
Giriş:  
  
Color = 'red' # str tipi değişken  
Sezon = 'yaz'  
Fiyat = 250 # int tipi değişken  
Pi = 3.14 # kayan tip değişken  
Color = 'blue' # Oluşturulan bir değişkene her zaman yeni bir değer atayabilirsiniz.  
Fiyat = 100 # 'fiyat' değeri değiştirildi  
Mevsim = 'kış'  
  
Print(renk, fiyat, sezon, sep=', ')  
Çıktı :  
  
Mavi, 100, kış  
Kazıma Zamanı! : Bu örneği sıfırdan çözün.  
  
  
  
💡İpuçları:  
Bir değişkene atanan son değerin geçerli olduğunu unutmayın.  
Python'da bir değişkenin değerini başka bir değişkene atamak mümkündür:  
  
Giriş:  
  
Bir = 5  
B = 55  
C = 555  
C = bir  
B = c  
A = b  
  
Yazdır(a, b, c, sep=', ')  
Çıktı :  
  
5, 5, 5  
⚠️ Kaçının ! :  
Unutmayın, yazdığınız kodda bir değişkenin tanımsız adını kullanırsanız, 'NameError' mesajı alırsınız.  
S: 'Değişken' nedir ve ona nasıl bir değer atarsınız?  
A: Değişken, bir değerin saklanabileceği ve daha sonra erişilebileceği belirlenmiş bir konumdur. Bir şeyler sakladığınız bir kutu hayal edin. Bu bir değişken.  
  
Python değişkenleri, bellek alanı ayırmak için açık bir bildirime ihtiyaç duymaz. Bir değişkene değer atadığınızda bildirim otomatik olarak gerçekleşir.  
  
Python'da bir değişken oluşturmak için tek yapmanız gereken değişken adını belirlemek ve ardından ona bir değer atamak.