

Checks the Python version

```
!python --version
```

```
Python 3.10.12
```

display the text using print method

```
print("Welcome to Python Programming Lab")
```

```
Welcome to Python Programming Lab
```

Separator in print method-- Specifies how to separate the objects, if there is more than one

Double-click (or enter) to edit

```
print("Hi","How are you doing?",sep="--")
```

```
Hi--How are you doing?
```

end=Specify what to print at the end. Default is '\n' (line feed)

```
print("Hello",end="")  
print("world")
```

```
Helloworld
```

```
print("Hello",end=",")  
print("world")
```


```
Hello,world
```

variables-Variables do not need to be declared with any particular type, and can even change type after they have been set. Variables are containers for storing data values.

- A variable can have a short name (like x and y) or a more descriptive name (age, carname, total_volume). Rules for Python variables:
- A variable name must start with a letter or the underscore character
- A variable name cannot start with a number

- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
- Variable names are case-sensitive (age, Age and AGE are three different variables)
- A variable name cannot be any of the Python keywords


```
x = 3          # x is of type int
x = "python"  # x is now of type str
print(x)
```

 python

```
myvar = "PYTHON"
my_var = "PYTHON"
_my_var = "PYTHON"
myVar = "PYTHON"
MYVAR = "PYTHON"
myvar2 = "PYTHON"
```

#Illegal variable names:

```
2myvar = "PYTHON"
my-var = "PYTHON"
my var = "PYTHON"
```

 File "["<ipython-input-17-aa1b75d8c15e>"](#), line 3
 2myvar = "PYTHON"
 ^
 SyntaxError: invalid decimal literal

Next steps: [Fix error](#)

Arithmetic Operations

```
x=2
y=3
print(x+y)
```

 5

```
print(x-y)
```

 -1

```
print(x/y)
```

```
➞ 0.6666666666666666
```

#divides x by y and rounds down to the nearest integer

```
print(x//y)
print(10/3)
print(10//3)
```

```
➞ 0
   3.3333333333333335
   3
```

Casting:If you want to specify the data type of a variable, this can be done with casting.

```
x = str(3)    # x will be '3'
y = int(3)    # y will be 3
z = float(3)  # z will be 3.0
```

```
int(3.555)
```

```
➞ 3
```

Get the Type You can get the data type of a variable with the `type()` function.

```
x = 5
y = "AIDSB"
print(type(x))
print(type(y))
```

```
➞ <class 'int'>
   <class 'str'>
```

Single or Double Quotes? String variables can be declared either by using single or double quotes:

```
x = "awesome"
# is the same as
x = 'good'
print(x)
```

```
➞ good
```

Case-Sensitive Variable names are case-sensitive.

```
a = 4
A = "list"
#A will not overwrite a
```

Many Values to Multiple Variables

```
x, y, z = "PYTHON", "DBMS", "SDC-I"
print(x)
print(y)
print(z)
```

```
➞ PYTHON
   DBMS
   SDC-I
```

One Value to Multiple Variables And you can assign the same value to multiple variables in one line:

```
x = y = z = "PYTHON"
print(x)
print(y)
print(z)
```

```
➞ PYTHON
   PYTHON
   PYTHON
```

In the print() function, you output multiple variables, separated by a comma:

```
x = "Python"
y = "is"
z = "awesome"
print(x, y, z)
```

```
➞ Python is awesome
```

Built-in Data Types In programming, data type is an important concept. Variables can store data of different types, and different types can do different things. Python has the following data types built-in by default, in these categories:

Text Type: str **Numeric Types:** int, float, complex **Sequence Types:** list, tuple, range **Mapping Type:** dict **Set Types:** set, frozenset **Boolean Type:** bool **Binary Types:** bytes, bytearray **None Type:** NoneType

Getting the Data Type You can get the data type of any object by using the `type()` function:

```
x = 5
print(type(x))
```

```
➞ <class 'int'>
```

Setting the Data Type In Python, the data type is set when you assign a value to a variable:

```
x = "Hello World"
```

```
#display x:
print(x)
```

```
#display the data type of x:
print(type(x))
```

```
➞ Hello World
   <class 'str'>
```

```
x = 20
```

```
#display x:
print(x)
```

```
#display the data type of x:
print(type(x))
```

```
➞ 20
   <class 'int'>
```

Start coding or [generate](#) with AI.

