Python 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 numbers of data types that are built-in by default like:

Data Types	Classes	Description
Numeric	int, float	holds numeric values
String	str	holds sequence of characters
Boolean	bool	holds either True or False

Examples:

Number1=3 >>>> int

Number2=2.9 >>>>float

X=”Hello” >>>> string

Y=true >>>> boolan

Note 1: In Python, the data type is set when you assign a value to a variable.

Note 2: Python has a set of keywords that are reserved words that cannot be used as variable names, function names, or any other identifiers.

Note 3: Python is a case-sensitive language. It considers that uppercase and lowercase characters are different.

Python Operators

Operators are used to perform operations on variables and values. In the example below, we use the + operator to add together two values:

Print (10 + 5)

Python divides the operators in the following groups:

Python Arithmetic Operators

Arithmetic operators are used with numeric values to perform common mathematical operations:

Operator	Name	Example
+	Addition	x + y
-	Subtraction	x - y
*	Multiplication	x * y
/	Division	x / y
%	Modulus	x % y
**	Exponentiation	x ** y

Python Assignment Operators

Assignment operators are used to assign values to variables:

Operator	Example	Same As
=	x = 5	x = 5
+=	x += 3	x = x + 3
-=	x -= 3	x = x - 3
*=	x *= 3	x = x * 3
/=	x /= 3	x = x / 3
%=	x %= 3	x = x % 3
**=	x **= 3	x = x ** 3

Python Logical Operators

Logical operators are used to combine conditional statements:

Operator	Description	Example
and	Returns True if both statements are true	x < 5 and x < 10
or	Returns True if one of the statements is true	x < 5 or x < 4
not	Reverse the result, returns False if the result is true	not(x < 5 and x < 10)

Python input() Function

Python input() function is used to take user input. By default, it returns the user input in form of a string.

Syntax:

input(prompt)

prompt [optional]: any string value to display as input message.

Example: input("What is your name? ")

Returns: Return a string value as input by the user.

By default input() function helps in taking user input as string. If any user wants to take input as int or float, we just need to typecast it.

Example:

```
color = input("What color is rose?: ")
print(color)
n = int(input("How many roses?: "))
print(n)
price = float(input("Price of each rose?: "))
print(price)
```

Output:

What color is rose?: red

red

How many roses?: 10

10

Price of each rose?: 12.9

12.9

Python Comments

Comments can be used to explain Python code.

Comments can be used to make the code more readable.

Comments can be used to prevent execution when testing code.

Creating a Comment

Comments starts with a #, and Python will ignore them:

Example:

```
#This is a comment
print("Hello, World!")
```

Comments can be placed at the end of a line, and Python will ignore the rest of the line:

Example:

```
print("Hello, World!") #This is a comment
```

A comment does not have to be text that explains the code, it can also be used to prevent Python from executing code:

Example:

```
#print("Hello, World!")
print("Ahmed, Classmate ")
```

Multiline Comments

To add a multiline comment, you could insert a # for each line:

Example:

```
#This is a comment
#written in
#more than just one line
print("Hello, World!")
```

or you can add a multiline string (triple quotes) in your code, and place your comment inside it:

Example:

```
"""
This is a comment
written in
more than just one line
"""
print("Hello, World!")
```