Python Variables and Data Types

---

1. Variables in Python

A **variable** is a name that refers to a value stored in memory. Python is **dynamically typed**, so you don't need to declare the type explicitly.

Creating Variables

```
x = 10              # Integer
name = "Alice"      # String
pi = 3.14           # Float
is_happy = True     # Boolean
```

- Variable names must start with a letter or underscore.
- They **cannot** start with a number or contain special characters like `@`, `$`, `%`.

---

2. Data Types in Python

Here are the **main built-in data types**:

Type	Description	Example
`int`	Integer numbers	`x = 5`
`float`	Floating point numbers	`pi = 3.14`
`str`	String (text)	`name = "Alice"`
`bool`	Boolean (True or False)	`is_valid = True`
`list`	Ordered, mutable collection	`fruits = ["apple", "banana"]`
`tuple`	Ordered, immutable collection	`point = (3, 4)`
`dict`	Key-value pairs	`person = {"name": "Bob", "age": 25}`
`set`	Unordered collection of unique items	`colors = {"red", "blue"}`

---

3. Type Checking and Casting

Check Data Type

```
x = 42
print(type(x))  # Output: <class 'int'>
```

Type Casting

Convert from one type to another:

```
x = str(10)     # "10"
y = int("5")    # 5
z = float("3.14") # 3.14
```

Example:

```
name = "John"
age = 25
height = 5.9
is_student = False

print("Name:", name)
print("Age:", age)
print("Height:", height)
print("Student:", is_student)
```

Reserved Words

Reserved words (also called keywords) are defined with predefined meaning and syntax in the language. These keywords have to be used to develop programming instructions. Reserved words can't be used as identifiers for other programming elements like name of variable, function etc.

>>> import keyword

>>> keyword.kwlist

Python Comments and Escape Sequences

---

1. Comments in Python

Comments are used to explain code and make it more readable. Python ignores them during execution.

➤ Single-line Comment

Use # before the comment.

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

➤ Multi-line Comment

Python doesn't have a built-in multi-line comment syntax, but you can use triple quotes as a workaround.

```
"""
This is a multi-line comment.
It spans multiple lines.
"""
print("Hello again!")
```

or

```
'''
This is a multi-line comment.
It spans multiple lines.
'''
print("Hello again!")
```

---

2. Escape Sequences in Python

Escape sequences are used to represent special characters in strings. They start with a backslash `\`.

Escape Sequence	Description	Example
`\n`	New line	`"Hello\nWorld"`
`\t`	Tab	`"Hello\tWorld"`
`\\`	Backslash	`"C:\\Users\\Name"`
`\'`	Single quote	`'It\'s okay'`
`\"`	Double quote	`"He said \"Hi\""`
`\r`	Carriage return	`"Hello\rWorld"`
`\b`	Backspace	`"Hello\bWorld"`
`\f`	Form feed	`"Hello\fWorld"`
`\ooo`	Octal value (e.g., `\101`)	`"\101"` → `'A'`
`\xhh`	Hex value (e.g., `\x41`)	`"\x41"` → `'A'`

Example:

```
print("Line1\nLine2")        # Prints on two lines
print("Path: C:\\ABC")  # Prints the backslash
print("Quote: \"Python\"")   # Includes double quotes in output
```

Operators

- Arithmetic Operators
- Comparison (i.e., Relational) Operators
- Assignment Operators
- Logical Operators
- Membership Operators
- Identity Operators
- Bitwise Operators

Arithmetic Operators:-

+,-,*,/,%,**,//

Example:-

```
a = 10

b = 3

c = a + b

print(c)

c = a - b

print(c)

c = a * b

print(c)

c = a / b

print(c)

c = a % b

print(c)

c = a**b

print(c)

c = a//b

print(c)
```

Comparison operators:-

==,!=,<,>,<=,>=

Example:-

```
a=10
b=5
if ( a == b ):
        print(" a is equal to b")
else:
        print(" a is not equal to b")
```

Assignment Operators:-

=,+=,-=,*=,/=,%=,**=,//=

Example:-

```
a=12
a+=12
print(a)
```

Logical Operators:-

and,or,not

Example:-

```
a=10
b=5
if(a == 10 or b < 20.0):
```

```
    print(“body of if”)
else:
     print(“body of else”)

#if not(a == 10 and b < 20.0):
#if (a == 10 and b < 20.0):
```

Membership Operators:-

in,not in

Example:-

```
a = 10
b = 25
list = [10, 23, 34, 46, 58 ]
if ( a in list ):
     print ("a is available in list”)
else:
     print ("a is not available in list”)
```

Identity operators:-

is,is not

Example:-

```
a=10
b=10

if( a is b ):
      print(“a and b memory location is same”)
else:
      print(“a and b memory location is not same”)
```

Bitwise Operators:-

&,|,^,~,<<,>>

Example:-

```
a = 10

b = 7

c = a & b    #and

c = a | b     #or

c = a ^ b     #xor

c = ~a #one’s compliment

c = a << 2 #Binary Left Shift Operator

c = a >> 2 #Binary Right Shift Operator
```