003 Python Variable

Variable and Value

- A variable is a memory location where a programmer can store a value. Example : roll_no, amount, name etc.
- Value is either string, numeric etc. Example : "Sara", 120, 25.36
- Variables are created when first assigned.
- Variables must be assigned before being referenced.
- The value stored in a variable can be accessed or updated later.
- No declaration required
- The type (string, int, float etc.) of the variable is determined by Python
- The interpreter allocates memory on the basis of the data type of a variable.

Python Variable Name Rules

- Must begin with a letter (a - z, A - B) or underscore (_)
- Other characters can be letters, numbers or _
- Case Sensitive
- Can be any (reasonable) length
- There are some reserved words which you cannot use as a variable name because Python uses them for other things.

Good Variable Name

- Choose meaningful name instead of short name. roll_no is better than rn.
- Maintain the length of a variable name. Roll_no_of_a-student is too long?

- Be consistent; roll_no or RollNo
- Begin a variable name with an underscore(_) character for a special case.

Python Assignment Statements

The assignment statement creates new variables and gives them values. Basic assignment statement in Python is :

Syntax:

```
<variable> = <expr>
```

Where the equal sign (=) is used to assign value (right side) to a variable name (left side). See the following statements :

```
>>> Item_name = "Computer" #A String

>>> Item_qty = 10 #An Integer

>>> Item_value = 1000.23 #A floating point

>>> print(Item_name)

Computer

>>> print(Item_qty)

10

>>> print(Item_value)

1000.23

>>>
```

One thing is important, assignment statement read right to left only.

Example:

a = 12 is correct, but 12 = a does not make sense to Python, which creates a syntax error. Check it in Python Shell.

```
>>> a = 12

>>> 12 = a

SyntaxError: can't assign to literal

>>>
```

Multiple Assignment

The basic assignment statement works for a single variable and a single expression. You can also assign a single value to more than one variables simultaneously.

Syntax:

```
var1=var2=var3...varn= = <expr>
```

Example:

```
x = y = z = 1
```

Now check the individual value in Python Shell.

```
>>> x = y = z = 1

>>> print(x)

1

>>> print(y)

1

>>> print(z)

1

>>>
```

Here is an another assignment statement where the variables assign many values at the same time.

Syntax:

```
<var>, <var>, ..., <var> = <expr>, <expr>, ..., <expr>
```

Example:

```
x, y, z = 1, 2, "abcd"
```

In the above example x, y and z simultaneously get the new values 1, 2 and "abcd".

```
>>> x,y,z = 1,2,"abcd"

>>> print(x)

1

>>> print(y)

2

>>> print(z)

abcd
```

You can reuse variable names by simply assigning a new value to them :

```
>>> x = 100

>>> print(x)

100

>>> x = "Python"

>>> print(x)

Python

>>>
```

Other ways to define value

```
>>> five_millions = 5_000_000
>>> five_millions
```

Output:

```
5000000
```

```
>>> small_int = .35

>>> small_int
```

Output:

```
0.35
```

```
>>> c_thousand = 10e3

>>> c_thousand
```

Output:

```
10000.0
```

Swap variables

Python swap values in a single line and this applies to all objects in python.

Syntax:

```
var1, var2 = var2, var1
```

Example:

```
>>> x = 10

>>> y = 20

>>> print(x)

10

>>> print(y)

20
```

```
>>> x, y = y, x

>>> print(x)

20

>>> print(y)

10

>>>
```

Local and global variables in Python

In Python, variables that are only referenced inside a function are implicitly global. If a variable is assigned a value anywhere within the function’s body, it’s assumed to be a local unless explicitly declared as global.

Example:

```
var1 = "Python"

def func1():

    var1 = "PHP"

    print("In side func1() var1 = ",var1)


def func2():

    print("In side func2() var1 = ",var1)

func1()

func2()
```

Output:

```
In side func1() var1 =  PHP
In side func2() var1 =  Python
```

You can use a global variable in other functions by declaring it as global keyword :

Example:

```python
def func1():
    global var1
    var1 = "PHP"
    print("In side func1() var1 = ",var1)


def func2():
    print("In side func2() var1 = ",var1)
func1()
func2()
```

Output:

```
In side func1() var1 =  PHP
In side func2() var1 =  PHP
```