Python: Variable Names

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.

Example: Legal variable names:

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

Example: Incorrect variable names:

```
2myvar = "kumar"
my_var = "kumar"
my var = "Kumar"
```

Multi Words Variable Names

Variable names with more than one word can be difficult to read.
There are several techniques you can use to make them more readable:

Camel Case

Each word, except the first, starts with a capital letter:

```
myVariableName = "Kumar"
```

Pascal Case

Each word starts with a capital letter:

```
MyVariableName = "Kumar"
```

Snake Case

Each word is separated by an underscore character:

```
my_variable_name = "Kumar"
```

Many Values to Multiple Variables

Python allows you to assign values to multiple variables in one line:

Example:

```
x, y, z = "Orange", "Banana", "Cherry"
Print (x)
Print (y)
Print (z)
```

ANS: Orange
Banana
Cherry

One Value to Multiple Variables

And you can assign the same value to multiple variable in one line:

Example:

```
x = y = z = "Orange"
Print (x)
Print (y)
Print (z)
```

ANS: Orange
orange
orange

Unpack a Collection

If you have a collection of values in a list, tuple etc. Python allows you to extract the values into variables. This is called unpacking.

Example: Unpack a list:

```
fruits = ["apple", "banana", "cherry"]
x, y, z = fruits
Print (x)
Print (y)
Print (z)
```

ANS: apple
banana
cherry

Output Variables

The Python Print() function is often used to output variables.

Example: `x = " Python is awesome"`
`Print (x)` → ANS: Python is awesome

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

Example:

```
X = "Python"
y = "is"
Z = "awesome"
Print(X,y,Z)   → ANS: Python is awesome
```

You can also use the + operator to output multiple variables:

Example:

```
X = "Python"
y = "is"
Z = "awesome"
Print(x+y+z) → ANS: Python is awesome
```

For numbers, the + character works as a mathematical operator:

Example:

```
X = 5
y = 10
Print(x+y)   → ANS: 15
```

In the Print() function, when you try to combine a string and a number with the + operator, Python will give you an error:

Example:

```
X = 5
y = "Kumar"
Print(x+y)   → ANS: TypeError
```

The best way to output multiple variable in the Print() function is to separate them with commas, which even support different data types:

Example:

```
X = 5
Y = "Kumar"
Print(x, y)
```

→ ANS: 5 Kumar

Global Variables

Variables that are created outside of a function (as in all of the examples above) are known as global variables. Global variables can be used by everyone, both inside of functions and outside.

Example:

Create a variable outside of a function, and use it inside the function.

```
x = "awesome"

def myfunc():
    Print("Python is " + x)

myfunc()
```

ANS: Python is awesome.