Python - Variables

Any value of certain type is stored in the computer's memory for processing. Out of available memory locations, one is randomly allocated for storage. In order to conveniently and repeatedly refer to the stored value, it is given a suitable name. A value is bound to a name by the assignment operator '='.

```
>>> myVar=21
```

Here myVar is an identifier (name) referring to integer value 21 (Python treats data value as a value). However, the same identifier can be used to refer to another value. For example, the below code will assign myVar as the name of a string value, as shown below.

```
>>> myVar="Steve"
```

So, the value being referred can change (or vary), hence it is called a variable. It is important to remember that a variable is a name given to a value, and not to a memory location storing the value.

Variables can be accessed by using their identifier (name) as shown below.

```
>>> myVar="Steve"
>>> myVar
'Steve'
```

Dynamic Typing

One of the important features of Python is that it is a dynamically-typed language. Programming languages such as C, C++, Java, and C# are **statically-typed languages**. A variable in these languages is a user-friendly name given to a memory location and is intended to store the value of a particular data type.

A variable in Python is not bound permanently to a specific data type. Instead, it only serves as a label to a value of a certain type. Hence, the prior declaration of variable's data type is not possible. In Python, the data assigned to a variable decides its data type and not the other way around.

In the following Python statements, a string value is assigned to a variable 'name'. We can test its type using the type() function.

```
>>> name="Steve"
>>> type(name)
<class 'str'>
```

Now the Python interpreter won't object if the same variable is used to store a reference to an integer.

```
>>> name=1234
>>> type(name)
<class 'int'>
```

We can see that the data type of a variable has now been changed to integer. This is why Python is called a **dynamically-typed language**.

Naming Conventions

Any suitable identifier can be used as a name of a variable, based on the following rules:

1. The name of the variable should start with either an alphabet letter (lower or upper case) or an underscore (_), but it cannot start with a digit.
2. More than one alpha-numeric characters or underscores may follow.
3. The variable name can consist of alphabet letter(s), number(s) and underscore(s) only. For example, myVar, MyVar, _myVar, MyVar123 are valid variable names but m*var, my-var, 1myVar are invalid variable names.
4. Identifiers in Python are case sensitive. So, NAME, name, nAME, and nAmE are treated as different variable names.

Starting the name with a single or double underscore has a special meaning in Python. More about this will be discussed in the chapter on object-oriented Programming.