Python is dynamically typed, which means that you don't have to declare what type each variable is. In Python, variables are a storage placeholder for texts and numbers. Variables are always assigned with the equal sign, followed by the value of the variable. Python reserved words can not be used as variable names. Variable values can be changed later on during the flow of a program.

You can use any letter, the special characters "_" and every number provided you do not start with it. To improve readability it is common practice to use lowercase with words separated by underscores

Note that variable names are case sensitive.

White spaces and signs with special meanings in Python, as "+" and "-" are not allowed.

Variable Assignment

In [16]:
```
#Use the assignment operator(=) to assign a value to a variable.
a = 67
```

In [17]:
```
print('a = ', a) #Let's check the value (a) holds after our assignment above.
```

```
a =  67
```

In [11]:
```
# Pyhton is case sensitive
a = 2
A = 4
#You will see the result are both different
print('a = ', a)
print('A = ', A)
```

```
a =  2
A =  4
```

Variable values can be changed later on during the flow of a program.

In [12]:
```
x = 4
y = x
x = 6
```

In [13]:
```
print ('x = ', x) #You notice variable x will become 6
print ('y = ', y) #but variable y will still hold the initial value of x
```

```
x =  6
y =  4
```

Naming Variables

In [18]:
```
@x = 5 #You will see that variables can't begin with some set of special chara
cters like @,&,%,$ etc.
```

```
  File "<ipython-input-18-ed75fe86e8fd>", line 1
    @x = 5 #You will see that variables can't begin with some set of special
 characters like @,&,%,$ etc.
       ^
SyntaxError: invalid syntax
```

In [22]:
```
#However your can begin with an underscore _
_b = 5
print ('_b = ', _b)
```

```
_b =  5
```

In [23]:
```
#You can't begin variable name with a number
1x = 4
```

```
  File "<ipython-input-23-b39722102d8c>", line 2
    1x = 4
     ^
SyntaxError: invalid syntax
```

Data Types in Python

In [27]:
```
x = 123                         # integer

y = 3.14                        # double float
z = "hello"                     # string
a = [0,1,2]                     # list
b = (0,1,2)                     # tuple
```

In [29]:
```
type(x)
```

Out[29]:
```
int
```

In [30]:
```
type(y)
```

Out[30]:
```
float
```

In [31]:
```
type(z)
```

Out[31]:
```
str
```

In [32]:
```
type(a)
```

Out[32]:
```
list
```

In [33]:
```
type(b)
```

Out[33]:
```
tuple
```

simultaneously multiple variable assignments.

In [34]:
```
a = b = c = 1 #Variable a,b and c are assigned to the same memory location,wit
h the value of 1
```

In [35]:
```
print ('a =', a)
```

```
a = 1
```

In [36]:
```
print ('b =', b)
```

```
b = 1
```

In [37]:
```
print ('c =', c)
```

```
c = 1
```