In [0]:
```
#This is a supplementary material to the lecture "Python Basics" to quickly revise, whenever needed
```

In [2]:
```
#variables
#you can create a variable as follows
a = 5                           #variable a is containing integer 5
print(a, type(a))
#no need to specify the data type like in other languages like c++ or java
#the same variable can be re-assigned another value of any other type as well
a = 'this is a sample text'     #variable a has been re-assigned to contain a string, string can be sp
ecified in single or double quotes, but they have to be consistent
print(a, type(a))
```

```
5 <class 'int'>
this is a sample text <class 'str'>
```

In [3]:
```
#taking input from user
#by default, the input taken from user is taken as string
print('input two numbers')
a = input()
```

```
input two numbers
5 10
```

In [4]:
```
#let's print a and see what does it contain
print(a, type(a))
```

```
5 10 <class 'str'>
```

In [5]:
```
#so we can see the the complete input has been taken as string
#we can now process it to extract the two numbers from this string as follows:
list1 = a.split(' ')     #this will split the input string across whitespaces and give list of strings
list1
```

Out[5]:
```
['5', '10']
```

In [6]:
```
#on a side note, list in python is just collection of data (may not be of same type)
example_list = [2, 'hello', 4.56, 7.0]
print(example_list)
#indexing of the elements in the list starts from 0
#any element can be acessed by its index like this
print(example_list[1])
#any element at a particular index can be updated to new value
example_list[1] = 'hey'
print(example_list[1])
```

```
[2, 'hello', 4.56, 7.0]
hello
hey
```

In [7]:
```
#we can now extract these two nos. as integer from this list
num1 = int(list1[0])
num2 = int(list1[1])
print(num1, num2)
```

```
5 10
```

In [8]:
```
#the above steps for taking 2 numbers input can be written in one line of code as follows:
print('enter two numbers')
n1, n2 = [int(item) for item in input().strip().split(' ')]   #here it means that for every item in th
e list created by the split(), convert it to an int, (we could specify any other expression also for e
ach item as per use case)
print('n1: ', n1, 'n2: ', n2)
```

```
enter two numbers
10 20
n1:  10 n2:  20
```

In [0]:
```
#string slicing
str = "this is a sample text"
```

In [10]:
```
#now, we can have a substring of the string str using slicing as follow:
slice1 = str[2:7]   #this will return the substring of str from index 2 till index 6 (end_index-1)
slice1
```

Out[10]:
```
'is is'
```

In [11]:
```
#there are other variants of this, if you want to specify, just the starting index or just the end ind
ex, you can do so
slice2 = str[:10]     #this will return substring of str from index 0 till index 9
slice3 = str[5:]      #this will return substring of str from index 5 to the end of the string str
slice4 = str[:]       #this will return complete string str, there is not much use of it as complete s
tring is str itself
print(slice2)
print(slice3)
print(slice4)
```

```
this is a
is a sample text
this is a sample text
```

In [0]:
```
#one thing to remember is that string are immutable, means you can not change a string, it can be reas
signed to a new string
#str[5] = 'q'  #will produce error
str = 'new sample text'   #this will work fine like any other assignment to the variable
```

In [13]:
```
#Tuples
#Tuples in python are like list as collection of data with one major difference is that, tuples are im
mutable
#and they are specified in parentheses instead of square brackets as in the case of list
#i.e. their elements can't be updated once initialized
example_tuple = (2, 'hello', 4.56, 7.0)
print(example_tuple)
#any element can be acessed by its index like this
print(example_tuple[1])
#following assignment will give an error
#example_tuple[1] = 'hey'
#however, it can be reassigned to new tuple as any other re-assignment
example_tuple = (4.0, 'hey', 65)
print(example_tuple)
```

```
(2, 'hello', 4.56, 7.0)
hello
(4.0, 'hey', 65)
```

In [0]:
```
#slicing in tuple works same as, we have seen for the strings
#give it a try by yourself
#Thanks, Happy coding!
```

In [0]:
```
#To download .ipynb notebook, right click the following link and click save as
https://ninjasfiles.s3.amazonaws.com/0000000000003216.ipynb
```