Introduction to Python

Class 5a

Functions

A function definition is introduced using a keyword def. def is followed by a function name and parenthesised list of formal parameters. Python doesn't require a return type explicitly, the default return type is NONE.

```
def summing(a, b, c):
    print(a + b + c)

summing(2,3,4)
```

```
def fibonacci(n):
    a, b=0, 1
    while a<n:
        print(a, end=' ')
        a, b=b, a + b

fibonacci(100)
```

In python, the parameters are passed by reference, this means that, **any changes made** to the values of the parameters inside a function, **will reflect back** on the original parameters too.

```
def function(li,a):
    li.append(3)
    a=5
    print(li,a)
list1=[1,2]
b=10
function(list1,b)
print(list1,b)
```

There are three types of arguments passing in python:

1. **Required arguments:** Arguments that must be passes to a function.
   The number of parameters in the function will be equal to the number of parameters

```
def sum_of_three(a,b,c):
s=a+b+c
print(s)
num1=23
num2=43
```

```
num3=12
sum_of_three(num1,num2,num3)
```

2. **Default arguments:** Arguments which assume a default value, when the value is not defined

```
def student_info(name,age,gender="M"):
    print("Name: ",name)
    print("Age: ",age)
    print("Gender: ",gender)
student_info("Rahul",19)
```

3. **Variable length arguments:** A function which works for varying number of arguments. The number of arguments are not defined unlike required and default arguments.

```
def make_sentence(*words):
    sentence=""
    for word in words:
        sentence+=word+' '
    print(sentence)
make_sentence("Hello", "World!") #Output: Hello World!
make_sentence("How","are","you?") #Output: How are you?
```

Anonymous Functions:

Anonymous functions are not defined by the general def keyword but instead defined by the lambda keyword. Such functions cannot contain multiple statements and cannot access any variables other than those in their parameter list. Lambda functions can take any number of parameters but can return only one.

Lambda definition doesn't include a "return statement, it always contains an expression which is returned. We don't have to assign it to a new variable at all.

```
val_of_square= lambda x: x*x
print(val_of_square(7))
print(val_of_square(13))
```

Anonymous functions are very useful mapping functions for data processing libraries such as **pandas and numpy.**

Recursive function:

A recursive function defined is a function defined in terms of itself via self-referencing. This means that the function will continue to call itself and repeat it's behaviour until some expression is met to return the result.

```
def fact(n):
    if n==1:
```

```
    return 1
  else:
    return n*fact(n-1)

print(fact(5))
```