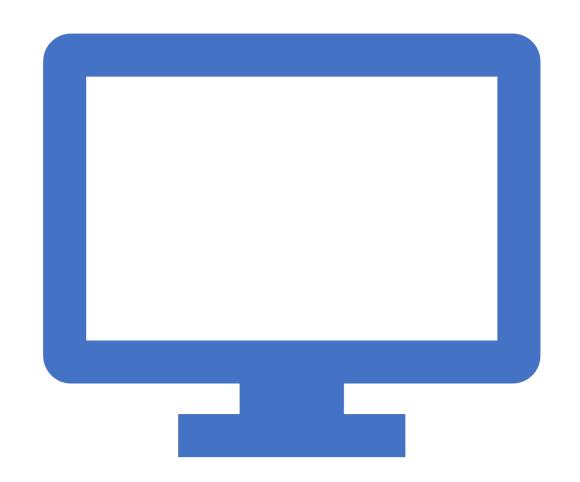


Python Part F

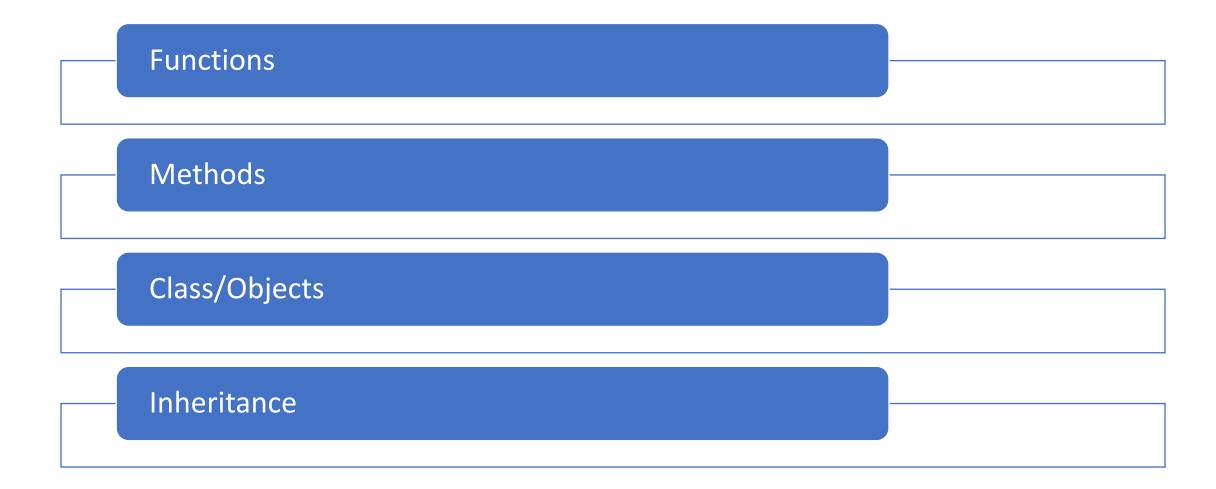
Part of Future Connect Media's IT Course

By Abhishek Sharma





Topics to be covered:



Python functions:

```
Future
Connect
Media
```

```
def my_function(firstname, surname):
    print(' my first name is ' + firstname + ' and surname name is ' + surname)
my_function('alex', 'canon')

my first name is alex and surname name is canon
```

Arbitrary argument:

```
def my_function(*cars):
    print("The fastest car is " + cars[0])
my_function("bmw", "jaguar", "mercedes")
The fastest car is bmw
```

Default parameter value:

```
def my_function(car='bmw'):
    print("The fastest car is " + car)

my_function("jaguar")
my_function()
my_function('mercedes')
```

```
The fastest car is jaguar
The fastest car is bmw
The fastest car is mercedes
```



Passing list as an argument:

```
def my_function(cars):
    for x in cars:
        print(x)
colour=['black','white','silver']
my_function(colour)
```

```
black
white
silver
```

Lambda function:

```
>>> a= lambda x, y: x*y
>>> print(a(7,8))
56
```



Python classes/objects:



```
class cars:
    def __init__(self, model, colour):
        self.model = model
        self.colour = colour
    def myfunction(self):
        print('The model of the car is ' + self.model)
a= cars('bmw 3 series', 'white')
a.myfunction()
```

The model of the car is bmw 3 series

#__innit__() function is used to assign value to the object.

#Current instance of the class is referenced by self parameter.

Python inheritance:

```
class cars:
    def __init__(self, model, colour):
        self.model = model
        self.colour = colour
class bmw(cars):
    def __init__(self, model, colour, interiorcolour):
        super(). init (model, colour)
        self.interiorcolour = interiorcolour
    def carappearance(self):
        print('car model is '+self.model+' ,colour is '+self.colour+
        ' and interior colour is '+self.interiorcolour)
x=bmw('bmw 3 series','white','red')
x.carappearance()
car model is bmw 3 series ,colour is white and interior colour is red
```

#parent class is 'cars'

#child class is 'bmw'

#super(): This function helps to inherit all the methods and properties from it's parent class. In this case 'class bmw' inherites from 'class cars'



 Python iterators: An object with a countable number of values is an iterator.



```
class num:
    def __iter__(self):
        self.x=1
        return self
    def __next__(self):
        if self.x<=15:</pre>
            y= self.x
            self.x += 2
            return y
        else:
            raise StopIteration
myclass = num()
myiter= iter(myclass)
                                     11
for y in myiter:
    print(y)
```

#__iter__(): This method is used for initializing.

#__next__(): This method is used to return the next value.

#StopIteration is used to stop the iteration from going on forever.