OBJECT ORIENTED PROGRAMMING WITH PYTHON

Second Class

1st Semester

FUNCTIONS

```
def  greet_member():
    print ('Hi there!')
    print(' Welcome to our class')


print("Start")
greet_member()
print("Finish")
```

- Function's name should be lower characters
- If multiple words separate them with underscore _
- Always and always use meaningful descriptive names
- Use parenthesis () followed by :

PARAMETERS

We can add the name of the user to add it to the greet messages

```
def  greet_member(name):
    print (f'Hi {name} !')
    print(' Welcome to our class')


print("Start")
greet_member("John")
greet_member("Mary")
print("Finish")
```

```
Start
Hi John
Welcome to our class
Hi Mary
Welcome to our class
Finish
```

KEYWORD ARGUMENTS

```
def  greet_member(first_name,last_name,age):
    print (f"Hi {first_name} {last_name} !")
    print(" Welcome to our class..")


print("Start")
greet_member(age=30,last_name='Smith',first_name='Sara')
print("Finish")
```

Here the sequence of the **keyword arguments** is not important, not like the Positional **arguments mentioned** earlier

KEYWORD ARGUMENTS

Note1:

We can use the **keyword arguments** when we have set of numerical arguments like shipping number and discount, in order to make the code **more readable.**

calc_cost (total=50,shipping=5,discount=0.1)

Note2:

We can use the **keyword arguments** and the **positional arguments** at the same time, only if the **positional arguments** is mentioned first.

greet_member("sara",age=30,last_name="Smith")

RETURN STATEMENT

Return function used to return value of the function, now let's write a function that calculate the square of the number

```
def  square(number):
        return  number*number


result= square(3)
print(result)
```

```
def  square(number):
        return  number*number


print(square(3))
```

RETURN STATEMENT

Now, if we removed the return from the previous function and execute

```
def  square(number):
        number*number


print(square(3))
```

None

From this we conclude that, by default all functions in Python return None. And we can change that using return statement

CREATING REUSABLE FUNCTIONS

Exercise:

Reorganize the Emoji's Converter code to function, since this program can be used in chat application, email applications and so on.

```
def emoji_converter(message):
    words = message.split(' ')
    emojis = {
        ":)":"☺",
        ":(":"☹"
    }
    output =""
    for word in words:
        output += emojis.get(word, word) + ' '
    return output


message=input("> ")
emoji_message=emoji_converter(message)
print(emoji_message)
```