1. Create a function named ‘factor’ that can only accept 1 argument. The function should return the factorial of that number.

```
# 5 --> 5*4*3*2*1
def factor(num):
    fact = 1
    if num<0:
        print('The factorial does not exist for negative numbers')
    elif num==0:
        print('The factorial of zero is 1')
    else:
        for i in range(1,num+1):
            fact = fact*i  # 1*1,1*1*2,1*1*2*3,
        print('The factorial of', num, 'is', fact)

factor(3)
```

```
 The factorial of 3 is 6
```

```
def factor(num):
    fact=1
    if num<0:
        print("Factorial has to be a positive number")
    elif num==0:
        print("Factorial of 0 is 1")
    else:
        for i in range(1,num+1):
            fact=fact*i
        print("Factor of ",num," is",fact)
```

```
def factor(num):
    fact = 1
    if num<0:
        print('The factorial does not exist for negative numbers')
    elif num==0:
        print('The factorial of zero is 1')
    else:
        for i in range(1,num+1):  # range(1,n) -->  1 to n-1
            fact = fact*i
        print('The factorial of', num, 'is', fact)
factor(int(input()))
```

```
 3
 The factorial of 3 is 6
```

2. Create a function named ‘check_string’, the function should accept a string data from the user and the function should check if the user input contains the letter ‘s’ in it. If it contains

the letter ‘s’ then print- ‘The string is containing the letter ‘s”, if not then print- ‘The string doesn’t contain the letter ‘s”.

```
def check_string():
    s = input('Enter a string value: ')
    c = 0
    for i in s:
        if i=='s':  # i= p, s==s
            print('The string contains s')
            c = c+1
            print(c)
        else:
            print('The string doesnot contain the letter s')


check_string()
```

```
Enter a string value: sushma eat's sushi in shop and sushi was super tasty
The string contains s
1
The string doesnot contain the letter s
The string contains s
2
The string doesnot contain the letter s
The string doesnot contain the letter s
The string doesnot contain the letter s
The string doesnot contain the letter s
The string doesnot contain the letter s
The string doesnot contain the letter s
The string doesnot contain the letter s
The string doesnot contain the letter s
The string contains s
3
The string doesnot contain the letter s
The string contains s
4
The string doesnot contain the letter s
The string contains s
5
The string doesnot contain the letter s
The string doesnot contain the letter s
The string doesnot contain the letter s
The string doesnot contain the letter s
The string doesnot contain the letter s
The string doesnot contain the letter s
The string contains s
6
The string doesnot contain the letter s
The string doesnot contain the letter s
The string doesnot contain the letter s
The string doesnot contain the letter s
The string doesnot contain the letter s
The string doesnot contain the letter s
The string doesnot contain the letter s
The string doesnot contain the letter s
```

```
The string contains s
7
The string doesnot contain the letter s
The string contains s
8
The string doesnot contain the letter s
The string doesnot contain the letter s
The string doesnot contain the letter s
The string doesnot contain the letter s
The string doesnot contain the letter s
The string contains s
9
The string doesnot contain the letter s
The string contains s
10
The string doesnot contain the letter s
The string doesnot contain the letter s
The string doesnot contain the letter s
The string doesnot contain the letter s
```

3. Create a class named ‘student’ and inside the class, create a function named ‘fun1’- this method should accept the user defined input and return that value. a. Create another method named- message() and that method should print the user defined input that we have defined in ‘fun1’.

```
class student:
    def __init__(self):
        self.n = input()
    def fun1(self):
        #n = int(input('Enter a value: '))
        return self.n
    def message(self):
        #k = self.fun1()
        return self.n
ob = student()
ob.message()
```

```
Shilpa
'Shilpa'
```

```
class student:
    def __init__(self):
        self.name=input("name of student")
    def disp (self):
        return(self.name)
o = student()
o.disp()
```

```
name of studentSumana
'Sumana'
```

```
class student:
        def fun1(self):
            x= int(input('please enter the input:'))
            return x
        def message(self):
            k = self.fun1()
            return k

obj = student()
obj.message()
```

```
please enter the input:5
5
```

4. Create a lambda function that should double or multiply the number (that we will be passing in the lambda function) by 2. Store the lambda function in a variable named ‘double_num’.

```
double_num = lambda n:n*2
double_num(5)
```

```
10
```

```
double_num(int(input()))
```

```
12
24
```

```
# palindrome --> Naman --> naman
```

5. Take the user input string and check whether that string is palindrome or not.

```
s = input('Enter a string: ')
s = s.lower()
if s==s[::-1]:
    print('s is palindrome')
else:
    print('s is not a palindrome')
```

```
Enter a string: Madam
s is palindrome
```

6. Create a class named ‘Super’ and inside that class define a user-defined function named fun1. a. Inside the ‘fun1’ function, pass the message “This is function 1 in the Super class.” in the print statement.

```
class Super:
    def fun1(self):
        print('This is function1 in the super class')
o = Super()
o.fun1()
```

```
This is function1 in the super class
```

7. Create another class named ‘Modified_Super’ and inherit this class from the Super class.

a. Inside the Modified_Super class, create a function named ‘fun1’ and pass the following message inside the print statement: ‘This is function 1 in the Modified Super class.’

b. Create another user-defined function named ‘fun2’ and pass the message: ‘This is the 2nd function from the Modified Super class’ in the print statement.

c. After that, now create an object for the Modified_Super class and call the fun1().

```
class Modified_Super(Super):
    def fun1(self):
        print('This is function1 in modified super class')
        Super.fun1(self)
    def fun2(self):
        print('This is function2 from modified super class')
ob = Modified_Super()
ob.fun1()
ob.fun2()
```

```
This is function1 in modified super class
This is function1 in the super class
This is function2 from modified super class
```

```
class Super:
    def __init__(self,name1,name2):
        self.name1="This is funtion 1 in the Modified Super Class"
        self.name2="This is funtion 2 in the Modified Super Class"

class Modified_Super(Super):

        def func1(self):
            return self.name1


        def func2(self):
            return self.name2

obj=Modified_Super('a','b')
obj.func1()
```

```
'This is funtion 1 in the Modified Super Class'
```

8. Create 2 methods named ‘Hello’. In the 1st Hello method, pass only one argument and pass this message: ‘This function only has 1 argument’. And in the 2nd Hello method, pass two arguments and pass this message: ‘This function has 2 arguments’.

a. Try to call both the methods and analyze the output of both the methods.

```
def Hello(a):
    print('This function only has one argument.')
def Hello(a,b):
    print('This function has two arguments.')
```

```
Hello(5,4)
```

```
 This function has two arguments.
```

9. Create a method named ‘Sum’ that can accept multiple user inputs.

Now add those userdefined input values using for loop and the function should return the addition of the numbers.

```
def sum(*num):   # 4,5,6
    total = 0
    for i in range(0, len(num)):  # len(num)=5, range(0,5)
        total = total + num[i]  # total = 0 + 2, 0 + 2 + 3,.......
    print(type(num))
    return total
```

```
sum(2,3,4,5,6)
```

```
 <class 'tuple'>
 20
```

10. Create a class named ‘Encapsulation’:

a. Inside the class, first create a constructor. Inside the constructor, initialize originalValue variable as 10.

b. After creating the constructor, define a function named ‘Value’ and this function should return the variable that we have initialized in the constructor.

c. Now create 2nd function named setValue, and pass an argument named ‘newValue’. The task of this function will be to replace the value of the originalValue variable by the value of the newValue variable.

```
class Encapsulation:
```

```
    def __init__(self):
        self.originalValue = 10
    def Value(self):
        return self.originalValue
    def setValue(self, newValue):
        self.originalValue = newValue
        return self.originalValue


obj = Encapsulation()


obj.setValue(int(input()))
```

```
 7
 7
```

```
obj.Value()
```

```
 7
```

```
ob2 = Encapsulation()


ob2.Value()
```

```
 10
```

```
class Encapsulation:

    def __init__(self):
        originalValue = 10

    def value(self):
        return self.originalValue

    def setValue(self, newValue):
        self.originalValue = newValue
        return self.originalValue


obj = Encapsulation()
# obj.value()
obj.setValue(5)
obj.value()
```

```
 5
```