Module 2: Function and Class

Assignment Contact us: support@intellipaat.com / © Copyright Intellipaat / All rights reserved Intel iPaat Python for Data Science Certification Course Problem Statement: You work in XYZ Corporation as a Data Analyst. Your corporation has told you to work on functions and classes. Tasks To Be Performed:

1. Create a function named 'factor' that can only accept 1 argument. The function should return the factorial of that number.
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"'.
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'.
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'.
5. Take user input string and check whether that string is palindrome or not.

In [ ]:
```
##1,
def factor(n):
    if n == 1:
        return 1
    else:
        return n * factor(n-1)
n =int(input('enter the number:'))
factor(n)
```

In [ ]:
```
##2,
def check_string(string):
    if 'sw' in string:
        return 'the string contains letter "s"
    else:
        return 'the string does not contains letter "s"
    check_string()
```

In [ ]:
```
##3,
class student:

    def fun1(self,n):

        self.n = n
        return n
    def massage(self):
        print(n)
obj1=student()
obj.fun1(120)
obj.massage()
```

In [ ]:
```
##4,
double_num =lambda n:n*2
double_num(8)
```

In [ ]:
```
##5,
n=input()
if n == n[::-1]:
    print("string is pal")
else:
    print("string is not pal")
```

Module 2: Functions

Assignment Contact us: support@intellipaat.com / © Copyright Intellipaat / All rights reserved Intel iPaat Python for Data Science Certification Course Problem Statement: You work in XYZ Corporation as a Python Developer. Your corporation has told you to work with the methods, and functions in Python. Tasks To Be Performed:

1. 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.
2. 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 2 nd 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().
3. Create 2 methods named 'Hello'. In the 1st Hello method, pass only one argument and pass this message: 'This function is only having 1 argument'. And in the 2nd Hello method, pass two arguments and pass this message: 'This function is having 2 arguments'. a. Try to call both the methods and analyze the output of both the methods.
4. Create a method named 'Sum' that can accept multiple user inputs. Now add those user defined input values using for loop and the function should return the addition of the numbers.
5. 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. Contact us: support@intellipaat.com / © Copyright Intellipaat / All rights reserved Intel iPaat Python for Data Science Certification Course c. Now create a 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 newValue variable

In [ ]:
```
##question 1 and 2
class super:
    def fun1(self):
        print("this is the function 1 in the super class")
class modified_super(super):
    def fun1(self):
        print("this is the function 1 msc")
    def fun2(self):
        print("this is the 2 nd function from the modifier super class")
obj=modified_super()
obj.fun1()
  ## both function have same name different values but it is execute recent one
```

In [ ]:
```
##3,
def hello(org):
    print("this function only has 1 arguments")
def hello(org1,org2):
    print("this function has 2 arguments")
hello(4,5)
```

In [ ]:
```
##4,
def sum1(list1):
    add=0
    for element in list1:
        add=add + int(element)
    return add
sum1(list(input().split()))
```

In [ ]:
```
##5,
class encapsulation:
    def__init__(self):
        self.original value=10
    def value(self):
        return self.original value
    def setvalue(self.newvalue):
        print('original',self.originalvalue)
        self.originalvalue=newvalue
        print(new;self.originalvalue)
obj1=encapsulation()
obj.setvalue()
```

Module 2: Import Assignment

Contact us: support@intellipaat.com / © Copyright Intellipaat / All rights reserved Intel iPaat Python for Data Science Certification Course Problem Statement: You work in XYZ Corporation as a Data Analyst. Your corporation has told you to work on importing various functions. Tasks To Be Performed:

1. Create a Python file named Module: a. Inside the file, define 4 methods named – addition, subtraction, multiplication, and division. b. Each method should only accept 2 arguments and should return the result of operation performed in each method. For e.g., addition() should return the sum of two arguments. c. Save the Module file in .py format.
2. Open a new python file and import the Module.py file a. Now call the 4 methods from the Module.py file, i.e., addition(), subtraction(), multiplication(), and division().
3. From the Module file, import only the addition() and pass the arguments so that it can display the result from the method.
4. From the Module file, import only the subtraction() and pass the arguments so that it can display the result from the method.
5. From the Module file, import both the multiplication() and division() and pass the arguments so that it can display the result from the methods

In [ ]:
```
##1,

a=float(input("enter the first number here"))
b=float(input("enter the second number here"))
add=a+b
sub=a-b
mul=a*b
div=a/b
print("addition is=",add)
print("subraction is=",sub)
print("multiplication is=",mul)
print("division is=",div)
```

Module 2: Inheritance

Assignment Contact us: support@intellipaat.com / © Copyright Intellipaat / All rights reserved Intel iPaat Python for Data Science Certification Course Problem Statement: You work in XYZ Corporation as a Data Analyst. Your corporation has told you to work with the inheritance of the classes. Tasks To Be Performed:

1. Create a class named parent_Class and inside the class, initialize a global variable num as 10 a. Create another class named child_Class and this class should be inherited from the parent class. b. Now create an object for the child_Class and with the help of child_Class object, display the value of 'num'.
2. Create three classes named A, B and C a. Inside the A class, create a constructor. Inside the constructor, initialize 2 global variables name and age. b. After initializing the global variables inside the constructor, now create a function named 'details' and that function should return the 'name' variable. c. Inside the B class, create a constructor. Inside the constructor, initialize 2 global variables name and id. d. After initializing the global variables inside the constructor, now create a function named 'details' and that function should return the 'name' variable. e. The C class should inherit from class A, and B. Inside the class C, create a constructor, and inside the constructor, call the constructor of class A. f. Now, create a method inside the class C, as get_details, and this function should return the value of name. g. Adast, create an object of class C, and with the help of the object, call the get_details(). Contact us: support@intellipaat.com / © Copyright Intellipaat / All rights reserved Intel iPaat Python for Data Science Certification Course h. After that, create another class named 'Super' and inside that class, create a method named 'final', and pass the below message in the print()- 'This is the final method from the super class.'. c. Now, create an object for the Super class and call all the 3 user defined methods, i.e., first(), second(), and final().
3. Create a class named 'Parent', and inside the class, create a function named 'fun1' and pass the following message in the print()- 'This is the message from the fun1.' a. Now create a class named 'Child1' and inside the class, create a method named 'fun2' and pass the following message in the print()- 'This is the message from the fun2'. b. After that, create another class named 'Child2' and inside the class, create a method named 'fun3' and pass the following message in the print()- 'This is the message from the fun3'. c. Now, create an object of Child2 class and with the help of the object, call the 'fun1' method from the 'Parent' class.
4. Create a class named 'Parent', and inside the class, create a function named 'fun1' and pass the following message in the print()- 'This is the message from the fun1.' a. Now create a class named 'Child' and inside the class, create a method named 'fun2' and pass the following message in the print()- 'This is the message from the fun2'. b. After that, create another class named 'Child2' and inside the class, create a method named 'fun3' and pass the following message in the print()- 'This is the message from the fun3'. c. Now create an object of Hybrid class and with the help of the object, call the 'fun1, 'fun2' and 'fun3' methods

In [ ]:
```
##1,
class parent_class:
    num=10
class child_class(parent_class):
    pass
child_obj=child_class()
print(child_obj.num)
```

In [ ]:
```
##2,
class A:
    def __init__(self):
        self.name = "vikas"
        self.age = 25

    def details(self):
        return self.name

class B:
    def __init__(self):
        self.name = "Doe"
        self.id = 154

    def details(self):
        return self.name

class C(A, B):
    def __init__(self):
        A.__init__(self)

    def get_details(self):
        return self.name

obj = C()
print(obj.get_details())
```

In [ ]:
```
##3,
class sub1:
    def first(self):
        print("This is the first function from Sub 1 class")
class sub2:
    def second(self):
        print("This is the second function from the Sub 2 class")
class Super(sub1,sub2):
    def final(self):
        print("This is the final method from the super class")
super_obj=Super()
super_obj.first()
super_obj.second()
super_obj.final()
```

In [ ]:
```
##4,
class parent:
    def fun1(self):
        print("This is the message from the fun1")
class child1:
    def fun2(self):
        print("This is the message from the fun2")
class child2(child1,parent):
    def fun3(self):
        print("This is the message from the fun3")
obj1=child2()
obj.fun1()
```

In [ ]:
```
##5,
class parent:
    def fun1(self):
        print("This is the message from the fun1")
class child:
    def fun2(self):
        print("This is the message from the fun2")
class hybrid(parent,child):
    def fun3(self):
        print("This is the message from the fun3")
obj_hybrid=hybrid()
obj_hybrid.fun1()
obj_hybrid.fun2()
```

Module 2: Case Study - 1

Contact us: support@intellipaat.com / © Copyright Intellipaat / All rights reserved Intel iPaat Python Certification Course Problem Statement: Consider yourself to be Sam who is a software engineer. He has been asked to build an employee management app. You have to use OOPs concept to build a few classes as outlined in the steps below. Tasks To Be Performed:

1. Create a class named Employee, with a constructor '**init**' method that accepts name and salary as parameters and set properties named name and salary.
2. Define **str** method in Employee class so that when someone tries to print the object the string Name: employee_name, Salary: employee_salary is printed with the actual employee name and salary.
3. Create another class named *Calculator*, with methods to add, subtract, multiply and divide two numbers.
4. These methods take two numbers as parameters.
5. These methods will be called by a method named execute command.
6. Execute command takes in 3 parameters command which is string that can be either 'add', 'sub', 'mul', 'div', and two numbers and it will call the appropriate method based on command parameter.

In [ ]:
```
##1,
class Employee:
    def __init__(self, name, salary):
        self.name = name
        self.salary = salary
```

In [ ]:
```
##2,
class Employee:
    def __init__(self, name, salary):
        self.name = name
        self.salary = salary

    def __str__(self):
        return f'name{self.name},{self.salary})'
Employee = Employee('vikas', 25000)
print(Employee)
```

In [ ]:
```
##3,
class Calculator:
    def add(self, x, y):
        print(x+y)


    def subtract(self, x, y):
        print(x-y)

    def multiply(self, x, y):
      print(x*y)

    def divide(self, x, y):
        print(x/y)

obj=Calculator()
obj.add(2,3)
obj.subtract(2,3)
obj.multiply(2,3)
```

In [ ]:
```
##4,5,6,
def add(num1, num2):
    return num1 + num2

def sub(num1, num2):
    return num1 - num2

def mul(num1, num2):
    return num1 * num2

def div(num1, num2):
    return num1 / num2

def execute_command(command, num1, num2):
    if command == 'add':
        return add(num1, num2)
    elif command == 'sub':
        return sub(num1, num2)
    elif command == 'mul':
        return mul(num1, num2)
    elif command == 'div':
        return div(num1, num2)
    else:
        return "Invalid command"
```