

**Usman Institute of Technology**

**Final Lab Examination**

**Spring 2020 Semester**

**Course**: – Introduction to Object Oriented Programming **Course Code**: CS-121

**Date**: August 18, 2020 **Batch**: 19B Software Engineering

**Paper: A (Roll No: 15B-085-SE Till 19B-030-SE)**

**Submission:** Sunday 16th August 2020, before 11:59 PM

**This is an open book examination**

|  |  |
| --- | --- |
| Student’s Name | MUHAMMAD OMER SIDDIQUI |
| UIT Roll Number | 19B-004-SE |
| Course Title | Introduction to Object Oriented Programming |
| Course Code | CS-121 |
| Date of the Examination | 8/20/2020 |
| Section | A |
| Faculty Member’s Name | FATIMA SHAKEEL |

**Ethical Compliance / Integrity Undertaking**

**I guarantee that all this is my independent work and is done without any unauthorized help. All activities are completed with full adherence to the “Ethics Policy” of the Institute. I understand that any breach would result in disciplinary action against me as per the Institute rules.**

**Please make sure to check the following before you proceed.**

**I have read and understood the ethical compliance / integrity undertaking and will comply.**

Instructions

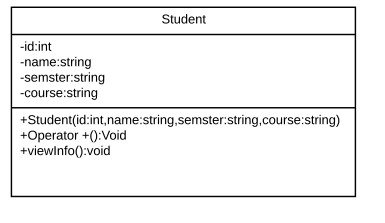
Please read the following instruction carefully before you proceed:

1. Please solve all questions on this word document and submit python code file.
2. Please download the paper immediately after it is made available to protect yourself from any power failure Internet issues.
3. Solve the paper on your computer and, once done, upload on Microsoft Teams assignment created to receive your paper. You would require the Internet only to download and upload the paper.
4. Once you are done with your paper, save it and upload.
5. This is an open book examination. You are allowed only to consult your textbook, reference books, and class notes. You are not allowed to use any other source.
6. All work submitted will tested with Turnitin. Any work considered suspicious may be tested through a viva.
7. Submit tasks in softcopy (code and screen shots in word document) and Code file on Sunday 16th August 2020 before 11:59 Pm. Failed to submit marked zero and no viva will be conducted and marked absent.
8. Code less than15% plagiarized will be marked out 30.
9. Code more than 15% plagiarized will be marked out of 15 (checking marks deducted to 50%)
10. Code more than 50% will be marked Zero. (Zero in viva and task both).
11. Viva will be conducted in groups time will be assigned later
12. Viva will be held on same lab timings.
13. Student need to share his/her screen, explain code and need to open his/her Web cam time of viva.
14. Student need to be present on time of viva otherwise marked absent.

**I have read and understood the instructions and will comply.**

****

**Task 1 (2 Marks):**



1. Implement above diagram.
2. Overload + to take names of courses in which student is enrolled.

class Student:

# initialization

def \_\_init\_\_(sf,id,name,semester,courses):

# attributes

sf.id =id

sf.name = name

sf.semester = semester

sf.course = courses

#methods

#performing method overloading

#concatenation of string

def \_\_str\_\_(sf):

return f"{sf.course}"

#addition

def \_\_add\_\_(sf, other):

sf.course = sf.course + other.course

return f"The name of the courses are {sf.course}"

#details

def View\_info(sf):

print("Identity: "+sf.id+"\nName: "+sf.name+"\nsemester: "+sf.semester+"\nCourse: "+sf.course)

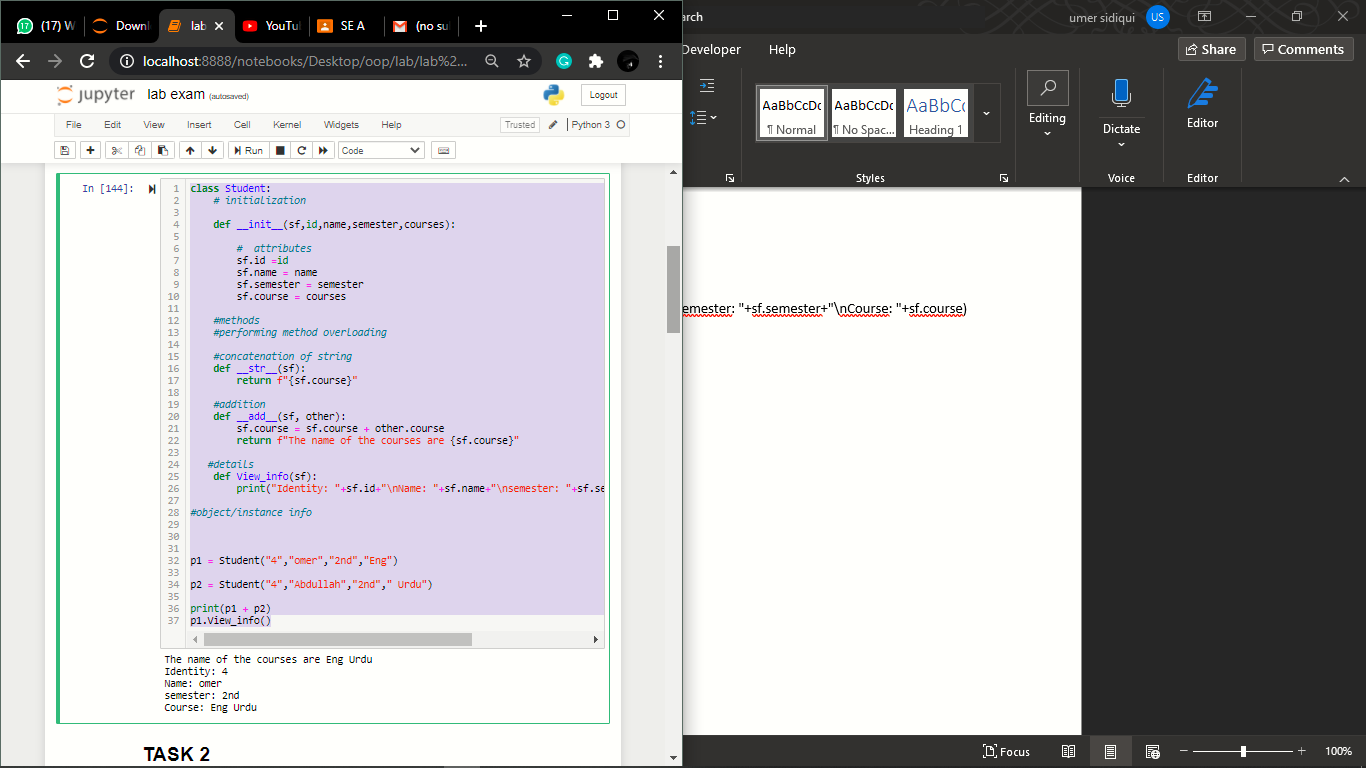
#object/instance info

p1 = Student("4","omer","2nd","Eng")

p2 = Student("4","Abdullah","2nd"," Urdu")

print(p1 + p2)

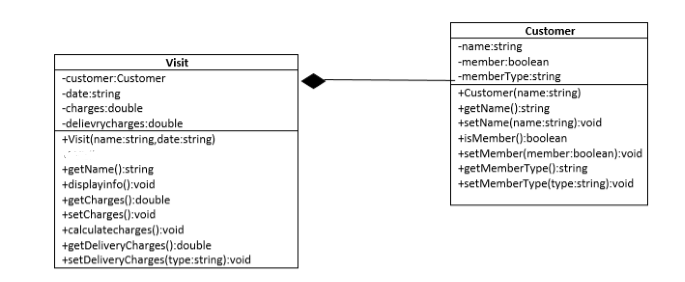
p1.View\_info()



**Task 2 (2 Marks):**

Create a class Visit and Customer and perform the composition

Create class Visit and Customer which contains instance variables and also implement following methods **printDetails()** to print the information



# creating visit class

class Visit:

#initialization

def \_\_init\_\_(sf,name,date,charges,deliverycharges,member,membertype):

#set attributes a/c to diagram

sf.name = name

sf.date = date

sf.charges = charges

sf.member = member

sf.membertype = membertype

sf.deliverycharges = deliverycharges\*2

sf.customer = Customer(sf.member,sf.membertype)

#info about the customer who visit

def Printdetails(sf):

print("Name: "+sf.name+"\nMember: "+sf.member+"\nMember Type: "+sf.membertype+

"\nDate: "+sf.date+"\nCharges: "+sf.charges+

"\nDelivery Charges: "+sf.deliverycharges)

# creating customer class

class Customer:

def \_\_init\_\_(sf,member,membertype):

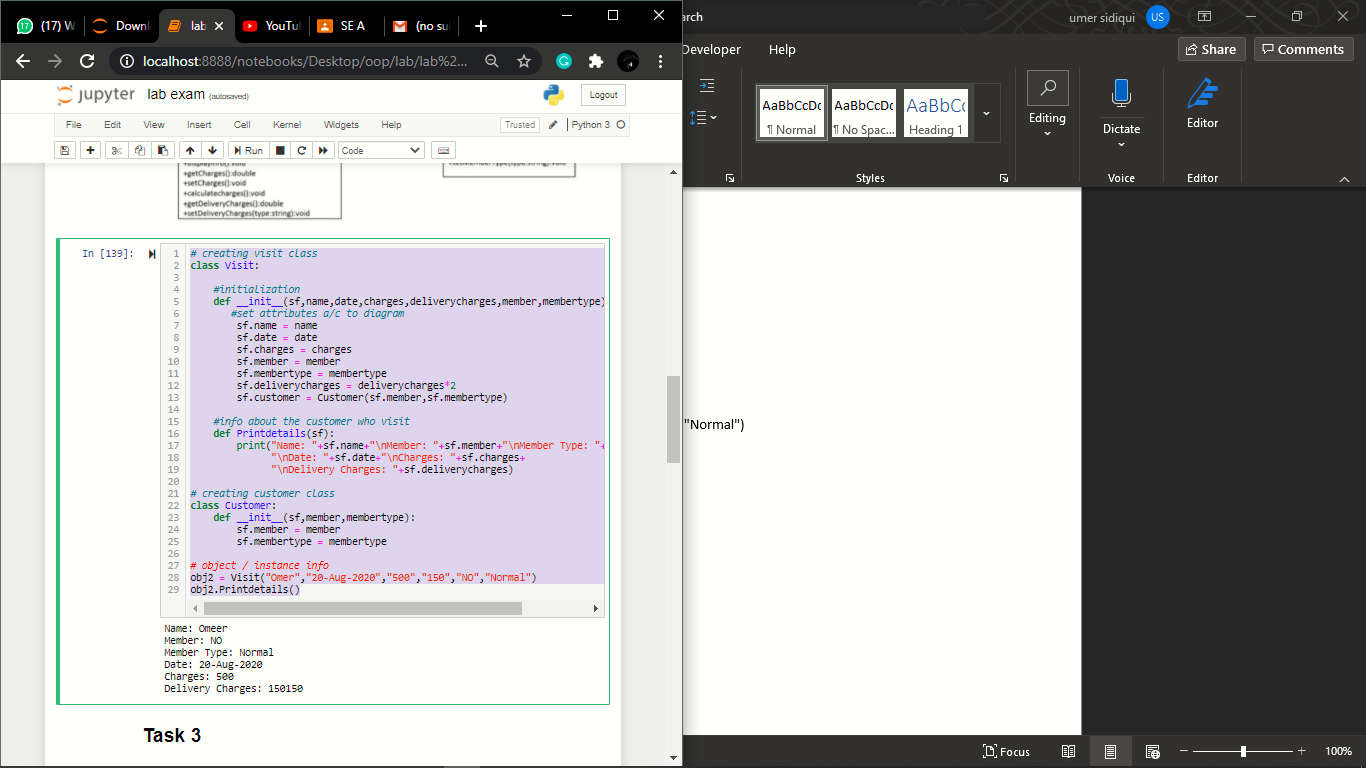
sf.member = member

sf.membertype = membertype

# object / instance info

obj2 = Visit("Omer","20-Aug-2020","500","150","NO","Normal")

obj2.Printdetails()



**Task 3 (8 Marks):**

Create 3 classes for given shapes and a parent class of Shape.

import math

# Creating shape class

class Shape:

def \_\_init\_\_(sf, color='black', filled=False):

sf.\_\_color = color

sf.\_\_filled = filled

def get\_color(sf):

return sf.\_\_color

def set\_color(sf, color):

sf.\_\_color = color

def get\_filled(sf):

return sf.\_\_filled

def set\_filled(sf, filled):

sf.\_\_filled = filled

# Creating rectangle class which inherit shape class

class Rectangle(Shape):

def \_\_init\_\_(sf, len, w):

super().\_\_init\_\_()

sf.len = len

sf.w = w

def get\_len(sf):

return sf.\_\_len

def set\_len(sf, len):

sf.\_\_len = len

def get\_br(sf):

return sf.\_\_w

def set\_br(sf, br):

sf.\_\_w = w

def get\_A(sf):

return sf.len \* sf.w

def get\_pm(sf):

return 2 \* (sf.len + sf.w)

# Creating circle class which inherit shape class

class Circle(Shape):

def \_\_init\_\_(sf, r):

super().\_\_init\_\_()

sf.\_\_r = r

def get\_r(sf):

return sf.\_\_r

def set\_r(sf, r):

sf.\_\_r = r

def get\_A(sf):

return math.pi \* sf.\_\_r \*\* 2

def get\_pm(sf):

return 2 \* math.pi \* sf.\_\_r

# Creating triangle class which inherit shape class

class Triangle(Shape):

def \_\_init\_\_(sf, r):

super().\_\_init\_\_()

sf.\_\_r = r

def get\_r(sf):

return sf.\_\_r

def set\_r(sf, r):

sf.\_\_r = r

def get\_A(sf):

return math.pi \* sf.\_\_r / 2

def get\_pm(sf):

return 2 \* math.pi \* sf.\_\_r

# # object / instance info

Object1 = Rectangle(9.6, 5.2)

print("The Area of rectangle is :", Object1.get\_A(),"\nThe Perimeter of rectangle is :", Object1.get\_pm(),"\nThe Color of rectangle is :", Object1.get\_color(),"\nThe rectangle is filled: ", Object1.get\_filled(),"\nThe rectangle is filled: ", Object1.get\_filled(),'\n',"\nThe Color of rectangle is :", Object1.get\_color())

print("\_"\*35)

# object / instance info

Object2 = Circle(22)

print("\nThe Area of circle is ", Object2.get\_A(),"\nThe Perimeter of circle is ",Object2.get\_pm(),"\nThe Color of circle is", Object2.get\_color(),"\nThe circle filled is ", Object2.get\_filled(),'\n',"\nIs the circle is filled ? ", Object2.get\_filled(),'\n',"\nThe Color of the circle is ", Object2.get\_color())

print("\_"\*35)

# object / instance info

Object3=Triangle(96)

print("\nThe A of Triangle is ", Object3.get\_A(),"\nThe Pm of Triangle is ",Object3.get\_pm(),"\nThe Color of Triangle ", Object3.get\_color(),"\nIs the Triangle is filled? ", Object3.get\_filled(),"\nTriangle is filled ? ", Object3.get\_filled(),"\nThe Color of Triangle is ", Object3.get\_color())

print("\_"\*35)

