USN | | | | | | | | | | |

Internal Assessment Test III – Jan 2023

Sub:	Application Development Using Python-Scheme and Solution				Sub Code:	18CS55	Branch:	ISE
Date:		Duration:	90 mins	Max Marks: 50	Sem/Sec:	V A,B & C		OBE

	Answer any FIVE FULL Questions	MARKS	CO	RBT

1a)	What is class? How do we define a class in python? How to instantiate the class and how class members are accessed?
	Class Definition [2 Mark]
	Defining Class[2 Marks]
	Instantiation of the class and class members accessing [2 Marks]
	Class: Class is a user-defined data type which binds data and functions together into single entity. Class is just a prototype (or a logical entity/blue print) which will not consume any memory.
	Defining Class:
	`class Point:`
	`  pass`
	
	Instantiation of the class and class members accessing
	`p=Point()`
	The process of creating a new object is called as ***instantiation*** and the object is ***instance*** of a class. When we print an object, Python tells which class it belongs to and where it is stored in the memory.
	`print(p)`
	`Class Point:`
	`  X=2`
	`  Y=3`
	
	`p1=Point()  #first object of the class`
	`print(p1.x, p1.y)  # prints 2 3`
	
	`p2=Point()  #second object of the class`
	`print(p2.x, p2.y)# prints 2 3`
		6	CO4	L2

1b)	Write a python program to add and multiply two complex number object using operator overloading concepts **Program with overloading method is must[4 marks]**	4	CO4	L3

```
import math
class Complex:
  """ This is a class adding compex numbers
  """

  def read_complex(self):
    self.rp=float(input("No's  real Part:"))
    self.ip=float(input("No's  Imaginary Part:"))

  def __str__(self):
    return "(%g,%g)"%(self.rp, self.ip)

  def __add__(self,n2):
    C=Complex()
    C.rp=self.rp+n2.rp
    C.ip=self.ip+n2.ip
    return C

C1=Complex()
C1.read_complex()

C2=Complex()
C2.read_complex()

C3=C1+C2
print(C1)
print(C2)
print(C3)
```

```
No's  real Part:10
No's  Imaginary Part:20
No's  real Part:-7
No's  Imaginary Part:13
(10,20)
(-7,13)
(3,33)
```

2a)	Write a Program that creates a class called Time with attributes hour, minute and second. Write the following functions. i)A function to read the attributes **[2.5 marks]** ii)A function to add two time objects and print the time in format(hours:minute:second) **[2.5 marks]**	5	CO4	L2

```
class Time:
  """Represents the time of a day Attributes: hour, minute, second
  """
  def readTime(self):
    self.hour=int(input("enter Hour"))
    self.minute=int(input("enter Minutes"))
    self.second=int(input("enter Seconds"))
  def printTime(self):
    print("%.2d:%.2d:%.2d" %(self.hour,self.minute,self.second))
  def add_time(self,t2):
      sum=Time()
      sum.hour = t1.hour + t2.hour
      sum.minute = t1.minute + t2.minute
      sum.second = t1.second + t2.second
      if sum.second >= 60:
        sum.second -= 60
        sum.minute += 1
      if sum.minute >= 60:
        sum.minute -= 60
        sum.hour += 1
      return sum
t1=Time()
t1.readTime()
print("Time1 is:")
t1.printTime()
t2=Time()
t2.readTime()
print("Time2 is:")
t2.printTime()
t3=t1.add_time(t2)
print("After adding two time objects:")
t3.printTime()
```

```
enter Hour2
enter Minutes34
enter Seconds40
Time1 is:
02:34:40
enter Hour5
enter Minutes45
enter Seconds34
Time2 is:
05:45:34
After adding two time objects:
08:20:14
```

2b)	Discuss type-based dispatch in python **Program[4 marks]** **Explanation [2 marks]**	6	CO4	L3

```
class Time:
 def __init__(self, h=0,m=0,s=0):
   self.hour=h
   self.min=m
   self.sec=s
 def time_to_int(self):
   minute=self.hour*60+self.min
   seconds=minute*60+self.sec
   return seconds
 def int_to_time(self, seconds):
   t=Time()
   minutes, t.sec=divmod(seconds,60)
   t.hour, t.min=divmod(minutes,60)
   return t
 def __str__(self):
   return "%.2d:%.2d:%.2d"%(self.hour,self.min,self.sec)
 def __eq__(self,t):
   return self.hour==t.hour  and self.min==t.min  and self.sec==t.sec
 def __add__(self,t):
   if isinstance(t, Time):
```

```
    return self.addTime(t)
  else:
    return self.increment(t)

 def addTime(self, t):
   seconds=self.time_to_int()+t.time_to_int()
   return self.int_to_time(seconds)

 def increment(self, seconds):
   seconds += self.time_to_int()
   return self.int_to_time(seconds)

 def __radd__(self,t):
   return self.__add__(t)

T1=Time(3,40)
T2=Time(5,45)
print("T1 is:",T1)
print("T2 is:",T2)
print("Whether T1 is same as T2?",T1==T2) #call for eq ()
T3=T1+T2
print("T1+T2 is:",T3)
T4=T1+75 #call for add ()
print("T1+75=",T4)
T5=130+T1          #call for radd ()
print("130+T1=",T5)
T6=sum([T1,T2,T3,T4])
print("Using sum([T1,T2,T3,T4]):",T6)
```

- When we try to perform addition, there are 3 cases –
  - Adding two time objects like T3=T1+T2.
  - Adding integer to Time object like T4=T1+75
  - Adding Time object to an integer like T5=130+T1

Each of these cases requires different logic. When first two cases are considered, the first argument will be T1 and hence self will be created and passed to add () method. Inside this method, we will check the type of second argument using isinstance() method. If the second argument is Time object, then we call addTime() method. In this method, we will first convert both Time objects to integer (seconds) and then the resulting sum into Time object again. So, we make use time_to_int() and int_to_time() here. When the 2nd argument is an integer, It is obvious that it is number of seconds. Hence, we need to call increment() method. Thus, based on the type of argument received in a method, we take appropriate action. **This is known as type-based dispatch.**

3a)	Explain operator overloading and polymorphism with example **Same answer as 2b) method with overloading add and sum method illustrate the concept of polymorphism. Explanation –[4 marks]** • we have implemented `__add__()` method (that is, overloading of + operator), the built- in `sum()` will is capable of adding multiple objects given in a sequence. This is due to ***Polymorphism*** in Python. • Consider a list containing Time objects, and then call `sum()` on that list as `T6=sum([T1,T2,T3,T4])` The `sum()` internally calls `__add__()` method multiple times and hence gives the appropriate result. Note down the square-brackets used to combine Time objects as a list and then passing it to `sum()`.	4	CO4	L3

3b)	Explain __init__ and __str__ method with an example Python Program.
	init method [2 marks]
	__str__ method [2 marks]
	Complete example [2 marks]
	`class Point:`
	`  def __init__(self,a,b):`
	`  self.x=a`
	`  self.y=b`
	`  def dist(self,p2):`
	`  d=math.sqrt((self.x-p2.x)**2 + (self.y-p2.y)**2)`
	`  return d`
	`  def __str__(self):`
	`  return "(%d,%d)"%(self.x, self.y)`
	`p1=Point(10,20)`
	`p2=Point(4,5)`
	`print("P1 is:",p1)`
	`print("P2 is:",p2)`
	`d=p1.dist(p2)`
	`print("Distance is:",d)`
	`P1 is: (10,20)`
	`P2 is: (4,5)`
	`Distance is: 16.15549442140351`
		6	CO4	L2

4a)	Explain parsing the element with beautiful soup module with code snippet for creating finding an element and getting data
	Explanation[2 marks]
	Program[3 marks]
	• Beautiful Soup is a module for extracting information from an HTML page (and is much better for this purpose than regular expressions).
	• The BeautifulSoup module's name is bs4 (for Beautiful Soup, version 4).
	• The bs4.BeautifulSoup() function needs to be called with a string containing the HTML it will parse.
	• The bs4.BeautifulSoup() function returns is a BeautifulSoup object.
	`import bs4`
	`exampleFile = open('example1.html')`
	`exampleSoup = bs4.BeautifulSoup(exampleFile.read())`
	`elems = exampleSoup.select('#author')`
	`print(elems)`
	`type(elems)`
	`print(len(elems))`
	`type(elems[1])`
	`elems[0].getText()`
	`str(elems[0])`
	`elems[0].attrs`
		5	CO5	L3

4b)	What methods the selenium's web element object have for simulating mouse clicks and keyboard keys explain with python code **Explanation [1.5 marks]** **Program[3.5 marks]** WebElement objects returned from the find_element_* and find_elements_* methods have a click() method that simulates a mouse click on that ele ment. This method can be used to follow a link, make a selection on a radio button, click a Submit button, or trigger whatever else might happen when the element is clicked by the mouse.	5	CO5	L2

Sending keystrokes to text fields on a web page is a matter of finding the <input> or <textarea> element for that text field and then calling the send_keys() method.

```
>>> from selenium import webdriver
>>> browser = webdriver.Firefox()
>>> browser.get('http://inventwithpython.com')
>>> linkElem = browser.find_element_by_link_text('Read It Online')
>>> type(linkElem)
<class 'selenium.webdriver.remote.webelement.WebElement'>
>>> linkElem.click()    # follows the "Read It Online" link
```

```
>>> from selenium import webdriver
>>> from selenium.webdriver.common.keys import Keys
>>> browser = webdriver.Firefox()
>>> browser.get('http://nostarch.com')
>>> htmlElem = browser.find_element_by_tag_name('html')
>>> htmlElem.send_keys(Keys.END)     # scrolls to bottom
>>> htmlElem.send_keys(Keys.HOME)    # scrolls to top
```

Clicking Browser Buttons

Selenium can simulate clicks on various browser buttons as well through the following methods:

`browser.back()` Clicks the Back button.

`browser.forward()` Clicks the Forward button.

`browser.refresh()` Clicks the Refresh/Reload button.

`browser.quit()` Clicks the Close Window button.

5a) How do we extract ,decrypt, copy and encrypt pdf files with example code — 4 — CO5 — L3

Full Program [4 marks]

```
import PyPDF2
pdfFileObj = open('pdf1.pdf', 'rb')
pdfReader = PyPDF2.PdfReader(pdfFileObj)
print(pdfReader.pages)
pageObj=pdfReader.pages[0]
print(pageObj.extract_text())
num = pdfReader.pages
for idx in range(num):
    # Get the page at index idx
    page = pdfReader.getPage(idx)
    # Add it to the output file
    out.addPage(page)
password = "pass"
out.encrypt(password)
# Open a new file "myfile_encrypted.pdf"
with open("myfile_encrypted.pdf", "wb") as f:
```

```
out.write(f)
file = PdfFileReader("myfile_encrypted.pdf")

# Check if the opened file is actually Encrypted
if file.isEncrypted:
    # If encrypted, decrypt it with the password
    file.decrypt(password)
    # Now, the file has been unlocked.
    # Iterate through every page of the file
    # and add it to our new file.
    for idx in range(file.numPages):
        # Get the page at index idx
        page = file.getPage(idx)

        # Add it to the output file
        out.addPage(page)
    # Open a new file "myfile_decrypted.pdf"
    with open("myfile_decrypted.pdf", "wb") as f:

        # Write our decrypted PDF to this file
        out.write(f)
    # Print success message when Done
    print("File decrypted Successfully.")
else:
    # If file is not encrypted, print the
    # message
    print("File already decrypted.")
pdfFileObj.close()
```

5b)	Create a program *multiplicationTable.py* that takes a number *N* from the com- mand line and creates an *N*×*N* multiplication table in an Excel spreadsheet.	6	CO5	L2

```python
import sys, openpyxl
from openpyxl.styles import Font
from openpyxl.utils import get_column_letter, column_index_from_string
n = 6
# Open a new workbook and swtich the active worksheet
wb = openpyxl.Workbook()
sheet = wb.active

# Print the headers
boldFont = Font(bold=True)
for i in range(2, n+2):
    multiplier = i - 1
    leftCell = 'A' + str(i)
    sheet[leftCell] = multiplier
    sheet[leftCell].font = boldFont
    rightCell = get_column_letter(i) + '1'
    sheet[rightCell] = multiplier
    sheet[rightCell].font = boldFont

# Print the multiplication table
for i in range(2, n+2):
    leftMultiplier = i - 1
    for j in range(2, n+2):
        rightMultiplier = j - 1
        cell = get_column_letter(j) + str(i)
        sheet[cell] = leftMultiplier*rightMultiplier

# Freeze the headers
sheet.freeze_panes = 'B2'
# Save to new Excel file in current working directory
wb.save('multiplicationTable.xlsx')
```

6a) Briefly describe the differences between the webbrowser, requests, BeautifulSoup, and selenium modules. — 4 — CO5 — L2

[4 points 4 marks]

- **webbrowser**. Comes with Python and opens a browser to a specific page.
- **Requests**. Downloads files and web pages from the Internet.
- **Beautiful Soup**. Parses HTML, the format that web pages are written in.
- **Selenium**. Launches and controls a web browser. Selenium is able to fill in forms and simulate mouse clicks in this browser.

6b) Consider below excel sheet. How do you use openpyxl module to programmatically add cells from B1 to B8 and put sum in B9 cell — 6 — CO5 — L3

[Full program 6 marks]

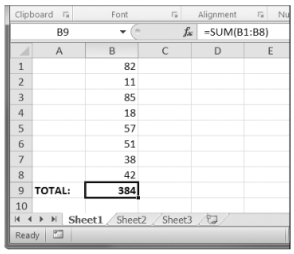

Figure 12-5: Cell B9 contains the formula =SUM(B1:B8), which adds the cells B1 to B8.

```python
import openpyxl

# Call a Workbook() function of openpyxl
# to create a new blank Workbook object
wb = openpyxl.Workbook()

# Get workbook active sheet
# from the active attribute.
sheet = wb.active

# writing to the cell of an excel sheet
sheet['A1'] = 200
sheet['A2'] = 300
sheet['A3'] = 400
sheet['A4'] = 500
sheet['A5'] = 600

# The value in cell A7 is set to a formula
# that sums the values in A1, A2, A3, A4, A5 .
sheet['A7'] = '= SUM(A1:A5)'

# save the file
wb.save("sum.xlsx")
```

HoD Signature CCI signature Course Instructor