Marking Scheme

Class XII

Computer Science (083)

Time Allowed: 3 hours **MM: 70**

Ques No	Question and Answers	Distribution of Marks	Total Marks
SECTION A			
1	False	1 mark for correct answer	1
2	Option b 6,20	1 mark for correct answer	1
3	Option c -244.0	1 mark for correct answer	1
4	PYTHON-is-Fun	1 mark for correct answer	1
5	Option b 8,15	1 mark for correct answer	1
6	Option a PAN	1 mark for correct answer	1
7	Option b `del D1["Red"]`	1 mark for correct answer	1
8	Option b	1 mark for correct answer	1

	ceieP0		
9	Option d Statement 4	1 mark for correct answer	1
10	Option b YELLOW* WHITE* BLACK* RED*	1 mark for correct answer	1
11	Option b Modulator	1 mark for correct answer	1
12	Option c global b	1 mark for correct answer	1
13	True	1 mark for correct answer	1
14	Option c A candidate key that is not a primary key is a foreign key.	1 mark for correct answer	1
15	circuit	1 mark for correct answer	1
16	Option c `seek()`	1 mark for correct answer	1

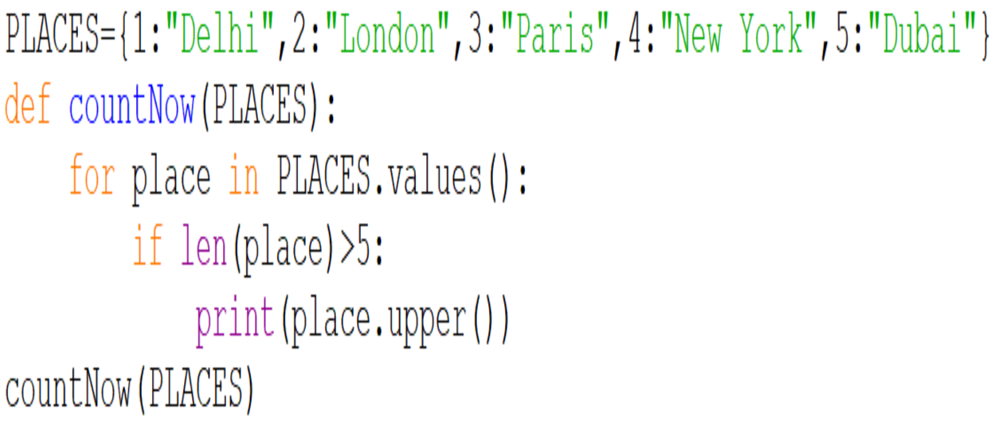
17	Option d A is false but R is True	1 mark for correct answer	1
18	Option b Both A and R are true but R is not the correct explanation for A	1 mark for correct answer	1
SECTION B			
19	(i) POP3 – Post Office Protocol 3 URL – Uniform Resource Locator (ii) HTML(Hyper text mark Up language) • We use pre-defined tags • Static web development language – only focuses on how data looks • It use for only displaying data, cannot transport data • Not case sensistive XML (Extensible Markup Language) • we can define our own tags and use them • Dynamic web development language – as it is used for transporting and storing data • Case sensitive	½ mark for each correct expansion 1 mark for any one correct difference No mark to be awarded if only full form is given	1+1=2

20	`def revNumber(num):`
	`    rev = 0`
	`    rem = 0`
	`    while num > 0:`
		½ mark for each	2

<table>
<tr>
<td></td>
<td>

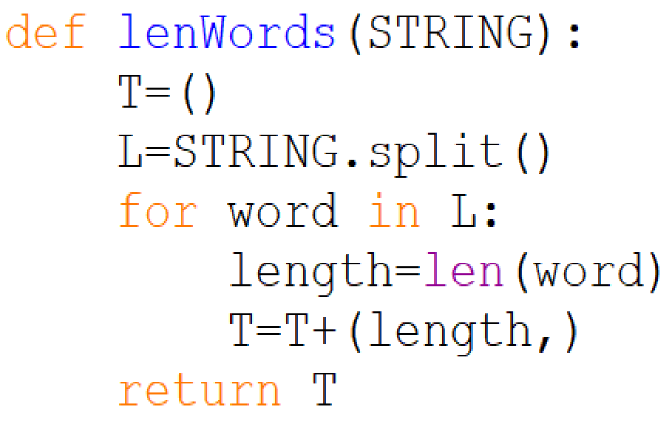
```
            rem =num %10
            rev = rev*10 + rem
            num = num//10
        return rev
print(revNumber(1234))
```

</td>
<td>correction made</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>



```
PLACES={1:"Delhi",2:"London",3:"Paris",4:"New York",5:"Dubai"}
def countNow(PLACES):
    for place in PLACES.values():
        if len(place)>5:
            print(place.upper())
countNow(PLACES)
```

OR



```
def lenWords(STRING):
    T=()
    L=STRING.split()
    for word in L:
        length=len(word)
        T=T+(length,)
    return T
```

Note: Any other correct logic may be marked

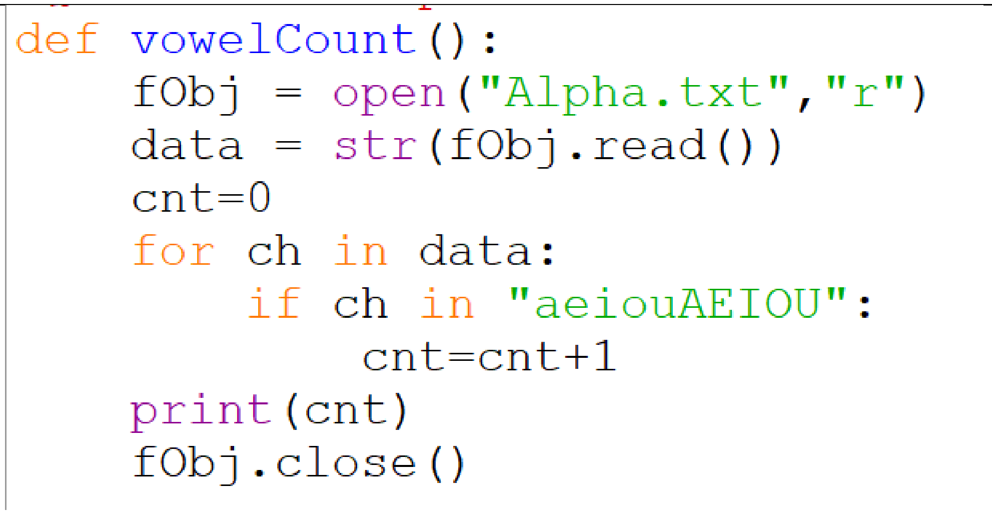
</td>
<td>½ mark for correct function header
½ mark for correct loop
½ mark for correct if statement
½ mark for displaying the output
½ mark for correct function header
½ mark for using split()
½ mark for adding to tuple
½ mark for return statement</td>
<td>2</td>
</tr>
</table>

22	4*L 33*4 21*S 10*6	½ mark for each correct line of output	2
23	(i) L1.insert(2,200) (ii) message.endswith('.')	1 mark for each correct statement	1+1=2

24	SQL Command to add primary key:
	ALTER TABLE Employee ADD EmpId INTEGER PRIMARY KEY;
	As the primary key is added as the last field, the command for inserting data will be:
	INSERT INTO Employee VALUES("Shweta","Production",26900,999);
	OR
	INSERT INTO Employee(EmpId,Ename,Department,Salary) VALUES(999,"Shweta","Production",26900);
		1 mark for correct ALTER TABLE command
		1 mark for correct INSERT command
			2

25	10.0$20 10.0$2.0###	1 mark for each correct line of output	2
SECTION C			
26	ND-*34	½ mark for each correct character	3
27			

<table>
<tr>
<td></td>
<td>
(i)
<table>
<tr><th>COUNT(DISTINCT SPORTS)</th></tr>
<tr><td>4</td></tr>
</table>
(ii)
<table>
<tr><th>CNAME</th><th>SPORTS</th></tr>
<tr><td>AMINA</td><td>CHESS</td></tr>
</table>
(iii)
<table>
<tr><th>CNAME</th><th>AGE</th><th>PAY</th></tr>
<tr><td>AMRIT</td><td>28</td><td>1000</td></tr>
<tr><td>VIRAT</td><td>35</td><td>1050</td></tr>
</table>
</td>
<td>1 mark for each correct output</td>
<td>1*3=3</td>
</tr>
<tr>
<td>28</td>
<td>
<pre>
def test():
    fObj1 = open("Alpha.txt","r")
    data = fObj1.readlines()
    for line in data:
        L=line.split()
        if L[0]=="You":
            print(line)
    fObj1.close()
</pre>
OR
</td>
<td>1 mark for correctly opening and closing files
½ mark for correctly reading data
1 mark for correct loop and if statement
½ mark for displaying data</td>
<td>3</td>
</tr>
</table>

	
	`def vowelCount():`
	`    fObj = open("Alpha.txt","r")`
	`    data = str(fObj.read())`
	`    cnt=0`
	`    for ch in data:`
	`        if ch in "aeiouAEIOU":`
	`            cnt=cnt+1`
	`    print(cnt)`
	`    fObj.close()`
	
	Note: Any other correct logic may be marked
		1 mark for correctly opening and closing the files
		
		½ mark for correctly reading data
		
		1 mark for correct loop and if statement
		
		½ mark for displaying the output.	

29	(i)
	`UPDATE Personal`
	`SET Salary=Salary*0.5`
	`WHERE Allowance IS NOT NULL;`
	
	(ii)
	`SELECT Name, Salary+Allowance AS`
	`"Total Salary" FROM Personal;`
	
	(iii)
	`DELETE FROM Personal`
	`WHERE Salary>25000`
		1 mark for each correct query	1*3=3

<table>
<tr>
<td>30</td>
<td>

```
travel = []
def Push_element(NList):
    for L in NList:
        if L[1] != "India" and L[2]<3500:
            travel.append([L[0],L[1]])


def Pop_element():
    while len(travel):
        print(travel.pop())
    else:
        print("Stack Empty")
```

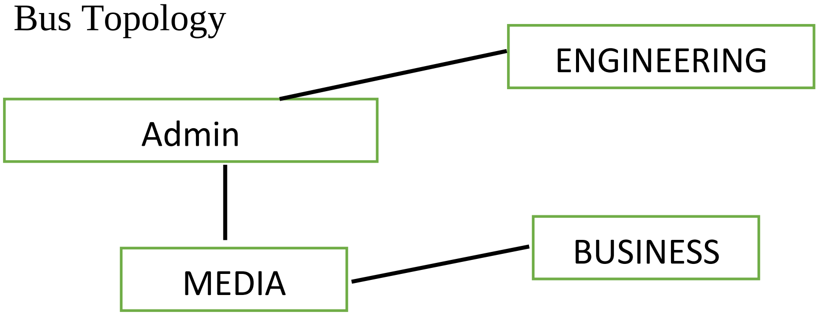
</td>
<td>1 ½ marks for each function</td>
<td>3</td>
</tr>
<tr>
<td colspan="4">

SECTION D

</td>
</tr>
<tr>
<td>31</td>
<td>

a)

Bus Topology



b) Switch

c) Admin block, as it has maximum number of computers.

d) Microwave

e) Firewall

</td>
<td>1 mark for each correct answer</td>
<td>1*5=5</td>
</tr>
<tr>
<td>32</td>
<td>

(i)

r+ mode:

- Primary function is reading
- File pointer is at beginning of file
- if the file does not exist, it results in an error

w+ mode:

</td>
<td>1 mark for each correct difference
(minimum two differences should be given)</td>
<td>2+3=5</td>
</tr>
</table>

- primary function is writing
- if the file does not exist, it creates a new file.
- If the file exists, previous data is overwritten
- File pointer is at the beginning of file

(ii)

```
def copyData():
    fObj = open("SPORT.DAT", "rb")
    fObj1 = open("BASKET.DAT","wb")
    cnt=0
    try:
        while True:
            data = pickle.load(fObj)
            print(data)
            if data[0] == "Basket Ball":
                pickle.dump(data,fObj1)
                cnt+=1
    except:
        fObj.close()
        fObj1.close()
    return cnt
```

½ mark for correctly opening and closing files

½ mark for correct try and except block

½ mark for correct loop

1 mark for correctly copying data

½ mark for correct return statement

OR

(Only for option ii)

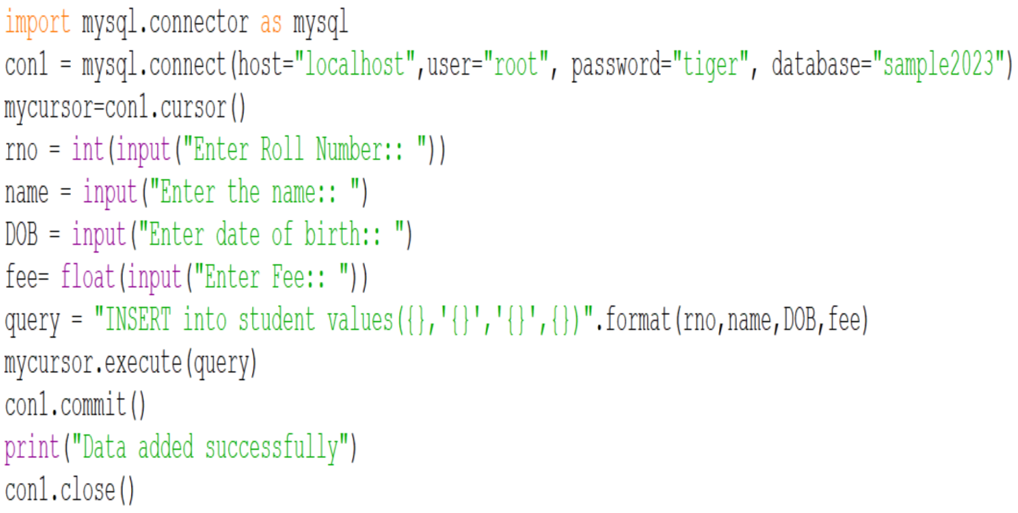
```
def Searchtype(mtype):
    fObj = open("CINEMA.DAT", "rb")
    try:
        while True:
            data = pickle.load(fObj)
            if data[2] == mtype:
                print("Movie number:",data[0])
                print("Movie Name:",data[1])
                print("Movie Type:",data[2])
    except EOFError:
        fObj.close()
```

½ mark for correctly opening and closing files

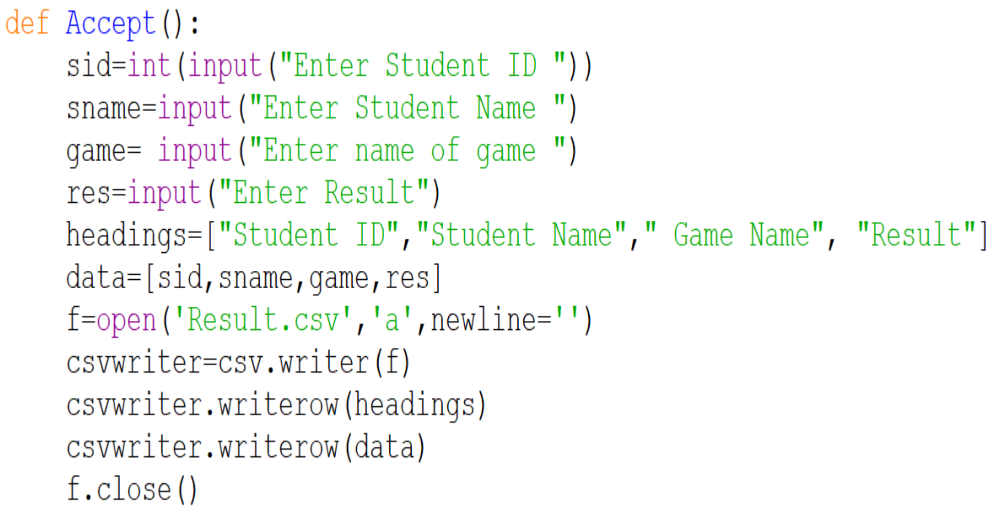
½ mark for correct try and except block

½ mark for correct loop

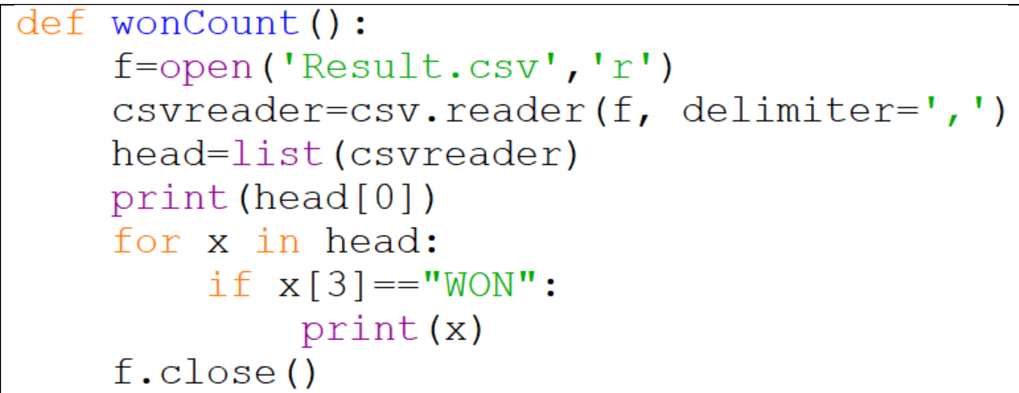
	Note: Any other correct logic may be marked	½ mark for correct if statement 1 mark for correctly displaying data	

33	(i) Domain is a set of values from which an attribute can take value in each row. For example, roll no field can have only integer values and so its domain is a set of integer values
	(ii)
	
	import mysql.connector as mysql
	con1 = mysql.connect(host="localhost",user="root", password="tiger", database="sample2023")
	mycursor=con1.cursor()
	rno = int(input("Enter Roll Number:: "))
	name = input("Enter the name:: ")
	DOB = input("Enter date of birth:: ")
	fee= float(input("Enter Fee:: "))
	query = "INSERT into student values({},'{}','{}',{})".format(rno,name,DOB,fee)
	mycursor.execute(query)
	con1.commit()
	print("Data added successfully")
	con1.close()
	Note: Any other correct logic may be marked
		½ mark for correct definition
		½ mark for correct example
		½ mark for importing correct module
		1 mark for correct connect()
		½ mark for correctly accepting the input
		1 ½ mark for correctly executing the query
		½ mark for correctly using commit()
			1+4=5

SECTION E

<table>
<tr><td>34</td><td>(i)
<pre>
SELECT PName, BName FROM PRODUCT P,
BRAND B WHERE P.BID=B.BID;
</pre>
(ii)
<pre>
DESC PRODUCT;
</pre>
(iii)
<pre>
SELECT BName, AVG(Rating) FROM PRODUCT
P, BRAND B
WHERE P.BID=B.BID
GROUP BY BName
HAVING BName='Medimix' OR
BName='Dove';
</pre>
(iv)
<pre>
SELECT PName, UPrice, Rating
FROM PRODUCT
ORDER BY Rating DESC;
</pre></td><td>1 mark for each correct query</td><td>1*4=4</td></tr>
<tr><td>35</td><td>
<pre>
def Accept():
    sid=int(input("Enter Student ID "))
    sname=input("Enter Student Name ")
    game= input("Enter name of game ")
    res=input("Enter Result")
    headings=["Student ID","Student Name"," Game Name", "Result"]
    data=[sid,sname,game,res]
    f=open('Result.csv','a',newline='')
    csvwriter=csv.writer(f)
    csvwriter.writerow(headings)
    csvwriter.writerow(data)
    f.close()
</pre></td><td>½ mark for accepting data correctly
½ mark for opening and closing file
½ mark for writing headings
½ mark for writing row</td><td>4</td></tr>
</table>

<table>
<tr>
<td></td>
<td>



```
def wonCount():
    f=open('Result.csv','r')
    csvreader=csv.reader(f, delimiter=',')
    head=list(csvreader)
    print(head[0])
    for x in head:
        if x[3]=="WON":
            print(x)
    f.close()
```

</td>
<td>½ mark for opening and closing file
½ mark for reader object
½ mark for print heading
½ mark for printing data</td>
<td></td>
</tr>
</table>