MARKING SCHEME

Class: XII Session: 2024-25

Computer Science (083)

Time allowed: 3 Hours **Maximum Marks: 70**

Q No.	SECTION A (21X1=21)	Marks
1.	False *(1 mark for correct answer)*	(1)
2.	(A) #THONPROGRAM *(1 mark for correct answer)*	(1)
3.	(A) not (True) and False *(1 mark for correct answer)*	(1)
4.	(B) ['I', 'ter', 'atio', 'al'] *(1 mark for correct answer)*	(1)
5.	ce lo *(1 mark for correct answer)*	(1)
6.	(B) False *(1 mark for correct answer)*	(1)
7.	(B) print(my_dict['apple', 'banana']) *(1 mark for correct answer)*	(1)
8.	(B) Removes the first occurrence of value x from the list *(1 mark for correct answer)*	(1)
9.	(D) t=tuple(1) *(1 mark for correct answer)*	(1)
10.	file.seek(0) (OR file.seek(0,0)) *(1 mark for correct answer)*	(1)
11.	False *(1 mark for correct answer)*	(1)
12.	(C) 12#15% *(1 mark for correct answer)*	(1)
13.	Alter (or Alter Table) *(1 mark for correct answer)*	(1)
14.	(A) Details of all products whose names start with 'App'	(1)

	(1 mark for correct answer)	
15.	(D) CHAR *(1 mark for correct answer)*	(1)
16.	(B) count() *(1 mark for correct answer)*	(1)
17.	(B) FTP *(1 mark for correct answer)*	(1)
18.	(B) Gateway *(1 mark for correct answer)*	(1)
19.	(B) Packet Switching *(1 mark for correct answer)*	(1)
20.	(B) Both A and R are true and R is not the correct explanation for A. *(1 mark for correct answer)*	(1)
21.	(C) A is True but R is False. *(1 mark for correct answer)*	(1)

Q No.	**SECTION B (7 X 2 =14)**	**Marks**
22.	A mutable object can be updated whereas an immutable object cannot be updated. Mutable object: [1,2] or {1:1,2:2} (Any one) Immutable object: (1,2) or ‘123’ (Any one) *(1 mark for correct difference)* *(½ x 2 = 1 Mark for selecting correct objects)*	(2)
23.	(I) Arithmetic operators: +,- (II) Relational operators: >, >= *(½ x 4 = 2 Marks for each correct operator)*	(2)
24.	(I) A) L1.count(4) OR B) L1.sort() *(1 mark for correct answer)* (II) A) L1.extend(L2)	(2)

<table>
<tr><td></td><td>OR
B) L2.reverse()
<i>(1 mark for correct answer)</i></td><td></td></tr>
<tr><td><b>25.</b></td><td>(A), (C)
<i>(½ x 2 = 1 Mark)</i>
Minimum and maximum possible values of the variable b: 1,6
<i>(½ x 2 = 1 Mark)</i></td><td>(2)</td></tr>
<tr><td><b>26.</b></td><td>Table: <b>Student</b>
<table>
<tr><th>ADMN</th><th>RollNo</th><th>Name</th><th>PhoneNo</th></tr>
<tr><td>124</td><td>1</td><td>Chavi</td><td>989899</td></tr>
<tr><td>235</td><td>2</td><td>Arpita</td><td>931124</td></tr>
<tr><td>276</td><td>3</td><td>Chavi</td><td>972457</td></tr>
</table>
Primary key: ADMN
Alternate keys: RollNo, PhoneNo
Total Candidate keys: 3
<i>(1 mark for the correct table)</i>
<i>(1 mark for number of candidate keys)</i></td><td>(2)</td></tr>
<tr><td><b>27.</b></td><td>(I)
A) UNIQUE
<b>OR</b>
B) NOT NULL
<i>(1 mark for correct answer)</i>
(II)
A) ALTER TABLE MOBILE DROP PRIMARY KEY;
OR
B) ALTER TABLE MOBILE ADD PRIMARY KEY (M_ID);
<i>(1 mark for correct answer)</i></td><td>(2)</td></tr>
<tr><td><b>28.</b></td><td>A) Advantage: Network extension is easy.
Disadvantage: Failure of switch/hub results in failure of the network.
<i>(1 mark for correct Advantage)</i>
<i>(1 mark for correct Disadvantage)</i>
<b>OR</b>
B) SMTP: Simple Mail Transfer Protocol.</td><td>(2)</td></tr>
</table>

	SMTP is used for sending e-mails from client to server. *(1 mark for correct expansion)* *(1 mark for correct usage)*	

Q No.	SECTION C (3 X 3 = 9)	Marks
29.	(A)	(3)

```
def show():
    f=open("Email.txt",'r')
    data=f.read()
    words=data.split()
    for word in words:
        if '@cmail' in word:
            print(word,end=' ')
    f.close()
```

(½ mark for correct function header)

(½ mark for correctly opening the file)

(½ mark for correctly reading from the file)

(½ mark for splitting the text into words)

(1 mark for correctly displaying the desired words)

OR

(B)

```
def display_long_words():
    with open("Words.txt", 'r') as file:
        data=file.read()
        words=data.split()
        for word in words:
           if len(word)>5:
               print(word,end=' ')
```

(½ mark for correct function header)

(½ mark for correctly opening the file)

(½ mark for correctly reading from the file)

(½ mark for splitting the text into words)

(1 mark for correctly displaying the desired words)

30. (A)

(I)

```
def push_book(BooksStack, new_book):
        BooksStack.append(new_book)
```

(II)

```
def pop_book(BooksStack):
    if not BooksStack:
        print("Underflow")
    else:
        return(BookStack.pop())
```

(III)

```
def peep(BooksStack):
    if not BooksStack:
        print("None")
    else:
        print(BookStack[-1])
```

(3x1 mark for correct function body; No marks for any function header as it was a part of the question)

(3)

OR

(B)

```
n=int(input("Enter an integer: "))
s=[] #stack
f=2
while n>1:
    if n%f==0:
    s.append(f)
    n//=f
    else: f+=1
while s:
    print(s.pop(),end=' ')
```

(½ mark for correct input)

(½ mark for correctly declaring an empty stack)

(1 mark for correctly pushing the factors on the stack)

(1 mark for correctly popping and displaying the factors)

31. (A)

(I) select Product, sum(Quantity) from orders
group by product having sum(Quantity)>=5;

(II) select * from orders order by Price desc;

(III) select distinct C_Name from orders;

(3x 1 mark for each correct query)

(3)

OR

(B)

(I) select quantity, count(*) from orders group by quantity;

(II) delete from orders where product = "Laptop";

(III) select sum(price) from orders where quantity is null;

(3x 1 mark for each correct query)

Q No. | **SECTION D (4 X 4 = 16)** | **Marks**

32. (A)

(I) ZeroDivisionError is raised when a statement tries to divide a number by zero. *(1 Mark for correct answer)*

(II)

```
try:
   a=int(input("Enter an integer: "))
   print("Reciprocal of the number =",1/a)
except ZeroDivisionError:
   print("Division by Zero is not allowed")
except:
   print("Some Error Ocurred")
```

(3x 1 mark for each correct part – try, except, except)

OR

(B)

(I) NameError is raised when an undefined identifier is used in the program. *(1 Mark for correct answer)*

(II)

```
try:
   a=eval(input("Enter an integer: "))
   print("Reciprocal of the number =",1/a)
except NameError:
   print("Some name is not defined")
except:
   print("Some Error Ocurred")
```

(3x1 Mark for each correct part – try, except, except)

(4)

33. (I)

```
def show():
   import csv
   f=open("happiness.csv",'r')
   records=csv.reader(f)
   next(records, None) #To skip the Header row
   for i in records:
      if int(i[1])>5000000:
         print(i)
```

(4)

```
        f.close()
```

(½ mark for opening in the file in right mode)
(½ mark for correctly creating the reader object)
(½ mark for correctly checking the condition)
(½ mark for correctly displaying the records)

(II)

```
    def Count_records():
        import csv
        f=open("happiness.csv",'r')
        records=csv.reader(f)
        next(records, None) #To skip the Header row
        count=0
        for i in records:
              count+=1
        print(count)
        f.close()
```

(½ mark for opening in the file in right mode)
(½ mark for correctly creating the reader object)
(½ mark for correct use of counter)
(½ mark for correctly displaying the counter)

Note (for both parts (I) and (II)**):**

(i) *Ignore* **import csv** *as it may be considered the part of the complete program, and there is no need to import it in individual functions.*

(ii) *Ignore next(records, None) as the file may or may not have the Header Row.*

34.

(I) Select * from FACULTY natural join COURSES where Salary<12000;

(II) Select * from courses where fees between 20000 and 50000;

(III) Update courses set fees=fees+500 where CName like '%Computer%';

(IV) (A) Select FName, LName from faculty natural join courses where Came="System Design";

OR

(B) Select * from FACULTY, COURSES;

(4x1 mark for each correct query)

(4)

35.

```
def Add_Item():
   import mysql.connector as mycon
   mydb=mycon.connect(host="localhost",user="root",
                 passwd="Pencil",database="ITEMDB")
   mycur=mydb.cursor()
   no=input("Enter Item Number: ")
   nm=input("Enter Item Name: ")
   pr=input("Enter price: ")
   qty=input("Enter qty: ")
   query="INSERT INTO stationery VALUES ({},'{}',{},{})"
```

(4)

```
    query=query.format(no,nm,pr,qty)
    mycur.execute(query)
    mydb.commit()
    mycur.execute("select * from stationery where price>120")
    for rec in mycur:
        print(rec)
```

(½ mark for correctly importing the connector object)
(½ mark for correctly creating the connection object)
(½ mark for correctly creating the cursor object)
(½ mark for correctly inputting the data)
(½ mark for correct creation of first query)
(½ mark for correctly executing the first query with commit)
(½ mark for correctly executing the second query)
(½ mark for correctly displaying the data)

Q No.	SECTION E (2 X 5 = 10)	Marks

36. **Note**: *For part (I), the student can mention any type of file with valid reason to support the choice. Answer with valid supporting reason should be considered Correct, and without a valid reason should be considered incorrect.* (5)

(I) Text file: A text file allows for easy maintenance of data, as it can be opened and manipulated with any text editor also.

(1 mark for correct answer)

(II)

```
def append():
    with open("Candidates.txt",'a') as f:
        C_id=input("Enter Candidate ID: ")
        C_nm=input("Enter Candidate name: ")
        C_dg=input("Enter Designation: ")
        C_ex=input("Enter Experience: ")
        rec=C_id+','+C_nm+','+C_dg+','+C_ex+'\n'
        f.write(rec)
```

(½ mark for opening in the file in right mode)
(½ mark for correctly inputting the data)
(½ mark for correctly writing the record in the file)
(½ mark for correctly closing the file, or ½ mark if the file was opened using with)

(II)

```
def display():
    with open("Candidates.txt") as f:
        for rec in f:
            data=rec.split(',')
            if float(data[-1])>10:
                print(rec.strip()) #OR print(rec)
```

(½ mark for opening the file in right mode)
(½ mark for correctly reading the data)
(½ mark for correctly checking the condition)
(½ mark for correctly displaying the records)

OR

(I) CSV File: A CSV file allows for easy maintenance of data, as it can be opened and manipulated with any spreadsheet application also.

(1 mark for correct answer)

(II)

```
def append():
    with open("Candidates.csv",'a',newline='') as f:
        C_id=input("Enter Candidate ID: ")
        C_nm=input("Enter Candidate name: ")
        C_dg=input("Enter Designation: ")
        C_ex=input("Enter Experience: ")
        rec=[C_id,C_nm,C_dg,C_ex]
        w=csv.writer(f)
        w.writerow(rec)
```

(½ mark for opening in the file in right mode)
(½ mark for correctly inputting the data)
(½ mark for correctly writing the record in the file)
(½ mark for correctly closing the file, or ½ mark if the file was opened using with)

(III)

```
def display():
    with open("Candidates.csv") as f:
        r=csv.reader(f)
        for rec in r:
            if float(rec[-1])>10:
                print(rec)
```

(½ mark for opening the file in right mode)
(½ mark for correctly reading the data)
(½ mark for correctly checking the condition)
(½ mark for correctly displaying the records)

OR

(I) Binary File: A binary file cannot be opened and manipulated with any general purpose application, and hence, it prevents any unintentional change in the data.

(1 mark for correct answer)

(II)

```
def append():
    with open("Candidates.dat",'ab') as f:
        C_id=int(input("Enter Candidate ID: "))
        C_nm=input("Enter Candidate name: ")
        C_dg=input("Enter Designation: ")
        C_ex=float(input("Enter Experience: "))
        rec=[C_id,C_nm,C_dg,C_ex]
        pickle.dump(rec,f)
```

(½ mark for opening in the file in right mode)
(½ mark for correctly inputting the data)
(½ mark for correctly writing the record in the file)
(½ mark for correctly closing the file, or ½ mark if the file was opened using with)

(III)

```
def display():
    with open("Candidates.dat",'rb') as f:
        while True:
            try:
                rec=pickle.load(f)
                if rec[-1]>10:
                    print(rec)
            except EOFError:
                break
```

(½ mark for opening the file in right mode)
(½ mark for correctly reading the data)
(½ mark for correctly checking the condition)
(½ mark for correctly displaying the records)

37. (I) MEDIA Block as it has the maximum number of Computers. (5)

OR

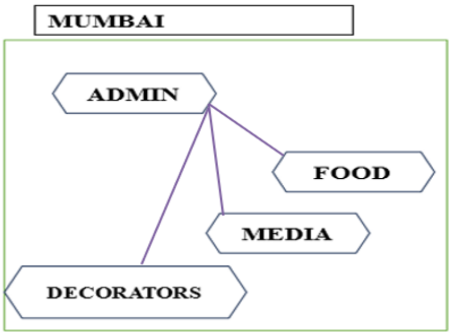
ADMIN Block as ADMIN block is generally the most secure.

(1 mark for correct answer)

(II) Switch

(1 mark for correct answer)

(III)



(or Any other correct layout)

Cable: Optical Fibre

(½ mark for correct layout + ½ mark for correct table type)

(IV) There is no requirement of the Repeat as the optical fibre cable used for the network can carry the data to much longer distances than within the campus.

(1 mark for correct answer)

(V) (A) a) Video Conferencing

OR

(B) LAN

(1 mark for correct answer)