COMMON PRE-BOARD EXAMINATION

COMPUTER SCIENCE-Code No. 083 MARKING SCHEME

Class-XII-(2025-26)

SET: 1

Time allowed: 3 Hrs. **Maximum Marks: 70**

General Instructions:

- This question paper contains 37 questions.
- All questions are compulsory. However, internal choices have been provided in some questions. Attempt only one of the choices in such questions.
- The paper is divided into 5 Sections- A, B, C, D and E.
- Section A consists of 21 questions (1 to 21). Each question carries 1 Mark.
- Section B consists of 7 questions (22 to 28). Each question carries 2 Marks.
- Section C consists of 3 questions (29 to 31). Each question carries 3 Marks.
- Section D consists of 4 questions (32 to 35). Each question carries 4 Marks.
- Section E consists of 2 questions (36 to 37). Each question carries 5 Marks.
- All programming questions are to be answered using Python Language only.
- In-case of MCQ, text of the correct answer should also be written.

Q.No.	Section-A (21 x 1 = 21 Marks) (1 mark for correct answer)	Marks
1	True	1
2	world-of-Python the world	1
3	a) True	1
4	b) Equi Join	1
5	c)z	1
6	31*84*136*	1
7	37.0	1
8	SELECT * FROM employees WHERE manager_id is NULL;	1
9	b) Error occurred Finally block	1
10	b) D@T@#ROCESSING	1
11	d) All the given	1
12	c) 5@@12##12	1
13	a) Same value stored repeatedly in different rows due to denormalized design	1
14	b) ['M', '', 'ss', 'ss', 'pp', '']	1
15	b) Cardinality = 12, Degree = 7	1

16	b) DROP	1
17	b) TELNET	1
18	c) IP address identifies the physical hardware of a device.	1
19	c) https	1
20	d) A is False but R is True.	1
21	a) Both A and R are True and R is the correct explanation for A.	1
Q No.	**Section-B (7 x 2=14 Marks)**	**Marks**
22	A. (see below)	2

22 A.

	List	Tuple
Mutability	Mutable — elements can be changed	Immutable — elements cannot be changed
Brackets Used	Written using [ ] with any correct example	Written using ( ) with any correct example

B. **Yes.** A tuple itself is immutable, but it can store a mutable object like a list, and the contents of that list can be changed.(with any correct example)
(1 mark for correct difference)
(1/2 mark for each correct example)

Marks: 2

23

```
def even_sum(num_list):                                    #1  Added colon (:)
    total = 0
    for n in num_list:
        if n % 2 == 0:                  #2 &3  Changed '=' to '==' & indentation
            total = total + n
    return total

numbers = [3, 8, 5, 12, 7, 10]
print("Sum of even numbers:" even_sum(numbers))            #4  Added comma (,)
```

(1⁄2 mark each for correcting 4 mistakes)

Marks: 2

24 A. {0: 1, 1: 1, 2: 1}
B.True
(1 mark for correct answer)

Marks: 2

25 A.

```
def removeduplicate(L):
    newlist = []
    for item in L:
        if item not in newlist:   # keep only first occurrence
            newlist.append(item)
    print("List after removing duplicates:", newlist)



# Prgm for Q25
# sample list L
L = [2, 5, 2, 9, 5, 1, 9]
removeduplicate(L)
```

Marks: 2

OR

B.

```
def updatemarks(marks, name, newmarks):
    if name in marks:            # check if student exists
        marks[name] = newmarks   # update marks
        print("Marks updated.")
    else:
        print(name," : Student not found")
 # Example use:
marks = {"Saju":85, "Neha":92, "Amit":76}
updatemarks(marks, "Neha", 95)     # existing
updatemarks(marks, "Sana", 88)     # not existing
print(marks)
```

(1/2 mark for function definition)
(1½ marks for the correct/similar logic)

26

1. 6
2. 1

(1 mark for the correct output)

2

27

```
A.I. SELECT * FROM EMPLOYEE
     ORDER BY SALARY DESC;
II. ALTER TABLE EMPLOYEE
    ADD EMAIL VARCHAR(50);
```

(1 mark for each correct answer.)

2

B.

Command	Purpose	Works on	Example
ALTER TABLE	Changes structure of a table	Columns / Constraints	ALTER TABLE EMP ADD AGE INT;
UPDATE	Changes data inside the table	Rows / Values	UPDATE EMP SET AGE = 25 WHERE ENO = 101;

(1 mark for correct difference)
(1/2 mark for each correct example)

28

A.I

- A switch is a network device that connects multiple computers in a LAN and forwards data only to the specific device (port) for which it is intended.
- It is intelligent and reduces network traffic.
- Forwards data only to the intended device, reducing traffic.

II.

- A hub is a basic network device that connects multiple computers in a LAN but sends incoming data to all connected devices, not just the intended one.
- It is not intelligent and causes more traffic.
- Broadcasts data to all connected devices, causing extra traffic.

(1 mark for each correct answer)

2

B.
1. WIDE AREA NETWORK AND PERSONAL AREA NETWORK
II.

Star Topology	Bus Topology
All devices connect to a central device (hub/switch).	All devices share a single main cable (backbone).
Failure of one device does not affect others.	Failure of the main cable stops whole network.
Easy to add/remove new devices.	Difficult to add new devices without disturbing the cable.

1 mark for correct expansion
1 mark for correct difference

SECTION C (3X3= 9 Marks)

29 A. — 3

```
def count_vowels():
    vowels = "aeiouAEIOU"
    count = 0
    with open("Data.txt", "r") as f:
        for line in f:
            for ch in line:
                if ch in vowels:
                    count += 1
    print("Total vowels:", count)

# call the function
count_vowels()
```

B.

```
def show_no_digit_lines():
    with open("Data.txt", "r") as f:
        for line in f:
            has_digit = False
            for ch in line:
                if ch.isdigit():
                    has_digit = True
                    break
            if not has_digit:
                print(line, end="")

# call the function
show_no_digit_lines()
```

(1/2 mark for correct function header)
(1/2 mark for correctly opening the file)
(1/2 mark for correctly reading from the file)

	(1/2 mark for nested loop) **(1/2 mark for correct use of counter variable)** **(1/2 mark for correct output)**	

30	
	```
	# Given list
	L = [("Ravi", 85), ("Neha", 92), ("Amit", 45), ("Sana", 78)]

	Student = []   # stack

	def Push_element():
	    for name, marks in L:
	        if marks > 90:          # condition as per reframed question
	            Student.append((name, marks))

	def Pop_element():
	    if not Student:
	        print("Stack Empty")
	    else:
	        while Student:
	            print(Student.pop())
	        print("Stack Empty")
	# --- Calling functions ---
	Push_element()
	Pop_element()
	```
	(1½ marks for each correct part)
		3

31	A. CS194 B. ['PY', 'C', 'PE'] **(1½ marks for each correct output)**	3
Q No.	**Section-D (4 x 4 = 16 Marks)**	**Marks**

32	A.
	```
	    I. SELECT Category, SUM(Quantity) AS TotalQty
	    FROM PRODUCTS
	    GROUP BY Category
	    HAVING SUM(Quantity) > 15;

	    II. SELECT ProductName
	    FROM PRODUCTS
	    WHERE ProductName LIKE '_a%';

	    III. SELECT *
	    FROM PRODUCTS
	    ORDER BY Price ASC;

	    IV. SELECT DISTINCT Category
	    FROM PRODUCTS;
	```
		4

	(4 x 1 mark for each correct query) B.Predict the Output I. Laptop II. 1 III. 900 IV.110 **4 x 1 mark for each correct output with table and header)**	

33	
	```
	import csv

	# I) Insert a new order into Orders.csv
	def InsertOrder():
	    with open("Orders.csv", "a", newline="") as f:
	        w = csv.writer(f)
	        oid  = input("Order ID : ")
	        item = input("Item     : ")
	        qty  = int(input("Quantity : "))
	        rate = float(input("Rate     : "))
	        w.writerow([oid, item, qty, rate])
	    print("Order Added")

	# II) Calculate total bill = sum of (Quantity × Rate) for all records
	def GenerateBill():
	    total = 0
	    try:
	        with open("Orders.csv","r") as f:
	            r = csv.reader(f)
	            for row in r:
	                qty  = int(row[2])
	                rate = float(row[3])
	                total += qty * rate
	    except FileNotFoundError:
	        print("File not found.")
	    return total
	```
	(1⁄2 mark for opening in the file in right mode)
	(1 mark for correctly creating the reader object)
	(1 mark for correctly using for loop)
	(1½ mark for correct logic and displaying the output)
	Note (for both parts (I) and (II)):
	Ignore import csv as it may be considered the part of the complete program
		4

34	1.SELECT * FROM DOCTOR D INNER JOIN PROJECTS P ON D.D_ID = P.D_ID WHERE D.Honorarium BETWEEN 40000 AND 55000;	4

	II.SELECT *
	FROM PROJECTS
	WHERE Budget NOT BETWEEN 60000 AND 100000;
	
	III.UPDATE PROJECTS
	SET Budget = Budget + (0.10 * Budget)
	WHERE PName LIKE '%AI%';
	
	IV.SELECT D.FName, D.LName
	FROM DOCTOR D
	INNER JOIN PROJECTS P
	ON D.D_ID = P.D_ID
	WHERE P.PName = 'Neuro Imaging';
	OR
	SELECT *
	FROM DOCTOR, PROJECTS;
	(4 x 1 mark for each correct query)	

35	import mysql.connector as mc
	
	con = mc.connect(
	    host="localhost",
	    user="librarian",
	    password="lib#2025",
	    database="LibraryDB")
	cur = con.cursor()
	cur.execute("DELETE FROM issued_books WHERE IssueID = 503")
	con.commit()
	
	if cur.rowcount == 1:
	    print("Record Deleted")
	else:
	    print("No single matching record found")
	
	cur.close()
	con.close()
	(1⁄2 mark for correctly importing the connector object)
	(1⁄2 mark for correctly creating the connection object)
	(1⁄2 mark for correctly creating the cursor object)
	(1 mark for correct creation of update query)
	(1 mark for correctly executing the query with commit)
	(1⁄2 mark for correctly closing the connection)
		4

Q No.	**Section-D (2 x 5 = 10 Marks)**	**Marks**
36	import pickle	2+3

```
# I) Add new fee records
def add_record():
    f = open("fees.dat","ab")
    rec = []
    Roll = int(input("Roll No: "))
    Name = input("Name: ")
    Course = input("Course: ")
    Fee = float(input("Fee Paid: "))
    rec = [Roll, Name, Course, Fee]
    pickle.dump(rec,f)
    f.close()
```

1/2 mark for correctly defining the function header)
(1/2 mark for correctly opening the file in append mode)
(1/2 mark for correctly taking user input)
(1/2 mark for using dump() method of the pickle module)

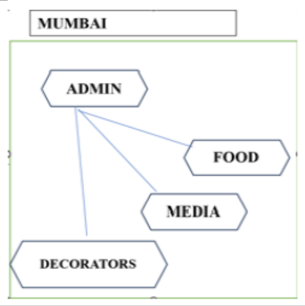
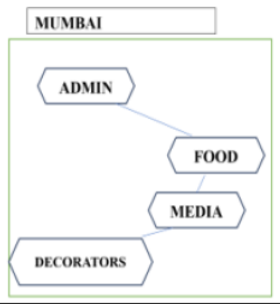
```
# II) Increase Fee_Paid by 10% for BTech
def update_btech_fee():
    f = open("fees.dat","rb")
    data = []
    try:
        while True:
            r = pickle.load(f)
            if r[2] == "BTech":
                r[3] = r[3] * 1.10   # +10%
            data.append(r)
    except EOFError:
        f.close()

    f = open("fees.dat","wb")
    for r in data:
        pickle.dump(r,f)
    f.close()
```

(1/2 mark for correctly defining the function header)
(1/2 mark for correctly opening the file)
(1 mark for using load() with while loop and try-except block)
(1 mark for checking the condition and updating the value)
Note: Note (for both parts (I) and (II)): (i) Ignore import pickle as it may be considered the part of the complete program.

37	I. The most appropriate location of the server inside the MUMBAI campus is ADMIN building due to the maximum number of computers in it. **½ mark for mentioning the branch and ½ mark for proper justification**	5

ii. Cable Layout

Star Topology (Based on server location)	Bus Topology (Based on minimum distance between branches)
	

1 mark for drawing any valid cable layout

III.. Switch or Hub

1 mark for suggesting the correct device

IV. c. Video Conferencing

1 mark for correct answer

V.

(a) WAN

(b) LAN

½ mark for mentioning WAN and ½ mark for mentioning LAN

***