END SEMESTER EXAMINATION (LAB)

CS23304- JAVA PROGRAMMING

SEMESTER – III **BATCH: Q**

Date: 5-11-2025 **Marks: 100**

Draw the following table in your first page

MARK SPLIT-UP

Q.NO.	COMMENTS	MARK
	Total	

1. You are developing a **multithreaded ticket booking system** for a concert. There are **100 tickets** available initially, and multiple users can book tickets at the same time through different devices (threads).

However:

- Each user can book **1–5 tickets** at a time.
- If not enough tickets remain, the system should show **"Tickets Sold Out"** or **"Not enough tickets available."**
- The system must ensure **thread safety**, i.e., the total tickets sold never exceeds 100.
- The system should display the remaining ticket count after each booking attempt.

Requirements:

- Create a class TicketCounter with: (15)
  - A variable availableTickets initialized to 100.
  - A method bookTickets(String userName, int numTickets) that:
    - Checks availability.
    - Deducts tickets if available.
    - Displays messages accordingly.
- Create a class UserBookingThread that extends Thread and: (15)
  - Takes the TicketCounter object and user details.
  - Calls bookTickets() in its run() method.
- Use the **synchronized** keyword to ensure multiple users cannot book tickets at the same time incorrectly. (10)
- Simulate **5–10 users** (threads) booking tickets concurrently. (10)

Sample Output

User A booked 5 tickets. Tickets remaining: 95
User B booked 3 tickets. Tickets remaining: 92
User C booked 10 tickets. Tickets remaining: 82
User D tried to book 30 tickets. Not enough tickets available!
...
User J tried to book 5 tickets. Tickets Sold Out!

2. You are hired to develop a simple **Student Records Management System** for a college. The college wants to **store, retrieve, and update student data** in a text file instead of a database.

Each student record contains:

StudentID, Name, Department, Marks

Example:

101, Alice Johnson, Computer Science, 89
102, Bob Smith, Electrical, 76
103, Carol White, Mechanical, 92
The data is stored in a file named "students.txt".

Requirements:

Create a Class for Student

Create a Student class with: (10)

- Attributes: studentId, name, department, marks
- Methods: constructor, getters/setters, and toString().

Implement File Operations

Create a class StudentFileManager with the following methods:

addStudent(Student s) **(5)**
Appends a new student record to students.txt.

displayAllStudents() **(5)**
Reads all records from students.txt and displays them neatly.

searchStudent(int studentId) **(10)**
Searches for a student by ID and displays the details.

updateStudentMarks(int studentId, int newMarks) **(10)**
Reads all data, updates the specific student's marks, and rewrites the file.

calculateAverageMarks() **(10)**
Calculates and displays the average marks of all students.

Sample Output:

All Students:
101 - Alice Johnson - Computer Science - 89
102 - Bob Smith - Electrical - 76
103 - Carol White - Mechanical - 92

Search by ID 102:
Student Found: Bob Smith, Department: Electrical, Marks: 76

Updating Marks...
Bob Smith's marks updated to 82 successfully.

Average Marks of all Students: 87.66