Final Assignment

CCP6224 – Object-Oriented Analysis and Design
Total Marks: 40%
Due Date: 16 Feb 2026, 11.59pm (firm deadline; no extensions)

Instructions:

1. **Group Work & Individual Evaluation:**
   - This is a group assignment (groups of 3 -5 students from the same tutorial section).
   - There must be work division among team members. You are advised to divide the project by modules first, then each team member responsible for a particular module design. Have a team discussion on how to integrate all the modules and create a grand design that can work seamlessly. Once this is finalized, then each team member can focus on the module and provide the implementation and integrate them.
   - During evaluation, you must be able to defend questions asked with regards to your module as well as overall questions. Do not give answers like we do this "together". It is not acceptable at this level of your degree path. This project is an Object-Oriented Design project where you must be able to show your skill in task division/modularization and then integration of the modules while still making the design intuitive and implementation 100% represent the design done.
   - **Important:** Evaluation will be done individually. Every group member must be able to explain and defend the design and implementation in a follow-up Q&A session.
   - You must form you group by 19 Dec 2025, else ask your tutor for assistance to find a team mate. Your tutor can match-make you to any available team.

2. **Plagiarism:**
   - Any instances of plagiarism will result in a zero mark without prior notice. Ensure all work is your own and any external help is appropriately acknowledged.
3. **Submission Format:**
   - Submit one zip folder with the naming convention: **GroupName.zip (Assignment GroupName as per the eBwise or Spreadsheet, i.e., Group-A.zip or Group-10.zip, etc.)**.
   - The folder must include all source code files, UML diagrams (use case, class, and sequence diagrams), and any supporting documentation.
   - Submission instruction will be given by your lecturer/tutor. If you dont follow the submission instructions, you will get zero marks.
   - The project is to be done using only Java Swing.
4. **Mandatory Interview Q&A Session:**
   - Each student must attend a physical interview session between **23 Feb – 6 Mar 2026** (details to be announced by your lecturer).
   - In this session, you will be asked questions about your design decisions, code structure, and how your solution supports future enhancements. Other related questions also can be asked.
   - For the interview, prepare yourself in line with the rubric below (Read the marking rubrics so you know how to score!).
   - Failure to attend the Q&A session will result in a zero mark for the assignment even if you submitted the assignment.

Case Study: Parking Lot Management System

Client: University Parking Lot Management Office

Project Overview:
The university's Parking Lot Management Office requires a standalone GUI application to design and implement a parking lot management system for a multi-level building. The system must handle vehicle parking, spot allocation, payment processing, and fine management.

Client Requirements:

1. **Parking Lot Structure**
   The parking lot has:
   - Multiple **floors** (e.g., 5 floors)
   - Each floor has multiple **parking spots** arranged in rows
   - **Parking spot types:**
     - **Compact:** For small vehicles (motorcycles, bicycles) - RM 2/hour
     - **Regular:** For regular cars - RM 5/hour
     - **Handicapped:** Reserved for handicapped vehicles - RM 2/hour (FREE only if handicapped card holder vehicle parks in handicapped spot)
     - **Reserved:** For VIP customers - RM 10/hour
   - Each spot has:
     - Spot ID (e.g., "F1-R1-S1" = Floor 1, Row 1, Spot 1)
     - Type (compact, regular, handicapped, reserved)
     - Status (available or occupied)
     - Current vehicle (if occupied)
     - Hourly rate (varies by type)

2. **Vehicle Management**

   The system must support:
   - **Vehicle types:**
     - **Motorcycle** - Can park in Compact spots only
     - **Car** - Can park in Compact or Regular spots
     - **SUV/Truck** - Can park in Regular spots only
     - **Handicapped Vehicle** - Can park in any spot, gets a discounted price of RM 2/hour only for a handicapped card holder
   - Each vehicle has:
     - License plate number
     - Vehicle type
     - Entry time
     - Exit time (null if still parked)
   - **Duration:**
     - Rounded up to the nearest hour (Ceiling rounding)

3. **Entry/Exit System**

   When a **vehicle enters:**
   1. The system shows available spots of suitable types
   2. User selects a spot
   3. System marks the spot as occupied
   4. System records entry time & assigns spot
   5. The system generates and displays a ticket with the spot location and entry time (recommended format: T-PLATE-TIMESTAMP)

   When a **vehicle exits:**
   1. User enters license plate number
   2. The system finds the vehicle and its entry time
   3. The system calculates parking duration in hours
   4. The system calculates the fee based on the spot type and duration

5. System checks if there are **unpaid fines** (from previous parkings)
6. System shows:
    - Hours parked
    - Parking fee
    - Any unpaid fines
    - Total payment due
7. The system accepts payment
8. System marks the spot as available
9. System generates exit receipt

4. Fine Management

Fines are charged if:

- The vehicle stays **more than 24 hours** in a parking lot
- The vehicle stays in a **reserved spot without a reservation**

Note: Fines are linked to the **license plate number**, not the ticket.

Fine calculation (admin should be able to choose ANY schemes):

Option A: Fixed Fine Scheme

- Flat RM 50 fine for overstaying

Option B: Progressive Fine Scheme

- First 24 hours: RM 50
- Hours 24-48: Additional RM 100
- Hours 48-72: Additional RM 150
- Above 72 hours: Additional RM 200

Option C: Hourly Fine Scheme

- RM 20 per hour for overstaying
- Fines are added to the customer's account
- Customers can pay fines when exiting
- If the customer leaves without paying the fine, the next exit will show the unpaid fine + the current parking fee.

5. Payment Processing

The system must:

- Accept cash payment
- Accept card payment
- Show receipt with:
    - Entry time, exit time
    - Duration (hours)
    - Parking fee breakdown (hours × hourly rate)
    - Any fines due
    - Total amount paid
    - Payment method used
    - Remaining balance (if any)

6. User Interface Requirements

The application must have:

1. **Admin Panel:**
    - View all floors and spots
    - View occupancy rate
    - View revenue (total fees collected)

- View vehicles currently parked
- View unpaid fines
- Choose a fine scheme (Should be applied in future entries only)

2. **Entry/Exit Panel:**

    - Vehicle entry interface
    - Vehicle exit interface
    - Spot selection for entry
    - Payment processing

3. **Reporting Panel:**

    - List of all vehicles currently in the lot
    - Revenue report
    - Occupancy report
    - Fine report (outstanding fines)

Assignment Questions

1. **UML Diagrams**

Analyze the client requirements and produce:

a) Use Case Diagram
- Show actors: Customer, Admin, System
- Show use cases: Park vehicle, Exit vehicle, Pay fine, View spots, Process payment, etc.
- Show relationships and relationships

b) Class Diagram
- Show all classes: ParkingLot, Floor, ParkingSpot, Vehicle, Motorcycle, Car, SUV, Ticket, Payment, Fine, etc.
- Show relationships: inheritance, composition, aggregation
- Show attributes and methods for each class
- Include your design pattern classes

c) Sequence Diagrams
- **Scenario 1:** Vehicle parks (entry process)
- **Scenario 2:** Vehicle exits and pays the parking fee
- **Scenario 3:** Vehicle has an unpaid fine and exits

2. **Design Pattern Application:**
   - Choose one design pattern from the following list: Composite, Adapter, Bridge, Façade, Iterator, Observer, Builder, Prototype, or Singleton.
   - Incorporate the chosen design pattern into your overall design and reflect its usage in your class diagram.
   - Provide a clear oral justification (during interview) for your selection, explaining how the pattern enhances the design and supports future expansion.

3. **Java Swing GUI Implementation:**
   Implement the complete system in Java with these requirements:

a) Core Features (Must have all):
1. Create a parking lot with multiple floors and spot types
2. Vehicle entry - search for available spots, park the vehicle
3. Vehicle exit - calculate fee, process payment
4. Fine management - detect overstaying, calculate fines
5. Report generation - show current vehicles, revenue, occupancy

b) Code Quality:
- Code must compile **without errors**
- Code must follow your UML design exactly
- Classes are organized correctly in packages
- Methods have appropriate visibility (public/private)
- Comments for complex logic
- No warnings during compilation

c) GUI Requirements:
- Use Java Swing
- Professional appearance (not fancy, just clean)
- Multiple panels/tabs for different operations
- Input validation with error messages
- Button clicks trigger appropriate actions
- Display results clearly to the user
- Your implementation should closely follow your UML design.

d) Data Management:
- Use any database you are comfortable with

4. **Some preparation questions before the interview session (Interviewer may ask some questions from here or any other questions)**

   a) **Future-Proof Design Analysis. How would you handle these new requirements:**
      - Add a new vehicle type (e.g., Bus)
      - Add a new parking spot type (e.g., Electric vehicle charging spot at RM 8/hour)
      - Add a new fine scheme (e.g., Maximum cap fine - fine cannot exceed RM 500)
      - Add a reservation system (customers can reserve spots in advance)

   Show with code examples how your design supports these without significant refactoring (during interview, if asked).

   b) Explain OOP principles in your design:
      - How did you use encapsulation?
      - How did you use inheritance?
      - How did you use polymorphism?

   c) Explain how your design pattern helps:
      - What would be difficult without the pattern?
      - How does the pattern make adding new features easier?

   d) Prepare to answer in-depth questions regarding:
      - Your design decisions and UML diagrams.
      - The rationale behind the chosen design pattern.
      - How does your implementation demonstrate object-oriented principles and prepare the system for future enhancements?
      - Your responses during the interview will be critical in assessing your individual understanding.

Good Luck!

Marking Rubrics:

Criteria	Score and Descriptors	Weight (%)	Marks
Feature Fulfillment	**Features required:** **1. The system should allow the administrator to define and manage a multi-level parking lot structure, including multiple floors, rows, and parking spots with different types (Compact, Regular, Handicapped, Reserved). Each spot must maintain information such as spot ID, type, status (available/occupied), current vehicle (if any), and hourly rate based on the spot type.** **2. The system should support a complete vehicle entry process, where the user can register a vehicle with its type (Motorcycle, Car, SUV/Truck, Handicapped Vehicle), view suitable available spots, select an appropriate spot, and have the system validate the selection, assign the spot, record the entry time, and generate a parking ticket showing the spot details and entry information.** **3. The system should support a complete vehicle exit and billing process. Given a license plate number, the system must locate the parked vehicle, compute the parking duration (using the defined time calculation rule), calculate the parking fee based on spot type and duration, detect and apply any applicable fines (e.g., overstaying or misuse of reserved spot), include any unpaid fines from previous visits, and then produce a detailed bill showing duration, fee breakdown, fines, total amount due, and payment confirmation before marking the spot as available.** **4. The system should provide clear reporting and administrative views that allow the admin to see vehicles currently parked, overall occupancy (by floor and/or spot type), total revenue collected from parking fees and fines, and a summary of outstanding unpaid fines. These reports should be accessible via an intuitive interface and reflect the current state of the system during the program run.** **If fulfilled all 4 features → 20 marks** **If fulfilled 3 features →15 marks** **If fulfilled 2 features → 10 marks** **If fulfilled 1 feature → 5 marks** **If none fulfilled → 0 mark** **Partial marks are possible if unsatisfactory.**	**20**	
UML Diagram Fulfillment	**• Use Case Diagram correctly represents system interactions: 5 marks** **• Class Diagram accurately depicts system's structure and relationships: 5 marks** **• Sequence Diagrams clearly illustrate object interactions for key processes: 5 marks** **Partial marks are possible if unsatisfactory.**	**15**	
Design Pattern Usage	**• Design pattern is correctly applied and well-justified: 10 marks** **• Design pattern usage shows some missing parts or weak justification: 5 marks** **• No design pattern used or incorrect application: 0 mark** **Partial marks are possible if unsatisfactory.**	**10**	
Future-proof Application Design	**• Design demonstrates strong object-oriented principles and scalability (e.g., proper use of design patterns, modularity, separation of concerns): 15 marks** **• Minor issues in ensuring scalability or adherence to OO principles: 10 marks** **• Major issues in design: 5 marks** **• No scalable design provided: 0 mark** **Partial marks are possible if unsatisfactory.**	**15**	
Q&A with Interviewer	**• Answers are correct, complete, and elaborated with clear explanations and appropriate terminology: 40 marks** **• Answers are correct but somewhat brief or lacking depth: 30 marks** **• Answers are generally correct but contain significant omissions or errors: 20 marks** **• Answers are vague or incorrect: 0 mark** **Partial marks are possible if unsatisfactory. Interviewer need to ask several questions based on student's work. The focus is on individual student's understanding of overall subject matter and the project design and implementation.**	**40**	