ourdiu.com | Shared by Robiul Karim Jesan

Daffodil International University

**Department of Computer Science and Engineering**

**Faculty of Science & Information Technology**

**Final Examination, Fall 2024**

**Course Code: CSE221 Course Title: Object Oriented Programming**

**Level_Term: L2_T2 Batch: 64 Section: All**

**Time: 2 Hours** **Marks: 40**

## Answer ALL Questions

***[The figures in the right margin indicate the full marks and corresponding course outcomes. All portions of each question must be answered sequentially.]***

| | | | |
|---|---|---|---|
| 1. | You are a software engineer working for the **Election Commission of Bangladesh.** Your role involves designing a system to manage election data, including candidates, voters, and voting machines.<br><br>The Election Commission needs a system to manage different types of candidates, such as **Local Candidate** and **National Candidate**. Each candidate type shares basic properties but has unique characteristics as well. | | |
| i. | **Design an abstract class** `Candidate` with:<br>o Common attributes: `candidateID`, `name`, and `party`.<br>o An abstract method `displayInfo()` to display detailed information about the candidate.<br>o A concrete method `showBasicInfo()` that displays `candidateID` and `name`. | 3 | CO2 |
| ii. | Create subclasses `LocalCandidate` and `NationalCandidate` that inherit from `Candidate`. Each subclass should:<br>o Override `displayInfo()` to include additional attributes relevant to that type, such as `constituency` for `LocalCandidate` and `region` for `NationalCandidate`. | 3<br>7 | CO2<br>CO3 |
| iii. | **Draw a UML class diagram** to represent the structure, showing relationships between `Candidate`, `LocalCandidate`, and `NationalCandidate`. | 7 | CO2 |
| iv. | Write Java code to create instances of each subclass, display basic information, and demonstrate polymorphism by calling `displayInfo()` on each candidate.<br><br>**[N.B. assume data types which is appropriate]** | | |
| 2. | The Election Commission also needs to manage different types of **Voting Machines** used in elections. Each machine type, such as **Electronic Voting Machine (EVM)** and **Ballot Voting Machine (BVM)**, has common operations as well as unique properties. | | |

ourdiu.com | Shared by Robiul Karim Jesan

| | | | |
|---|---|---|---|
| | i. **Define an interface** `MachineOperations` with:<br>o A method `startOperation()` to initiate the voting process.<br>o A method `endOperation()` to end the voting process. | 3 | CO2 |
| | ii. Create an abstract class `VotingMachine` that implements `MachineOperations` and includes:<br>o Common attributes: `machineID` and `location`.<br>o A concrete method `showMachineInfo()` to display the machine's ID and location. | 3 | CO2 |
| | iii. Implement subclasses `EVM` and `BVM` that inherit from `VotingMachine`. Each subclass should:<br>o Override `startOperation()` and `endOperation()` to print unique messages for each type of machine.<br>o Include an additional attribute unique to each subclass, such as `batteryLevel` for `EVM` and `ballotCapacity` for `BVM`. | 4 | CO3 |
| | iv. **Draw a UML class diagram** showing the interface `MachineOperations`, abstract class `VotingMachine`, and its subclasses `EVM` and `BVM`.<br><br>[N.B. assume data types which is appropriate] | | |
| 3. | Reflect on your course project experience in the Object-Oriented Programming (OOP) in Java course. Describe how you used object-oriented principles (such as encapsulation, inheritance, polymorphism, or abstraction) to design and implement a solution for your project. In your answer, include: | | |
| | i. **Overview of the Project:** Briefly describe the project and the main objectives it aimed to achieve. | 2 | |
| | ii. **Application of OOP Principles:** Explain which OOP principles you applied and how each one contributed to structuring your solution. Provide examples from your code or design. | 3 | CO4 |
| | iii. **Challenges and Improvements:** Describe any challenges you faced when implementing these principles and discuss how you addressed them. If applicable, suggest any improvements you would make to the design based on what you learned. | 2 | |