**USN** 

**First/Third Semester B.E. / B.Arch. Semester End Examination, January-March 2022-23**

**Object Oriented Programming using JAVA**

**Time: 3 Hours Max. Marks: 100**

| ***Instructions:*** | ***1.*** | ***From Part A answer any 5 questions each Question Carries 6 Marks.*** |
| --- | --- | --- |
|  | ***2.*** | ***From Part B answer any one full question from each unit and each Question Carries 10 Marks.*** |
|  | ***3.*** | ***From Part C answer any one full question and each Question Carries 20 Marks.*** |
|  | ***4.*** | ***Make suitable assumptions wherever necessary.*** |

| **PART A** | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| **Answer any Five.** | | | **L** | **CO** | **PO** | **M** |
| **1.** | **List the Key attributes of Java. Explain any 5 of them** | | **(L2)** | **1** | **1** | **6** |
| **2.** | **Distinguish between a class and an object? Give the general form to create them with suitable example .** | | **Dist** | **1, 2** | **1,12** | **6** |
| **3.** | **What is instance variable hiding? Explain with a suitable example, how do you access the hidden instance variables?** | | **(L2)** | **1** | **1,12** | **(6)** |
| **4.** | **(i)Explain with an example inheritance and its benefits in object-oriented programming**  **(ii)Explain with an example the difference between single inheritance and multiple inheritance.** | | **(L2)** | **2** | **1,12** | **(6)** |
| **5** | **What is the difference between method overloading and method overriding, and how are they used to achieve polymorphic behavior in object-oriented programming?** | | **(L2)** | **2** | **1,12** | **(6)** |
| **6.** | **Define a string in Java. Explain any 4 methods that operate on string with an example** | | **(L2)** | **1** | **1,12** | **(6)** |
| **7.** | **What is garbage collection? Explain the general form and the importance of finalizer with an example** | | **(L2)** | **1** | **1,12** | **(6)** |
| **PART B(minimum L3 level questions)** | | | | | | |
|  |  | **UNIT - I** | **L** | **CO** | **PO** | **M** |
| **8** | **a.** | **Implement a Java class ArraySum which includes**   1. **a method sumA() to find the sum of the elements in an array ‘a’ with elements 1,2,3,4,5 initialized by a default constructor** 2. **a show() method to display the array elements and sum.** | | | | |
|  |  |  | **(L3)** | **1,2** | **1,12** | **( 6 )** |
|  | **b.** | **Implement a Java class MatrixSum() and perform the following.**   1. **Read the elements of 3X3 matrix.** 2. **Display the Matrix in matrix format.** 3. **Compute the sum and print the result of matrix.** | | | | |
|  |  |  | **(L3)** | **1,2** | **1,12** | **(4 )** |
|  |  | **OR** |  |  |  |  |
| **9** | **a.** | **Write a java program to display all the odd numbers between 1 to n using advanced for loop & if statement.** | | | | |
|  |  |  | **(L3)** | **1,2** | **1,12** | **( 5 )** |
|  | **b.** | **Develop a program to implement a Queue using arrays. Write methods (i)Add (ii)Delete (iii)Display.** |  |  |  |  |
|  |  |  | **(L3)** | **1,2** | **1,12** | **(5)** |
|  |  | **UNIT – II** | **L** | **CO** | **PO** | **M** |
| **10** | **a.** | **Write a class “SortingClass” that has three overloaded methods to sort data of int, float and String types. Demonstrate its working by passing an array of all the above three types. Use any sorting method.** | | | | |
|  |  |  | **(L3)** | **1, 2** | **1, 3, 12** | **6** |
|  | **b.** | **Demonstrate the concept of class and its members by writing a Java program to define a class Lamp. It can be in an on or off state. You can turn on and turn off lamp.** | | | | |
|  |  |  | **(L3)** | **1, 2** | **1, 2** | **4** |
|  |  | **OR** |  |  |  |  |
| **11** | **a.** | **Write a Java program to represent a Complex number. Include member functions to:**  **1. Initialize a complex number to a default value of zero (default constructor)**  **2. Initialize a complex number to a user defined value (parameterized constructor)**  **3. Add two complex numbers and return the result.**  **4. Subtract two complex numbers and return the result.**  **5. Display a complex number.** | | | | |
|  |  |  | **(L3)** | **1,2** | **1, 3, 12** | **6** |
|  | **b.** | **A company has 10 zonal sales offices in four zones namely, North, East, West and South. The company wants to organize the sales data of each of the office in each zone and find answers to queries such as,**  **1. Which office has performed the highest sales in each zone?**  **2. What is the average sales done by all the offices in each zone?**  **3. Which office among each zone has a poor sale?** | | | | |
|  |  |  | **(L3)** | **1, 2** | **1, 3, 12** | **4** |
|  |  | **UNIT - III** | **L** | **CO** | **PO** | **M** |
| **12** | **a.** | **Design an object-oriented library management system. The system should be able to keep track of books, their authors, and their availability. It should allow users to borrow and return books.**  **Create a Book class that stores information about the book, including its title, author, ISBN, and availability status.**  **Create an Author class that stores information about the author, including their name and contact information.**  **Create a Library class that stores information about all the books in the library and manages their availability status.**  **Implement a borrowing system that allows users to borrow books if they are available and return books when they are done.**  **Use object-oriented principles.** | | | | |
|  |  |  | **(L3)** | **1,2** | **1,3,12** | **6** |
|  | **b.** | **Develop an object-oriented python program to demonstrate inheritance as below.**  **MyScientificCalc implements member functions sin, cos, tan of a number x which inherits from a Calculator class (which has member functions add, subtract, multiply and divide). Instantiate an object of MyScientificCalc and show all possible operations.**  **Assume syntax for finding sin/cos/tan of a number x is result = sin(x) where x is in degrees. Inheritance diagram is as below.** | | | | |
|  |  |  | **(L3)** | **1,2** | **1,3,12** | **5** |
|  |  | **OR** |  |  |  |  |
| **13** | **a.** | **Develop a Java program with the definition as below.**  **The employee has different designations (teacher, principal).**  **The salary calculation should be based on the employee's designation.**  **The Principal gets an additional 20% (of basic Salary)**   | **Employee** | | --- | | **-name: String|**  **-basicSalary: double** | | **+Employee(name: String, basicSalary: double)**  **+getName(): String**  **+getBasicSalary(): double**  **+calculateSalary(): double** | |  | | **Teacher** | | **+Teacher(name: String, basicSalary: double)** | | **+calculateSalary(): double** | |  | | **Principal** | | **+Principal(name: String, basicSalary: double)** | | **+calculateSalary(): double** | |  | | | | | |
|  |  |  | **(L3)** | **1,2** | **1,3,12** | **6** |
|  | **b.** | **Develop a program that uses inheritance to model the relationship between different types of vehicles such as cars and trucks.**   1. **Create a base class called Vehicle with common attributes such as make, model, year, and mileage, and a method to calculate the total distance traveled based on the mileage and years of use.** 2. **Create subclasses for Car and Truck each inheriting from the Vehicle class and having additional attributes and methods specific to their vehicle type.** 3. **The Car subclass should have a method to calculate the total cost of ownership based on purchase price, fuel economy, and maintenance costs.** 4. **The Truck subclass should have a method to calculate the maximum weight it can carry based on the engine size and number of axles.** 5. **Demonstrates the functionality of each subclass method.** | | | | |
|  |  |  | **(L3)** | **( )** | **( )** | **5** |
|  |  | **UNIT - IV** | **L** | **CO** | **PO** | **M** |
| **14** | **a.** | **Write a java program to illustrate exception handling that handles the array index out of bound. Describe several commonly used methods defined by Throwable.** | | | | |
|  |  |  | **(L3)** | **(4)** | **(3)** | **(10)** |
|  |  | **OR** |  |  |  |  |
| **15** | **a.** | **Write a java program to demonstrate the implementation of creating packages.** | | | | |
|  |  |  | **(L3)** | **(4)** | **(3)** | **(10)** |
|  |  | **UNIT -V** | **L** | **CO** | **PO** | **M** |
| **16** | **a.** | **Write a java program to create an arraylist that adds five elements ‘A’,’B’,’C’,’D’,’E’and deletes the elements which is stored at array index 2 and 4.** | | | | |
|  |  |  | **(L3)** | **(5)** | **(3)** | **(10)** |
|  |  | **OR** |  |  |  |  |
| **17** | **a.** | **Demonstrate the Implementation of Lambda functions with example.** | | | | |
|  |  |  | **(L3)** | **(5)** | **(3)** | **(10)** |
| **PART C** | | | **L** | **CO** | **PO** | **M** |
| **18** | **a.** | **Devise an interface IndCurr to declare plus method. IndMoney class with two integer instance variables rupees and paise. Add a constructor with two parameters for initializing an IndMoney object. The constructor should check the paise value is between 0 and 99 and, if not, transfer some of the paise to the rupees variable to make it between 0 and 99. Override plus method of IndCurr in the class that takes an IndMoney object as a parameter. It creates and returns a new IndMoney object representing the sum of the object whose plus() method is being invoked and the parameter. It does not modify the values of the two existing objects. It should also ensure that the value of the paise instance variable of the new object is between 0 and 99. For example, if x is an IndMoney object with 12 rupees and 80 paise, and if y is an IndMoney object with 8 rupees and 90 paise, then x.plus(y) will return a new IndMoney object with 21 rupees and 70 paise. Also, create an IndMoneyDemo driver class that tests the IndMoney class** | | | | |
|  |  |  | **(L4)** | **2, 5** | **1, 2, 3,12** | **8** |
|  | **b.** | **Analyze the below code, write the output by putting the below values and justify your answer**  **i. n=0 ii. n=2** | | | | |
|  |  |  | **(L4)** | **3** | **1, 2, 12** | **6** |
|  | **c.** | **Analyze the following by example program**  **i. To append the element ‘a’ at third position.**  **ii. To insert an element into the array list at the first position.**  **iii. To remove the third element from a array list.** |  |  |  |  |
|  |  |  | **(L4)** | **5** | **3** | **6** |
|  |  | **OR** |  |  |  |  |
| **19** | **a.** | **Analyze the problem definition and develop a complete Java.program.**  **A company manufactures different types of electronic devices such as smartphones, laptops, and tablets. Each electronic device has a unique model number, brand name, and price. Additionally, smartphones have unique features like screen size, camera quality, and battery life, while laptops have unique features like screen size, RAM, and hard disk capacity. Tablets also have unique features like screen size, battery life, and storage capacity.**  **The company wants to design a software system to keep track of their inventory and sales. The software system should have the following functionalities:**   1. **Add a new electronic device to the inventory** 2. **Remove an electronic device from the inventory** 3. **Search for an electronic device by model number or brand name** 4. **Display the inventory list with all the details of each electronic device** 5. **Calculate the total revenue generated by the sale of electronic devices** |  |  |  |  |
|  |  |  | **(L4)** | **3** | **1,2,12** | **10** |
|  | **b.** | **Design a basic banking system that can handle two types of accounts: Savings Account and Current Account. Both account types should have the following attributes:**   * **Account Number: A unique identifier for each account.** * **Account Holder Name: The name of the person who owns the account.** * **Account Balance: The current balance in the account.**   **In addition to the above attributes, the Savings Account should have the following attributes:**   * **Interest Rate: The rate at which the account earns interest.** * **Minimum Balance: The minimum balance required to avoid fees.**   **And the Current Account should have the following attribute:**   * **Overdraft Limit: The maximum negative balance that can be held in the account.**   **Your implementation should use the following classes:**   * **Account: A base class that holds the common attributes of all accounts.** * **SavingsAccount: A subclass of Account that holds the additional attributes of a savings account.** * **CurrentAccount: A subclass of Account that holds the additional attributes of a current account.**   **Your implementation should also provide the following functionality:**   * **Deposit and withdraw money from an account.** * **Transfer money between accounts.** * **Calculate the interest earned by a savings account.** * **Check if a current account is overdrawn.** * **Print the details of an account, including the account number, account holder name, and account balance.**   **Implement a complete Java program for the above system using lambda expressions and collections, wherever appropriate.** |  |  |  |  |
|  |  |  | **L4** | **3** | **1,2,12** | **10** |