|  |  |
| --- | --- |
| **Name** | Shubhan Singh |
| **UID no.** | 2022300118 |
| **Experiment No.** | 4-C/2 |

|  |  |
| --- | --- |
| **PROBLEM STATEMENT :** | Implement the previous problem(cart problem) using OOP in Java |
| **THEORY:** | Classes and inheritance in Java:  In Java, classes and inheritance are two fundamental concepts that form the basis of object-oriented programming (OOP). Classes define the properties and behavior of objects, while inheritance allows you to create new classes that inherit the attributes and methods of existing classes.  A class in Java is a blueprint or template that defines the properties and behavior of objects. It includes variables, constructors, and methods that can be used to create objects. For example, you could create a class called "Car" that includes variables for the car's make, model, and year, as well as methods for accelerating and braking.  Inheritance allows you to create new classes that inherit the attributes and methods of existing classes. This means that you can create a new class that has all the same properties and behavior as an existing class, but with additional features or modifications. The existing class is called the superclass, and the new class is called the subclass. The subclass can override the superclass's methods or add new methods of its own.  To implement inheritance in Java, you use the "extends" keyword to indicate that a subclass is inheriting from a superclass. For example, you could create a subclass called "SUV" that extends the "Car" class. The SUV class would inherit all the properties and methods of the Car class, but could also have additional properties and methods specific to SUVs. |
| **PROGRAM:** | import java.util.\*; //Import class  // Restock class to handle restocking of items and returning total cost of different types of items class Restock {  Scanner scan = new Scanner(System.in);   // restock method to take input of perishable and non-perishable item prices  public void restock(int[][] *a*) {  System.out.println("Enter the price of all the perishable items one by one : ");  for (int i = 0; i < 4; i++) {  System.out.printf("%d : ", (i + 1));  *a*[0][i] = scan.nextInt();  }  System.out.println("Enter the price of non perishable items one by one : ");  for (int i = 0; i < 4; i++) {  System.out.printf("%d : ", (i + 1));  *a*[1][i] = scan.nextInt();  }  }   // getTotal\_cost method to calculate the total cost of all the items  int getTotal\_cost(int[][] *a*) {  int sum = 0;  for (int j = 0; j < 2; j++) {  for (int i = 0; i < 4; i++) {  sum += *a*[j][i];  }  }  return sum;  }   // getTotal\_perishable\_cost method to calculate the total cost of perishable items  int getTotal\_perishable\_cost(int[][] *a*) {  int sum = 0;  for (int i = 0; i < 4; i++) {  sum += *a*[0][i];  }  return sum;  }   // getTotal\_costliest\_nonperishable\_cost method to calculate the cost of costliest non-perishable item  int getTotal\_costliest\_nonperishable\_cost(int[][] *a*) {  int sum = *a*[1][0];  for (int i = 1; i < 4; i++) {  if (sum < *a*[1][i]) {  sum = *a*[1][i];  }  }  return sum;  }  }  // main class to handle multiple carts public class Supercart {  public static void main(String[] *args*) {  Scanner scan = new Scanner(System.in);  int[][] Conveyor\_belt = new int[2][4];  System.out.print("Enter number of carts : ");  int x = scan.nextInt();   // Creating array of Restock objects for each cart  Restock[] R = new Restock[x];   // Restocking of items for each cart  for (int i = 0; i < x; i++) {  R[i] = new Restock();  System.out.println("Cart no : " + (i + 1));  R[i].restock(Conveyor\_belt);  }   // Loop to perform different operations on the carts  do {  System.out.println("Choose an operation :\n1: Total cost of Cart\n2: Total cost of perishable\n3: Costliest non perishable item");  int choice = scan.nextInt();  switch (choice) {  case 1 -> {  // case 1 to calculate total cost of cart  System.out.println("Enter the cart number");  int y = scan.nextInt();  System.out.printf("The total cost of Cart %d is %d\n", y, R[y - 1].getTotal\_cost(Conveyor\_belt));  }  case 2 -> {  // case 2 to calculate total cost of perishable items  System.out.println("Enter the cart number");  int y = scan.nextInt();  System.out.printf("The total cost of perishable items in Cart %d is %d\n", y, R[y - 1].getTotal\_perishable\_cost(Conveyor\_belt));  }  case 3 -> {  // case 3 to calculate the costliest nonperishable items  System.out.println("Enter the cart number");  int y = scan.nextInt();  System.out.printf("The costliest non perishable item in the Cart %d is %d\n", y, R[y - 1].getTotal\_costliest\_nonperishable\_cost(Conveyor\_belt));  }  default -> System.out.println("Invalid Input!!");  }  System.out.println("Enter 5 to continue or 0 to exit");  } while (scan.nextInt() == 5);  } } |
| **RESULT:** | |