| **Name:** | Mahadev Balla |
| --- | --- |
| **UID:** | 2023300010 |
| **Experiment No.** | 5A |

| **AIM:** | Implement a program to demonstrate single / multilevel inheritance. |
| --- | --- |
| **Program 1** | |
| **PROBLEM STATEMENT :** | Define parent class "Employee" that has 3 private attributes String name, String id, int age. Employee has constructor with 3 arguments that set value of name, id, age. It also has getter and setter methods for all 3 private attributes. Class "SalariedEmployee" is a sub class of Employee and has 1 private attribute empSalary. "SalariedEmployee" can be permanent or on contract and has constructor SalariedEmployee(String name, String id, int age, double empSalary) to set the values. Constructor SalariedEmployee must call the superclass constructor to set name, id, age and call setter method to set the salary. Employee salary is empSalary + 2000(allowance) if he is a permanent employee else Employee salary is empSalary (no allowance). Accept the details of atleast 5 employees from the user and print details of the employee on the output screen. Then print the details of the employee with highest salary. |
| **PROGRAM:** | import java.util.\*;  class Employee{  private String name, id;  private int age;    Employee(String name, String id, int age){  this.name = name; this.id = id; this.age = age;  }    public void setName(String n){ name = n; }  public void setId(String i){ id = i; }  public void setAge(int a){ age = a; }  public String getName(){ return name; }  public String getId(){ return id; }  public int getAge(){ return age; }  }  class SalariedEmployee extends Employee{  private double empSalary;  private int temp;    public void setSalary(double salary){ empSalary = salary; }  public double getSalary(){ return empSalary; }  public void setTemp(int t){ temp = t; }  public double getTemp(){ return empSalary; }    SalariedEmployee(String name, String id, int age, double empSalary, int temp){  super(name, id, age);  setSalary(empSalary);  setTemp(temp);  }    public void printDetails(){  if(temp==1){  System.out.printf("%-20s %-20s %-20s %-20s %-20s\n", getName(), getId(), getAge(), (empSalary + 2000), "Permanent");  }  else{  System.out.printf("%-20s %-20s %-20s %-20s %-20s\n", getName(), getId(), getAge(), empSalary, "On Contract");  }  }  }  class employ{  public static void main(String args[]){  Scanner sc = new Scanner(System.in);  System.out.print("Enter the number of employees : ");  int n = sc.nextInt();  double a = 0, salary;  int b = 0, age;  sc.nextLine();  if(n>=5){  SalariedEmployee e[] = new SalariedEmployee[n];  for(int i=0; i<n; i++){  System.out.println("Enter details of employee " + (i+1));  System.out.print("Name : ");  String name = sc.nextLine();  System.out.print("Id : ");  String id = sc.nextLine();  do{ System.out.print("Age : ");  age = sc.nextInt();  if(age<=0){System.out.println("Age can't be negative/zero. Please enter a valid input!!");}  }  while(age<=0);  sc.nextLine();  do{System.out.print("Salary : ");  salary = sc.nextDouble();  if(salary>0 && salary<=2000){System.out.println("Salary can't be less than or equal to allowance. Please enter a valid input!!");}  else if(salary<=0){System.out.println("Salary can't be negative/zero. Please enter a valid input!!");}  }  while(salary<=2000);  if(a<salary){ a = salary; b = i; }  System.out.print("Are you a permanent employee ?\n(1 = Yes, Any other integer = No) : ");  int temp = sc.nextInt();  sc.nextLine();  e[i] = new SalariedEmployee(name,id,age,salary,temp);  }  System.out.printf("\n%-20s %-20s %-20s %-20s %-20s\n", "Name", "Id", "Age", "Salary", "Permanent/On Contract");  for(int j=0; j<e.length; j++){  e[j].printDetails();  }  System.out.println("\nDetails of employee with highest salary -\n");  System.out.printf("%-20s %-20s %-20s %-20s %-20s\n", "Name", "Id", "Age", "Salary", "Permanent/On Contract");  e[b].printDetails();  }  else{  System.out.println("You must enter details of atleast 5 employees.");  }  }  } |
| **RESULT:** | |
| **Program 2** | |
| **PROBLEM STATEMENT :** | The cost of stock on each day is given in an array A[] of size N.  Day 1 price in first location, day 2 price in second location etc. Find all the days on which you buy and sell the stock any number of time so that in between those days your profit is maximum.A new transaction can only start after previous transaction is complete. Person can hold only one share at a time. Create class Stock that has name of stock and array of prices. Also it has input method that initialises the predicted price of the stock in an array of length N. Create class Transaction that is sub class of Stock class. It has method findMaximumProfit method.  Example  Stock Prices: {1, 5, 2, 3, 7, 6, 4, 5}  Total profit earned is 10  Buy on day 1 and sell on day 2  Buy on day 3 and sell on day 5  Buy on day 7 and sell on day 8 |
| **PROGRAM:** | import java.util.\*;  class Stock {  private String name;  private double a[];  Stock(String n, double b[]) {  this.name = n;  this.a = b.clone();  }  public String getName() {  return name;  }  }  class Transaction extends Stock {  Transaction(String n, double a[]) {  super(n, a);  }  public void findMaxProfit() {  double max = 0;  int buyDay = 0;  int sellDay = 0;  for (int i = 0; i < a.length - 1; i++) {  if (a[i + 1] > a[i]) {  max += a[i + 1] - a[i];  if (buyDay == 0)  buyDay = i + 1;  sellDay = i + 2;  }  }  System.out.println("Max. profit in " + getName() + " : " + max);  if (max > 0) {  System.out.println("Buy on day " + buyDay + " and sell on day " + sellDay);  } else {  System.out.println("No profit can be made.");  }  }  }  class stocks {  public static void main(String args[]) {  Scanner sc = new Scanner(System.in);  System.out.print("Enter the number of stocks : ");  int n = sc.nextInt();  sc.nextLine();  Transaction t[] = new Transaction[n];  for (int i = 0; i < n; i++) {  System.out.print("Enter the name of stock " + (i + 1) + " : ");  String name = sc.nextLine();  double a[] = new double[8];  System.out.println("Enter the prices for stock " + name + " -");  for (int j = 0; j < 8; j++) {  System.out.print("Day " + (j + 1) + " : ");  a[j] = sc.nextDouble();  }  sc.nextLine();  t[i] = new Transaction(name, a);  }  for (int i = 0; i < t.length; i++) {  t[i].findMaxProfit();  }  }  } |
| **RESULT:** | |
| **CONCLUSION:** | Studied the implementation of arrays of objects to solve the given problems. |