Name	Shubhan Singh
UID no.	2022300118
Experiment No.	5-B

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 5 employees and print details of the employee with highest salary.* *Create class Tester with main method*
THEORY:	**Super and Final keywords in Java:** **1. 'super'** Keyword: The **super** keyword is used to refer to the superclass (parent class) of a class. It can be used in two ways: a. Accessing superclass members: By using the **super** keyword, you can access the methods or variables of the superclass that have been overridden or hidden by the subclass. This is useful when you want to invoke the superclass implementation of a method. b. Invoking superclass constructors: The **super** keyword can also be used to invoke the constructor of the superclass. It is used when you want to initialize the inherited members of the subclass using the superclass

	constructor. **2. 'final'** Keyword: The **final** keyword is used to declare entities (classes, methods, and variables) that cannot be modified or overridden. a. Final classes: When a class is declared as **final**, it cannot be subclassed or extended. This is useful when you want to prevent any further inheritance from a class. b. Final methods: When a method is declared as **final**, it cannot be overridden by any subclass. This is useful when you want to ensure that the implementation of a method remains the same across all subclasses. c. Final variables: When a variable is declared as **final**, its value cannot be changed once assigned. It becomes a constant. This is useful when you want to create a variable whose value should not be modified.
PROGRAM:	``` class Employee{ private String name,id; private int age; Employee(){}; // CONSTRUCTOR Employee(String name, String id, int age) { this.name = name; this.id = id; this.age = age; } // SETTERS public void setname(String name) { this.name = name; } public void setid(String id) { this.id = id; } public void setage(int age) { this.age = age; } // GETTERS public String getname() { return this.name; } public String getid() { return this.id; } public int getage() { return this.age; } ```

```
}
class SalariedEmployee extends Employee{
    private double empsalary;
    private int status;

    // CONSTRUCTOR OF CHILD CLASS
    SalariedEmployee(String name, String id, int age, double
empsalary) {
        super(name, id, age);
        setempSalary(empsalary);
    }

    // SETTER
    public void setempSalary(double empsalary) {
        this.empsalary = empsalary;
    }

    public void setstatus(int status) {
        this.status = status;
    }

    // GETTER
    public double getempSalary() {
        return this.empsalary;
    }

    public int getstatus() {
        return this.status;
    }
    // METHOD FOR ALLOWANCE
    public double Allowance() {

        if (this.status == 1) {
            this.empsalary = this.empsalary + 2000;
        }
        return this.empsalary;
    }
}
public class Employee_tester {
    public static void main(String[] args) {
        String name, id;
        int age;
        double empsalary;
        int status;
        Scanner sc = new Scanner(System.in);

        // CREATING AN ARRAY OF OBJECTS OF CHILD CLASS
        SalariedEmployee[] employee = new
SalariedEmployee[5];
        for (int i = 0; i < 5; i++) {
            System.out.printf("Enter the Name of Employee %d
:", i + 1);
            name = sc.nextLine();
            System.out.printf("Enter the ID of Employee %d
:", i + 1);
```

```
            id = sc.nextLine();
            System.out.printf("Enter the Age of Employee %d
:", i + 1);
            age = sc.nextInt();
            System.out.printf("Enter the Salary of Employee
%d :", i + 1);
            empsalary = sc.nextDouble();
            // TRANSFERRING THE DATA TO THE "ith" OBJECT OF
CHILD CLASS
            employee[i] = new SalariedEmployee(name, id, age,
empsalary);

            // SETTING THE STATUS SEPARATELY
            System.out.printf("Enter the Status of Employee
%d (1 for permanent 0 for commissioned):", i + 1);
            status = sc.nextInt();
            employee[i].setstatus(status);
            // CALLING THE ALLOWANCE METHOD TO CALCULATE THE
FINAL SALARY
            employee[i].Allowance();

            // BUFFER CLEAR
            sc.nextLine();
        }

        // TO FIND THE PERSON WITH MAXIMUM INCOME
        double max = 0;
        int index = 0;
        for (int i = 0; i < 5; i++) {
            if (max < employee[i].Allowance()) {
                max = employee[i].Allowance();
                // INDEX VARIABLE STORES THE "NUMBER" OF THE
EMPLOYEE WITH MAX INCOME
                index = i;
            }
        }
        // DISPLAYING THE DETAILS OF THE EMPLOYEE WITH MAX
INCOME
        System.out.printf("\nThe max salary is: %f", max);
        System.out.printf("\nEarned by the %dth Employee",
index + 1);
        System.out.printf("\nName: %s",
employee[index].getname());
        System.out.printf("\nID: %s",
employee[index].getid());
        System.out.printf("\nSalary: %f",
employee[index].Allowance());
        System.out.printf("\nAge: %d",
employee[index].getage());
        if (employee[index].getstatus() == 1)
            System.out.printf("\nStatus: Permanent");
        else
            System.out.println("Status: Commissioned");
        sc.close();
    }
}
```

RESULT:

```
Enter the Name of Employee 1 :Shubhan Singh
Enter the ID of Employee 1 :48422
Enter the Age of Employee 1 :18
Enter the Salary of Employee 1 :324822
Enter the Status of Employee 1 (1 for permanent 0 for commissioned):1
Enter the Name of Employee 2 :Vikas Kumar
Enter the ID of Employee 2 :32422
Enter the Age of Employee 2 :19
Enter the Salary of Employee 2 :39922
Enter the Status of Employee 2 (1 for permanent 0 for commissioned):0
Enter the Name of Employee 3 :Suryansh
Enter the ID of Employee 3 :32421
Enter the Age of Employee 3 :19
Enter the Salary of Employee 3 :392333
Enter the Status of Employee 3 (1 for permanent 0 for commissioned):0
Enter the Name of Employee 4 :Siddhesh
Enter the ID of Employee 4 :23211
Enter the Age of Employee 4 :14
Enter the Salary of Employee 4 :343847
Enter the Status of Employee 4 (1 for permanent 0 for commissioned):1
Enter the Name of Employee 5 :Nakshatra Shegaonkar
Enter the ID of Employee 5 :13223
Enter the Age of Employee 5 :18
Enter the Salary of Employee 5 :568339
Enter the Status of Employee 5 (1 for permanent 0 for commissioned):1

The max salary is: 574339.000000
Earned by the 5th Employee
```

```
The max salary is: 574339.000000
Earned by the 5th Employee
Name: Nakshatra Shegaonkar
ID: 13223
Salary: 576339.000000
Age: 18
Status: Permanent
Process finished with exit code 0
```