



Programming Test Results (With Test Cases)

Result Summary

Field	Value
Test ID	41183
Student ID	29111
Programs (with test cases)	1
Total Test Cases	5
Test Cases Passed	5
Fully Passed Programs	1
Partially Passed Programs	0
Failed Programs	0
Overall % (with test cases)	100.00%
Grade	Outstanding

Programs With Test Cases

#	Program Name	Total TC	Passed	Success Rate	Score /10	Submitted At	Attempts
1	EmployeeSalarySlip	5	5	100.0%	10	12/1/2025, 10:41:51 AM	0

Program Details (With Test Cases)

Program 1: EmployeeSalarySlip

Languages: java

Score (010): 10 / 10
Test Case Summary: Total: 5 Passed: 5
Failed: 0 Success: 100.0%
Attempts: 0
Submitted At: 12/1/2025, 10:41:51 AM
Description: Create a Employee Salary calculation Application project by using Method Overriding Concept to display different kinds of salary for PartTimeEmployee and FullTimeEmployee. Validate all the inputs properly and generate error message, if any input is not appropriate.

Create a BLC class called Employee

Fields :

```
id int protected  
name String protected
```

Use a parameterized constructor to initialize all the fields,

Methods :

1) Method Name : calculateSalary()
Argument : No Argument
Return Type : double
Access modifier : public
In this method return 0.0 for generic salary

Create another BLC class FullTimeEmployee which is sub class of Employee.

Field :

```
protected double salary;
```

Take a parameterized constructor to initialize super class and sub class properties. Validate all the inputs properly and generate error message, if any input is not appropriate.

[See the Test cases for Input Validation]

Method :

1) Method Name : calculateSalary()
Argument : No Argument

Return Type : double
Access modifier : public

In this overridden method return the salary of employee as a non static variable

Create another BLC class PartTimeEmployee which is sub class of Employee

Field :
protected double hourlyRate;
protected int hoursWorked;

Take a parameterized constructor to initialize super class and sub class properties.
Validate all the inputs properly and generate error message, if any input is not appropriate.

[See the Test cases for Input Validation]

Method :

1) Method Name : calculateSalary()

Argument : No Argument

Return Type : double

Access modifier : public

In this overridden method return the salary of employee as based on number of hours, he has

worked in the Organization.

Create an ELC class EmployeeSalarySlip with main method to test this application.

Constraints:

-

Sample Input:

Test Case 1 : ----- ***Salary Calculation Menu*** 1) FullTime Employees 2) PartTime Employees Please select the Employee type 1 Enter Fulltime Employee Id :101 Enter Fulltime Employee Name :Scott Enter the Salary :90000 Scott Salary is :90000.0

Sample Output:

Scott Salary is :10500.0

Explanation:

Based on the test cases develop the switch case based on the choice only it will take the required type input.

Solution Code

```
import java.util.*;
```

```

public class EmployeeSalarySlip{
    public static void main(String args[]){
        Scanner sc = new Scanner(System.in);
        int choice = sc.nextInt();

        switch(choice){
            case 1->{
                int id = sc.nextInt();
                String name = sc.next();
                double salary = sc.nextDouble();
                FullTimeEmployee fte = new FullTimeEmployee(id,name,salary);
                System.out.println(name+ " Salary is : "+fte.calculateSalary());
            }

            case 2->{
                int id = sc.nextInt();
                String name = sc.next();
                double hourlyRate = sc.nextDouble();
                int hoursWorked = sc.nextInt();
                PartTimeEmployee pte = new
PartTimeEmployee(id,name,hourlyRate,hoursWorked);
                System.out.println(name+ " Salary is :" +pte.calculateSalary());
            }
        }
    }
}

class Employee {
    protected int id;
    protected String name;

    public Employee(int id, String name){
        this.id = id;
        this.name = name;
    }

    public double calculateSalary(){
        return 0.0;
    }
}

```

```
class FullTimeEmployee extends Employee{
    protected double salary;

    public FullTimeEmployee(int id , String name, double salary){
        super(id, name);
        if(salary<=0){
            System.out.println("Salary can't be negative!!!!");
            System.exit(0);
        }
        this.salary = salary;
    }

    public double calculateSalary(){
        return this.salary;
    }
}

class PartTimeEmployee extends Employee{
    protected double hourlyRate;
    protected int hoursWorked;

    public PartTimeEmployee(int id , String name, double hourlyRate, int hoursWorked){
        super(id, name);
        if(hourlyRate<=0){
            System.out.println("Employee hourly rate can't be zero OR Negative");
            System.exit(0);
        }
        this.hourlyRate = hourlyRate;
        this.hoursWorked = hoursWorked;
    }

    public double calculateSalary(){
        return hourlyRate*hoursWorked;
    }
}
```