

Programming Test Results (With Test Cases)

Result Summary

Field	Value
Test ID	39154
Student ID	29195
Programs (with test cases)	1
Total Test Cases	7
Test Cases Passed	7
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	Submitted At	Attempts
1	TesterDayScholarHosteller	7	7	100.0%	10	13/11/2025, 13:48:55	0

Program Details (With Test Cases)

Program 1: TesterDayScholarHosteller

Field	Value
Program No.	1
Program Name	TesterDayScholarHosteller
Description	<p>Inheritance - Student(DayScholar and Hosteller) Payment Details</p> <p>A class Student is given to you. It contains the following implementation.</p> <p>Instance Variables: studentId : int , name :String , examFee:double</p> <p>Methods: displayDetails(): String , payFee() :double</p> <p>Define parameterized constructor and a toString method.</p> <p>Create two sub classes of Student called DayScholar and Hosteller. Add the following</p>

Field	Value
	<p>implementations to each class.</p> <p>DayScholar:</p> <p>Instance Variables : transportFee:double ,</p> <p>Methods: Define parameterized constructor</p> <p>Hosteller:</p> <p>Instance Variables: hostelFee :double</p> <p>Methods: Define parameterized constructor</p> <p>Add the following methods in BOTH classes.</p> <p>Method: displayDetails(): This method should return a string containing the details of the student in the following format:</p> <p>Student [name=John Smith,studentId=123,examFee=100.0] OR</p> <p>DayScholar[transportFee=500, name=John Smith,studentId=123,examFee=100.0] and so on.</p> <p>Method: payFee(amount): This method takes amount as parameter that represents the fee provided. You must calculate and then return the remaining amount paid in excess. If excess is paid, the returned amount will be negative. The total fees that each student must pay must include all the fees applicable to that Student(examFees, transportFees, hostelFees as applicable). Subtract the given amount from total fee for each student and return the remaining amount.</p> <p>An ELC class TesterDayScholarHosteller is given to you with a main method. Use this class to test your solution's</p> <p>classes and methods.</p> <p>Example Output:</p> <pre>Student[name=John Smith,studentId=1,examFee=25000.0] DayScholar[TransportFee=5000.0,name=Brian Lara,studentId=2,examFee=25000.0] Remaining amount to pay is: 12000.0 Hosteller[HostelFee=8000.0,name=Virat Kohli,studentId=3,examFee=25000.0] All Fees are clear</pre> <p>Test Cases 1:-</p> <p>-----</p> <p>Enter 1 for DayScholar, 2 for Hosteller: 1 Enter studentId 23 Enter name James Enter examFee 8000 Enter transportFee 2500 Enter amountPaid 10500</p>

Field	Value
	<p>Expected Output :-</p> <pre>DayScholar[TransportFee=2500.0,name=James,studentId=23,examFee=8000.0] All Fees are clear</pre> <p>=====</p> <p>Test Cases 2:-</p> <p>-----</p> <pre>Enter 1 for DayScholar, 2 for Hosteller: 1 Enter studentId 45 Enter name Kim Enter examFee 5000 Enter transportFee 2000 Enter amountPaid 6000</pre> <p>Expected Output :-</p> <pre>DayScholar[TransportFee=2000.0,name=Kim,studentId=45,examFee=5000.0] Remaining amount to pay is: 1000.0</pre> <p>=====</p> <p>Test Cases 3:-</p> <p>-----</p> <pre>Enter 1 for DayScholar, 2 for Hosteller: 2 Enter studentId 45 Enter name uma Enter examFee 5000 Enter HostelFee 4500 Enter amountPaid 8000</pre> <p>Expected Output :-</p> <pre>Hosteller[HostelFee=4500.0,name=uma,studentId=45,examFee=5000.0] Remaining amount to pay is: 1500.0</pre>
Constraints	
Sample Input	<pre>1 22 James 5000 2500 7500</pre>
Sample Output	<pre>DayScholar[TransportFee=2500.0,name=James,studentId=22,examFee=5000.0] All Fees are clear</pre>

Field	Value
Explanation	-
Language(s)	java
Total Test Cases	7
Test Cases Passed	7
Test Cases Failed	0
Success Rate	100.0%
Score (0–10)	10
Attempts	0
Submitted At	13/11/2025, 13:48:55

Code

```

import java.util.*;
public class TesterDayScholarHosteller{
    public static void main(String [] args ){
        Scanner sc = new Scanner(System.in);
        int choice = Integer.parseInt(sc.next());
        if(choice >2){
            System.out.println("Invalid Choice!");
            System.exit(0);
        }
        int id = Integer.parseInt(sc.next());
        String name =sc.next();
        double examFee =  Double.parseDouble(sc.next());
        if(examFee<0){
            System.out.println("Exam Fee should be Positive.");
            System.exit(0);
        }
        else if(id<0){
            System.out.println("Id Should be positive.");
            System.exit(0);
        }
    }
    double totalAmount=0;
    switch(choice){
        case 1 ->{
            double transportFee = Double.parseDouble(sc.next());
            totalAmount = Double.parseDouble(sc.next());
            if(totalAmount<=0){
                System.out.println("amount should be Positive");
                System.exit(0);
            }
            DayScholar d = new DayScholar(id,name,examFee,transportFee);
            System.out.println(d.displayDetails());
            System.out.println(d.calculate(totalAmount));
        }
        case 2 -> {
            double hostellFee = Double.parseDouble(sc.next());
            totalAmount = Double.parseDouble(sc.next());
            if(totalAmount<=0){
                System.out.println("amount should be Positive.");
                System.exit(0);
            }
            Hosteller h = new Hosteller(id,name,examFee,hostellFee);
            System.out.println(h.displayDetails());
            System.out.println(h.cal(totalAmount));
        }
        default ->{
        }
    }
    // sc.close();
}
}

class Student{
protected int sId;
protected String name;
protected double examFee;

Student(int sId,String name,double examFee){
    this.sId=sId;
    this.name=name;
    this.examFee = examFee;
}
Generated on 13/11/2025, 13:51:14
public String displayDetails(){
    return "";
}

```