| Name | Registration No. |
|---|---|
| Abdul Rehman Irfan | BSE213051 |

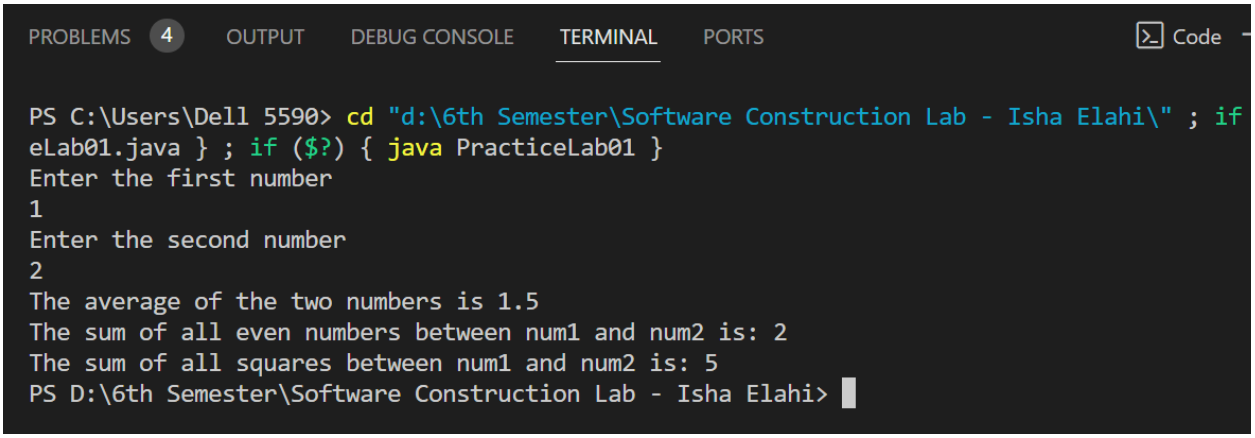
| | |
|---|---|
| Assignment No: | No # 1 |
| Date: | March 31 2024 |
| Subject: | SCD Lab |
| Submitted to: | Miss Isha Ilahi |

Capital University of Science and Technology

**Practice Task 1:**

Executing Basic program Write a java program that will ask the user to enter two numbers (integers). Then perform following tasks: 1) Calculate average of given numbers. 2) Sum of all even numbers between num1 and num2. 3) The sum of all squares between num1 and num2.

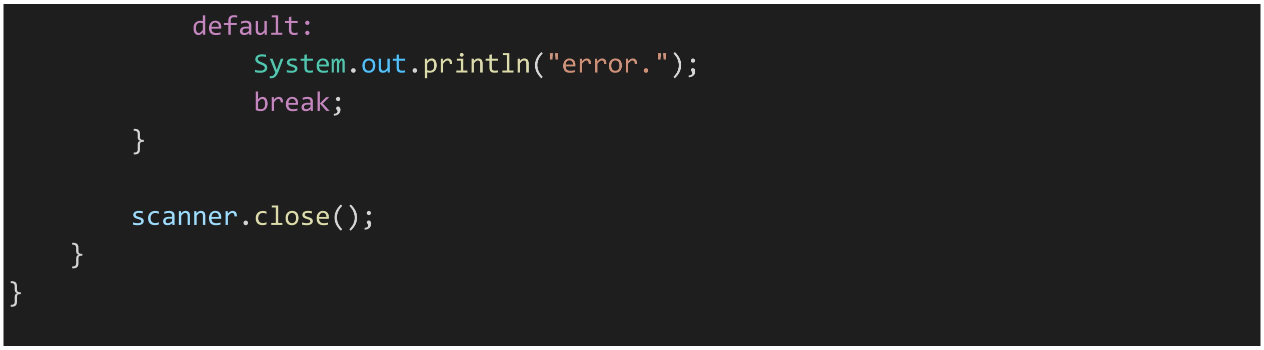
```
import java.util.Scanner;

public class PracticeLab01 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter the first number");
        int num1 = scanner.nextInt();

        System.out.println("Enter the second number");
        int num2 = scanner.nextInt();

        int start = num1 < num2 ? num1 : num2;
        int end = num1 > num2 ? num1 : num2;

        double average = (num1 + num2) / 2.0;
        System.out.println("The average of the two numbers is " + average);

        int evenSum = 0;
        for (int i = start; i <= end; i++) {
            if (i % 2 == 0) {
                evenSum += i;
            }
        }
        System.out.println("The sum of all even numbers between num1 and num2 is:
" + evenSum);

        int squareSum = 0;
        for (int i = start; i <= end; i++) {
            squareSum += i * i;
        }
        System.out.println("The sum of all squares between num1 and num2 is: " +
squareSum);

        scanner.close();
    }
}
```



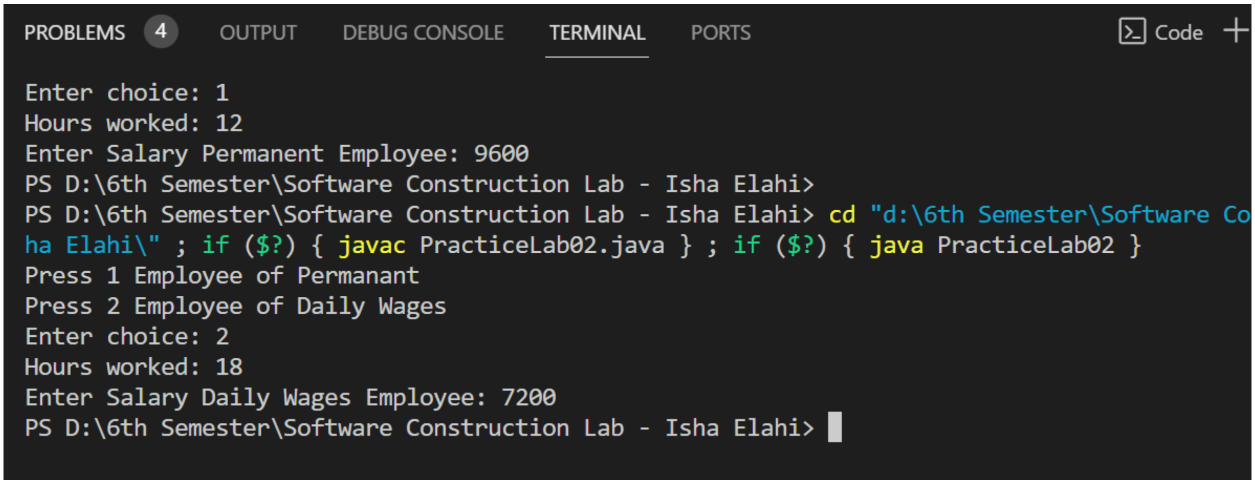
```
PS C:\Users\Dell 5590> cd "d:\6th Semester\Software Construction Lab - Isha Elahi\" ; if
eLab01.java } ; if ($?) { java PracticeLab01 }
Enter the first number
1
Enter the second number
2
The average of the two numbers is 1.5
The sum of all even numbers between num1 and num2 is: 2
The sum of all squares between num1 and num2 is: 5
PS D:\6th Semester\Software Construction Lab - Isha Elahi>
```

**Practice Task 2:**
Write a program that displays the following menu to the user: Press 1 for Permanent Employee Press 2 for Daily Wages Employee After selecting the appropriate employee calculate salary for the employees. The daily wages employees are paid 400 per hour while permanent employees are paid 800 per hour. First you need to ask for the number of hours for which the employee has worked so far and then calculate and display the salary of each type of employee. Perform the above-mentioned task using switch statement.

```
import java.util.Scanner;

public class PracticeLab02 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Press 1 Employee of Permanant");
        System.out.println("Press 2 Employee of Daily Wages");
        System.out.print("Enter choice: ");
        int choice = scanner.nextInt();

        System.out.print("Hours worked: ");
        int hoursWorked = scanner.nextInt();

        switch (choice) {
            case 1:
                int salaryPermanent = 800 * hoursWorked;
                System.out.println("Enter Salary Permanent Employee: " +
salaryPermanent);
                break;
            case 2:
                int salaryDailyWages = 400 * hoursWorked;
                System.out.println("Enter Salary Daily Wages Employee: " +
salaryDailyWages);
                break;
```

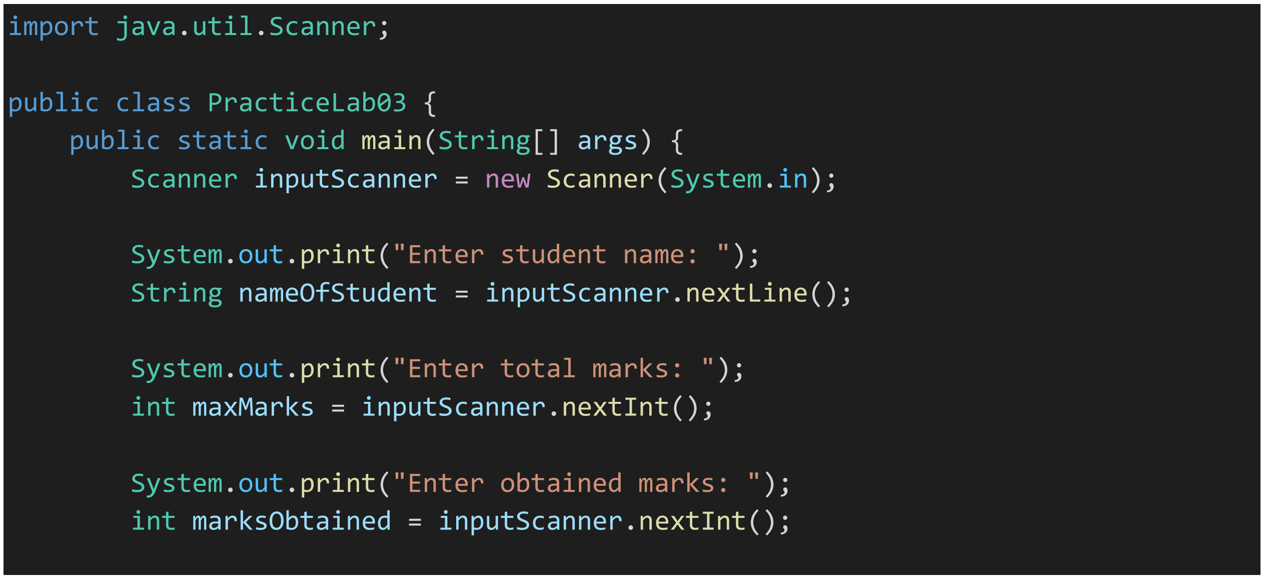


```
            default:
                System.out.println("error.");
                break;
        }

        scanner.close();
    }
}
```

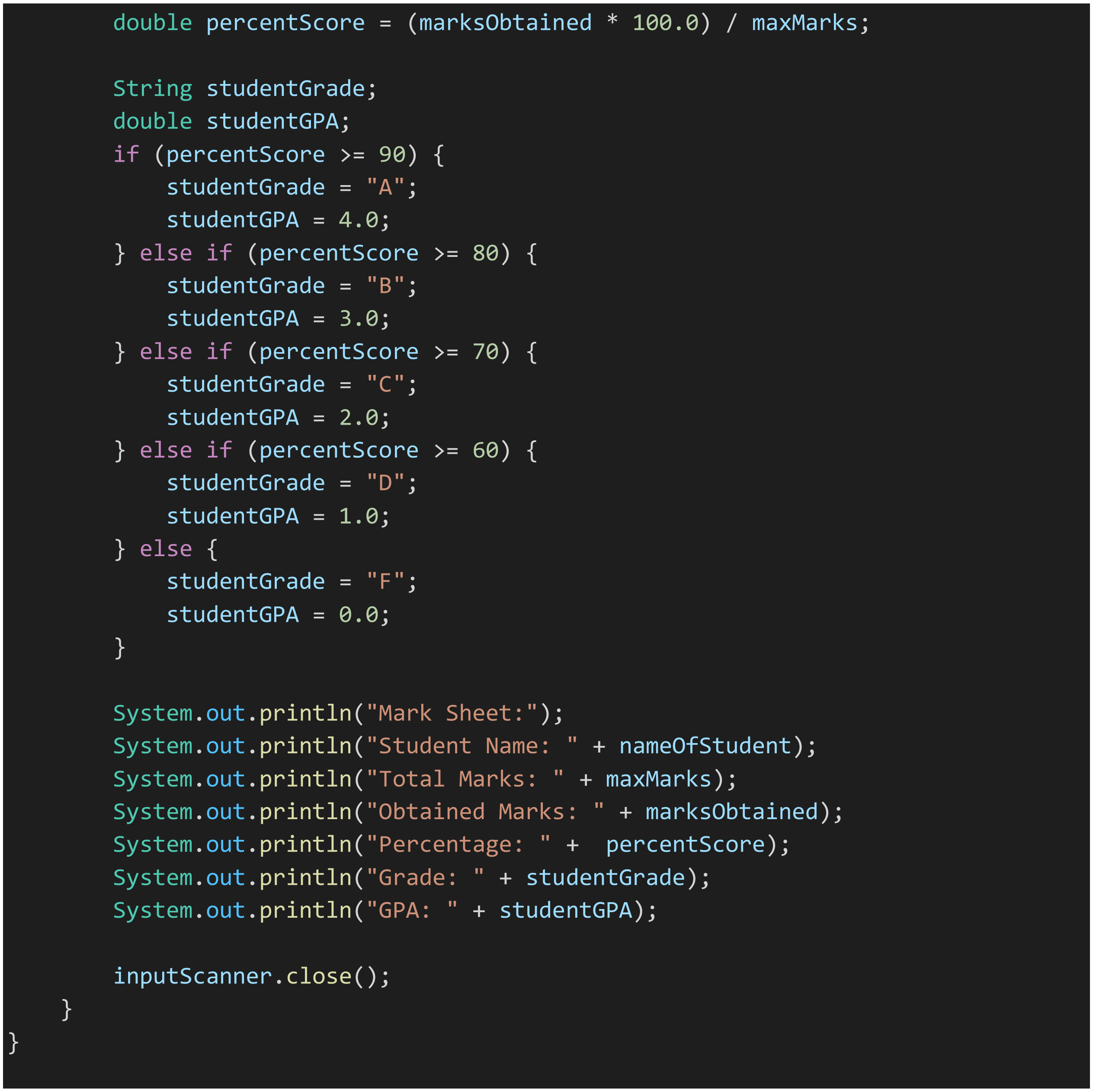


```
Enter choice: 1
Hours worked: 12
Enter Salary Permanent Employee: 9600
PS D:\6th Semester\Software Construction Lab - Isha Elahi>
PS D:\6th Semester\Software Construction Lab - Isha Elahi> cd "d:\6th Semester\Software Co
ha Elahi\" ; if ($?) { javac PracticeLab02.java } ; if ($?) { java PracticeLab02 }
Press 1 Employee of Permanant
Press 2 Employee of Daily Wages
Enter choice: 2
Hours worked: 18
Enter Salary Daily Wages Employee: 7200
PS D:\6th Semester\Software Construction Lab - Isha Elahi>
```

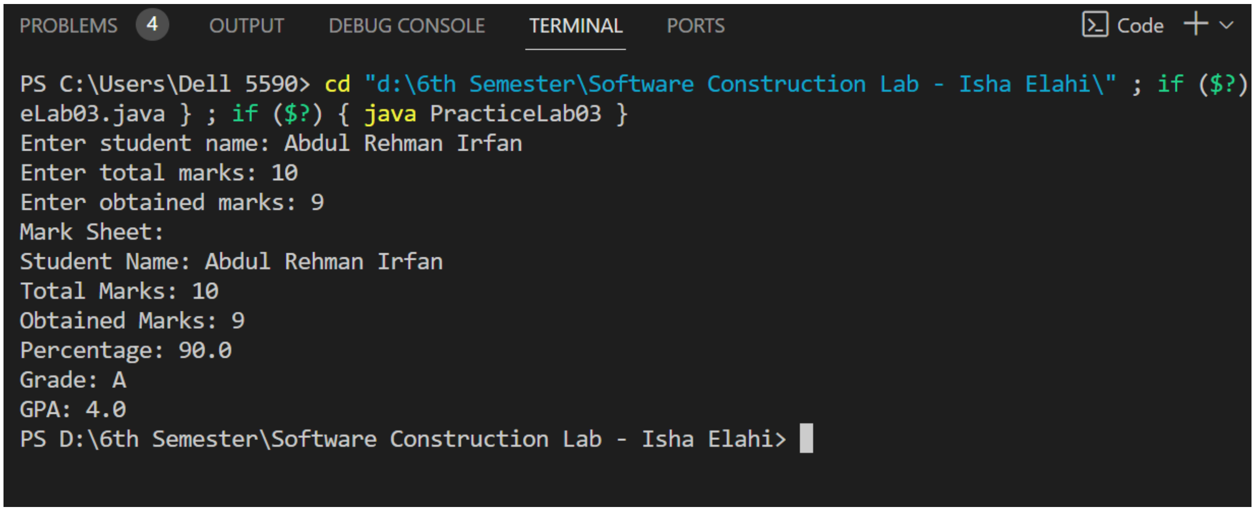
**Practice Task 3:**
Create a design for the mark sheet by taking runtime value of student name, total marks, obtained marks and calculate its percentage, grade and GPA. Use good practices of programming that we have studied and ensure that the outcomes should be presented in a proper Viewable approach.



```
import java.util.Scanner;

public class PracticeLab03 {
    public static void main(String[] args) {
        Scanner inputScanner = new Scanner(System.in);

        System.out.print("Enter student name: ");
        String nameOfStudent = inputScanner.nextLine();

        System.out.print("Enter total marks: ");
        int maxMarks = inputScanner.nextInt();

        System.out.print("Enter obtained marks: ");
        int marksObtained = inputScanner.nextInt();
```



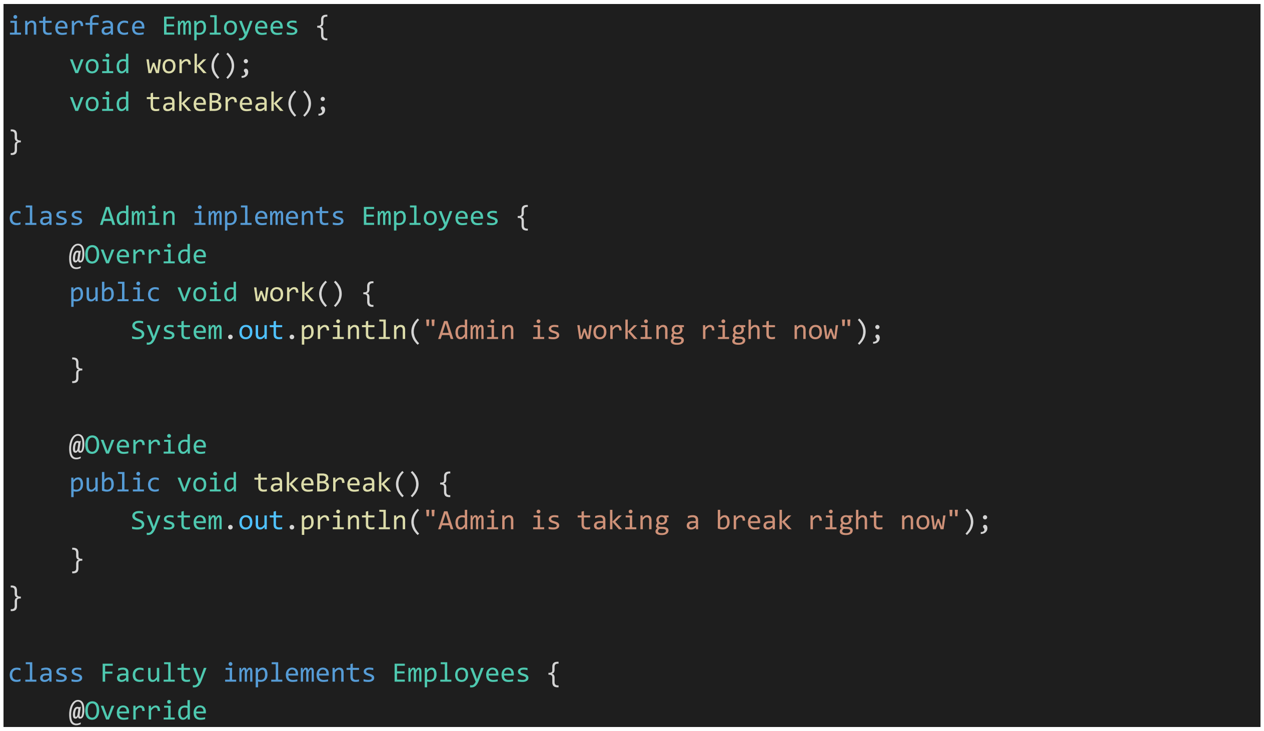
```
        double percentScore = (marksObtained * 100.0) / maxMarks;

        String studentGrade;
        double studentGPA;
        if (percentScore >= 90) {
            studentGrade = "A";
            studentGPA = 4.0;
        } else if (percentScore >= 80) {
            studentGrade = "B";
            studentGPA = 3.0;
        } else if (percentScore >= 70) {
            studentGrade = "C";
            studentGPA = 2.0;
        } else if (percentScore >= 60) {
            studentGrade = "D";
            studentGPA = 1.0;
        } else {
            studentGrade = "F";
            studentGPA = 0.0;
        }

        System.out.println("Mark Sheet:");
        System.out.println("Student Name: " + nameOfStudent);
        System.out.println("Total Marks: " + maxMarks);
        System.out.println("Obtained Marks: " + marksObtained);
        System.out.println("Percentage: " +  percentScore);
        System.out.println("Grade: " + studentGrade);
        System.out.println("GPA: " + studentGPA);

        inputScanner.close();
    }
}
```



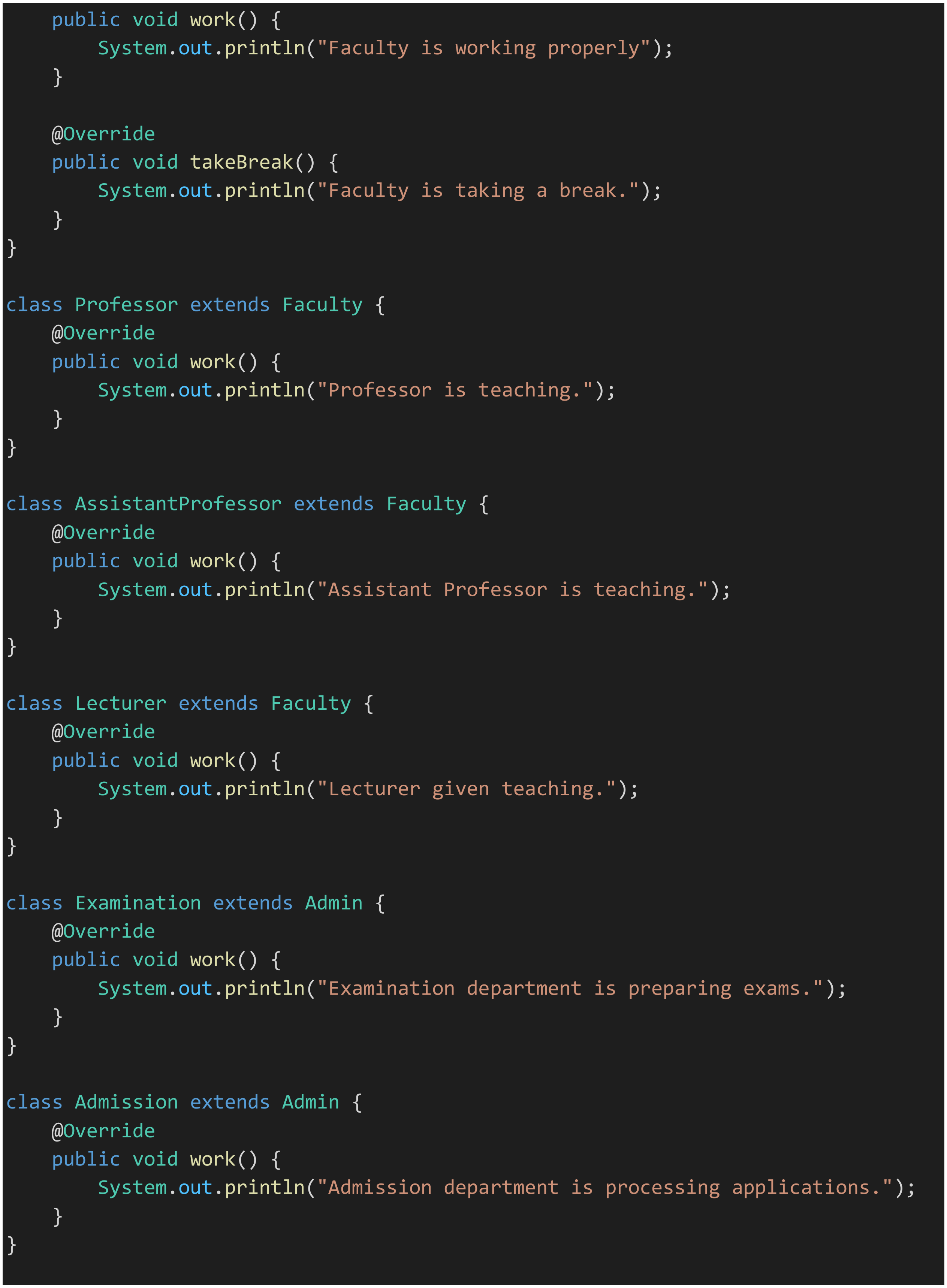
```
PS C:\Users\Dell 5590> cd "d:\6th Semester\Software Construction Lab - Isha Elahi\" ; if ($?)
eLab03.java } ; if ($?) { java PracticeLab03 }
Enter student name: Abdul Rehman Irfan
Enter total marks: 10
Enter obtained marks: 9
Mark Sheet:
Student Name: Abdul Rehman Irfan
Total Marks: 10
Obtained Marks: 9
Percentage: 90.0
Grade: A
GPA: 4.0
PS D:\6th Semester\Software Construction Lab - Isha Elahi>
```

**Practice Task 4:**
Setting Up Class Structure Convert the following scenario into java code. Design a university management system, an interface of ‘Employees’ that will have two/three method declarations. This interface will be implemented by two classes i.e. Admin and Faculty. The faculty class will be extended by Professor, Assistant Professor, and Lecturer. On the other hand, admin class will be extended by examination and admission classes. All these inheritances will be called in the main class by instantiation. (The internal mechanism or method implementation is choice dependent; you can perform whatever you want).

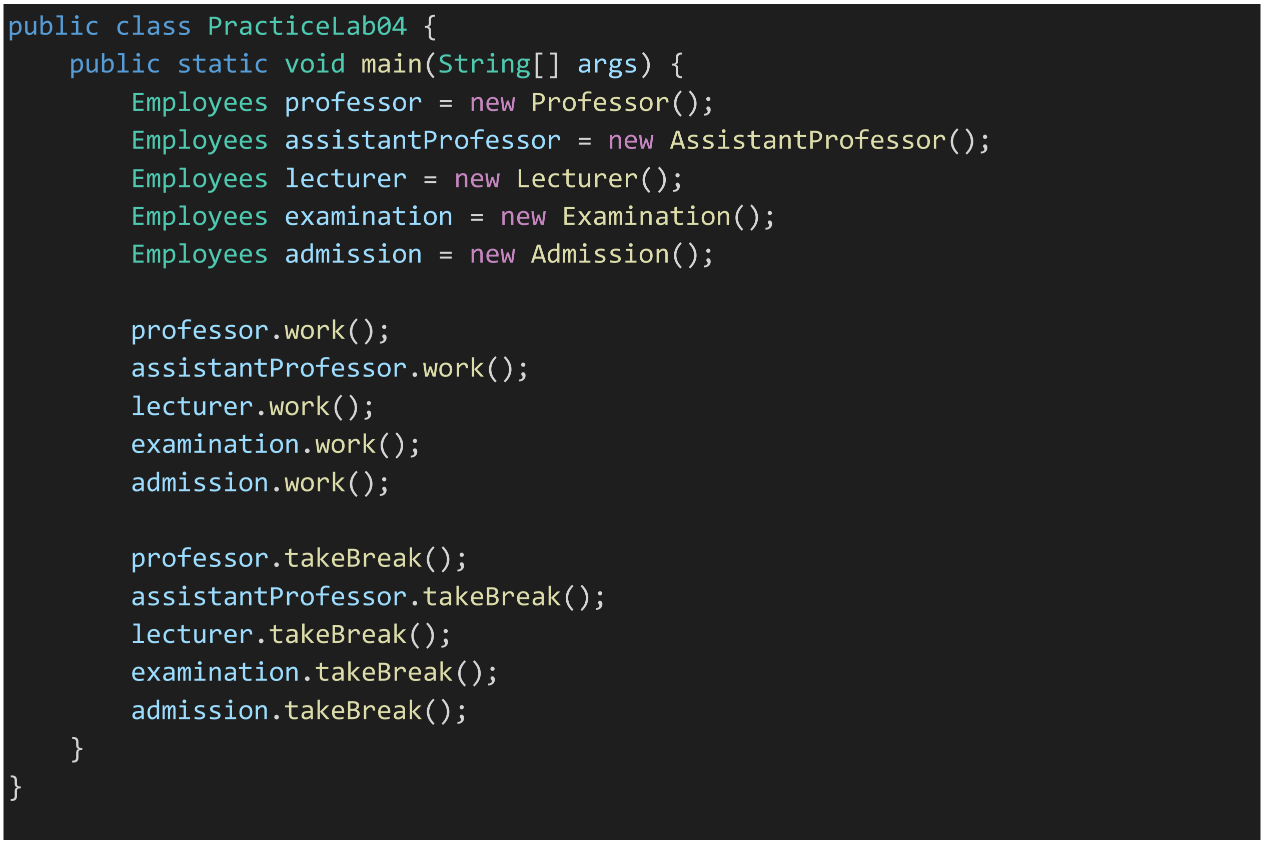
Note: I have done this code all classes in one file



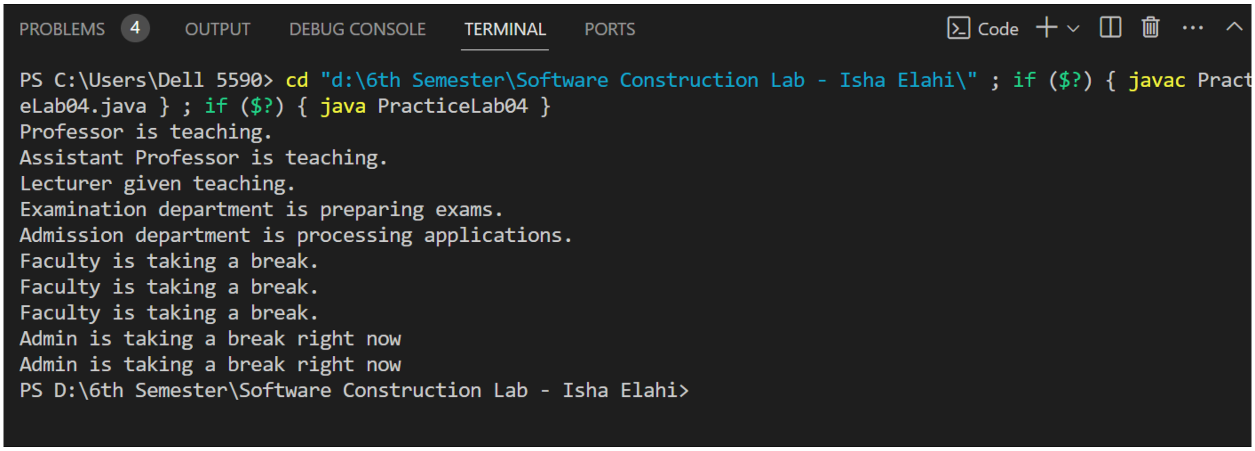
```java
interface Employees {
    void work();
    void takeBreak();
}

class Admin implements Employees {
    @Override
    public void work() {
        System.out.println("Admin is working right now");
    }

    @Override
    public void takeBreak() {
        System.out.println("Admin is taking a break right now");
    }
}

class Faculty implements Employees {
    @Override
```



```java
    public void work() {
        System.out.println("Faculty is working properly");
    }

    @Override
    public void takeBreak() {
        System.out.println("Faculty is taking a break.");
    }
}

class Professor extends Faculty {
    @Override
    public void work() {
        System.out.println("Professor is teaching.");
    }
}

class AssistantProfessor extends Faculty {
    @Override
    public void work() {
        System.out.println("Assistant Professor is teaching.");
    }
}

class Lecturer extends Faculty {
    @Override
    public void work() {
        System.out.println("Lecturer given teaching.");
    }
}

class Examination extends Admin {
    @Override
    public void work() {
        System.out.println("Examination department is preparing exams.");
    }
}

class Admission extends Admin {
    @Override
    public void work() {
        System.out.println("Admission department is processing applications.");
    }
}
```



```java
public class PracticeLab04 {
    public static void main(String[] args) {
        Employees professor = new Professor();
        Employees assistantProfessor = new AssistantProfessor();
        Employees lecturer = new Lecturer();
        Employees examination = new Examination();
        Employees admission = new Admission();

        professor.work();
        assistantProfessor.work();
        lecturer.work();
        examination.work();
        admission.work();

        professor.takeBreak();
        assistantProfessor.takeBreak();
        lecturer.takeBreak();
        examination.takeBreak();
        admission.takeBreak();
    }
}
```



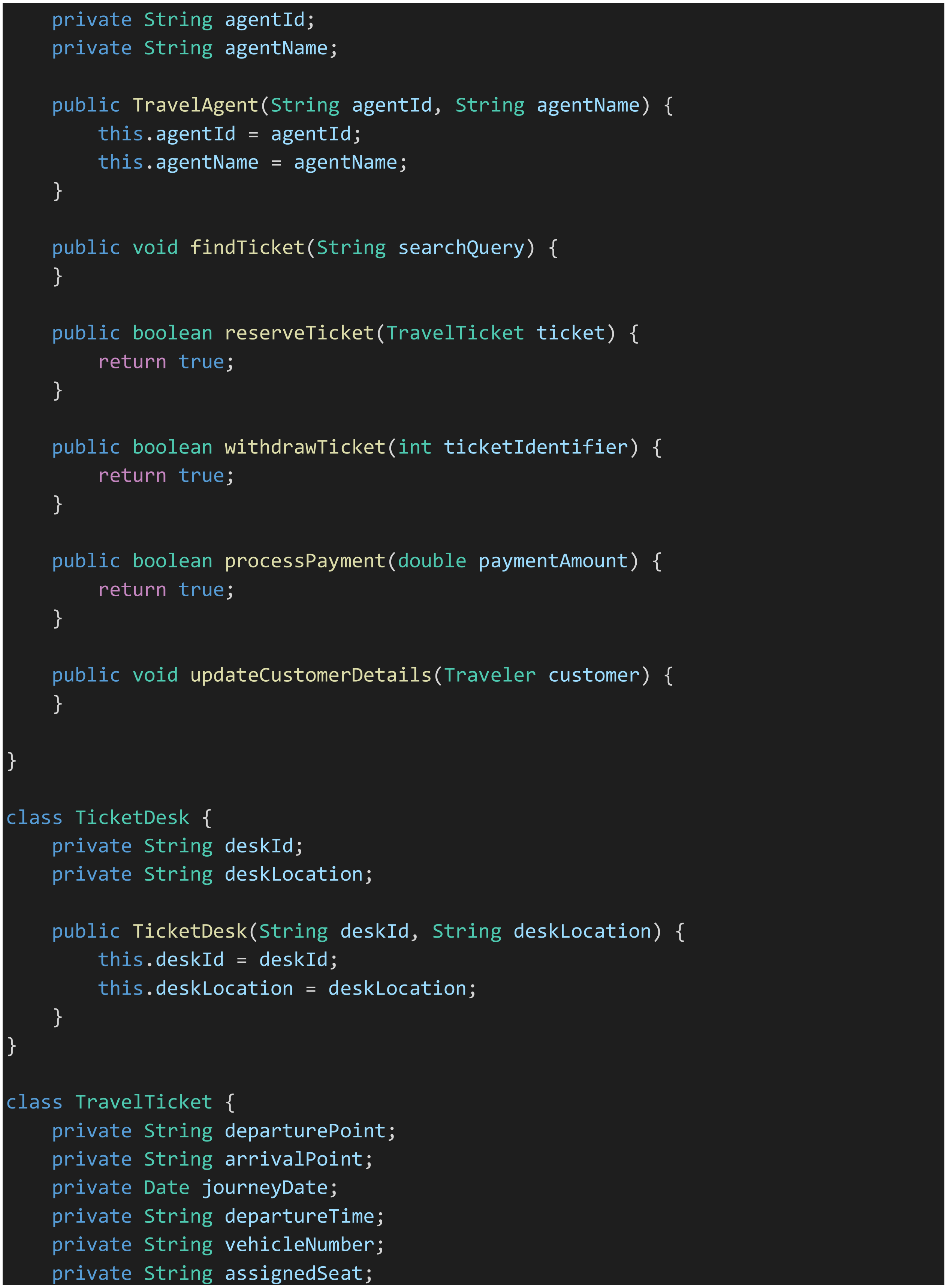
```
PS C:\Users\Dell 5590> cd "d:\6th Semester\Software Construction Lab - Isha Elahi\" ; if ($?) { javac Pract
eLab04.java } ; if ($?) { java PracticeLab04 }
Professor is teaching.
Assistant Professor is teaching.
Lecturer given teaching.
Examination department is preparing exams.
Admission department is processing applications.
Faculty is taking a break.
Faculty is taking a break.
Faculty is taking a break.
Admin is taking a break right now
Admin is taking a break right now
PS D:\6th Semester\Software Construction Lab - Isha Elahi>
```

**Practice Task 5:**

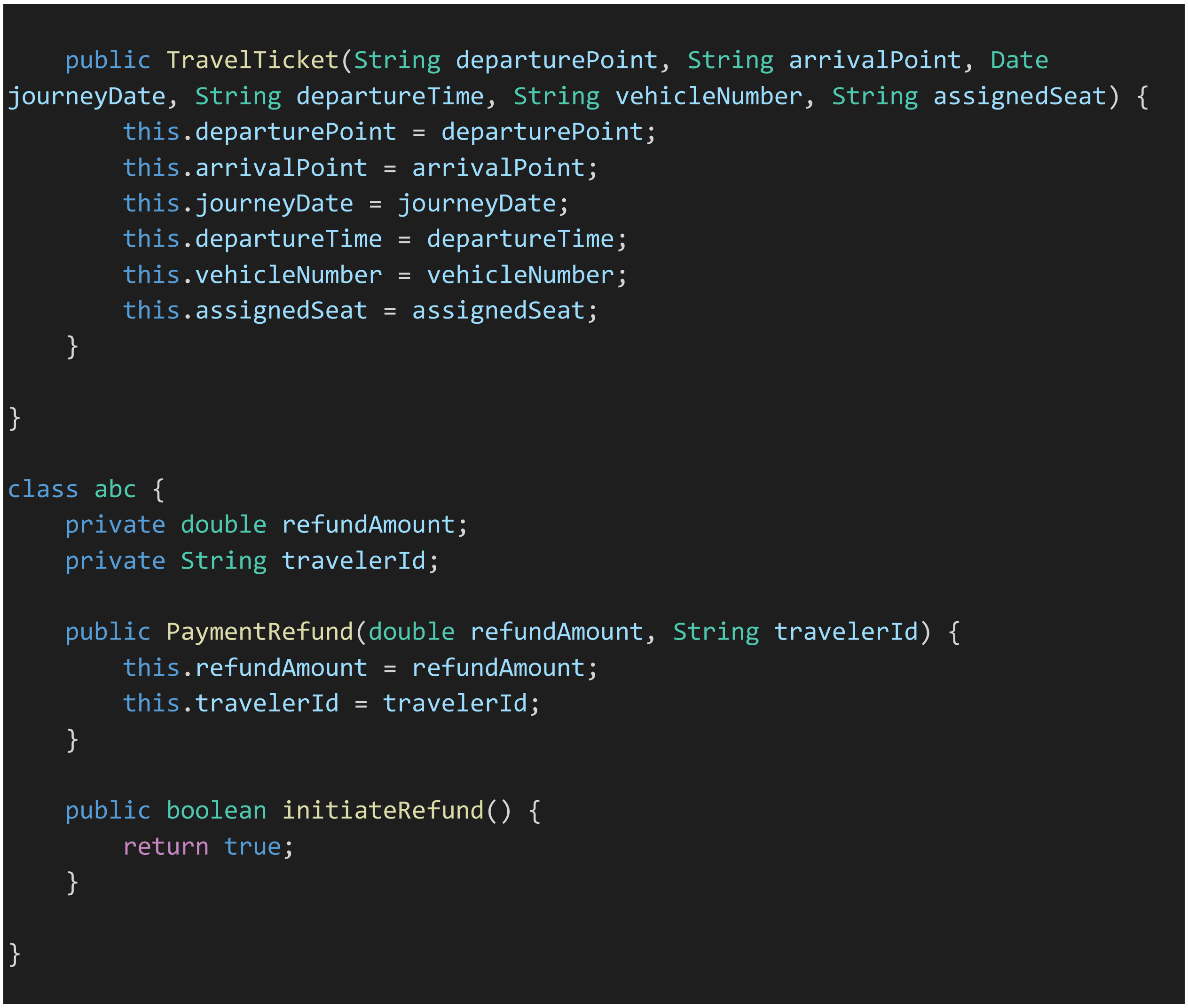
Setting Up Class Structure- Mapping Design Elements to Code

Note: Here code is generic and simple

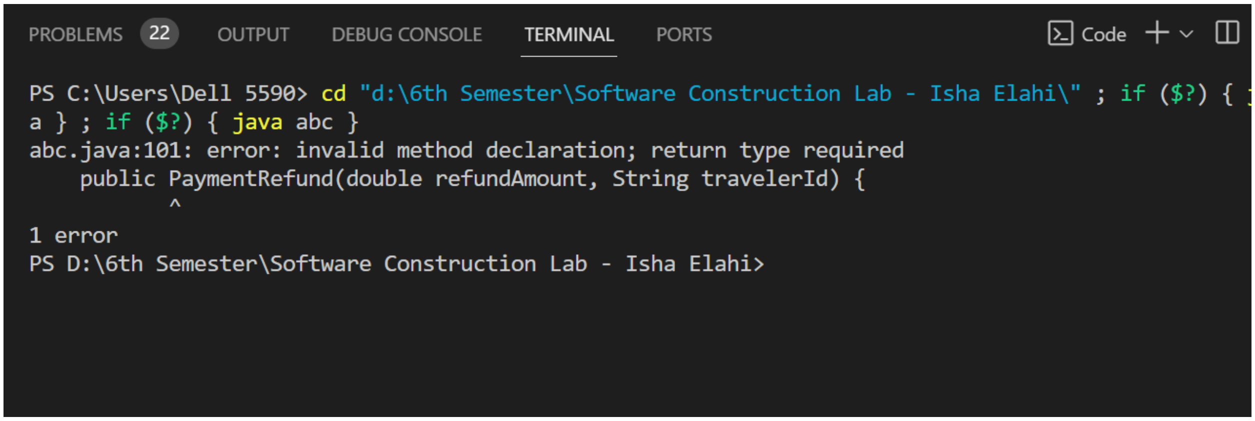
```
import java.util.Date;

interface TicketOperations {
    void findTicket(String searchQuery);
    boolean reserveTicket(TravelTicket ticket);
    boolean withdrawTicket(int ticketIdentifier);
    boolean processPayment(double paymentAmount);
    void updateCustomerDetails(Traveler customer);
}

class Traveler {
    private String travelerId;
    private String fullName;
    private String residentialAddress;
    private String contactNumber;
    private int currentAge;

    public Traveler(String travelerId, String fullName, String
residentialAddress, String contactNumber, int currentAge) {
        this.travelerId = travelerId;
        this.fullName = fullName;
        this.residentialAddress = residentialAddress;
        this.contactNumber = contactNumber;
        this.currentAge = currentAge;
    }

    public void provideDetails(String fullName, String residentialAddress, String
contactNumber, int currentAge) {
        this.fullName = fullName;
        this.residentialAddress = residentialAddress;
        this.contactNumber = contactNumber;
        this.currentAge = currentAge;
    }

    public void alterDetails(String fullName, String residentialAddress, String
contactNumber, int currentAge) {
        provideDetails(fullName, residentialAddress, contactNumber, currentAge);
    }

}

class TravelAgent implements TicketOperations {
```



```
    private String agentId;
    private String agentName;

    public TravelAgent(String agentId, String agentName) {
        this.agentId = agentId;
        this.agentName = agentName;
    }

    public void findTicket(String searchQuery) {
    }

    public boolean reserveTicket(TravelTicket ticket) {
        return true;
    }

    public boolean withdrawTicket(int ticketIdentifier) {
        return true;
    }

    public boolean processPayment(double paymentAmount) {
        return true;
    }

    public void updateCustomerDetails(Traveler customer) {
    }

}

class TicketDesk {
    private String deskId;
    private String deskLocation;

    public TicketDesk(String deskId, String deskLocation) {
        this.deskId = deskId;
        this.deskLocation = deskLocation;
    }
}

class TravelTicket {
    private String departurePoint;
    private String arrivalPoint;
    private Date journeyDate;
    private String departureTime;
    private String vehicleNumber;
    private String assignedSeat;
```



```
    public TravelTicket(String departurePoint, String arrivalPoint, Date
journeyDate, String departureTime, String vehicleNumber, String assignedSeat) {
        this.departurePoint = departurePoint;
        this.arrivalPoint = arrivalPoint;
        this.journeyDate = journeyDate;
        this.departureTime = departureTime;
        this.vehicleNumber = vehicleNumber;
        this.assignedSeat = assignedSeat;
    }

}

class abc {
    private double refundAmount;
    private String travelerId;

    public PaymentRefund(double refundAmount, String travelerId) {
        this.refundAmount = refundAmount;
        this.travelerId = travelerId;
    }

    public boolean initiateRefund() {
        return true;
    }

}
```



```
PS C:\Users\Dell 5590> cd "d:\6th Semester\Software Construction Lab - Isha Elahi\" ; if ($?) {
a } ; if ($?) { java abc }
abc.java:101: error: invalid method declaration; return type required
    public PaymentRefund(double refundAmount, String travelerId) {
           ^
1 error
PS D:\6th Semester\Software Construction Lab - Isha Elahi>
```