Group-2

Find MemberShip Category Count

Grade settings: Maximum grade: 100
Disable external file upload, paste and drop external content: Yes
Run: Yes **Evaluate**: Yes
Automatic grade: Yes **Maximum execution time**: 32 s

Find Membership Category Count

ZEE Shopping mall wanted to know how many Members are available for each of their Membership category. The member ship category is of three types (Gold, Silver and Platinum).

You being their software consultant have been approached by them to develop an application which can be used for managing their business. You need to implement a java program using thread to find out the count of members in each membership category. Membership details should be obtained from the user in the console.

Component Specification: Member (Model Class)

Type(Class)	Attributes	Methods	Responsibilities
Member	String memberId String memberName String category	Include getters and setter method for all the attributes. Include a three argument constructor in the given order – memberId, memberName and category.	Set the values for all the attributes via constructor.

Note: The class and methods should be declared as public and all the attributes should be declared as private.

Requirement 1: Count the number of members

Count the number of members available in the memberList based on the membership category to be searched and set the value to count attribute.

Component Specification: ZEEShop (Thread Class)

Component Name	Type(Class)	Attributes	Methods	Responsibilities
	ZEEShop	String memberCategory	Include getters and setter	Set the values for all the

		int count List<Member>memberList	method for all the attributes. Include a two argument constructor with arguments – memberCategory and memberList.	attributes via constructor.
Count the number of members	ZEEShop		void run()	Count the number of members based on the Membership category and set the value to the countattribute.

Note: The class and methods should be declared as public and all the attributes should be declared as private.

Create a class called **Main** with the main method and get the inputs like **number of members**, **member details**, **number of times Membership category needs to be searched** and **Membership category to be searched** from the user**.**

The member details will be in the form of String in the following format **memberId:memberName:category**.

Parse the member details and set the values for all attributes in **Member** class using **constructor**.

Invoke the ZEEShop thread class for each memberCategory and count the number of members in that category and display the count as shown in the sample input and output.

Assumption: The **memberCategory** is **case –sensitive** and will be of only three values – **Platinum** or **Gold** or **Silver**.

Note:

In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user and the remaining text represents the output.

Ensure to follow object oriented specifications provided in the question description.
Ensure to provide the names for classes, attributes and methods as specified in the question description.

Adhere to the code template, if provided.

Sample Input / Output 1:

Enter the number of Members:

5

Enter the Member Details:

101:Tom:Gold

Enter the Member Details:

102:Sam:Gold

Enter the Member Details:

103:John:Silver

Enter the Member Details:

104:Rose:Platinum

Enter the Member Details:

105:Tint:Silver

Enter the number of times Membership category needs to be searched:

4

Enter the Category

Gold

Enter the Category

Silver

Enter the Category

Platinum

Enter the Category

Gold

Gold:2

Silver:2

Platinum:1

Automatic evaluation[+]

Member.java

```
1
2 public class Member {
3
4         private String memberId;
5         private String memberName;
6         private String category;
7
8         public String getMemberId() {
9                 return memberId;
10        }
11        public void setMemberId(String memberId) {
12                this.memberId = memberId;
13        }
14        public String getMemberName() {
15                return memberName;
16        }
17        public void setMemberName(String memberName) {
18                this.memberName = memberName;
19        }
20        public String getCategory() {
21                return category;
22        }
23        public void setCategory(String category) {
24                this.category = category;
25        }
26
27        public Member(String memberId, String memberName, String category) {
28                super();
29                this.memberId = memberId;
30                this.memberName = memberName;
31                this.category = category;
32        }
33
34
35 }
```

Main.java

```
1 import java.util.*;
2 public class Main {
3
4   public static void main(String args[]){
5      // Fill the code here
6      List<Member> memberList = new ArrayList<Member>();
7      Scanner scan = new Scanner(System.in);
8      System.out.println("Enter the no of Members");
9      int memberCount = scan.nextInt();
10      String tempIp;
11      while(memberCount>0){
12         System.out.println("Enter the member details");
13         tempIp = scan.next();
14         String tempArr[] = tempIp.split(":");
15         memberList.add(new Member(tempArr[0],tempArr[1],tempArr[2]));
16         memberCount--;
17      }
18      System.out.println("Enter the number of times Membership category needs to be searched");
19      int noOfTimes = scan.nextInt();
20      String[] tempArr = new String[noOfTimes];
```

```
        for(int index=0;index<noOfTimes;index++){
            System.out.println("Enter the category");
            tempArr[index] = scan.next();
        }
        int countArr[] = new int [noOfTimes];
        for(int i=0; i<noOfTimes;i++){
            ZEEShop thread = new ZEEShop(tempArr[i],memberList);
            thread.run();
            /*try{
                thread.join();
            }catch(InterruptedException e){

            }*/
            countArr[i] = thread.getCount();
        }
        for(int i=0;i<noOfTimes;i++){
            System.out.println(tempArr[i]+ ":"+countArr[i]);
        }
        scan.close();
        /*List<ZEEShop> zList = new ArrayList<ZEEShop>()
        for(int i = 0;i<count;i++){
            ZEEShop zs = new ZEEShop(category , memList);
            zList.add(zs);
        }
        for(ZEEShop z: zeelist){
            z.start();
            try{
                z.join();
            }catch(Exception e){
                e.printStackTrace();
            }
        }*/
    }
}

```

ZEEShop.java

```
import java.util.*;
public class ZEEShop extends Thread {
    // Fill the code here
    private String memberCategory;
    private int count;
    private List<Member> memberList;
    public ZEEShop(String memberCategory, List memberList){
        super();
        this.memberCategory = memberCategory;
         this.memberList = memberList;
    }
     public int getCount(){
         return count;
     }
     public String getMemberCategory(){
         return memberCategory;
     }
     public List<Member> getMemberList(){
         return memberList;
     }
     public void setMemberCategory(String memberCategory){
         this.memberCategory = memberCategory;
     }
     public void setMemberList(List<Member> memberList){
         this.memberList = memberList;
     }
     public void setCount(int count){
         this.count = count;
     }
```

```
30    public void run(){
31
32      synchronized(this)
33      {
34      for(Member m : memberList){
35          if(m.getCategory().equals(memberCategory))
36              count++;
37      }
38
39    }
40    }
41 }
42
```

Grade

Reviewed on Friday, 7 January 2022, 7:25 PM by Automatic grade
Grade 100 / 100
Assessment report
[+]**Grading and Feedback**

Grade Calculation

Grade settings: Maximum grade: 100
Disable external file upload, paste and drop external content: Yes
Run: Yes **Evaluate**: Yes
Automatic grade: Yes **Maximum execution time**: 32 s

Grade Calculation

Rita is working as a science teacher in an International school. She is the Class Teacher of class V and was busy in calculating the grade for each student in her class, based on his/her total marks obtained in SA1 assessment.

Since she found it very difficult to calculate the grade, she approached you to develop an application which can be used for completing her task faster. You need to implement a java program using thread to calculate the grade for each student. Student details should be obtained from the user in the console.

Requirement 1: Calculate the grade for each student.

Calculate the grade based on total marks (sum of all marks) as shown below obtained by each student and set the same in result attribute for respective student.

Total Marks	Grade
400 to 500	A
300 to 399	B
200 to 299	C

Less than 200	E

Assumption: Each student will have only five subjects and marks of each subject will be greater than or equal to 0 and lesser than or equal to 100. Hence the maximum Total marks obtained by each student will be 500. And the minimum Total marks obtained by each student will be 0.

Component Specification: GradeCalculator (Thread Class)

Component Name	Type(Class)	Attributes	Methods	Responsibilities
	GradeCalculator	String studName char result int[] marks	Include getters and setter method for all the attributes. Include a two argument constructor in the given order – studName and marks.	Set the values for all the attributes via constructor.
calculate the grade for each student	GradeCalculator		public void run()	Calculate the grade based on total marks and set the same to result attribute.

Note: The class and methods should be declared as public and all the attributes should be declared as private.

Create a class called **Main** with the main method and get the inputs like **number of threads** and **Student details** from the user**.**

The student details will be in the form of String in the following format **studName:mark1:mark2:mark3:mark4:mark5**.

Parse the student details and set the values of studName and marks attributes in **GradeCalculator** thread class using **constructor**.

Invoke the **GradeCalculator** thread class to calculate the grade based on total marks and set the same to result attribute.

Display the Student name and Grade obtained by each student as shown in the sample input and output.

Note:

In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user and the remaining text represents the output.

Ensure to follow object oriented specifications provided in the question description.
Ensure to provide the names for classes, attributes and methods as specified in the question description.

Adhere to the code template, if provided.

Sample Input / Output 1:

Enter the number of Threads:

4

Enter the String:

Jeba:100:80:90:40:55

Enter the String

David:10:8:9:40:5

Enter the String

Adam:90:80:90:50:75

Enter the String

Rohit:99:99:99:99:99

Jeba:B

David:E

Adam:B

Rohit:A

Automatic evaluation[+]

Main.java

```
1 import java.util.Scanner;
2 import java.io.BufferedReader;
3 import java.io.InputStreamReader;
4 public class Main {
5         public static void main(String[] args)  throws Exception {
6             BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
7             System.out.println("Enter the number of Threads");
8             int th=Integer.parseInt(br.readLine());
9             GradeCalculator obj=null;
10            String str="";
11            String[] details=new String[th];
```

```
            for(int i=0;i<th;i++)
            {
                System.out.println("Enter the String");
                str=br.readLine();
                details[i]=str;
            }
            for(int i=0;i<th;i++)
            {
                String sp[]=details[i].split(":");
                int k=0;
                int arr[]=new int[sp.length];
                for(int j=1;j<sp.length;j++)
                    arr[k++]=Integer.parseInt(sp[j]);
                obj=new GradeCalculator(sp[0],arr);
                obj.start();
                try{
                        Thread.sleep(1000);
                    }
                catch(Exception e)
                    {
                        System.out.println(e);
                    }
            }
            //Fill your code here

        }

}
```

GradeCalculator.java

```

public class GradeCalculator extends Thread{
 private String studName;
 private char result;
 private int[] marks;
 public String getStudName()
 {
    return studName;
 }
  public void setStudName()
  {
     this.studName=studName;
  }
  public char getResult()
  {
     return result;
  }
  public void setResult(char result)
  {
     this.result=result;
  }
  public int[] getMarks()
  {
     return marks;
  }
  public void setMarks(int[] marks)
  {
     this.marks=marks;
  }
  public GradeCalculator(String studName,int[] marks)
  {
     this.studName=studName;
     this.marks=marks;
  }
  public void run()
  {
```

```
37      int sum=0;
38      int[] score=getMarks();
39      for(int i=0;i<score.length;i++)
40      sum=sum+score[i];
41      if((400<=sum)&&(sum<=500))
42      System.out.println(getStudName()+":"+'A');
43      if((300<=sum)&&(sum<=399))
44      System.out.println(getStudName()+":"+'B');
45      if((200<=sum)&&(sum<=299))
46      System.out.println(getStudName()+":"+'C');
47      if(sum<200)
48      System.out.println(getStudName()+":"+'E');
49   }
50 }
```

Grade

Reviewed on Friday, 7 January 2022, 7:24 PM by Automatic grade
Grade 100 / 100
Assessment report
[+]**Grading and Feedback**

Query Data Set

Grade settings: Maximum grade: 100
Disable external file upload, paste and drop external content: Yes
Run: Yes **Evaluate**: Yes
Automatic grade: Yes **Maximum execution time**: 32 s

Query Data Set

Jackson is pursuing his Bachelor's degree in TekSpec University, Alaska. His professor has given him a weekend assignment to develop an application in java using inner class concept.

You being his best friend, help him in completing his weekend assignment. You need to implement a java program using inner class concept to display the details available in primaryDataSet, secondaryDataSet and Query.

Requirement 1: Display the details in primaryDataSet, secondaryDataSet and Query

Component Specification: Query (Model Class)

Component Name	Type(Class)	Attributes	Methods	Responsibilities
	Query	String queryId String queryCategory DataSet primaryDataSet	Include getters and setter method for all the attributes.	

		DataSet secondaryDataSet		
	DataSet	String theatreId String theatreName String location int noOfScreen double ticketCost	Include getters and setter method for all the attributes.	The DataSet class must be available only to Query class. So the DataSet class should come as a Inner class in Query class.
Display the details	Query	toString()		Override the toString method to produce the output as specified in the sample output

Note: The class and methods should be declared as public and all the attributes should be declared as private.

Create a class called **TestApplication** with the main method and get the inputs for primary data set and secondary data set like **theatreId**, **theatreName**, **location**, **noOfScreen** and **ticketCost**, and details of Query like **queryId** and **queryCategory** from the user**.**

Display the details of primary data set, secondary data set and Query as shown in the sample input and output.

Note:

In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user and the remaining text represents the output.

Ensure to follow object oriented specifications provided in the question description.
Ensure to provide the names for classes, attributes and methods as specified in the question description.

Adhere to the code template, if provided.

Sample Input / Output 1:

Enter the Details for primary data set

Enter the theatre id

PNR6001

Enter the theatre name

KV cinemas

Enter the location

Chennai

Enter the no of screens

8

Enter the ticket cost

120

Enter the Details for secondary data set

Enter the theatre id

RNV5001

Enter the theatre name

Inoxe

Enter the location

Bangalore

Enter the no of screens

5

Enter the ticket cost

150

Enter the query id

Q510

Enter the query category

DML

Primary data set

Theatre id : PNR6001

Theatre name : KV cinemas

Location : Chennai

No of Screen : 8

Ticket Cost : 120

Secondary data set

Theatre id : RNV5001

Theatre name : Inoxe

Location : Bangalore

No of Screen : 5

Ticket Cost : 150

Query id : Q510

Query category : DML

Automatic evaluation[+]

Query.java

```
1
2 //Write the required business logic as expected in the question description
3
4 public class Query {
5    private String queryId;
6    private String queryCategory;
7    private DataSet primaryDataSet;
8    private DataSet secondaryDataSet;
9
10    @Override
11    public String toString()
12    {
13       String g="";
14       g+=("Primary data set"+"\n");
15       g+=("Theatre id :"+primaryDataSet.getTheatreId()+"\n");
16       g+=("Theatre name :"+primaryDataSet.getTheatreName()+"\n");
17       g+=("Location :"+primaryDataSet.getLocation()+"\n");
18       g+=("No of Screen :"+primaryDataSet.getNoOfScreen()+"\n");
19       g+=("Ticket Cost :"+primaryDataSet.getTicketCost()+"\n");
20
21       g+=("Secondary data set"+"\n");
22       g+=("Theatre id :"+secondaryDataSet.getTheatreId()+"\n");
23       g+=("Theatre name :"+secondaryDataSet.getTheatreName()+"\n");
24       g+=("Location :"+secondaryDataSet.getLocation()+"\n");
25       g+=("No of Screen :"+secondaryDataSet.getNoOfScreen()+"\n");
26       g+=("Ticket Cost :"+secondaryDataSet.getTicketCost()+"\n");
27       g+=("Query id : "+queryId+"\n");
28       g+=("Query category : "+queryCategory+"\n");
29
30       return g;
31    }
32    public class DataSet{
```

```
33      private String theatreId;
34      private String theatreName;
35      private String location;
36      private int noOfScreen;
37      private double ticketCost;
38
39      public double getTicketCost()
40      {
41          return ticketCost;
42      }
43      public void setTicketCost(double a)
44      {
45          ticketCost=a;
46      }
47
48      public int getNoOfScreen()
49      {
50          return noOfScreen;
51      }
52      public void setNoOfScreen(int a)
53      {
54          noOfScreen=a;
55      }
56      public String getLocation()
57      {
58          return location;
59      }
60      public void setLocation(String a)
61      {
62          location=a;
63      }
64      public String getTheatreName ()
65      {
66          return theatreName;
67      }
68      public void setTheatreName(String a)
69      {
70          theatreName=a;
71      }
72
73      public String getTheatreId()
74      {
75          return theatreId;
76      }
77      public void setTheatreId(String a)
78      {
79          theatreId=a;
80      }
81  }
82  public void setSecondaryDataSet(DataSet pD)
83  {
84      this.secondaryDataSet=pD;
85  }
86  public DataSet getSecondaryDataSet()
87  {
88      return this.secondaryDataSet;
89  }
90  public void setPrimaryDataSet(DataSet pD)
91  {
92      this.primaryDataSet=pD;
93  }
94  public DataSet getPrimaryDataSet()
95  {
96      return this.primaryDataSet;
97  }
98    public void setQueryId (String queryId)
```

```
99     {
100        this.queryId=queryId;
101     }
102       public void setQueryCategory(String queryCategory)
103       {
104          this.queryCategory=queryCategory;
105       }
106       public String getQueryId()
107       {
108          return this.queryId;
109       }
110       public String getQueryCategory()
111       {
112          return this.queryCategory;
113       }
114
115 }
```

TestApplication.java

```
1  import java.util.*;
2  public class TestApplication {
3     //Write the required business logic as expected in the question description
4     public static void main (String[] args) {
5        Scanner sc= new Scanner (System.in);
6        Query q= new Query();
7        Query.DataSet pd= q.new DataSet();
8        Query.DataSet sd= q.new DataSet();
9        System.out.println("Enter the Details for primary data set");
10        System.out.println("Enter the theatre id");
11        pd.setTheatreId(sc.nextLine());
12        System.out.println("Enter the theatre name");
13        pd.setTheatreName(sc.nextLine());
14        System.out.println("Enter the location");
15        pd.setLocation(sc.nextLine());
16        System.out.println("Enter the no of screens");
17        pd.setNoOfScreen(sc.nextInt());
18        System.out.println("Enter the ticket cost");
19        pd.setTicketCost(sc.nextDouble());
20        System.out.println("Enter the Details for secondary data set");
21        System.out.println("Enter the theatre id");
22
23        String id2=sc.next();
24        //System.out.println(id2);
25        sd.setTheatreId(id2);
26        System.out.println("Enter the theatre name");
27        sc.nextLine();
28        sd.setTheatreName(sc.nextLine());
29        System.out.println("Enter the location");
30        String gll=sc.nextLine();
31        sd.setLocation(gll);
32        System.out.println("Enter the no of screens");
33
34        //System.out.println(gll);
35        //String pp=sc.nextLine();
36        //System.out.println(pp);
37
38        sd.setNoOfScreen(sc.nextInt());
39        System.out.println("Enter the ticket cost");
40        sd.setTicketCost(sc.nextDouble());
41        sc.nextLine();
42        System.out.println("Enter the query id");
43        q.setQueryId(sc.nextLine());
44        System.out.println("Enter the query category");
45        q.setQueryCategory(sc.nextLine());
46
47        q.setSecondaryDataSet(sd);
```

```
48          q.setPrimaryDataSet(pd);
49          System.out.println(q.toString());
50      }
51  }
```

Grade

Reviewed on Friday, 17 December 2021, 6:59 PM by Automatic grade
Grade 100 / 100
Assessment report
[+]**Grading and Feedback**

Retrieve Flights Based on Source and Destination

Grade settings: Maximum grade: 100
Disable external file upload, paste and drop external content: Yes
Run: Yes **Evaluate**: Yes
Automatic grade: Yes **Maximum execution time**: 32 s

Retrieve Flights Based on Source and Destination

Zaro Flight System wants to automate the process in their organization. The flight details are available in the database, the customer should have the facility to view flights which are from a particular source to destination.

You being their software consultant have been approached by them to develop an application which can be used for managing their business. You need to implement a java program to view all the flight based on source and destination.

Component Specification: Flight (Model Class)

Type(Class)	Attributes	Methods	Responsibilities
Flight	int flightId String source String destination int noOfSeats	Include getters and setter method for all the attributes. Include a five argument constructor in the given order – flightId, source, destination, noOfSeats and flightFare.	

	double flightFare		

Note: The class and methods should be declared as public and all the attributes should be declared as private.

Requirement 1: Retrieve all the flights with the given source and destination

The customer should have the facility to view flights which are from a particular source to destination. Hence the system should fetch all the flight details for the given source and destination from the database. Those flight details should be added to a ArrayList and return the same.

Component Specification: FlightManagementSystem

Component Name	Type(Class)	Attributes	Methods	Responsibilities
Retrieve all the flights with the given source and destination	FlightManagement System		public ArrayList<Flight> viewFlightBySourceDestination(String source,String destination)	This method should accept a Source and a destination as parameter and retrieve all the flights with the given source and destination from the database. Return these details as ArrayList<Flight>.

Note: The class and methods should be declared as public and all the attributes should be declared as private.

The **flight** table is already created at the backend. The structure of flight table is:

Column Name	Datatype
flightId	int
source	varchar2(30)
destination	varchar2(30)
noofseats	int

flightfare	number(8,2)

Sample records available in **flight** table are:

Flightid	Source	Destination	Noofseats	Flightfare
18221	Malaysia	Singapore	50	5000
18222	Dubai	Kochi	25	50000
18223	Malaysia	Singapore	150	6000
18224	Malaysia	Singapore	100	7000

To connect to the database you are provided with **database.properties** file and **DB.java** file. **(Do not change any values in database.properties file)**

Create a class called **Main** with the main method and get the inputs like **source** and **destination** from the user**.**

Display the details of flight such as flightId, noofseats and flightfare for all the flights returned as ArrayList<Flight> from the method **viewFlightBySourceDestination** in **FlightManagementSystem** class.

If no flight is available in the list, the output should be "**No flights available for the given source and destination**".

Note:

In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user and the remaining text represents the output.

Ensure to follow object oriented specifications provided in the question description.
Ensure to provide the names for classes, attributes and methods as specified in the question description.

Adhere to the code template, if provided.

Sample Input / Output 1:

Enter the source

Malaysia

Enter the destination

Singapore

Flightid Noofseats Flightfare

18221 50 5000.0

18223 150 6000.0

18224 100 7000.0

Sample Input / Output 2:

Enter the source

Malaysia

Enter the destination

Dubai

No flights available for the given source and destination

Automatic evaluation[+]

Flight.java

```
1
2 public class Flight {
3
4       private int flightId;
5       private String source;
6       private String destination;
7       private int noOfSeats;
8       private double flightFare;
9       public int getFlightId() {
10                      return flightId;
11      }
12      public void setFlightId(int flightId) {
13                      this.flightId = flightId;
14      }
15      public String getSource() {
16                      return source;
17      }
18      public void setSource(String source) {
19                      this.source = source;
20      }
21      public String getDestination() {
22                      return destination;
23      }
24      public void setDestination(String destination) {
25                      this.destination = destination;
26      }
27      public int getNoOfSeats() {
28                      return noOfSeats;
29      }
30      public void setNoOfSeats(int noOfSeats) {
31                      this.noOfSeats = noOfSeats;
32      }
33      public double getFlightFare() {
34                      return flightFare;
35      }
36      public void setFlightFare(double flightFare) {
37                      this.flightFare = flightFare;
```

```
38          }
39          public Flight(int flightId, String source, String destination,
40                                      int noOfSeats, double flightFare) {
41                      super();
42                      this.flightId = flightId;
43                      this.source = source;
44                      this.destination = destination;
45                      this.noOfSeats = noOfSeats;
46                      this.flightFare = flightFare;
47          }
48
49
50
51 }
52
```

FlightManagementSystem.java

```
1  import java.sql.Connection;
2  import java.sql.ResultSet;
3  import java.sql.SQLException;
4  import java.util.ArrayList;
5  import java.util.List;
6  import java.sql.PreparedStatement;
7  public class FlightManagementSystem {
8      public ArrayList <Flight> viewFlightBySourceDestination(String source,String destination){
9          Connection conn = null;
10         ResultSet Rs = null;
11         String sql = "select * from flight where source=? and destination=? order by flightid";
12         ArrayList<Flight> flight = new ArrayList<>();
13         try{
14             conn = DB.getConnection();
15             PreparedStatement ps = conn.prepareStatement(sql);
16
17             ps.setString(1, source);
18             ps.setString(2, destination);
19
20             Rs=ps.executeQuery();
21         while(Rs.next()){
22             Flight F = new Flight(Rs.getInt(1),source,destination,Rs.getInt(4),Rs.getInt(5));
23             flight.add(F);
24         }
25         }catch(ClassNotFoundException e){
26             e.printStackTrace();
27         }catch(SQLException e){
28             e.printStackTrace();
29         }
30
31         return flight;
32     }
33 }
```

Main.java

```
1  import java.util.Comparator;
2  import java.util.Scanner;
3  import java.util.ArrayList;
4
5
6  public class Main{
7      public static void main(String[] args){
8          Scanner sc=new Scanner(System.in);
9          // fill your code here
10         System.out.println("Enter the source");
11         String source=sc.nextLine();
12         System.out.println("Enter the destination");
13         String destination=sc.nextLine();
```

```
        ArrayList<Flight> flight = new
FlightManagementSystem().viewFlightBySourceDestination(source,destination);
        if(flight.isEmpty())
        {
            System.out.println("No flights available for the given source and destination");

        }
        else
        {
            System.out.println("Flightid Noofseats Flightfare");
            for(Flight f : flight)
            {
                System.out.println(f.getFlightId()+" "+f.getNoOfSeats()+" "+f.getFlightFare());
            }
        }


    }
}
```

DB.java

```
import java.io.FileInputStream;
import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.util.Properties;

public class DB {

        private static Connection con = null;
        private static Properties props = new Properties();


    //ENSURE YOU DON'T CHANGE THE BELOW CODE WHEN YOU SUBMIT
        public static Connection getConnection() throws ClassNotFoundException, SQLException {
            try{

                                    FileInputStream fis = null;
                                    fis = new FileInputStream("database.properties");
                                    props.load(fis);

                                    // load the Driver Class
                                    Class.forName(props.getProperty("DB_DRIVER_CLASS"));

                                    // create the connection now
            con =
DriverManager.getConnection(props.getProperty("DB_URL"),props.getProperty("DB_USERNAME"),props.getPro
perty("DB_PASSWORD"));
            }
            catch(IOException e){
                e.printStackTrace();
            }
                    return con;
        }
}

```

database.properties

```
#IF NEEDED, YOU CAN MODIFY THIS PROPERTY FILE
#ENSURE YOU ARE NOT CHANGING THE NAME OF THE PROPERTY
#YOU CAN CHANGE THE VALUE OF THE PROPERTY
#LOAD THE DETAILS OF DRIVER CLASS, URL, USERNAME AND PASSWORD IN DB.java using this
properties file only.
#Do not hard code the values in DB.java.

```

```
 7 DB_DRIVER_CLASS=com.mysql.jdbc.Driver
 8 DB_URL=jdbc:mysql://localhost:3306/${sys:DB_USERNAME}
 9 DB_USERNAME=${sys:DB_USERNAME}
10 DB_PASSWORD=${sys:DB_USERNAME}
11
```

Grade

Reviewed on Wednesday, 12 May 2021, 6:31 AM by Automatic grade
Grade 100 / 100
Assessment report
[+]**Grading and Feedback**