AMRITA SCHOOL OF ENGINEERING

21AIE111
DATA STRUCTURES AND ALGORITHMS

BRANCH: Computer Science Engineering-AIE

SEMESTER-2

UNDER THE GUIDANCE OF: Mr. Ullas S.

Batch: E4

TEAM MEMBERS:

M Ramya Sree – BL.EN.U4AIE21072

Mettu Siddhartha- BL.EN.U4AIE21080

Poli Vamsi Vardhan Reddy – BL.EN.U4AIE21101

INDEX

INTRODUCTION

The Train Reservation and Management System handles train reservations for public interactions and services. The project's objective is to learn how to utilize a railway booking system and manage passengers' travel arrangements. This project's interface also offers a simple method to book and cancel reservations.This system is basically concerned with the reservation and cancellation of railway tickets to the passengers. The need of this system arose because as is the known fact that India has the largest railway network in the whole of the world and to handle it manually is quite a tough job. By computerizing it, we will be able to overcome many of its limitations and will be able to make it more efficient.

To be more specific, our system is limited in such a way that a train starting from a particular source will have a single destination.

Description

Linked Lists:

A linear data structure used to store the elements in contiguous locations is called a Linked List in Java. It has addresses and pointers that are used to link the elements, and each element in the linked list consists of two parts, namely the data part and the address part. The data part is the value of the element, and the address part consists of the pointers and addresses that are used to link the elements. Each element in the list is called a node.

The syntax to define a Linked list in Java is as follows:

LinkedList <data_type> linkedlistname = new LinkedList<data_type>();

where data_type is the data type of the elements to be stored in the linked list,

linkedlistname is the name of the linked list.

The usage of a linked list allows dynamic insertion and deletion of elements into the linked list. Because of this feature, linked lists are preferred over arrays.

Working Of Linked List in Java Is as Follows:

- A linked list in Java is a dynamic data structure whose size increases as you add the elements and decreases as you remove the elements from the list.
- The elements in the linked list are stored in containers.
- The list holds the link to the first container.
- All the containers have a link to the next container in the list.
- Whenever you add an element to the list using add() operation, a new container is created, and this container is linked to the other containers in the list.

Types of linked lists:

- Singular Linked List
- Doubly Linked List
- Circular Linked List

Singly linked list:

- The type of linked list consisting of a sequence of nodes where each node consists of data and a link to the next node, that can be traversed from the first node of the list (also called as head) to the last node of the list (also called as Tail) and is unidirectional is called Singly Linked list.
- The above figure demonstrates a singly linked list.
- Each element in the list is called a node.
- A node is made of two parts, namely data and pointer.
- Data is the data stored in the   and the pointer is the next node in the list.
- The first node in the list is referred to as the head of the list.
- The last node in the list is the tail, and it points to NULL.

The syntax to define a node in a singular linked list is as follows:

```
public class SinglyLinkedList

{

    class Node

{

        int data;

        Node next;
```

```
                    public Node(int data)
                {
                            this.data = data;
                            this.next = null;
                    }
                  }
                }
```

The basic functions being performed by our system are:

1. RESERVATION
2. CHECKING
3. CANCELATION

These functions will be handled with the help of following sub functions: -

- It reserves and cancels seats for the passenger.
- It contains information about the trains.
- It contains information about the passenger.
- It contains the details of reservation destinations, any concessions etc.

Doubly linked list:

- This type of a linked list consists of a sequence of nodes where each node consists of data and two pointers, namely the previous pointer pointing to the previous node and the subsequent pointer that points to the next node that is part of the list. This can be traversed from the first node of the list to the last node of the list and vice versa, and this is called Doubly Linked list.

- The above figure demonstrates a doubly linked list.
- Data is the data stored in the node and each node consists of two pointers namely the previous pointer and the next pointer.
- The previous pointer points, as the name suggests to the previous node that is part of the list.
- The pointer after the current one, points to the next node on the list.

The syntax to define a node in a doubly linked list is as follows:

```
public class DoublyLinkedList

{

   class Node

   {

      int data;

      Node previous;

      Node next;

      public Node(int data)

      {
            this.data = data;

        }

    }

}
```

Circular linked list:

It is the type of linked list consisting of a sequence of nodes where each node consists of data and a link to the next node and the last node in the list (also called as tail) that points to the first node in the list (also called as head) is called as Circular Linked List.

The syntax to define a node in a circular linked list is as follows:

```
public class CircularLinkedList

{

    public class Node

    {

        int data;

        Node next;

        public Node(int data)

        {

            this.data = data;

        }
    }

}
```

Services

- Inquiring about the availability of the selected destination location.
- Booking tickets for the trip.
- Choosing the passengers' preferred coach.
- Cancelation of pre-booked reservations
- Complaints about any difficulties faced when purchasing or cancelling tickets.

Booking a ticket

- Firstly, choose the book a reservation option from the given choices.
- Choose the destination place.
- After Choosing the destination place, a train id will be given which helps for further process.
- Then enter the number of passengers and their names by entering the train id.
- Finally choose the seating class.
- Following the conclusion of the booking, you will be notified that "Your Ticket Was Confirmed."

Canceling a reservation

- To begin, select the cancel a reservation option from the available options.
- Enter the train id that the client was given during the booking procedure.
- Then type in the names of the passengers whose reservations to be cancelled.
- A message indicating the passenger's reservation has been cancelled will be displayed.

Characteristics of the proposed system

- Searching of the trains is so easy:

It is simple to find the desired train since the train code and train number are available; you do not need to memorise them; you only need to pick the destination place.

- Reduce the possibility to make mistake:
  Due to excessive amount of work the employers tend to do mistakes by manual form. Here the chance of mistake is minimum.
- Reservation can be done very easily:
  The entire procedure is simple and consists of only a few steps. To execute the process, no extensive expertise is required.

Code

```
import java.util.Scanner;
public class train
{
  Node head;

  static Scanner sc = new Scanner(System.in);

  static train l_list1 = new train();
  static train l_list2 = new train();
  static train l_list3 = new train();
  static train l_list4 = new train();

   static class Node
   {
      String data;
      Node next;
      Node(String d)
      {
         data = d;
         next = null;
      }
   }
   static void type ()
   {
         System.out.println("Choose the coach:");
         System.out.println("a.Sleeper");
      System.out.println("b.General");
      char choice1 = sc.next().charAt(0);
              switch(choice1) {
                case 'a' :
                  System.out.println("sleeper coach is booked");

                  break;
                case 'b':
                   System.out.println("general coach is booked");
                           break;
                default :
                           System.out.println("Invalid coach");
                           break;
              }

   }


   static void bookingtrain()
   {
```

```
            System.out.println("Choose your destination place");
            System.out.println("a.Banglore");
        System.out.println("b.Mysore");
        System.out.println("c.Mumbai");
        System.out.println("d.Pune");
        System.out.println("e.Chennai");
        System.out.println("f.Pondicherry");
        System.out.println("g.Kochi");
        System.out.println("h.Delhi");
        System.out.println("i.Agra");


                char choice = sc.next().charAt(0);
                switch(choice) {
                  case 'a' :
                    System.out.println("your selected destination is BANGLORE"+'\n'+"Train
id:T1");
                    break;
                  case 'b':
                     System.out.println("your selected destination is MYSORE"+'\n'+"Train
id:T1");
                                break;
                  case 'c' :
                    System.out.println("your selected destination is MUMBAI"+'\n'+"Train
id:T2");
                    break;
                  case 'd' :
                    System.out.println("your selected destination is PUNE"+'\n'+"Train id:T2");
                    break;
                  case 'e' :
                    System.out.println("your selected destination is CHENNAI"+'\n'+"Train
id:T3");
                    break;
                  case 'f' :
                                System.out.println("your selected destination is
PONDHICHERRY"+'\n'+"Train id:T3");
                                break;
                  case 'g' :
                                System.out.println("your selected destination is Tirupati"+'\n'+"Train
id:T3");
                                break;
                  case 'h' :
                                System.out.println("your selected destination is DELHI"+'\n'+"Train
id:T4");
                                break;
                  case 'i' :
                                System.out.println("your selected destination is AGRA"+'\n'+"Train
id:T4");
                                break;
                  default :
```

```
                System.out.println("Invalid destination");
                break;
           }
        System.out.println("Enter the Train id provided:");
        String q = sc.next();
        if (q.equals("T1")||q.equals("t1"))
        {
           System.out.println("Enter the number of passengers you want to book for Train T1");
           int w = sc.nextInt();

           for (int i = 0; i < w; i++)
           {
              System.out.println("Enter the  name of passenger: " + (i + 1));

              String n = sc.next();
              Node new_node = new Node(n);
              new_node.next = null;

              if (l_list1.head == null)
              {
                 l_list1.head = new_node;
              }
              else
              {

                 Node last = l_list1.head;
                 while (last.next != null)
                 {
                    last = last.next;
                 }
                 last.next = new_node;
              }
           }
        }
        else if (q.equals("T2")||q.equals("t2"))
        {
           System.out.println("Enter the number of passengers you want to book for Train T2");
           int w = sc.nextInt();
           for (int i = 0; i < w; i++)
           {
              System.out.println("Enter the name of passenger: " + (i + 1));

              String m = sc.next();
              Node new_node = new Node(m);
              new_node.next = null;

              if (l_list2.head == null)
              {
                 l_list2.head = new_node;
              }
```

```
            else
            {
                Node last = l_list2.head;
                while (last.next != null)
                {
                    last = last.next;
                }
                last.next = new_node;
            }
        }
    }
    else if (q.equals("T3")||q.equals("t3"))
    {
        System.out.println("Enter the number of passengers you want to book for Train T3");
        int w = sc.nextInt();
        for (int i = 0; i < w; i++)
        {
            System.out.println("Enter the name of passenger: " + (i + 1));

            String m = sc.next();
            Node new_node = new Node(m);
            new_node.next = null;

            if (l_list3.head == null)
            {
                l_list3.head = new_node;
            }
            else
             {

                Node last = l_list3.head;
                while (last.next != null)
                {
                    last = last.next;
                }


                last.next = new_node;
            }
        }
    }

    else if (q.equals("T4")||q.equals("t4"))
    {
        System.out.println("Enter the number of passengers you want to book for Train t4");
        int w = sc.nextInt();
        for (int i = 0; i < w; i++)
        {
            System.out.println("Enter the name of passenger: " + (i + 1));
            String m = sc.next();
            Node new_node = new Node(m);
            new_node.next = null;

            if (l_list4.head == null)
            {
               l_list4.head = new_node;
            }
            else
            {
               Node last = l_list4.head;
               while (last.next != null)
               {
                  last = last.next;
               }
               last.next = new_node;
            }
         }
      }
      else{
         System.out.println("The entered Train number is not available");
      }

   }


   static void printNames()
   {
      System.out.println("Enter the Train number for which you want to print the Passenger
list: ");
      String e = sc.next();


      if (e.equals("T1") || e.equals("t1"))
      {
         Node currNode = l_list1.head;
         if (l_list1.head == null)
         {
            System.out.println("No Passengers");
         }
         else
         {
            System.out.print("Names of Passengers in T1: ");


            while (currNode != null)
            {

               System.out.print(currNode.data + " ");
            currNode = currNode.next;
        }
    }
}


else if (e.equals("T2")||e.equals("t2"))
{
    Node currNode = l_list2.head;
    if (l_list2.head == null)
    {
        System.out.println("No Passengers");
    }
    else
    {
        System.out.print("Names of passengers in T2: ");
        while (currNode != null)
        {

            System.out.print(currNode.data + " ");

            currNode = currNode.next;
        }
    }
}
else if (e.equals("t3")||e.equals("T3"))
{
    Node currNode = l_list3.head;
    if (l_list3.head == null)
    {
        System.out.println("No Passengers");
    }
    else
    {
        System.out.print("Names of passengers in T3: ");


        while (currNode != null)
        {
            System.out.print(currNode.data + " ");

            currNode = currNode.next;
        }
    }

}

else if (e.equals("t4")||e.equals("T4"))
{
```

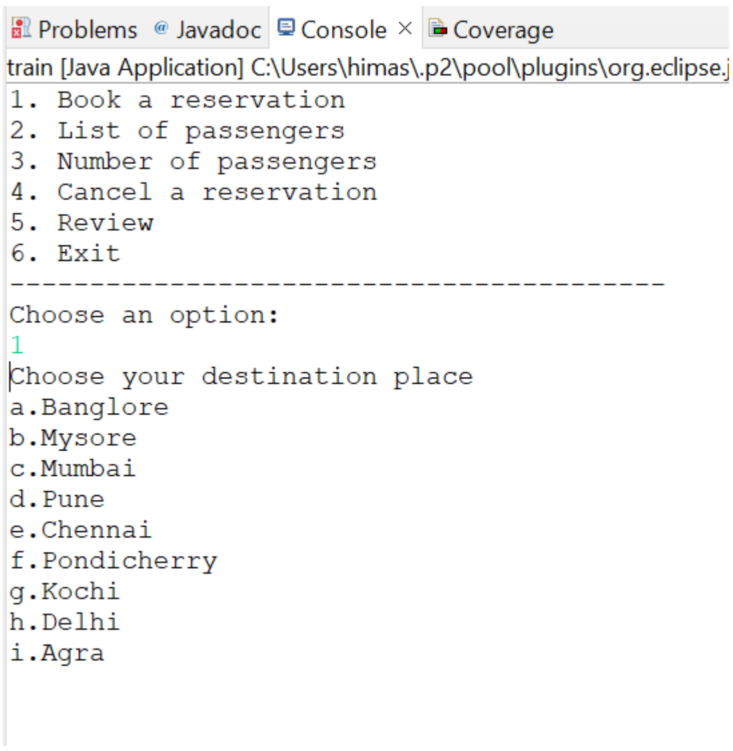
```
            Node currNode = l_list4.head;
            if (l_list4.head == null)
            {
                System.out.println("No Passengers");
            }
            else
            {
                System.out.print("Names of passengers in T4: ");


                while (currNode != null)
                {
                    System.out.print(currNode.data + " ");

                    currNode = currNode.next;
                }
            }

        }
        else
        {
            System.out.println("The entered Train number is not available");
        }
    }
    static void printSize()
    {
        System.out.println("Enter the flight number for which you want to print the number of
passengers: ");
        String e = sc.next();
        if (e.equals("T1")||e.equals("t1"))
        {
            Node currNode = l_list1.head;
            if (l_list1.head == null)
            {
                System.out.println("0");
            }
            else
            {
                int count = 0;

                while (currNode != null)
                {
                    count++;
                    currNode = currNode.next;
                }
                System.out.println("Number of passengers: "+count);
            }

        }
        else if (e.equals("T2")||e.equals("t2"))
```

```
{
   Node currNode = l_list2.head;
   if (l_list2.head == null)
   {
      System.out.println("0");
   } else
   {
      int count = 0;


      while (currNode != null)
      {
         count++;
         currNode = currNode.next;
      }
      System.out.println("Number of passengers: " + count);
   }
}

else if (e.equals("t3")||e.equals("T3"))
{
   Node currNode = l_list3.head;
   if (l_list3.head == null)
   {
      System.out.println("0");
   } else {
      int count = 0;

      while (currNode != null) {
         count++;
         currNode = currNode.next;
      }
      System.out.println("Number of passengers: " + count);
   }
}

else if (e.equals("t4")||e.equals("T4")) {
   Node currNode = l_list4.head;
   if (l_list4.head == null)
   {
      System.out.println("0");
   } else
   {
      int count = 0;


      while (currNode != null)
      {
         count++;
         currNode = currNode.next;
```

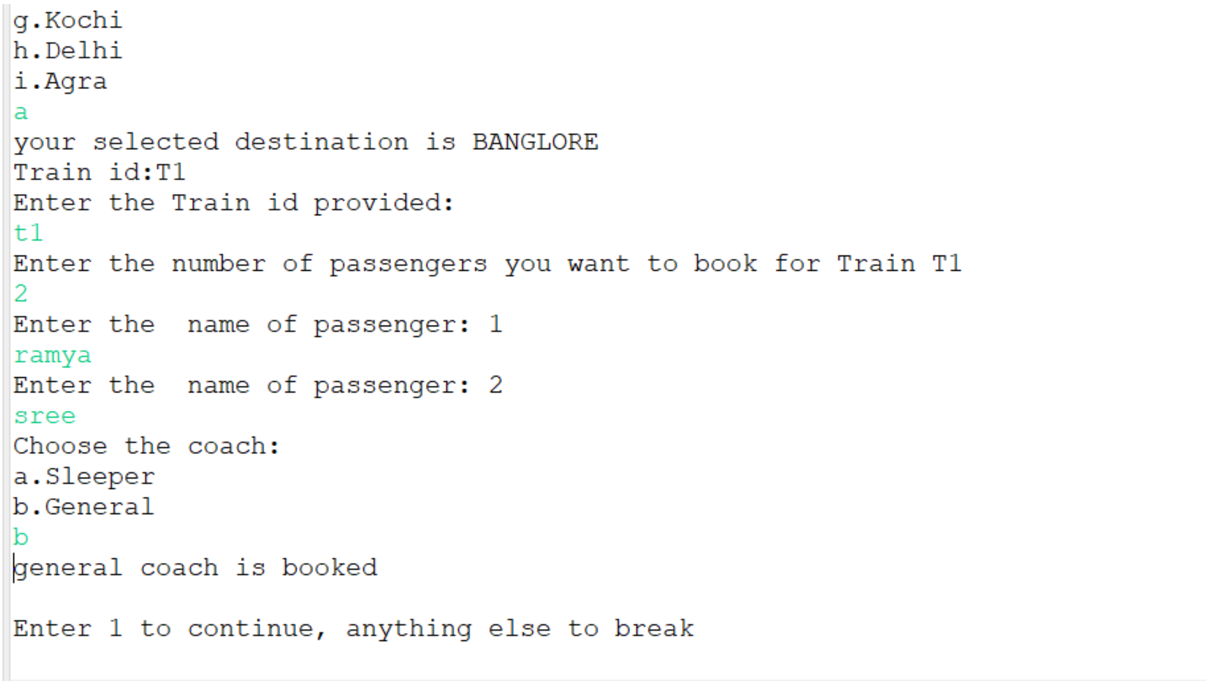
```
            }
            System.out.println("Number of passengers: " + count);
        }
    }
    else
    {
        System.out.println("The entered Train number is not available");
    }
}

static void deletingReservation()
{
        if (l_list1.head == null)
    {
        System.out.println("No reservations are booked in this train to cancel");
    }
        else
        {
 try
  {
    System.out.println("Enter the name of passenger");
    Scanner sc = new Scanner(System.in);
    String name = sc.nextLine();
    Node point = l_list1.head;
    Node rev = null;


    if(point.data.equals(name))
    {
        l_list1.head = point.next;
    }
    else
    {
        while (!(point.data.equals(name)))
        {
            rev = point;
            point = point.next;
        }
        rev.next = point.next;
    }


   }
    catch (NullPointerException e)
    {
        return;
    }
        }

}
```

```
static void deletingReservation1()
{
        if (l_list1.head == null)
    {
        System.out.println("No reservations are booked in this train to cancel");
    }
        else
        {
                try
                {
                System.out.println("Enter the name of passenger");
                Scanner sc = new Scanner(System.in);
                String name = sc.nextLine();
                Node point = l_list2.head;
                Node rev = null;
                if(point.data.equals(name))
                {
                    l_list2.head = point.next;
                }
                else
                {
                    while (!(point.data.equals(name)))
                    {
                        rev = point;
                        point = point.next;
                    }
                    rev.next = point.next;
                }
            }
                catch (NullPointerException e)
                {
                    return;
                }
        }
}
static void deletingReservation2()
{
        if (l_list1.head == null)
    {
        System.out.println("No reservations are booked in this train to cancel");
    }
        else
        {
                try
                {
                            System.out.println("Enter the name of passenger");
                            Scanner sc = new Scanner(System.in);
                            String name = sc.nextLine();
                            Node point = l_list3.head;
                            Node rev = null;
                        if(point.data.equals(name))
                                {
                                    l_list3.head = point.next;
                                }
                        else
                        {
                            while (!(point.data.equals(name)))
                            {
                                rev = point;
                                point = point.next;
                            }
                            rev.next = point.next;
                        }
                }
                catch (NullPointerException e)
                {
                    return;
                }

            }

    }
    static void deletingReservation3()
    {
            if (l_list1.head == null)
        {
            System.out.println("No reservations are booked in this train to cancel");
        }
            else
            {
                        try
                         {
                            System.out.println("Enter the name of passenger");
                            Scanner sc = new Scanner(System.in);
                            String name = sc.nextLine();
                            Node point = l_list4.head;
                            Node rev = null;
                            if(point.data.equals(name))
                            {
                                l_list4.head = point.next;
                            }
                            else
                            {
                                while (!(point.data.equals(name)))
                                {
                                    rev = point;
                                    point = point.next;
                                }
                                rev.next = point.next;
                            }
                    }
                       catch (NullPointerException e)
                       {
                          return;
            }
         }
   }



      public static void main (String[]args)
      {
        while (true)
        {
          System.out.println("1. Book a reservation");
          System.out.println("2. List of passengers");
          System.out.println("3. Number of passengers");
          System.out.println("4. Cancel a reservation");
          System.out.println("5. Review");
          System.out.println("6. Exit");
          System.out.println("----------------------------------------");

          System.out.println("Choose an option: ");
          int u = sc.nextInt();

          if (u == 1)
          {
             bookingtrain();
             type();

          }

          else if (u==2)
          {
             printNames();
          }
          else if (u==3)
          {
             printSize();
          }
          else if (u==4)
          {
                   System.out.println("Enter u r train number");
                   String t=sc.next();
                   if (t.equals("T1")||t.equals("t1"))
                   {
               deletingReservation();
                   }
                   else if(t.equals("T2")||t.equals("t2"))
                   {
                        deletingReservation1();
                }
                else if(t.equals("T3")||t.equals("t3"))
                {
                        deletingReservation2();
                }
                else if(t.equals("T4")||t.equals("t4"))
                {
                        deletingReservation3();
                }
                else
                        System.out.println("Invalid Train number");


            }
            else if (u==5)
            {
                Scanner sc = new Scanner(System.in);
              System.out.println("Enter your review");
              sc.nextLine();
              System.out.println("Thanks for your valuable review");

            }

            else if (u==6)
            {
              System.out.println("Terminated");

            }

            else
            {
              System.out.println("Wrong Input");
              break;
            }
            System.out.println();
            System.out.println("Enter 1 to continue, anything else to break");
            int i = sc.nextInt();
            if (i != 1)
            {
              System.out.println("Terminated");
              break;

            }
          }
        }
    }
```

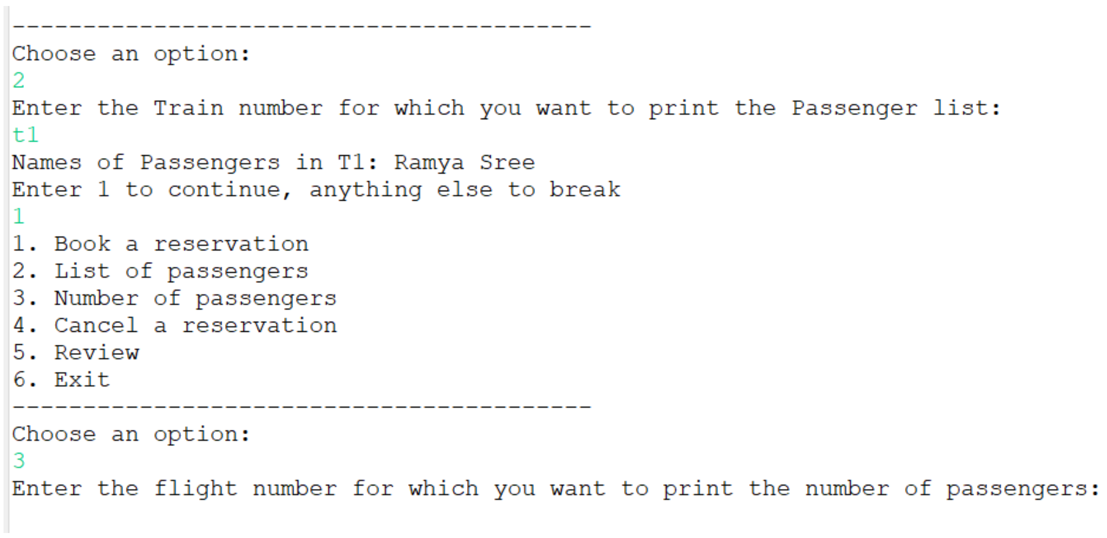
Output Screenshots



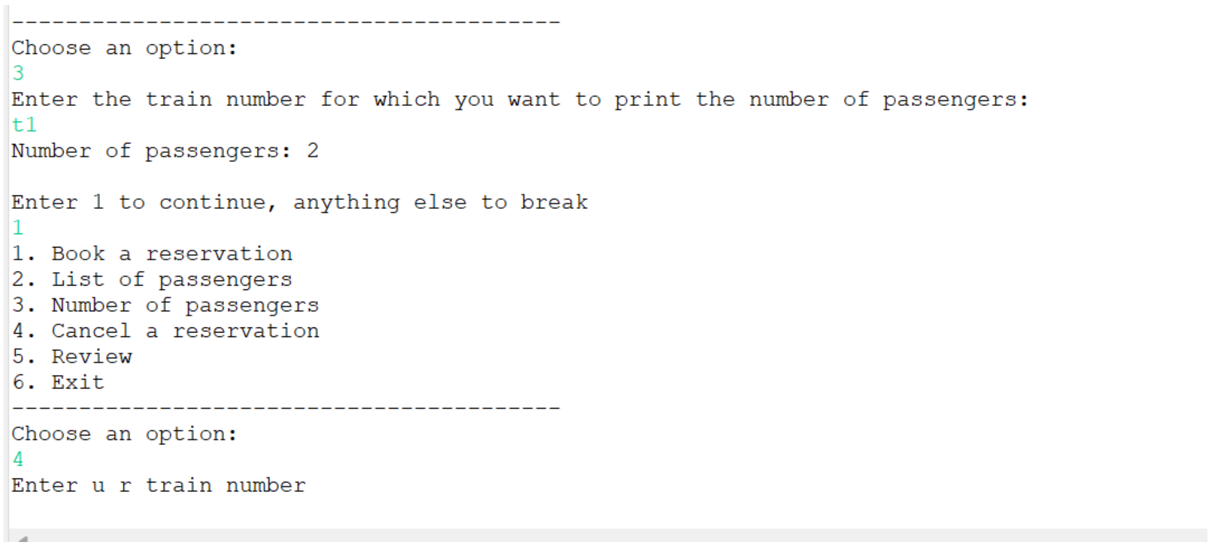
```
train [Java Application] C:\Users\himas\.p2\pool\plugins\org.eclipse.j
1. Book a reservation
2. List of passengers
3. Number of passengers
4. Cancel a reservation
5. Review
6. Exit
-----------------------------------------
Choose an option:
1
Choose your destination place
a.Banglore
b.Mysore
c.Mumbai
d.Pune
e.Chennai
f.Pondicherry
g.Kochi
h.Delhi
i.Agra
```



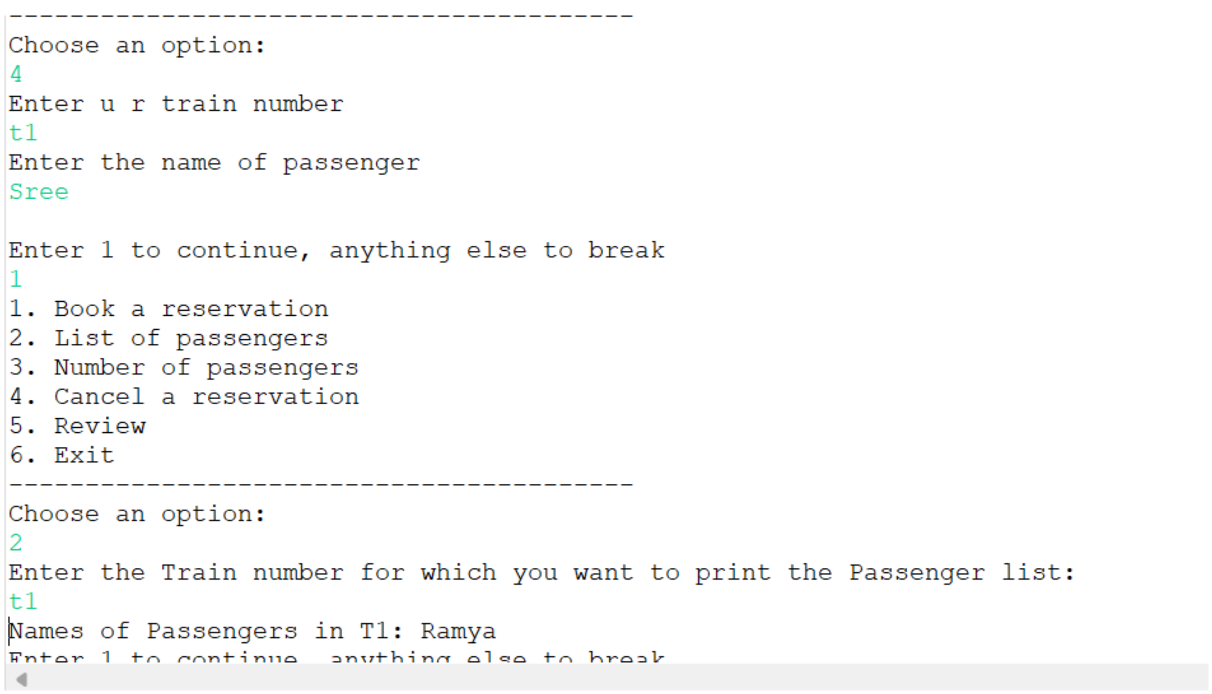
```
g.Kochi
h.Delhi
i.Agra
a
your selected destination is BANGLORE
Train id:T1
Enter the Train id provided:
t1
Enter the number of passengers you want to book for Train T1
2
Enter the  name of passenger: 1
ramya
Enter the  name of passenger: 2
sree
Choose the coach:
a.Sleeper
b.General
b
general coach is booked

Enter 1 to continue, anything else to break
```



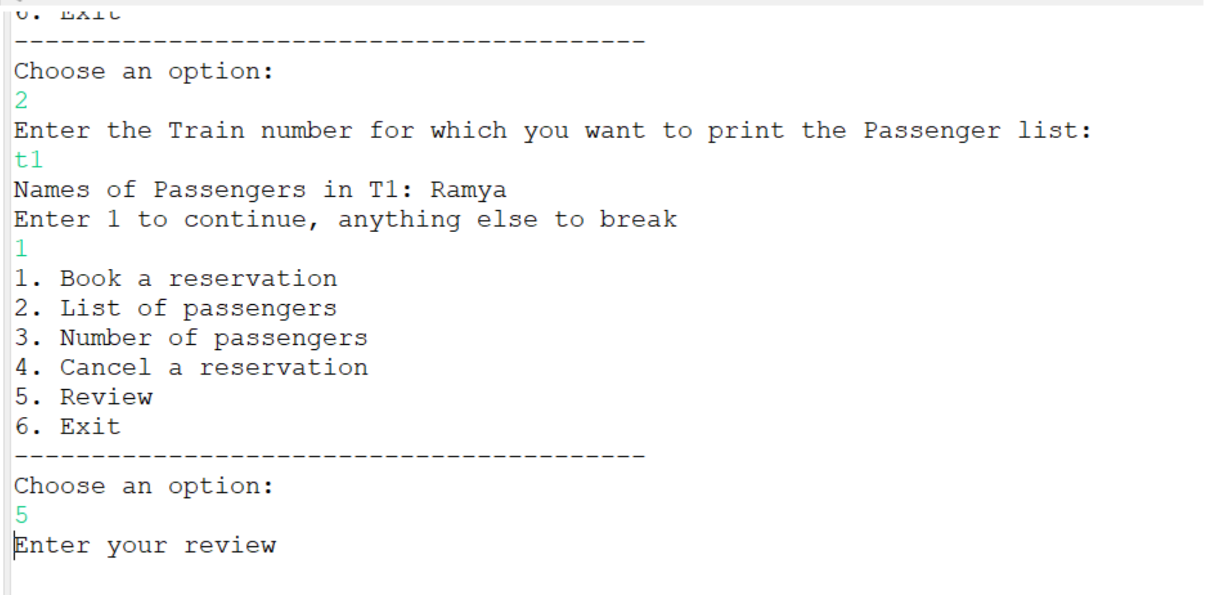
```
-----------------------------------------
Choose an option:
2
Enter the Train number for which you want to print the Passenger list:
t1
Names of Passengers in T1: Ramya Sree
Enter 1 to continue, anything else to break
1
1. Book a reservation
2. List of passengers
3. Number of passengers
4. Cancel a reservation
5. Review
6. Exit
-----------------------------------------
Choose an option:
3
Enter the flight number for which you want to print the number of passengers:
```



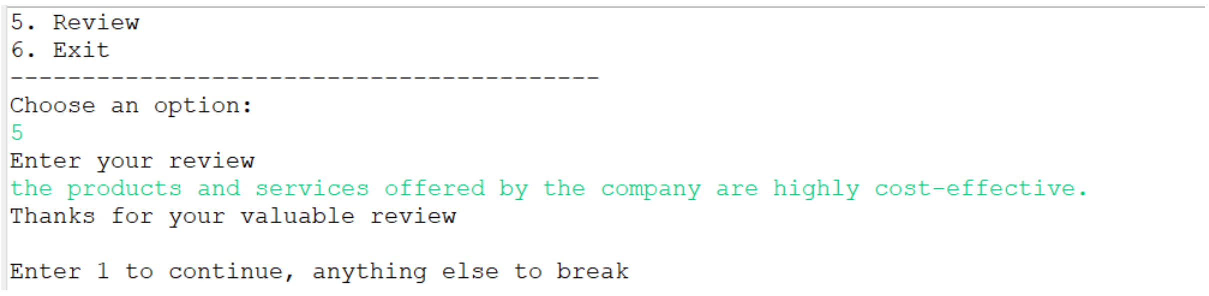
```
-----------------------------------------
Choose an option:
3
Enter the train number for which you want to print the number of passengers:
t1
Number of passengers: 2

Enter 1 to continue, anything else to break
1
1. Book a reservation
2. List of passengers
3. Number of passengers
4. Cancel a reservation
5. Review
6. Exit
-----------------------------------------
Choose an option:
4
Enter u r train number
```



```
-----------------------------------------
Choose an option:
4
Enter u r train number
t1
Enter the name of passenger
Sree

Enter 1 to continue, anything else to break
1
1. Book a reservation
2. List of passengers
3. Number of passengers
4. Cancel a reservation
5. Review
6. Exit
-----------------------------------------
Choose an option:
2
Enter the Train number for which you want to print the Passenger list:
t1
Names of Passengers in T1: Ramya
Enter 1 to continue, anything else to break
```



```
6. Exit
-----------------------------------------
Choose an option:
2
Enter the Train number for which you want to print the Passenger list:
t1
Names of Passengers in T1: Ramya
Enter 1 to continue, anything else to break
1
1. Book a reservation
2. List of passengers
3. Number of passengers
4. Cancel a reservation
5. Review
6. Exit
-----------------------------------------
Choose an option:
5
Enter your review
```



```
5. Review
6. Exit
-----------------------------------------
Choose an option:
5
Enter your review
the products and services offered by the company are highly cost-effective.
Thanks for your valuable review

Enter 1 to continue, anything else to break
```

Thank you