CSD201 - Assignment 1

Train Booking System using Linked List Data Structure

INTRODUCTION

Your first assignment in this block will be using linked list data structure for implementing a Train Booking System (TBS) in Java language. TBS manages information about trains, customers, and train booking. This information includes:

About a train:

- **tcode** (string): the code of the train (this should be unique for the train).
- **train_name** (string): the name of the train.
- **seat** (integer): the number of seats in the train (seat > 0).
- **booked** (integer): the number of booked seats in the train (booked >= 0 and booked ≤ seat).
- **depart_time** (double): The departure time of the train (depart_time >= 0).
- **depart_place** (string): The place of departure of the train.

About a customer:

- **ccode** (string): the code of the customer (this should be unique for the customer).
- **cus_name** (string): the name of the customer.
- **phone** (string): The phone number of the customer (must contain digits only).

About Booking:

- **tcode** (string): the code of the train to be booked.
- **ccode** (string): the code of the customer.
- **seat** (integer): the number of seats to be booked on the train.

YOUR TASKS

You should use 3 linked lists, each one is used to store data for trains, customers, or train booking items. You should create linked lists from scratch, do not use list structures available in Java like ArrayList, Vector, or LinkedList classes. On running, your program displays the menu as below:

Train list (8 marks):

1. Load data from file
2. Input & add to the end
3. Display data
4. Save train list to file
5. Search by tcode
6. Delete by tcode
7. Sort by tcode
8. Add after position k

9. Delete the node before the node having tcode = xCode

Customer list (1 mark):

1. Load data from file
2. Input & add to the end
3. Display data
4. Save customer list to file
5. Search by ccode
6. Delete by ccode

Booking list (1 mark):

1. Input data
2. Display data with available seats
3. Sort by tcode + ccode

TASKS EXPLANATION

Train list (8 marks):

1. **Load data from file**
   Allow a user to input the file name that contains information of trains. The content of the text file may be:

```
B03 | Sug | 12 | 3 | 11   | PA
B01 | Mil | 10 | 5 |  5.7 | PC
B02 | App |  5 | 2 |  4   | PB
B05 | Roo |  7 | 6 | 15   | PE
B07 | Bee | 11 | 3 | 12   | PF
B04 | Boo |  9 | 5 |  5   | PD
```

   The first line means that: tcode = B03, train_name = Sug, seat = 12, booked = 3, depart_time = 11 and depart_place = PA. The task of this option is to read rows from the text file and add to the end of the list. If the list is not empty, then the program asks the user if he/she wants to keep the existing data or not.

2. **Input & add to the head**
   Allow a user to add new information about a train. After checking validation of data (including that the tcode could not be duplicated), the train is added to the head of the list.

3. **Display data**
   Display data in format:

```
tcode  |  train_name  |  seat  |  booked  |  depart_time  |  depart_place  |
available_seat
```

where available_seat = seat - booked. For example, after loading the above file, this option gives the output below:

```
tcode|Train_name|Seat|booked|depart_time|depart_place|available_seat
-----------------------------------------------------------------
B03 |  Sug   | 12 |  3     | 11        |     PA     |     9
B01 |  Mil   | 10 |  5     |  5.7      |     PC     |     5
B02 |  App   |  5 |  2     |  4        |     PB     |     3
B05 |  Roo   |  7 |  6     | 15        |     PE     |     1
B07 |  Bee   | 11 |  3     | 12        |     PF     |     8
B04 |  Boo   |  9 |  5     |  5        |     PD     |     4
```

4. **Save train list to file**
   Allow a user to input the file name and save the train list to the file. The information and format like the option 1.3.

5. **Search by tcode**
   Write the function:

   ```
   Node search(String xCode) {}
   ```

   which returns reference to the node whose info contains the train with tcode = xCode. Allow a user to input the tcode to be searched and display the result: found or not found.

6. **Delete by tcode**
   Write the function:

   ```
   void dele(String xCode) {}
   ```

   which deletes the node whose info contains the train with tcode = xCode. Allow a user to input the tcode to be deleted and then delete the train having that tcode.

7. **Sort by tcode**

8. **Add after position k**
   The position of the first element is 0, the second's is 1. Allow a user to input data for a train and add the train after position k.

9. **Delete the node before the node having tcode = xCode**
   Allow a user to input xCode and delete the node before the node having tcode = xCode.

Customer list (1 mark):

1. **Load data from file**
   Allow a user to input the file name that contains information of customers. The content of the file may

be:

```
C03 | Hoa  | 1902
C01 | La   | 1901
C02 | Canh | 1903
C05 | Cay  | 1910
```

The first line means that: ccode = C03, name = Hoa, phone = 1902

2. **Input & add to the end**
3. **Display data**
4. **Save customer list to file**
5. **Search by ccode**
6. **Delete by ccode**

Booking list (1 mark):

1. **Input data**
   Allow a user to input booking item. When running, the screen looks like:

```
Enter train code:
Enter customer code:
Enter number of seats to be booked:
```

After the user enters tcode and ccode, the program checks and acts as follows:

- If tcode not found in the Train list or ccode not found in the customer list then data is not accepted.
- If both tcode and ccode are found in the booking list then data is not accepted.
- If tcode and ccode found in Trains and customers lists but booked = seat then inform the user that the train is exhausted.
- If tcode or ccode found and in the Train list booked < seat and k is the entered seat then if k <= seat - booked then data is accepted and added to the end of the Booking list.

2. **Display booking data**
3. **Sort by tcode + ccode**

Submission Requirements

Zip your project with name format ID_NAME_ASM01.zip Ex: SE189009_HuynhNgocDung_ASM01.zip Note: You need to compress file in .zip format. Do not change RAR file to ZIP

Assignment assessment

You will be asked to modify immediately and to explain your assignment in the classroom to be sure that you are really the author of the assignment you submitted.