DSA - Assignment 1 (Deadline: the first slot of 6th week)
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 small Train Booking System (TBS) in Java language.

TBS manages information about trains, passengers and train booking items. These information are:

About a train:

1. tcode (string): the code of the train (this should be unique for the train).
2. name (string): the name of the train.
3. dstation (string): the departing station of the train
4. astation (string): the arriving station of the train
5. dtime (double): the departing time of the train ($0 \le dtime \le 24$)
6. seat (integer): the number of seats in the train ($seat > 0$).
7. booked (integer): the number of seats which are booked ($0 \le booked \le seat$).
8. atime (double): the arriving time of the train ($dtime \le atime \le 24$).

About a passenger:

1. pcode (string): the code of the passenger (this should be unique for the customer).
2. name (string): the name of the passenger.
3. phone (string): The phone number of the passenger (must be unique and contains digits only).

About booking:

1. bcode (string): the code of the bus to be booked.
2. pcode (string): the code of the passenger
3. odate (date): the date when customer book the bus
4. paid (date): the state of payment: 0 – is not paid; 1 – is paid;
5. seat (integer): the number of seats to be booked (must be greater than 0).

YOUR TASKS: You should use 3 linked lists, each one is used to store data for trains, passengers or 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:

Bus list (4 marks):

1.1. Load data from file
(load train list from trains.txt)

1.2. Input & add to the end
(input and validate train data, then add the train to the end of the list)

1.3. Display data
(display info of all trains in the train list)

1.4. Save bus list to file
(save the train list to file trains.txt)

1.5. Search by tcode
(input tcode to be searched, then return the found train data or not found)

1.6. Delete by tcode
(input tcode, then delete found train;
Remember to delete all related records in booking list first)

1.7. Sort by tcode
(dislay trains in ascending order of the tcode)

1.8. Input & add to beginning
(input and validate train data, then add the train to the begin of the list)

1.9. Add before position k
(input and validate train data, then add the train to the position k-1 of the list)
1.10. Delete position k
(delete the train to the position k of the list)
1.11. Search by name
(input name to be searced, then return the found train data or not found)
1.12. Search booked by tcode
(input tcode to be searced, then return the train data or not found;
Then list all passengers who booked the train)

Passenger list (1.5 mark):
2.1. Load data from file
(load passenger list from passengers.txt)
2.2. Input & add to the end
(input and validate passenger data, then add the passenger to the end of the list)
2.3. Display data
(display info of all passengers in the passenger list)
2.4. Save passenger list to file
(save the passenger list to file passengers.txt)
2.5. Search by pcode
(input pcode to be searched, then return the found passenger data or not found)
2.6. Delete by pcode
(input pcode, then delete found passenger;
Remember to delete all related records in booking list first)
2.7. Search by name
(input name to be searced, then return the found passengers data or not found)
2.8. Search trains by pcode
(input pcode to be searced, then return the found passenger data or not found;
Then list all trains booked by passenger)

Booking list (1.5 mark):
3.1. Load data from file
(load booking list from bookings.txt)
3.2. Book bus
(input tcode, pcode; check if train and passenger exsist; check if booking seat is less than or equals to seat of found train; odate to today, and paid to 0; add booking to the end of the booking list; substract booking seat from train seat; add booking seat to train booked;)
3.3. Display data
(display info of all bookings in the booking list)
3.4. Save booking list to file
(save the booking list to file bookings.txt)
3.5. Sort by tcode + pcode
(display orders in the descending of tcode first, then in the descending of the pcode)
3.6. Pay booking by tcode + pcode
(input tcode and pcode to be searched booking, if booking is found and is not paid, then set paid to to 1;)

Directly modify coding and answer questions at classroom (3 marks)

Submission Requirements

Create the directory with a name like **<class>_<roll number>_<name>_ASS1**,

e.g. **SE0508_HE180045_QuangTV_ASS1 (1)**

The (1) directory contains the following files:

1. The run.bat file to run your program.
2. Your source code files.
3. Your input test files.

The statements in run.bat file may be:

```
cls
javac Main.java
java Main
pause
del *.class
```

Compress the folder (1) to .zip file (with the same name) and upload to Assignment1 in edunext.

Assignment assessment

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