CS 305: HW3 Flight tracker
Fall 2018

During the semester free tutoring is available to you for CS 305, please see tutors if you need to. If you have hard time in the class, e-mail the tutor fellow Alex Schendel: schendel21@up.edu for an appointment.

Due Date: NLT Sunday, October 28th 2018 at Midnight. Your code files (.h and .c files), makefile, 10 sample inputs that represent different flight lists you used for testing your code. In your archive include answer questions in a pdf or docx file. Submit your HW as a single zip file to Moodle.

Your task is to create a sorted list (by flight number) of flights that are either read from a file (flights.txt for example) or read into the program from the command line.

The command line instructions include
./flight

```
a <flight information>   # add a flight, no duplicate flight numbers are allowed
d <flight number>        # for deleting a flight with the given <flight number:int>
s <file name>            # saves the current content of the LList into a file
p                        # print the list of flights
q                        # quit the flight tracker and deallocate memory
```

The flight data you need to keep track of are stored in a struct called `flight` that has the following fields:
<airlines: a string of two chars, flight number: int, arrival time:int, departure time:int>

All flights and the associated flight information has to be stored in a dynamically allocated link list, where the `LLNode` struct has two fields
<next: LLNode*, data: flight*>

Operational requirements:
1. After every valid operation (a, d), the program prints the list of flights.
2. If a valid operation (a, d) cannot be carried out OR operation does not exist (not a or d), print appropriate error message. Also if the flight data to be stored contains information with incorrect format or is missing, print an error message and omit the insertion of the flight to the list.
3. If you delete a flight or quit the program the link list must be deallocated – each data struct of type `flight` as well as the `LLNode` struct have to be property de-allocated from the heap
4. The program will allow for any number of flights to be entered and stored in the llist for tracking.
5. The input file (for example flights.txt) is an optional parameter to the program – if the file name is passed into the program at the execution time, the file contains flights to be inserted to the llist. If no file is specified, read instructions with data from the command line.

The input file contains one flight's information per line, and after you read the line into a struct `flight`, then insert this struct into the llist of flights.
Usage: `./flight <file name>` OR `./flight`
6. Inserting of a flight has to be in a non-decreasing order of the flight number.
7. Insertion of a flight has to be a recursive algorithm.
8. You are not allowed to use any arrays to store your data, only malloc-ed structs.
9. You must keep the linked list in a sorted order – when read and insert flights, they have to be inserted into the correct place in the linked list. Also, when you delete a flight, the list should still be in order.
10. Print each flight as you read it into your program, then print the final sorted list before ending your program.
11. Functions should be used to insert, print, save, and delete the flight structures.
12. **You will be graded on the overall design of your code.** Design your program by breaking the overall program logic into logical blocks or task that are defined as functions in your code. Try to reuse as much code as possible. Use the previous HWs as your guide or an illustration of how to design your program's functional structure. The following are only hints:
You should use the following functions at a minimum. Feel free to alter these functions prototypes if you need to, to suit your program.

```
void insert(struct LLNode *, struct flight *);
void delete(struct LLNode *, int);
void save(char *);
void printList(struct LLNode *);
...
```

Useful function and function parameters for this assignment:
check out different formating for
– scanf/fscanf and the associated printf – %d vs %i, %c vs %s
atoi – convert a string to an integer
readline – read a line
strtok – tokenize a line

Grading criteria:

- (10 points) Code Operation: Does code do what is listed in the specification? Note: code that does NOT compile will receive 0 points for code operation.
  - 1 points print
  - 5 points insert
  - 2 points delete
  - 1 point free
  - 1 point makefile (works for compiling and cleaning)
- (5 points) Summary report: Completeness, correctness, and clarity

Sample input/output:

```
Inserting flight: AA 2415 2015 2135
Inserting flight: AM 45 1500 1615
Inserting flight: DT 123 1230 1320
Inserting flight: FR 1440 1000 1100
Inserting flight: SW 13 800 905
Inserting flight: JB 3626 721 830
Inserting flight: UN 4452 740 840
Inserting flight: AA 7786 2254 2132
Inserting flight: AM 2184 2225 2345
Inserting flight: DT 596 1551 1630
Inserting flight: FR 5378 1150 1300
Inserting flight: SW 9959 1601 1710
Inserting flight: JB 8849 2028 2145
Inserting flight: UN 6110 805 1000
Inserting flight: AA 1381 857 1110
Inserting flight: AM 7333 1142 1300
Inserting flight: DT 1857 958 1540
Inserting flight: FR 8011 1721 1835
Inserting flight: SW 8524 2006 2100
Inserting flight: JB 7098 1452 1550
Inserting flight: UN 7916 1405 1530
Inserting flight: HN 876 1159 1430
Inserting flight: DL 345 730 815
Inserting flight: AL 366 800 950
Inserting flight: RS 7322 745 1000
Inserting flight: JK 23 1230 1405
Inserting flight: LO 238 600 745
Inserting flight: OP 8763 1600 1740
Inserting flight: LK 987 935 1200
Inserting flight: MN 678 1020 1130
Inserting flight: LN 543 2200 2330

Flight list:
SW 13 800 905
JK 23 1230 1405
AM 45 1500 1615
DT 123 1230 1320
LO 238 600 745
DL 345 730 815
AL 366 800 950
LN 543 2200 2330
DT 596 1551 1630
MN 678 1020 1130
HN 876 1159 1430
LK 987 935 1200
AA 1381 857 1110
FR 1440 1000 1100
DT 1857 958 1540
AM 2184 2225 2345
AA 2415 2015 2135
JB 3626 721 830
UN 4452 740 840
FR 5378 1150 1300
UN 6110 805 1000
JB 7098 1452 1550
RS 7322 745 1000
AM 7333 1142 1300
AA 7786 2254 2132
UN 7916 1405 1530
FR 8011 1721 1835
SW 8524 2006 2100
OP 8763 1600 1740
JB 8849 2028 2145
SW 9959 1601 1710
Enter or flight number to delete or enter 'q' or 'Q' to exit: 23
Deleting flight number 23

Flight list:
SW 13 800 905
AM 45 1500 1615
DT 123 1230 1320
LO 238 600 745
DL 345 730 815
AL 366 800 950
LN 543 2200 2330
DT 596 1551 1630
MN 678 1020 1130
HN 876 1159 1430
LK 987 935 1200
AA 1381 857 1110
FR 1440 1000 1100
DT 1857 958 1540
AM 2184 2225 2345
AA 2415 2015 2135
JB 3626 721 830
UN 4452 740 840
FR 5378 1150 1300
UN 6110 805 1000
JB 7098 1452 1550
RS 7322 745 1000
AM 7333 1142 1300
AA 7786 2254 2132
UN 7916 1405 1530
FR 8011 1721 1835
SW 8524 2006 2100
OP 8763 1600 1740
JB 8849 2028 2145
SW 9959 1601 1710
```

HW 2 Report Guidelines and Format – use the template provided below. Include the questions in your write-up.

Name: **CS 305 HW 3 Report**

1. Questions (include these in your write-up):

1a. (.5 point) If your program does not meet the specifications, please note the differences and anything that is not working correctly. If it works, write “Meets all specifications.”

1b. (.5 point) Copy and paste your terminal window contents from running the flight tracker on the 10 flight list files (similar to what is copied/pasted above for sample output). Make your test files as simple as possible to demonstrate program’s operation – both checking for error and correct operation.

2c. (3 point) Provide a design document that shows the function, function calls, structs, and any necessary details that explain the design and operation of your program.

2d. (.5 point) Do you have test cases for different possibilities of no inputs, incorrect input configurations in the file or from the command line? If not, state the cases that you did not test.

3a. (.25 pt) How much time did you spend in total on this homework assignment (including the report)?

3b. (.25 pt) What was the most challenging part for you when completing this assignment?

Appendix A: (copy this statement if it applies to you) I verify that the code and this write-up were authored by me. I have documented the help I have received in comments in the code files.

Appendix B: Copy and paste your source files .c, your header files .h and makefile here (use Courier New 8pt font so the characters line up correctly)