Assignment2
Deadline: 11:59PM Mar 10

Requirements
In this assignment, you are required to write a program that reads the content of a file and writes a sorted output to a file. You will be working in a group of 2. The number of lines in the file is not known (it could be thousands of lines). The format of each line is:

```
>> cat input.txt
Mary Jackson Feb-2-1990 4.0 I 60
Jack He Feb-3-1990 2.45 D
Mike Johnson Sep-2-1980 3.125 D
Jane Zhang Mar-2-1970 3.8 I 120
```

You must use a quick sort to implement the sorting algorithm. Your pivot must be random. Below is the sorting criteria where 1 is the first criteria to use and use the next criteria if the current criteria results in a tie.

1. Year of birth
2. Month of birth
3. Day of birth
4. Last name (alphabetical order)
5. First name (alphabetical order)
6. GPA
7. TOEFL (no TOEFL score is provided, then domestic students takes precedent)
8. Domestic > International

The birthday has the month written in a string. Only the following abbreviation will be used for input and must be used for the output.

```
Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec
```

It must be an ascending order. For example using the sorting criteria 1, someone born earlier must be outputted first. All sorting criteria must follow this ascending order. Below are additional specifications of this assignment.

1. An international student has a TOEFL score ranging from 0-120.
2. DomesticStudent struct does not have the TOEFL field.
3. I stands for an international student and D stands for a domestic student in status column.
4. If the format of a student does not conform the specified format, your program must print the appropriate message and exit.
   ```
   //replace XXX with an appropriate message. Make sure that the word ERROR are all capitalized.
   printf("COMP2510ERROR: XXX");
   exit(1);
   ```
5. The first argument is the intput file, the second argument is the output file name, and the third argument is the option (specification 6). The run command will be

```
./<name of executable> <input file> <output file> <option>
```

6. The option field is the command line is an integer (1, 2 or 3). Here are descriptions of options.
    a. Option 1 only saves the sorted output of domestic students (no international students in the output file)
    b. Option 2 only saves the sorted output of international students (no domestic students in the output file)
    c. Option 3 only saves the sorted output of all students
7. You must handle corner cases including intput format errors with the output message as in specification 4.
8. Assume that everyone has a first name and last name. No middle name and names are case insensitive, but the output must have the same capitalization as the input
9. GPA ranges from 0.0 to 4.3 and we consider up to 2 decimal places. If more than 2 decimal places, please round off to the nearest 2 decimal places.
10. Birthday must be displayed with three letter abbreviation as shown above with the first letter being the capital.
11. Year of birth will range from 1950 to 2010
12. The number of lines are unknown, but each line is guaranteed to be maximum 100 characters.

Some students took the course multiple times, so there can be multiple entries with the same name and the same birthday. You should handle it, so that only the highest entry shows up in the output file.

Now since we don't know the total number of lines, we have to come up with some storage size first. A typical strategy is to start with a small array size and double the size as the capacity is filled out. Start with a size of 2 for this assignment and double it as it gets filled in. Below is the code snippet of how you can achieve this.

```
int capacity = 2; //initial small size
int used = 0; //no items in use yet
int *intArray = (int *)malloc(sizeof(int)*capacity);
/**create an array of size 2
   after filling up the array, keep increasing used each time you
   assign an item
**/
if (used == capacity) {
     capacity = capacity * 2; //double the capacity
     //request an array of size 4, keeping the items
     intArray = (int *)realloc(sizeof(int)*capacity);
}
```

Restrictions

- Your output must have the same content format as the input.
- No trailing spaces or extra new line characters at the end of file as it will fail the diff
- You must only use a quick sort

Grading
I will provide sample input and output files. Make sure to test against those to ensure that your program output format works. Any grading failure due to not following specifications will result in 0. For full marks this week, you must:

- (1 point) Correctly submit A number file of you and your partner
- (1 point) Not having any files in github other than assignment2.c and AXXXX.txt
- (6 point) Generate a correct solution (including correct memory allocation and deallocation) to the problem(s) in this lab
- (2 point) handle error cases

Submission Files

- You must deliver only one .c file named: **assignment2.c** (do not capitalize)
- Two AXXXX.txt files if you have a partner (empty file, but with your A number as file name. Make sure to include 0's, match this A number with your A number in learning hub, and have .txt extension)
- Github: https://classroom.github.com/a/qfy1tE6P