Lab #9

Summer 2023

Requirements

In this lab, you will cover executing an efficient search on a sorted array. Remember that not all of the lab's requirements may be possible without the use of helper functions. In this lab you are given the following struct definition:

```
typedef struct {

        unsigned int flightNumber;
        float distance;

        unsigned short passengers;

} Flight;
```

1.1 makeArray

```
void * makeArray(int arraySize, int elementSize)
```

Info: This function will take an array size, as well as the size of each element in the array. It will allocate an array with the given size, and store the size before the start of the array as an int. If allocating the array was successful, it will return a pointer to the array, otherwise it will return NULL.

1.2 getSize

```
int getSize(void *array)
```

Info: This function takes an array which was allocated with makeArray, and returns the size stored before the array.

1.3 searchFlights

```
// O(log(n))

int searchFlights(Flight *array, Flight *query)
```

Info: This function performs a *binary search* based on the "distance" member on the given struct array using **recursion**. This function will return the index of the query struct (0-based) when it is successfully located, or **-1** on error.

1.4 compareFlights

```
// O(1)
int compareFlights(Flight *a, Flight *b)
```

Info: This function compares the two structs given by their **distance** members. It should return a *strictly negative number* if a < b, a *strictly positive number* if a > b, or 0 if they are equal.

1.5 freeArray

```
void freeArray(void *array)
```

Info: This function takes an array which was allocated with makeArray, and frees the memory allocated to the array.

Submission Information

Submit this assignment by using the mucsmake command.

Use the following submit command on tc.rnet:

mucsmake <assignment> <filename>

For example:

mucsmake lab9 lab9.c

Rubric: 15 points

1. Write required *makeArray* function
   * 1 points
2. Write required *getSize* function
   * 1 points
3. Write required *searchFlights* function
   * 10 points
4. Write required *compareFlights* function
   * 2 points
5. Write required *freeArray* function
   * 1 points

Notice:

1. Do NOT change the given .h file or function prototype.
2. All of your lab submissions must compile under GCC using the $-Wall$ and $-Werror$ flags to be considered for a grade.
3. You are expected to provide proper documentation in every lab submission, in the form of code comments.
4. Make sure to reference the source of any code that was not created independently by yourself. For example, if you used code which was presented in class lectures, the source would be something like "CS 2050 Course Notes by Jim Ries".