Prelab: Week of February 3rd

Implement a function that will allocate (and return) a float array containing values read from a file:

```
float * readFloatFileIntoArray(FILE *fptr, int *length)
```

The first entry in the file will be (*is assumed to be*) an integer specifying how many floats will follow. That integer will tell you how big your array needs to be. You'll allocate that array and then read and store each of the float values from the file into the array, which will be returned to the calling program. Okay, so what is the second parameter? Well, that's how we can tell the calling program how many elements there are in the returned array. That's obviously something the calling program will need to know!

QUESTION: Suppose `malloc` fails (always check!), how will the calling program know if your function succeeded or failed?

Because the user (i.e., the calling program) will eventually need to free the array, create a function called `freeFloatArray` that will not only do that but will also set the user's array variable/pointer to NULL so that s/he doesn't accidently reference memory that has been freed. This function will not return anything and thus should be declared "`void freeFloatArray`". You'll need to figure out the rest of the prototype for the single parameter.

HINT 1: The user will have to pass his/her array variable *by reference* to `freeFloatArray`.
HINT 2: Here's where you'll need to make use of ** when you pass the array pointer by reference.

In summary, you will provide two functions. Be sure to document your two functions so that a user will understand what each does, how to use it, and any other necessary information such as possible error conditions.

- JKU