Lab 2: binary files and structs

Due by midnight, Wednesday 9/19

> If you're stuck on the ID3 tag part of the project, this might give you some ideas.

In this lab, you can get some practice with a simple struct and reading/writing instances of it to/from a file.

Getting the materials

First, **make a `lab2` directory** in your `~/private/cs0449` directory like you did for lab1.

Then, `cd` **into it** and run this command to get the files:

```
cp ~jfb42/public/cs449/lab2/* .
```

> Don't forget the space and period in the above command.

- `lab2.c` **is where you'll do your work.**
- `driver.c` is a small program I wrote to use your code.
- `lab2.h` is a *header file* which both C files use. (We'll learn about those.)
- `food.db` is the little data file which the driver program uses to exercise your code.

Compiling and running

Compiling a C program with two files is simple – just list both programs on the GCC commandline.

```
gcc -Wall --std=c99 -o lab2 lab2.c driver.c
```

> Do not list `lab2.h` on the command line. We only compile C files.

If you do this with the unmodified files, the resulting `./lab2` executable will do nothing.

What to do

Implement the functions in `lab2.c`. Follow the directions in the comments, and remove the comments as you go.

Start from the top function (`open_database`) and work your way down. As you do so, you will be able to use more and more features of the driver program.

Hints:

- There is not actually a lot of code to write here! But like a simple recipe, the ingredients matter. Getting the details right is important.
- Make a copy of the `food.db` file in case you mess it up. If you still mess it up, `cp ~jfb42/public/cs449/lab2/food.db .` again.
- Make sure you get the order of parameters to `fseek` right. I screwed it up when I made this. :P
- If your program gets stuck in an infinite loop, `ctrl+C` will terminate it.

Submission

Remember the instructions you learned last time?

Follow those, but replace `lab1` with `lab2`.