SCC110: Software Development Term 1. C Programming.

Assessed Exercise: Santa Claus is coming to town
Moodle Submission Deadline: 16:00 Friday Week 10
Assessment Mode: IN LAB ASSESSMENT IN YOUR OWN WEEK 10 LAB SESSION

Aims

This **assessed exercise** is designed to test your understanding of all the software development concepts we've seen in the lectures and labs, and their application using C. The assignment is separated into incremental tasks, which will result in marks being awarded for the successful completion of each task.

Santa Claus is coming to town. He knows if you've been good or bad, he knows if you've been naughty or nice, he's making a list, he's checking it twice.

In this assessed exercise we've given you exclusive access to Santa's list, stored in a file called "`santaslist.txt`" (make sure you save the file into the same directory as your program). This file consists of 0 or more lines, each of which has the following format (*no header is included*).

```
surname,firstname,behaviourscore,lat,long,giftweight
```

Field	**Description**
`surname, firstname`	surname and firstname are both limited to 20 characters each
`behaviourscore`	is an integer (from -10 = naughty to +10 = nice)
`lat, long`	are latitude and longitude both represented as doubles
`giftweight`	is a float (weight in Kg)

An example line might be as follows:

```
Scrooge,Ebenezer,-10,54.05262333494372,-2.804919966042138,0.1
```

This indicates that Ebenezer Scrooge has been very bad, lives in Lancaster and can expect to receive a total of 0.1Kg of gifts when Santa visits.

The Program

Your task is to write a program that can perform a range of calculations on Santa's list to help him and the elves with their planning.

Important points:

1. The program should be **your own work**—this is *not a group exercise!*—do not share all or part of your solution with anyone, and do not post it online (*not even to github!*). You *will be required* to submit your code for plagiarism checking.
2. If you need help with your programming please seek help from the tutors or the lab demonstrators—*not from colleagues or the Internet*.
3. You must attend on-line **your specified** practical class in week 10, assuming this does not conflict with your allocated travel window. You will only be able to access the questions **during the practical classes** in week 10, and you can **only make one overall attempt**. *Please let us know if you are not able to attend your allocated day/time.*
4. You will need a working program at the start of week 10 *so please be sure to join the practical class with a working program*.

5. Ensure you read the specification very carefully and that your functions are declared exactly as specified.

Part 1—The Main Program

- You should write a program that opens the file and that can call the functions you write as your answers in Parts 2-3 to test that your program is working satisfactorily. Your main function will not be marked—it is just to help you develop your program. *See resources below for important pointers!*

Part 2—Analysing the list to help Santa

Santa needs to understand a little more about who he is delivering to and how naughty or nice they've been.

- Write a function that take as a parameter a pointer to the file that you have opened and returns the number of people on Santa's list.

  ```
  int number_of_people(FILE *)
  ```

- Write a pair of functions that take as parameters a pointer to the file that you have opened and that then count the number of ``nice'' and ``naughty'' people. Where nice people have a `behaviourscore` is greater than zero, and naughty people a score less than zero.

  ```
  int number_of_nice_people(FILE *)

  int number_of_naughty_people(FILE *)
  ```

- Write a function that takes as a parameter a pointer to the file that you have opened. The function should return the average behaviourscore for all entries in the file.

  ```
  float average_behaviourscore(FILE *)
  ```

- Write a function that takes as a parameter a pointer to the file that you have opened. The function should return the *lat* and *long* of the location of the naughtiest person on the list (return the first person if there are multiple people who have been equally naughty).

  ```
  void naughtiest_person(FILE *, double *, double *)
  ```

Part 3—Take off from Lapland

Santa has a limited number of sleighs and needs to plan his deliveries carefully.

- Write a function that takes as a parameter a pointer to the file that you have opened. The function should return the total weight of gifts Santa has to deliver this Christmas.

  ```
  float total_weight(FILE *)
  ```

- Each sleigh can only carry a maximum of 1500Kg. Write a function that takes as a parameter a pointer to the file that you have opened and returns the total number of sleighs Santa will have to use to deliver gifts this Christmas.

  ```
  int calculate_sleighs_needed(FILE *)
  ```

- If the airport at Santa's secret base can only launch 1 sleigh every 90 seconds write a function that calculates how long will it take to launch the required number of sleighs this Christmas (*rounded up to the nearest minute*).

  ```
  int calculate_time_needed(FILE *)
  ```

- Write a function that takes as a parameter a pointer to the file that you have opened and returns the *lat* and *long* of the address with the most number of people living there.

  ```
  void largest_gathering(FILE *, double *, double *)
  ```

Week 10's Practical

In the week 10 practical you will have to modify your program and then submit the complete program *for plagiarism checking* and then answer a series of on-line questions during the practical class. The questions are designed to test your program and to assess your understanding of the program that you have written and of programming in general. You will be required to use Coderunner to test your functions. The more of the functions you are able to implement, naturally the more of the questions you'll be able to answer and the higher the potential for marks.

We recommend generating your own test files to ensure your program is well tested in order to catch errors in advance of your marking session.

Resources

The best preparation for this exercise is the week 8 access log task, in which we make good use of the `stdio.h` library including `FILE *`, `fopen`, `fgets` and `fclose`.

- We've seen 'scanf' to gather input from the user. You may find `fscanf` useful, the variant of scanf for file input (manual page).
- Note there are many useful 'format' options for fscanf, particularly useful are %d (decimal/int), %f (float), and %lf (long float, or double).

```
fscanf(boysAndGirls, "%[^,],%[^,],%d,%lf,%lf,%f ", surname, firstname,
                &behaviourScore, &latitude, &longitude, &giftWeight)
```

- For instance, would read from a file handle (boysAndGirls), reading 6 fields separated by commas (,). Note we've had to use %[] instead of %s to read the characters until 'not ,' (^,). The last space 'swallows' end of line whitespace including the \n.
- For your driver function (main), you will need to open the file. You may find it helpful to 'seek back to the start' between functions, e.g. using 'rewind' (manual page).