Fitness Tracking Lab – Arrays of Structures

General Information

This lab emulates part of the code for a fitness tracker app. It determines how many equivalent miles have been walked.

Detailed Information

The code of this program could be used in conjunction with a wearable fitness tracker such as a FitBit™ and/or a phone app where people choose from a list of activities. The app might even be synched with the wearable. Since we don't have these input devices (the wearable and/or phone app), our program will get its input from a file. However, the program we will write is similar to the code that would be part of a fitness tacker app.

For some real input, you each need to track your activity for two weeks and enter it in the spreadsheet provided to you where you'll keep track of activities and minutes spent. Use the codes below to indicate the activity.

Activity	**Steps in 1 minute**	**Code**
Assembly Line	85	0
Basketball (shooting baskets)	136	1
Basketball game	242	2
Bicycling under 10mph	121	3
Bicycling	242	4
Cooking	61	5
Dance	167	6
Football	242	7
Hiking	182	8
House cleaning	91	9
Gardening	99	10
Miniature golf	91	11
Racquetball	212	12
Rowing	212	13
Running 6 mph	303	14
Running 7 mph	348	15
Running 8 mph	409	16
Shopping	70	17
Soccer	212	18
Softball	152	19
Stairs	273	20
Swimming laps	212	21
Tennis	212	22
Walking 3mph	100	23
Walking 4mph	152	24
Weight lifting	121	25
Yoga	76	26

Cynthia Arnold

Here’s the beginning of my sample spreadsheet. It lists my name and height in inches (62.5) followed by the number of activities on the next lines, 4, in this case. The following rows list activities. So, 24 is walking 4 mph above, and I did this for 60 minutes. I lifted weights (code=25) for 45 minutes, I played tennis for 55 minutes, and I practiced yoga (code=26) for 30 minutes.

Cindy	Arnold	62.5	4
24	60		
25	45		
22	55		
26	30		

The program starts by asking the user for the name of an input file.

- If the file doesn’t exist, print the following error message where ________ is the file name and end the program.
  `File ________ does not exist. Please contact the administrator.`
- If the file is empty, print the following error message where ________ is the file name and end the program.
  `File _________ is empty. Please contact the administrator.`

If the file exists and is non-empty, it will be in the following format with the data in it from all students in the class. So, it will have the data for more than one person in it. I will create this text file by adding all of your data to one spreadsheet then saving it from MS Excel as a space delimited, formatted text file (*.prn). So, the text file created from the MS Excel spreadsheet shown above with just my data in it would look like:

```
     Cindy Arnold 62.5 4
     24 60
     25 45
     22 55
     26 30
```

The program prints a welcome message then reads the file into an array of structures (assuming the file exists and is non-empty; otherwise see above). The program then sorts the array by the number of equivalent miles walked per person in descending order (largest first), calculates total miles walked by everyone, prints a table formatted as shown below and goodbye message, and ends. Names will be no longer than 20 characters. Total miles will be less than 99,999.99.

```
Welcome to the PSCC Fitness Tracker.
Input file: tracking.prn

                       Tracking Results
     Name                                        Miles
     --------------------------------------------------
     Bilbo                Baggins                165.90
     Rick                 Grimes                  75.40
     Cindy                Arnold                  11.61
     Zelda                Hyrule                   1.99
     ==================================================
     TOTAL                                       254.90
Thank you for using the PSCC Fitness Tracker.
```

Use the at least following constants and structure definition for the array of structures.

```
const int MAX_NAME_LENGTH = 21;
const int MAX_ACTIVITIES = 365;
const int MAX_PARTICIPANTS = 40;
struct Participant {
     char firstName[MAX_NAME_LENGTH];
     char lastName[MAX_NAME_LENGTH];
     double height;                           // in inches
     int numActivities;
     int activities[MAX_ACTIVITIES];          // parallel array with
     int minutes[MAX_ACTIVITIES];             //   minutes array
     double totalMiles;

};
```

Other constant examples:

```
const int NUM_ACTIVITIES = 27;

and
const int ASSEMBLY_LINE = 0;
const int BASKETBALL_SHOOTING = 1;
```

etc.

or
`const int ACTIVITY_STEPS[NUM_ACTIVITIES] = {85, 136,` etc.
so that `ACTIVITIES[BASKETBALL_GAME]` or `ACTIVITIES[participants[i].activities[j]]` is `242` (the number of steps for 1 minute of a basketball game) assuming `participants[i].activities[j]` is a basketball game.

Assume the distance in feet in one person's step, *f*, is given by the following formula where *h* is the person's height in inches.

$$f = (0.413 * h) /12$$

Recall there are 5280 feet in a mile.

Relevance

The code we are writing is similar to code in a lot of apps that help people track their activities. I use MyFitnessPal™ sometimes to track my activities. There are even wearable device such as a Fitbit™ that track steps. Our program just doesn't have a good interface for input such as a nice GUI or a blue-tooth enabled wearable. Even without the nice input, we could still add a lot more functionality to this program such as the ability to create a new activity other than the ones listed above. We could add calculations such as calories burned in which case might want more data such as weight, age, gender, and even heart rate. We could even add food tracking to calories consumed and compare that to calories burned.