CSCI 135 Section 05 – Fall 2014

Programming Assignment #1

Program Description

Your program will read the attributes for two bodies from a file and then perform computations/manipulations of those attributes. Your program's output should be written to the standard output stream, except for error messages, which should be written to the standard error stream.

Input

The input for the program will come from a file named `"data.txt"` which the program should expect is in the current working directory. The program will need to open and read the data from the file. If the file does not exist or cannot be opened for any reason, the program should display an error message on the standard error stream and exit. The input file will have the format described below; your code does not need to check for incorrectly formatted data.

data.txt file format

The data file contains a seed value for the random number generator. It also contains the mass, x-coordinates and y-coordinates of two bodies:

```
SEED_VALUE
MASS_BODY1 XCOORD_BODY1 YCOORD_BODY1
MASS_BODY2 XCOORD_BODY2 YCOORD_BODY2
```

- The seed value will be a whole number, greater than or equal to zero
- The values for mass, x-coordinate and y-coordinate of each body will be floating-point values
- Mass will be a value greater than zero (units are kilograms)
- Each of the coordinates may be either a positive or negative value (units are meters)

In addition to the input data coming from a file, running certain tasks will require input from the user (described within the User Interface section below).

Immediately after reading the data from the input file, you should set the random seed value as described in the next section below.

Computational Tasks

Your program must perform the following tasks, implemented by functions that you define:

Set random seed value: given the seed value from the input file, set the seed value for the random number generator in your program. If the input value is zero, set the seed to the current time. If the input value is greater than zero, then use the value provided as the seed.

Calculate the gravitational attractive force between two bodies: given the mass for body #1 (m_1), the mass for body #2 (m_2) and the distance between them (d), the gravitational attractive force (F) between the two bodies can be calculated with the following formula:

$$F = G \times \frac{m_1 m_2}{d^2}$$

Where G in the above formula is the universal gravitational constant:

$$G = 6.674 \times 10^{-11} (\frac{m^3}{kg \cdot sec^2})$$

The gravitational force will be in Newtons, where one Newton (N) is:

$$N = kg \times \frac{m}{sec^2}$$

Calculate the distance between two bodies: given the x- and y-coordinates for two bodies, calculate the distance between them. The distance will be in meters.

Update the coordinates of a body: given a distance and direction (in degrees), you must calculate and update the x- and y-coordinates of a body. The direction will be a whole number from 0 to 359. Consider 0 degrees equal to the x-axis in the positive direction; 90 degrees equal to the y-axis in the positive direction; 180 degrees equal to the x-axis in the negative direction; and 270 degrees equal to the y-axis in the negative direction.

User Interface

After you have read the data from the file and set the random number seed, the program should display the following menu:

```
1     Calculate the gravitational attractive force between the two bodies
2     Calculate the distance between the two bodies
3     Move body #1 a random distance and direction
4     Move body #2 a random distance and direction
5     Move body #1 a user defined distance and direction
6     Move body #2 a user defined distance and direction
7     Exit the program
```

The program should then prompt the user for a selection, waiting until they enter a valid choice. If the user enters 7, the program should output a goodbye message and then quit. If the user enters anything besides a valid selection (from 1 to 7), the program should output an error message and ask for a new selection.

After a valid selection is entered (1-6), the program should perform the requested task then redisplay the menu and await another selection. Before the tasks 1-6 are to be performed, the program should output the current attributes for both bodies (mass, x- and y-coordinates should be displayed with 2 decimal places of precision):

```
Body #1: mmmm.mm kg @ (xxxx.xx, yyyy.yy)
Body #2: mmmm.mm kg @ (xxxx.xx, yyyy.yy)
```

For task 1, the result of computing the gravitational attractive force should be displayed (with 5 decimal places of precision):

```
Gravitational Force between bodies: ffff.fffff Newtons
```

For task 2, the result of computing the distance should be displayed (with 5 decimal places of precision):

```
Distance between bodies: dddd.ddddd Meters
```

For tasks 3 and 4, the distance and direction for the move should be randomly generated. The distance should be between 10.0 and 1000.0. The direction should be in degrees; a whole number from 0 to 359. The randomly chosen information should be displayed as well as the new position for the particular body (distance, x- and y-coordinates should be displayed with 2 decimal places of precision):

```
Moving Body #N dddd.dd meters in direction zzz degrees
Updated position for Body #N: (xxxx.xx, yyyy.yy)
```

For tasks 5 and 6, the user should enter the distance and direction for the move. The distance should be between 10.0 and 1000.0. The direction should be in degrees; a whole number from 0 to 359. If the user enters an invalid value for either, the program should ask them again until a valid value is entered. The information for the move should be displayed as well as the new position for the particular body (distance, x- and y-coordinates should be displayed with 2 decimal places of precision):

```
Moving Body #N dddd.dd meters in direction zzz degrees
Updated position for Body #N: (xxxx.xx, yyyy.yy)
```

Programming Rules

Your program must adhere to the programming rules as described in the "Programming Rules and Guidelines" document posted on Blackboard. Of note, or in addition:

You must document your program (preamble, function pre- and post-conditions, comments throughout the code, and appropriately choosing variable/function names.

You must implement and use at least four functions that perform the four **Computational Tasks** described above. You are free to implement and use additional functions as you see fit.

Each of the menu tasks 3-6, as described in the **User Interface** section, should utilize the single function you've written to "update the coordinates of a body."

You may not use any global variables, unless they are constants.

You are expected to handle a missing input file gracefully, as described in the **Input** section intro above. Otherwise, you may assume the input file will be correctly formatted.

All output should be sent to the standard output stream, except for error messages. Any error messages you display should be sent to the standard error stream.

Grading

- Does the program compile (on one of the cslab machines): **15%**
- Correctness and implementation of the four functions: **35%**
    - Set random seed value (5%)
    - Calculate the gravitational attractive force between two bodies (15%)
    - Calculate distance between the two bodies (5%)
    - Update the coordinates of a body (10%)

- Correctness and implementation of the file I/O operations: 5%
- Correctness and implementation of the main function and any other functions in the program: 30%
- Documentation: 10%
- Style (variable naming, indentation, etc…): 5%

What to submit and when

This assignment is due by the end of the day (i.e. 11:59PM, EST) on October 9, 2014. You will submit a single source code file (.cpp) for this assignment. The name for your file should be **`lastname_firstname_hwk1.cpp`** (replace **`lastname`** and **`firstname`** with your last and first name). Submit the file to the assignment in our Blackboard course. Do not paste the code into your submission. You should test your program on one of the cslab machines to make sure it compiles and runs correctly there.