Introduction to C - Programming Assignment #3
More Car Accessories!
Due date: ***October 13, 2010 – 11:59pm***

Note: In this assignment, you are required to write three programs.

Objectives
1. Learn how to write and use loops.
2. Review the use of if statements.

References
CKnights: Sections 4,5,6 GuhaDraft: Chapter 4

Problem A: Real-Time Fuel Efficiency Calculation
So far, your fuel efficiency calculator only works for making the calculation over the duration of a trip. But, you are too impatient to wait for this information until the end of the trip. Instead, you would like updates about your fuel efficiency for every five minute interval of your trip. In this program, you will ask the user how long their trip was (which is guaranteed to be a positive multiple of five minutes), the radius of their tire, and then for each five minute interval, you will prompt the user to enter the number of revolutions of their wheels and how much fuel their car consumed in the five minute interval. After each set of input, you should output the user's fuel efficiency for that five minute interval. After data for the last interval is entered, your program should ***also*** report fuel efficiency data for the whole trip.

Input Specification
For the first two initial values entered,
1. The trip length will be entered in minutes and be a positive integer multiple of five.
2. The radius of the car's tires will be a positive real number in inches.

For each five minute interval, both pieces of data will adhere to the following specifications:

3. The number of revolutions the car's tires make will be a positive integer.
4. The amount of gas, in gallons, the car uses will be a positive real number.

Output Specification
For each five minute interval, output a single line with the following format:

```
Time A-B minutes: Your car averaged XX.XX mpg.
```

where A and B are the appropriate multiples of 5 and XX.XX represents the fuel efficiency of the user's car during that particular time interval, in miles per gallon.

At the very end, output an extra line which has the trip average data with the following format:

```
For the whole trip, your car averaged XX.XX mpg.
```

where XX.XX represents the fuel efficiency of the user's car for the whole trip, in miles per gallon.

Output Sample
Below is one sample output of running the program. Note that this sample is NOT a comprehensive test. You should test your program with different data than is shown here based on the specifications given above. In the sample run below, for clarity and ease of reading, the user input is given in *italics* while the program output is in bold.

Sample Run #1
```
How long is your trip, in minutes?
10
What is the radius of your tires, in inches?
15

During time interval #1, how many revolutions did your car's tires
make?
10000
During time interval #1, how many gallons of gas did your car use?
0.75
Time 0-5 minutes: Your car averaged 19.83 miles per gallon.

During time interval #2, how many revolutions did your car's tires
make?
20000
During time interval #2, how many gallons of gas did your car use?
0.50
Time 5-10 minutes: Your car averaged 59.50 miles per gallon.

For the whole trip, your car averaged 35.70 mpg.
```

Problem B: Printing Out a Tire

You are sick and tired of calculating fuel efficiency, so you decide that your car's computer needs a feature that's more fun. You decide it would be cool to write a program to print out what a tire looks like, at least in two dimensions. There are two pieces of input you receive from the user: the number of "pixels" the outer radius of the tire is, *r1*, and the number of "pixels" the inner radius of the tire is, *r2*. Using these two pieces of information, you are to print out a grid with the size *2*r1+1* by *2*r1+1*, that displays the wheel. In particular, wheel "pixels: will be designated by the character '*'. These will be the "pixels" that are in between *r2* and *r1* "pixels" away from the center of the grid. "Pixels" that are farther away from the center of the grid than this will represented by the ' ' character. Pixels that are less than *r2* "pixels" away from the center will represent the tire and will be drawn with the '+' character. Finally, "pixels" that are one pixel or closer to the center of the grid (there will always be five of these), will be drawn with the 'X' character.

Input Specification

1. *r1*, the outer radius, will be a positive integer less than 50.
2. *r2*, the innter radius, will be a positive integer less than *r1*.

Output Specification

1. Wheel characters will be represented by the character '*'.
2. Tire characters will be represented by the character '$'.
3. Axle characters will be represented by the character '+'.

The "grid" that gets outputted should be indexed from –r1 to r1, in both rows and columns. The final product should resemble what a wheel looks like, viewed from the side of a car. **Hint: Loop through each grid square, and when you get to that square, decide which character is supposed to be printed based on that square's distance from the center square, (0,0).**

Output Samples

Here are two sample outputs of running the program. Note that these samples are NOT a comprehensive test. You should test your program with different data than is shown here based on the specifications given above. The user input is given in *italics* while the program output is in bold.

Sample Run #1

```
Enter the outer radius of your wheel?
20
Enter the inner radius of your wheel?
15

             *************
           *****************
         *********************
        ***********************
      ***************************
     *********$$$$$$$$$$$*********
    ********$$$$$$$$$$$$$$$********
    *******$$$$$$$$$$$$$$$$$*******
   ******$$$$$$$$$$$$$$$$$$$$$******
  ******$$$$$$$$$$$$$$$$$$$$$$$******
  ******$$$$$$$$$$$$$$$$$$$$$$$******
 ******$$$$$$$$$$$$$$$$$$$$$$$$$******
 *****$$$$$$$$$$$$$$$$$$$$$$$$$$$*****
******$$$$$$$$$$$$$$$$$$$$$$$$$$$******
*****$$$$$$$$$$$$$$$$$$$$$$$$$$$$$*****
*****$$$$$$$$$$$$$$$$$$$$$$$$$$$$$*****
*****$$$$$$$$$$$$$$$$$$$$$$$$$$$$$*****
*****$$$$$$$$$$$$$$$$$$$$$$$$$$$$$*****
*****$$$$$$$$$$$$$$+$$$$$$$$$$$$$$*****
*****$$$$$$$$$$$$$+++$$$$$$$$$$$$$*****
*****$$$$$$$$$$$$$$+$$$$$$$$$$$$$$*****
*****$$$$$$$$$$$$$$$$$$$$$$$$$$$$$*****
*****$$$$$$$$$$$$$$$$$$$$$$$$$$$$$*****
*****$$$$$$$$$$$$$$$$$$$$$$$$$$$$$*****
*****$$$$$$$$$$$$$$$$$$$$$$$$$$$$$*****
******$$$$$$$$$$$$$$$$$$$$$$$$$$$******
 *****$$$$$$$$$$$$$$$$$$$$$$$$$$$*****
 ******$$$$$$$$$$$$$$$$$$$$$$$$$******
  ******$$$$$$$$$$$$$$$$$$$$$$$******
  ******$$$$$$$$$$$$$$$$$$$$$$$******
   ******$$$$$$$$$$$$$$$$$$$$$******
    *******$$$$$$$$$$$$$$$$$*******
    ********$$$$$$$$$$$$$$$********
     *********$$$$$$$$$$$*********
      ***************************
        ***********************
         *********************
           *****************
             *************
```

<u>Sample Run #2</u>

```
Enter the outer radius of your wheel?
14
Enter the inner radius of your wheel?
6

        ***********
      ***************
     *****************
    *******************
   *********************
  ***********************
 *************************
 *************************
**********$$$$$$$**********
*********$$$$$$$$$*********
********$$$$$$$$$$$********
********$$$$$$$$$$$********
********$$$$$+$$$$$********
********$$$$+++$$$$********
********$$$$$+$$$$$********
********$$$$$$$$$$$********
********$$$$$$$$$$$********
*********$$$$$$$$$*********
**********$$$$$$$**********
 *************************
 *************************
  ***********************
   *********************
    *******************
     *****************
      ***************
        ***********
```

Problem C: Game for the Kids

Your car's computer is doing a great job keeping track of gas mileage and even has a neat feature that prints out a wheel, but it would be nice if the kids could use it in a constructive fashion. Write a program that drills the kids with multiplication problems. In particular, ask the user how many problems for the game, and then give the user that many random multiplication problems, where each number being multiplied is in between 0 and 12, inclusive. While the user is doing the problems, if they answer incorrectly, tell them so, and allow them to answer again. Do not move onto the next problem until they have correctly answered the current one. At the end of the game, when the user has correctly solved all the given multiplication problems, output the amount of time they spent.

How to calculate time spent for a segment of code in a C program:

In order to calculate how much time something takes, you can use the time function. In particular, the function call time(0) returns an int that represents the number of seconds after the birth of the Unix operating system. In order to effectively use this, you must call the function twice: once right before you start what you want to time, and once right afterwards. Subtract these two values to obtain the amount of time a segment of code took. Here is a short example:

```
int start = time(0);
// Insert code you want to time here.
int end = time(0);
int timespent = end - start;
printf("Your code took %d seconds.\n", timespent);
```

Input Specification

The number of problems to answer entered by the user will always be a positive integer less than 50.

Output Specification

For each correct response, simply move onto the next problem. For each incorrect response, output a message with the following format.

```
Incorrect, try again. AxB =
```

where A and B are the two numbers to multiply from the original problem.

After all the problems are completed, output a single line with the following format:

```
You completed X problems in Y seconds.
```

where X is the number of problems solved and Y is the number of seconds it took the user to solve them.

Output Samples
Here is one sample output of running the program. Note that this sample is NOT a comprehensive test. You should test your program with different data than is shown here based on the specifications given above. The user input is given in *italics* while the program output is in bold.

Sample Run #1
```
How many problems do you want?
5
Answer: 3x9 = 27

Answer: 4x6 = 42
Incorrect, try again.
Answer: 4x6 = 24

Answer: 12x11 = 132

Answer: 8x2 = 16

Answer: 7x5 = 35

You completed 5 problems in 17 seconds.
```

Deliverables
Three source files:

1) *realtimempg.c*, for your solution to Problem A
2) *wheel.c,* for your solution to Problem B
3) *multgame.c,* for your solution to Problem C

All files are to be submitted over WebCourses.

Restrictions
Although you may use other compilers, your program must compile and run using Dev C++. Each of your three programs should include a header comment with the following information: your name, course number, section number, assignment title, and date. Also, make sure you include comments throughout your code describing the major steps in solving the problem.

Grading Details
Your programs will be graded upon the following criteria:

1) Your correctness
2) Your programming style and use of white space. Even if you have a plan and your program works perfectly, if your programming style is poor or your use of white space is poor, you could get 10% or 15% deducted from your grade.
3) Compatibility to Dev C++ (in Windows). If your program does not compile in this environment, **the maximum credit you will receive is 50%.**