CS 101, Spring 2014 — May 8th — Exam 4

Note: For all of the programming questions, you should use `scanf` to read the input value(s) required by the program.

Note: Make sure your programs produce the output in **exactly** the format described, including capitalization and punctuation. You may not receive credit for programs that produce incorrectly-formatted output.

Getting started: Start **Cygwin Terminal** and **Notepad++**. (Note: do *not* open any other programs.) Your instructor will give you the name of a zip file. In Cygwin Terminal, run the following commands:

```
cd h:
mkdir -p CS101
cd CS101
curl -O http://faculty.ycp.edu/~dhovemey/spring2014/cs101/assign/zipfile
unzip zipfile
cd CS101_Exam4
```

Note that in the `curl` command, the `-O` has the letter 'O', not the digit '0'.

Substitute the name of the zip file for *zipfile*.

Editing code: Use Notepad++ to open the source file (e.g., `question1.cpp`) referred to in the question. Do not open any files other than the ones for the exam.

Compiling: To compile the program for Question 1, run the following command in Cygwin Terminal:

```
make question1.exe
```

Change the number as appropriate for the other questions (e.g., `question2.exe`).

Running: To run the program for Question 1, run the following command in Cygwin Terminal:

```
./question1.exe
```

Change the number as appropriate for the other questions (e.g., `question2.exe`).

To submit: In Cygwin Terminal, run the command

```
make submit
```

Enter your Marmoset username and password when prompted.

Good luck!

Question 1. [25 points] Complete the program in `question1.cpp` by writing a prototype and definition for the `swap_values` function. This function should take two parameters, each of which is a pointer to an `int` variable, and swap the values of the variables pointed to by the two pointers.

When you compile and run the program (`question1.exe`), the `main` function will prompt the user to enter two integer values, call the `swap_values` function to swap them, and then print out the updated values. (You should look at the `main` function, since it will help you understand what the `swap_values` function is intended to do.)

Example run (user input in **bold**):

```
Enter two integers: 17 42
After calling swap_values: 42 17
```

Important: Do *not* modify the `main` function in any way.

Question 2. [25 points] In the file `question2.cpp`, code is provided which declares two arrays, each with 10 integer elements, and prompts the user to enter values to store in each array.

Complete the program so that after the input is read from the user, the program prints the following lines of output:

1. The text `First:`, followed by the elements of the first array
2. The text `Second:`, followed by the elements of the second array
3. The text `Difference:`, followed by the elements in the first array are *not* present in the second array.

Example run (user input in **bold**):

```
Enter values for array 1: 1 2 3 4 5 6 7 8 9 10
Enter values for array 2: 2 3 1 9 5 6 4 5 6 4

First: 1 2 3 4 5 6 7 8 9 10
Second: 2 3 1 9 5 6 4 5 6 4
Difference: 7 8 10
```

Important: Make sure that each line of output is in *exactly* the format described above.

Hint: Start by determining whether array 2 contains the first element of array 1. Then, determine whether array 2 contains the second element of array 1. Can you generalize this process to check all elements of array 1?

Question 3. [25 points] The program `question3.cpp` defines the following struct types:

```
struct Point {
    double x, y;
};
```

```
struct Circle {
    struct Point center;
    double radius;
};
```

Write a definition of the `distance` and `is_inside` functions. They have the following prototypes:

```
double distance(const struct Point *p1, const struct Point *p2);
bool is_inside(const struct Point *p, const struct Circle *c);
```

The `distance` function should return the distance between the two `struct Point`s pointed to by the function's parameters. The distance between two points is given by the formula

$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

where x_1, y_1 and x_2, y_2 are the coordinates of the two points.

The `is_inside` function should return true if the `struct Point` pointed to by `p` is inside the `struct Circle` pointed to by `c`. A point is inside a circle if the distance between the point and the center of the circle is less than or equal to the radius of the circle.

A `main` function is provided. It reads the x/y coordinates of a point and the center and radius of a circle, storing them in `struct Point` and `struct Circle` variables. It then

- calls `distance` to compute the distance between the point and the center of the circle,
- calls `is_inside` to determine whether the point is inside the circle, and
- prints the results of the calls

Example run (user input in **bold**):

```
Point x/y: 1.2 2.5
Circle center x/y: 8.3 2.3
Circle radius: 7.9
distance returned 7.102816
is_inside returned true
```

Important: Don't modify the `main` function in any way.

Hint: You can use the `sqrt` function to compute the square root of a `double` value.

Hint: You may find it useful to use the `distance` function in your definition of the `is_inside` function. To do this, you will need to pass the address of the `center` field of the circle.

Question 4. [25 points] Complete the program in `question4.cpp` so that it uses the integer value entered by the user to print a figure consisting of `X` and `-` characters as shown in the examples below (user input in **bold**):

```
Enter size: 5
X-X-X
-X-X-
X-X-X
-X-X-
X-X-X

Enter size: 7
X-X-X-X
-X-X-X-
X-X-X-X
-X-X-X-
X-X-X-X
-X-X-X-
X-X-X-X
```

The program only needs to support input values which are both positive and odd.