CS 101, Spring 2014 — April 29th — Exam 3 Name: _____

Question 1. [5 points] What output is printed by the following program (which begins on the left and continues on the right)?

```
#include <stdio.h>

void f(int a[]) {
    a[0] = 33;
}
```

```
int main(void) {
    int b[2] = { 11, 22 };
    f(b);
    printf("%i %i\n", b[0], b[1]);
    return 0;
}
```

Question 2. [2 points] Circle **True** or **False**: If `arr` is an array with 10 elements, then `a[0]` is a valid element of the array.

Question 3. [2 points] Circle **True** or **False**: If `arr` is an array with 10 elements, then `a[10]` is a valid element of the array.

Question 4. [2 points] Circle **True** or **False**: It is possible to return an array from a function (as the return value of the function).

Question 5. [2 points] Circle **True** or **False**: If a function takes a one-dimensional array as a parameter, it is possible to call that function on an array with any number of elements, as long as the element type of the array matches.

Question 6. [2 points] Circle **True** or **False**: For a function to know how many elements an array passed as a parameter has, a separate integer parameter is required.

Question 7. [5 points] Consider the following struct type definition:

```
struct Rectangle {
    int x, y, width, height;
};
```

In the partially-specified code below, what code could be substituted for *HERE* so that the call to `printf` prints the area of the rectangle entered by the user (*i.e.*, the product of the rectangle's width and height)?

```
struct Rectangle r;
printf("Enter x, y, width, and height: ");
scanf("%i %i %i %i", &r.x, &r.y, &r.width, &r.height);

int area = HERE;
printf("Area is %i\n", area);
```

Question 8. [5 points] What output is printed by the following program (which begins on the left and continues on the right)?

```
#include <stdio.h>

struct Point {
    int x, y;
};

void transpose(struct Point p) {
    int tmp = p.x;
    p.x = p.y;
    p.y = tmp;
}
```

```
int main(void) {
    struct Point q;
    q.x = 3;
    q.y = 4;
    transpose(q);
    printf("%i %i\n", q.x, q.y);
    return 0;
}
```

Question 9. [10 points] A complex number has the form $a + bi$, where a is the *real* component, b is the *imaginary* component, and $i^2 = -1$. The *magnitude* of a complex number $a + bi$ is defined as $\sqrt{a^2 + b^2}$.

A struct type to represent complex numbers might be defined as:

```
struct Complex { double real, imag; };
```

The `real` field stores the real component of the complex number, and the `imag` field stores the imaginary component.

In the space below, write the definition of a function called `complex_magnitude` that takes a `struct Complex` parameter and returns the magnitude of that complex number as a `double`. Here is an example of how the function might be used:

```
struct Complex c;
c.real = 3.0;
c.imag = 4.0;
double magnitude = complex_magnitude(c);
printf("%lf\n", magnitude); // prints 5.000000
```

Your code should use the `sqrt` function to compute the square root of a `double` value.

[Write your definition for the `complex_magnitude` function below.]

Question 10. [5 points] What output is printed by the program below?

```
#include <stdio.h>

int main(void) {
    int a = 44, b = 55;
    int *p = &a;

    printf("%i\n", *p);
    p = &b;
    printf("%i\n", *p);

    return 0;
}
```

Question 11. [5 points] What output is printed by the program below?

```
#include <stdio.h>

int main(void) {
    int a = 44, b = 55;
    int *p = &a;

    printf("%i\n", a);
    *p = b;
    printf("%i\n", a);

    return 0;
}
```

Question 12. [5 points] What output is printed by the program below (which begins on the left and continues on the right)?

```
#include <stdio.h>

void f(int *p) {
    *p = *p * 3;
}
```

```
int main(void) {
    int a = 4;
    f(&a);
    printf("%i\n", a);
    return 0;
}
```

Programming Questions

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_Exam3
```

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., `question13.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 13, run the following command in Cygwin Terminal:

```
make question13.exe
```

Change the number as appropriate for the other questions (e.g., `question14.exe`).

Running: To run the program for Question 13, run the following command in Cygwin Terminal:

```
./question13.exe
```

Change the number as appropriate for the other questions (e.g., `question14.exe`).

To submit: In Cygwin Terminal, run the command

```
make submit
```

Enter your Marmoset username and password when prompted.

Good luck!

Question 13. [25 points] In the file `question13.cpp`, define the `first_occurrence` function. Its prototype is

```
int first_occurrence(double arr[], int num_elts, double sval);
```

The function takes an array of `double` values, `arr` (whose number of elements is specified as `num_elts`), and a specified "search" value `sval`. The function should return the index of the first element in the array whose value is equal to `sval`, or -1 if the array does not have any elements equal to `sval`.

You can test your implementation of the function by compiling and running `question13.exe`. It allows you to enter a series of `double values` and a search value, calls `first_occurrence`, and prints a message indicating what value was returned by `first_occurrence`.

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

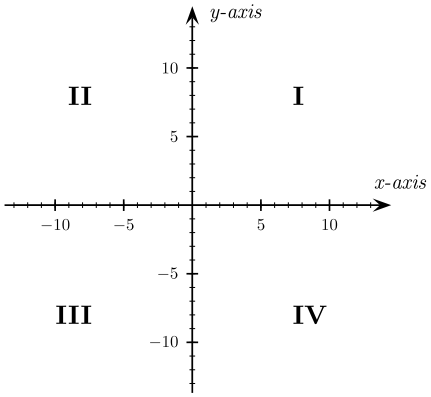
```
How many values? 10
Enter the values: 0.3 7.4 1.1 4.8 6.6 8.7 9.2 0.0 4.8 4.9
What search value? 4.8
first_occurrence returned 3

How many values? 6
Enter the values: 3.4 8.2 4.9 2.7 1.1 6.7
What search value? 9.8
first_occurrence returned -1
```

Important: Do not change any code in the `main` function.

Hints:

- Use a loop to check each element of the array to see if it is equal to the search value
- Make sure that the function only returns `-1` if it has checked every element of the array without finding the search value
- Make sure that if the array contains two or more occurrences of `sval` that the index of the *first* occurrence is returned

Question 14. [25 points] The quadrants of the x/y coordinate plane are traditionally numbered as follows (source: Wikipedia):



In the file `question14.cpp`, define the `find_quadrant` function. It has the following prototype:

```
int find_quadrant(struct Point p);
```

The function should return the quadrant that contains the point represented by the parameter `p`, which is a `struct Point`: 1 for quadrant I, 2 for quadrant II, 3 for quadrant III, and 4 for quadrant IV. As a special case, the function should return 0 if `p` is the origin ($x = 0$ and $y = 0$.)

The `struct Point` type is defined as follows: `struct Point { double x, y; };`

You can test your implementation of the function by compiling and running `question14.exe`. The program prompts the user to enter x and y coordinates (as `doubles`), and then prints the result returned by `find_quadrant`.

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

```
Enter x and y: -1.3 2.6
find_quadrant returned 2

Enter x and y: 4.7 -11.4
find_quadrant returned 4

Enter x and y: 0.0 0.0
find_quadrant returned 0
```

Important: Do not change any code in the `main` function.

Hints:

- You might find nested `if/else` statements useful