Assignment 6

FUNCTIONS AGAIN

1- Read Chapter 7, pages 384-411

2- Go to the book's site on this topic, click on and read the overview and take the self-test , at

http://oc.course.com/computerscience/malikcpp4e/index.cfm?page=chapter-selftests&chapter=7

3. Answer the following questions based on your readings, google searches and reflection about the topic of functions:

Given the code answer the questions:

```
1.  #include <iostream>
2.  using namespace std;
3.
4.
5.  int testing (int a, float b);
6.
7.   int main()
8.  {
9.     int x;
10.    float y;
11.    cout << "Enter an integer and a decimal number"<< endl;
12.    cin >> x >> y;
13.    cout << testing (x, y);
14.   return 0;
15.   }
16.
17.   int testing (int c, float d)
18.    {
19.      //if d is greater than c, return the integer part of d, else return c
20.
21.       if (d > c)
22.         return (int) d;
23.       else
24.         return c;
25.  }
```

Q 1. Is line 5 a call to function **testing**, a definition of function **testing** or the prototype of function **testing**?

Q2. What are the names of the parameters to fucntion **testing**? remember the parameters are the ones that appear in the definition, not in the prototype

Q3. If we change the names of the parameters, will function **testing** still work? That is, if we rewrite the code and replace them throughout the function with the names number1 and number2, would the computation and results be the same? Try it

Q4. Is d inside the function the same variable as (another name for) the caling variable y, or not? In

other words, if we assigned a value to d inside the function, at the end of the execution of test, would the value of y be changed too?

Q5. Given the functions below, explain the difference in behavior when these two functions are called using the concept of **calling by-reference** vs **calling by-value**:

```
#include <iostream>
using namespace std;


void change (int & val) {

        val = 27;
    }
```

and

```
  void change2 (int val) {
     val = 27;
    }


       int main() {

        int x = 99;

         change2(x);

         cout << "After calling change2() , the value of x is " << x << endl;

        change(x) ;

        cout << "After calling change() , the value of x is " << x << endl;

        return 0;
```

Q6. Using the notion of **scope** explain what instances of variable x are displayed AND WHY.

```
   #include <iostream>
   using namepsace std;

  int x = 12;

   int main() {

     int x = 9;

      { int x = 10

    cout << x << endl;
       }
```

```
 cout << x << endl;
   return 0;
}
```