CS 271 Lab #4
Working with Functions and Arrays

Programs you submit for this assignment must compile on the Linux systems in SH 118B and SH 118 using the standard gcc compiler.

Programs that do not compile will receive a grade of zero.

Documentation and Style:

- Header comments and inline comments are required.
- Your program must adhere to the course style guidelines for naming, indentation, and spacing, etc.

Before you begin working on this assignment, read the Chapter 6 Notes (in the Canvas Modules).

Then read the Notes on Random Number Generation. This material is in chapter 5 of the textbook, also.

lab4.h

- Create a file called lab4.h.
- **If you're using Kate, save the file before you start typing code.**
- Insert a preprocessor wrapper to prevent this header file from being included twice.
- Insert the prototypes for the following functions:

```
int linearSearch( int array[ ], int arraySize, int key );
```

If key is found in the array, returns the index. If it is not found, returns -1.

```
int binarySearch ( int array [ ], int arraySize, int key );
```

If key is found in the array, returns the index. If it is not found, returns -1.

```
void printIntArray ( int array [ ], int arraySize );
```

Print the elements of the array, each element should be right-justified using a field width of 5. Print 10 elements per line. (The last line may not have 10 elements.)

```
void bubbleSort ( int array[ ] , int arraySize );
```

Sort the array in non-descending order.

lab4.c

- Create a file called lab4.c.
- **If you're using Kate, save the file before you start typing code.**
- Write the implementations of the functions listed above.
- The description on the previous page should be sufficient explanation for linearSearch, binarySearch, and printIntArray.

  For the bubbleSort function, you must use the bubble sort algorithm. Here's the pseudocode:

```
BubbleSort( int a[], int n)

   for i = 1 to n-1
      sorted = true
      for j = 0 to n-1-i
         if a[j] > a[j+1]
            temp = a[j]
            a[j] = a[j+1]
            a[j+1] = temp
            sorted = false
         end if
      end for
      *** you can terminate the i loop if sorted is still true
          when it reaches this point
   end for
```

test4.c

- Create a file called test4.c.
- **If you're using Kate, save the file before you start typing code.**
- Write a main function to test the functions above. Your main method must do the following:
  a) Declare an array of 100 integers.
  b) Seed the random number generator with the time function.
  c) Fill the array with random integers between 1 and 1000.
  d) Print the array using the printIntArray function.
  e) Use linearSearch to search for 30, 86, and 87.
  f) Use bubble sort to sort the array.
  g) Print the array using the printIntArray function.
  h) Use binarySearch to search for 11, 28, 74, and 99.

makefile

- Create a file called makefile.
- **If you're using Kate, save the file before you start typing code.**
- Create a makefile to compile test4.c and lab4.c (these should be separate commands), then link the program components into an executable called test4. You must have an "all" target that produces the executable.

Test thoroughly and debug as needed to make sure that the functions work correctly.