# CSC 225

**Homework 2**

Due: Wednesday, November 9, 2022 at 11:59pm

# Instructions

|  |
| --- |
| 1. **You must work on this assignment individually.** You may ask another person to look over your project for errors, but they cannot instruct you the correct way to write. (For example: the person helping can say “The problem is in the ‘switch’ statement” or “Your logic is incorrect in the second ‘if’ statement.”) In addition, you may not use code from other sources, such as the Web or other students. 2. You will need to upload the .cpp and .h files from your solution to eLearn. |

Write a complete C++ program that analyzes the following sorts by keeping track of how long they the take to sort arrays of ints of size 5000, 10000, and 100000, as well as the number of swaps and comparisons they take in doing so:

* Insertion sort
* Selection sort
* Quicksort
* Shell sort (using gaps of n/2, n/4, n/8, ..., 1)
  + For 20, the gaps would be 10, 5, 2, and 1
* Shell sort (using gaps based on 2k - 1)
  + For 20, the gaps would be 15, 7, 3, and 1
    - These will be easiest to generate in increasing order, and then reversing the array

You should manually complete a table in Word or Excel showing how long each algorithm takes to sort a random array of each of the above sizes, as well as the number of comparisons and swaps performed.  The numbers in your table should be an average of at least three runs of your program for each size array.  The program itself does not need to generate the table or the averages.

Specifications:

* The program should ask the user the size of the array to be sorted; it could be any valid number up to the maximum, not just those specified for testing
* Your gap arrays will not need to hold more than 20 elements
* When generating random numbers for the arrays, just use rand(); do not use %
* The only things that should be printed are the times, swaps, and comparisons for each sort
* Use the same numbers for each sort.  To do this, keep a master array and then copy the values from the master array to the array to be sorted before you run each sort
  + Do not have more than two large arrays
* In order to keep track of the numbers of swaps and comparisons, you will need to do the following:
  + Declare a structure that contains two variables of type long long (2 words) that will store the swaps and comparisons
    - You will need five variables of this type declared in your main
  + Pass a structure as a reference parameter to each of your sort functions
  + Increment the variables within the structures when appropriate within your sort functions
    - This is similar to what you did in the Binary Search Lab in section 12.4
* You will also need a function that is passed a structure by reference and initializes both variables within the structure to zero

In addition, answer the following questions in the document that also contains your table:

1. Which sort was the fastest?
2. Which sort was the slowest?
3. Which had the most swaps?
4. Which had the most comparisons?
5. Which had the fewest swaps?
6. Which had the fewest comparisons?
7. Which gap selection method of Shell sort had more swaps and comparisons?
8. Which gap selection method of Shell sort was faster?
9. Explain why your answers to 7 and 8 could be the way they are.
10. Use the times from your Insertion and Selection sorts to demonstrate that they are indeed O(N2).  Use the relationship between the factor the array size increases by and the factor the time increases by.

In addition to the program functioning correctly as described above, you will also be graded on:

* Using appropriate variable names and data types
  + Limit globals
  + Use constants appropriately
  + Name constants using all capital letters
* Using function prototypes
* Using comments appropriately, including header comments
* Formatting your code appropriately using tabs and blank lines
* Having user-friendly and well-formatted input and output
* Appropriate data validation

Additional information:

* You should only write one shell sort function, but you will need to generate two different gap arrays, and call the sort function twice, once with each array
* Both gap arrays can be created using while loops.
  + Each loop will need a variable for the location the gap value should go in the gap array. This variable will be incremented each time through the loop. It will be useful as the number of gaps when calling shell sort.
  + For n/2:
    - You will need a variable for the gap value, which will be divided by two each time through the loop.
  + For powers of 2:
    - Generate it in reverse (ascending), and then reverse the array
    - You will need to initialize the first value in the array to1since it does not fit the pattern
    - You will need a variable for the power of two, which will be multiplied by two each time through the loop.
    - To reverse the array, use a for loop that swaps ends of the array. So, if there were seven elements in the array, it would swap locations
      * 0 and 6
      * 1 and 5
      * 2 and 4
* Make sure you understand the above requirements before you begin. Dr. Thompson can provide any clarifications you need.