# COP3530 – Data Structures and Algorithms I - Fall 2016

## Programming Project#1

*** Due 09/21/2016, Wednesday, 05:00 p.m.**

*Instructions:*

*This is a programming project to be done and submitted individually. Please make sure your solutions are original, in your own words, expressing your own ideas. Any copying from others' home-works, text books or online sources will be penalized. I'll be checking the code for plagiarism, so, be advised.*

*IMPORTANT: You can use any IDE for creating your programs. Create the programs and save them. Compile and run the programs. Save the screenshot/s of the successfully run programs and submit the screenshot/s with the corresponding source code files on the Dropbox. Do not submit any executable (.exe) files.*

*Programs should be well-commented to make their understanding easy. Variable-names should be so chosen that it is easy to visualize what is going on, inside the program. Points would be taken off for poor documentation and readability.*

*It is highly recommended that students keep a soft copy of the assignment files with them. Any queries can be mailed to amishra@uwf.edu.*

*Any exceptions to this should be pre-approved by the Instructor; late submissions (only the ones pre-approved by the instructor before the deadline) are liable to be penalized.*

Previous instructions about coding submissions stay as they are.

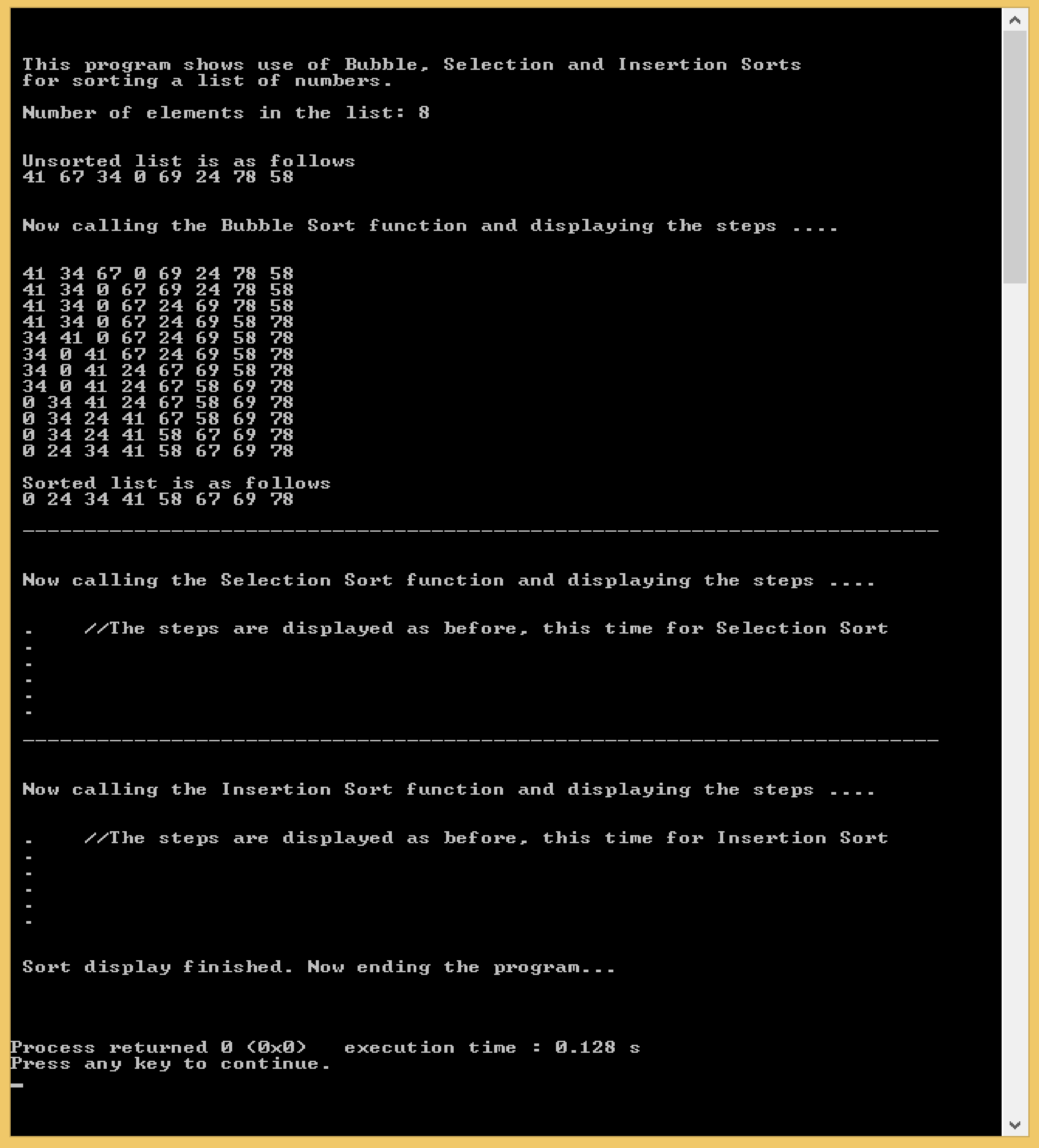
**WHAT TO TURN IN**:

Upload through the "Assignments" page on Dropbox, all the files in your submission. Submit **each** of your C programs: one program for each of the above exercises. In addition, submit **all relevant screen captures of your successfully running programs** as jpeg files using the same name (for example, the screen captures for successfully running *sumOfCubes.c* should be named as *SumOfCubes1.jpg*, *SumOfCubes2.jpg* – if more than one relevant screen captures are included). Do not submit the object files or executables.

**In order for your submission to be counted as on time, it should be submitted before the indicated deadline on the due date unless otherwise specified. The penalty for late submissions is 5% for each day late, and late submissions need to be preapproved by the instructor (they should NOT be automatically assumed as granted). The very latest that you may submit a late lab is one week after the submission was due.**

1. Write a complete C program with a function to perform Bubblesort, Selection sort and Insertion sort on a list of numbers. You should generate the list inside the main( ) function and pass it on to the different sort functions for comparison, one by one. When a sort function runs, display the important steps (just like in the examples we did in the class). After each sort function has sorted the list, the control transfers back to the main( ) function and the sorted list is printed out from within main( ). The algorithm should also be able to handle duplicate elements. (**5 points x 3 for the three algorithms, 5 points for displaying the steps correctly = Total 20 points**)

   Make your program output to look like the following sample run (try out different sized lists):



```


 This program shows use of Bubble, Selection and Insertion Sorts
 for sorting a list of numbers.

 Number of elements in the list: 8


 Unsorted list is as follows
 41 67 34 0 69 24 78 58


 Now calling the Bubble Sort function and displaying the steps ....


 41 34 67 0 69 24 78 58
 41 34 0 67 69 24 78 58
 41 34 0 67 24 69 78 58
 41 34 0 67 24 69 58 78
 34 41 0 67 24 69 58 78
 34 0 41 67 24 69 58 78
 34 0 41 24 67 69 58 78
 34 0 41 24 67 58 69 78
 0 34 41 24 67 58 69 78
 0 34 24 41 67 58 69 78
 0 34 24 41 58 67 69 78
 0 24 34 41 58 67 69 78

 Sorted list is as follows
 0 24 34 41 58 67 69 78
 ___________________________________________________________________________


 Now calling the Selection Sort function and displaying the steps ....


 .    //The steps are displayed as before, this time for Selection Sort
 .
 .
 .
 .
 .

 ___________________________________________________________________________


 Now calling the Insertion Sort function and displaying the steps ....


 .    //The steps are displayed as before, this time for Insertion Sort
 .
 .
 .
 .
 .


 Sort display finished. Now ending the program...



Process returned 0 (0x0)   execution time : 0.128 s
Press any key to continue.
_
```

***