CS 3130 Design and Analysis of Algorithms Fall 2015

Programming Project #1 [100 points].

Due date: Monday, September 28.

Part A (some elementary sorting algorithms).

1. [25 points] Write a program to implement four sorting algorithms: (a) bubble sort with swaps counting, (b) cocktail (or shaker) sort with swaps counting, (c) and (d) two versions of Shell sort (with two different sequences of "gaps").
   Each algorithm must be implemented as a function. Run your program for an array containing 100 random integers. Show the original and sorted arrays in your output.
   REMEMBER to display arrays in the form of a square table with aligned columns!

2. [15 points] Write a program to compare efficiency of four sorting algorithms from part (1). To do that, you will have to use some function to measure the time of execution of a code fragment. Apply each sorting method to 10 different arrays of random integers of each of the following sizes: 500, 1000, 2500 and 5000. The output of the program is average execution times for all sizes.

3. [20 points] In this part of the project you are supposed to analyze the results of parts (A1) and (A2). Show the average running times of all four algorithms in form of the tables and in the graphic format as functions of the input size. Explain how your results correspond to the theoretical estimates.

Part B (elementary searching algorithms).

1. [10 points] Write a program to implement a linear search and binary search algorithms. As input arrays, use sorted arrays that are an output from part (A) of your project. Search for a random number that is between the smallest and biggest value of your sorted array. The output in this part is one sorted array, the key value you were searching for, and the result of your search using both algorithms. You are NOT supposed to take an advantage of an array to be sorted when implementing a linear search!

2. [15 points] Write a program to compare efficiency of two searching methods. Apply each of them to 10 sorted arrays of random integers of each of the following sizes: 500, 1000, 2500 and 5000. The output of the program is average execution times for all sizes.

3. [15 points] In this part of the project you are supposed to analyze the results of parts (B1) and (B2). Show the average running times of both algorithms in form of tables and in the graphic format as functions of the input size. Explain how your results correspond to the theoretical estimates.

   ATTENTION:

   Parts (A1), (A2), (B1), (B2) of this project must be submitted electronically (instructions will be posted on My Gateway later). Parts (A3) and (B3) must be submitted in person in the class, typed or ACCURATELY written. I will not grade messy writing!