Algorithms and data structures

Labwork 5: Searching and Sorting Algorithms

Dr. Doan Nhat Quang: doan-nhat.quang@usth.edu.vn

1. Searching and Sorting

Please use Google classroom for labwork submission; join the class by the following link: https://classroom.google.com/c/NTQ1NjMwOTMwNzU0?cjc=gqtgfva

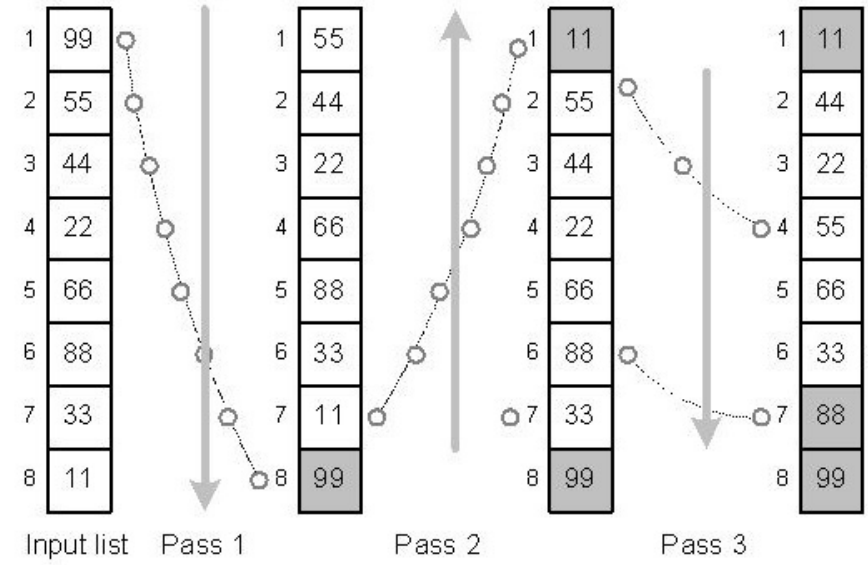
There are 6 Labworks in this course. After each lab work session:

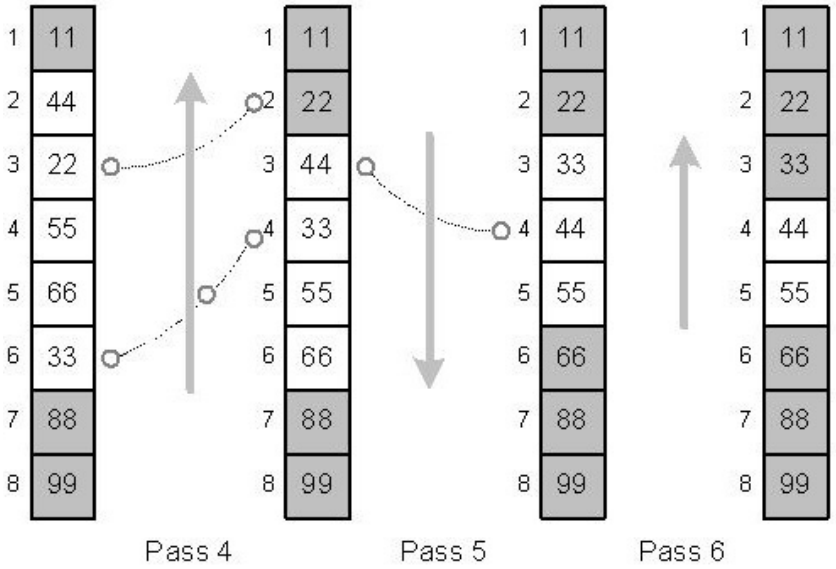
- You need to complete one of the given exercises and upload your files to the assignment "Labwork No. - Group No. - Version 1 (Attendance)". Submission must be done within 30 mins after the lab; otherwise, it will be considered a late submission.
- You will have one week (or 7 days) to complete the remaining exercises and upload your files to the assignment "Labwork No. - Group No. - Version 2 (Complete)"
- Compress all code source files in a zip file and rename it as FULLNAME-ID-Lab#no.zip (e.g NguyenVanA-070-Lab1.zip). Save your files according to the exercise number i.e Ex1.cpp, Ex2.c, etc. Incorrect filenames will result in no score for the respective exercises.
- Only code source files (.c or .cpp) should be in the zip files. Other files (.exe, .o) MUST be removed from the zip file.
- Copy/Paste from any source is not tolerated. The penalty will be applied for a late submission.

NOTE: You must follow the guide. Incorrect zip file names, zip files containing other files (.exe), and copy/pasting lead to heavy penalties.

Exercise 1:
In this problem, we would like to implement a variation of the Insertion Sort algorithm. The algorithm differs from a bubble sort in that it sorts in both directions on each pass through the list. The algorithm is illustrated in the following figure:

- For the first step, we perform insertion sort from the index 1 to n (n is the number of elements in the array).





- In the next pass, we perform a reserved bubble sort from the index n to 1.
- The process is repeated until all the array is sorted.

Propose a pseudo-code to complete the Insertion Sort algorithm. Implement and test this algorithm in C/C++. Analyze and compute the complexity of this algorithm in the best, average and worst scenarios.

Exercise 2:

Re-implement Exercise 1 using a linear data structure: List, Stack, Queue. Justify your choice of data structure.