**GC UNIVERSITY LAHORE**

**Department of Computer Science**

**DSA Mid Programming Tasks Part-1**

Note: Tasks are NOT TO BE SHARED (copied) from/to anyone. Even TRY NOT to discuss your solution idea with any other student. You can discuss teacher for clue on any problem, but solution MUST be your own. Such approach is very important if you want to get full benefit from DSA course and Lab.

When developing solution, Do not mix tasks to avoid confusion. Try to concentrate on ONE TASK ONLY. Some tasks depend on solution of previous tasks. If you cannot solve one task, leave it for some time and then start again.

Do as much paper work, as possible.

All tasks are to be developed in VC++. Develop separate project for each task or sub-tasks.

1. Write a program that performs and compares all three sequential sorts i.e Bubble sort, selection sort and insertion sort. Use array sizes of 1M, 100M, 400M, and higher sizes. Use counter variable to count number of external and internal loop iterations, number of comparisons, number of swaps and shifts etc. Also use time header file to calculate actual number of milliseconds consumed in sorting each array with different sorts. Then compare which sort is best. Use large random number arrays. Also find the limitations on array sizes and faults of random number generator.
2. Write a program, that finds min, second min, max, second max, mean, median, mode of very large random array. Write program to find if sum of any two elements Ai and Aj also lies in Array A. i.e. Find Ak such that Ak = Ai + Aj.
3. Write program to find any two numbers Ai and Aj such their sum Ai + Aj is minimum of all such other pairs . (without sorting)
4. Write a program that finds all duplicates in an array (without sorting). It also prints number (frequency) of each repeated element. Use counters to find number of iterations of such program. Can you find some efficient solution to find frequency of elements in a large array.
5. Write a program that prints all unique elements of an array i.e. those elements which appear in array only once. (no sorting)
6. Write a program that finds and prints common elements of two different random arrays A and B, and also stores them in another array C.
7. A and B are two sorted arrays. Ranges of both arrays overlaps (one array may start from 40, 45, 46, . .55,….61,, . . 90, Other array may start from 10, 12, 13, .55, ….61….. 69, 70, 71 etc). There are some common elements in both arrays. There may be element repetition within an array. Write a program that efficiently inserts-inorder all elements of Array B into array A.
8. A and B are two sorted arrays. Range of both arrays overlaps. There are some common elements in both arrays. Write a program that combines both arrays into a single array C in order. Common element is selected only once.
9. Write a program that finds median of an array. Then median is swapped with middle index of array. Then arranges all other elements around median such that all smaller elements shifted/swapped before median and all larger elements shifted/swapped after median.
10. We use random number generator provided in C standard library, but rand and srand functions have their inner faults, like semi-sequence, repetition and range issues. Find some other random number generator from C library and internet and use them. Then Develop your own random number generator which is more reliable. You can use already supplied random numbers (Hint: One method is to generate random numbers from rand, but place them on random positions in array, instead of loop. Another method is to randomly shuffle any two index of array. You can test these hints to develop more random number schemes).
11. Develop and show complete classes for following General data structures. Your classes should support all types of insert, (insertlast, insertfirst, insertafter, insertbefore, insertInorder, etc) and all types of delete, search and modify operations (According to nature of that structure)
    1. General flexible array.
    2. General ordered flexible array.
    3. Single, double and circular linked lists.
    4. Stack as fixed array and as singly, double and circular linked lists.
    5. Simple Queue, Circular Queue, Priority Queue, Double Ended Queue, Double Ended Circular Priority Queue( All type to be implemented in Fixed and Flexible array, Single and Double lists).
12. Develop a simple stack and use it manually to play Tower of Hanoi game. Your game will ask number of plates in initial stack. Then in a loop, you will enter which stack to pop and plate number will be printed on screen. Then you will decide to push that plate number into which stack. Your program will print contents of all three stacks after each step.
13. Develop a pre-compiler syntax checker, that opens a .cpp code file, and checks if parenthesization is correct or not. It uses a stack internally, as discussed in class.
14. Your stack contains some random numbers. Using another empty stack and ONE or TWO temporary variables, develop an algorithm that sorts initial stack – but without sorting-. Only push and pop functions can be used. Only temporary variables can be compared. You can use peek( ) function for some simplification.
15. Develop an automatic, program that solves tower of Hanoi puzzle, by generating stacks push and pop statements and performing them automatically.
16. Develop a complete Matrix class. The class should comprise of a dynamic 2D array and should support common matrix operations, like input, output, matrix addition, subtraction, multiplication, transpose, determinant etc. Class should also check some basic features of matrix. i.e square or non-square matrix, upper triangular, lower triangular, singular or non singular, symmetric, non-symmetric or skew symmetric. Compatibility of two matrices for multiplication. Comparison of two matrices for equality, or approximate-equality. Check multiplicative associativity, commutativity and distributive properties of matrices. i.e. apply operations on both sides of property laws and check if answer is same or not.

This is first part of Pre-Mid tasks. Submission and Deadline: 12 December 2022.

Part-2 will be assigned soon.

**Saeed ur Rehman Turk**

(Assistant Professor)