National Institute of Technology Calicut
Department of Computer Science and Engineering
Third Semester B. Tech.(CSE)
CS2092D Programming Laboratory
Assignment 4

Submission deadline (on or before):

- 03.09.2023, 11:00 AM

Policies for Submission and Evaluation:

- You must submit your assignment in the Eduserver course page, on or before the submission deadline.
- Ensure that your programs will compile and execute without errors in the Linux platform.
- During the evaluation, failure to execute programs without compilation errors may lead to zero marks for that evaluation.
- Detection of ANY malpractice related to the lab course can lead to awarding an F grade in the course.

Naming Conventions for Submission

- Submit a single ZIP (.zip) file (do not submit in any other archived formats like .rar, .tar, .gz). The name of this file must be

  `ASSG<NUMBER>_<ROLLNO>_<FIRST-NAME>.zip`

  (Example: *ASSG*1_*BxxyyyyCS_LAXMAN.zip*). DO NOT add any other files (like temporary files, input files, etc.) except your source code, into the zip archive.

- The source codes must be named as

  `ASSG<NUMBER>_<ROLLNO>_<FIRST-NAME>_<PROGRAM-NUMBER>.c`

  (For example: *ASSG*1_*BxxyyyyCS_LAXMAN*_1.*c*). If you do not conform to the above naming conventions, your submission might not be recognized by our automated tools, and hence will lead to a score of 0 marks for the submission. So, make sure that you follow the naming conventions.

Standard of Conduct

- Violation of academic integrity will be severely penalized. Each student is expected to adhere to high standards of ethical conduct, especially those related to cheating and plagiarism. Any submitted work MUST BE an individual effort. Any academic dishonesty will result in zero marks in the corresponding exam or evaluation and will be reported to the department council for record keeping and for permission to assign F grade in the course. The department policy on academic integrity can be found at: http://cse.nitc.ac.in/sites/default/files/Academic-Integrity_new.pdf.

General Instructions

- Programs should be written in C language and compiled using C compiler in Linux platform. **Submit the solutions to questions from 1 to 4 through the submission link in Eduserver**.

QUESTIONS

1. Given an array, Arr of n positive real numbers between 0 and 1 with 2-digit precision (ex: 0.32, 0.45, 0.98, etc.), Use the **Bucket Sort** algorithm to sort the given numbers in ascending order.

   Input Format:

   - First line contains an integer indicating the number of elements $n \in [1, 100]$
   - Second line contains n elements $Arr[i] \in [0, 1]$ with 2 digit precision, separated by 1 space

   Output Format:

   - A single line containing the elements of Arr in sorted order with 2-digit precision (Use %.2f while printing the elements)

   Sample Input 1:
   10
   0.23 0.58 0.66 0.32 0.64 0.20 0.11 0.55 0.98 0.22

   Sample Output 1:
   0.11 0.20 0.22 0.23 0.32 0.55 0.58 0.64 0.66 0.98

2. Given an array, Arr of positive numbers of size n, Use the **Radix Sort** algorithm to sort the given numbers in ascending order.

   Input Format:

   - First line contains an integer n indicating the number of elements in the array Arr. ($1 <= n <= 10^3$).
   - Second line contains the elements in the array Arr. The elements take the values in the range $[0 - 10^4]$.

   Output Format:

   - Numbers in the array Arr in ascending order.

   Sample Input 1:
   4
   1 9 345 2

   Sample Output 1:
   1 2 9 345

3. Given a character array of size N, sort the elements of the array in descending order using **Quick Sort** and print the sorted array. Sort the characters on the basis of their ASCII value.

   Input Format:

   - First line consist of an integer $N \in [1, 1000]$ denoting size of array.
   - Second line consists of a sequence of character array elements separated by 1 space each. Array elements are English alphabets in the range $[a - z, A - Z]$.

   Output Format:

   - Single line containing elements of array sorted in descending order.

   Sample Input 1:
   4
   a c d b

Sample Output 1:
d c b a

Sample Input 2:
6
a B q T i j

Sample Output 2:
q j i a T B

4. As a renowned scientist, you've developed a cutting-edge robot that excels not only in technology but also in understanding and implementing complex tasks. Having completed your work on the robot's construction, you're now focused on assessing its capabilities and efficiency. Your expertise in sorting algorithms has led you to devise a unique challenge for your robot, one that involves both sorting and energy conservation.
Task Description:
Your robot has been assigned the task of arranging a collection of books in a specific order while minimizing energy consumption. The arrangement is to follow a precise pattern: the smallest book should be placed first, followed by the largest book, then the second smallest, and so on. The robot is equipped with a limited battery capacity, and every time it arranges/swaps a book with another book, it consumes one unit of energy. Once it runs out of energy it stops. When the books are swapped at least one of the books gets placed in the correct position. The robot's primary operation involves swapping two books to achieve the desired arrangement. The swapping process is designed to maintain the order of the previously correctly arranged books, ensuring that no other books' correct arrangement is disturbed. This task is not only a test of the robot's sorting capabilities but also a demonstration of its energy-efficient decision-making.

Input Format:

- The first line contains an integer $N \in [1, 10^4]$, indicating the total number of books.
- The second line contains a list of integers representing the number of pages a book has $Pages \in [1, 10^6]$
- The third line contains an integer $Energy \in [1, 10^4]$, indicating energy capacity.

Output Format:

- Single line of an integer representing final arrangement by the robot.

Sample Input 1:
9
52 3 4 654 6 467 57 4 2
4

Sample Output 1:
2 654 3 467 6 4 57 4 52

Sample Input 2:
4
1 1 2 4
2

Sample Output 2:
1 4 1 2