# Department of Computing

# School of Electrical Engineering and Computer Science

**CS-250: Data Structure and Algorithms**

**Class: BESE** **13A**

**Lab 10:  Open Ended Lab**

**Name:** Aimen Munawar

**Class:** BESE-13-A

**CMS ID:** 415867

**Date: 1st December, 2023**

**Time: 10:00 am – 12:50 am**

# Lab Engineer: Anum Asif

# Lab 10: Open Ended Lab

**Introduction**

This is an open ended lab where you have to choose your own unique dataset and implement your own methodology based on your understanding of the problem.

**Objectives**

Objective of this open ended lab is to implement a problem using the methodology of your choice.

**Tools/Software Requirement**

VisualStudioC++

**Helping Material**

Lecture slides, text book

**Description**

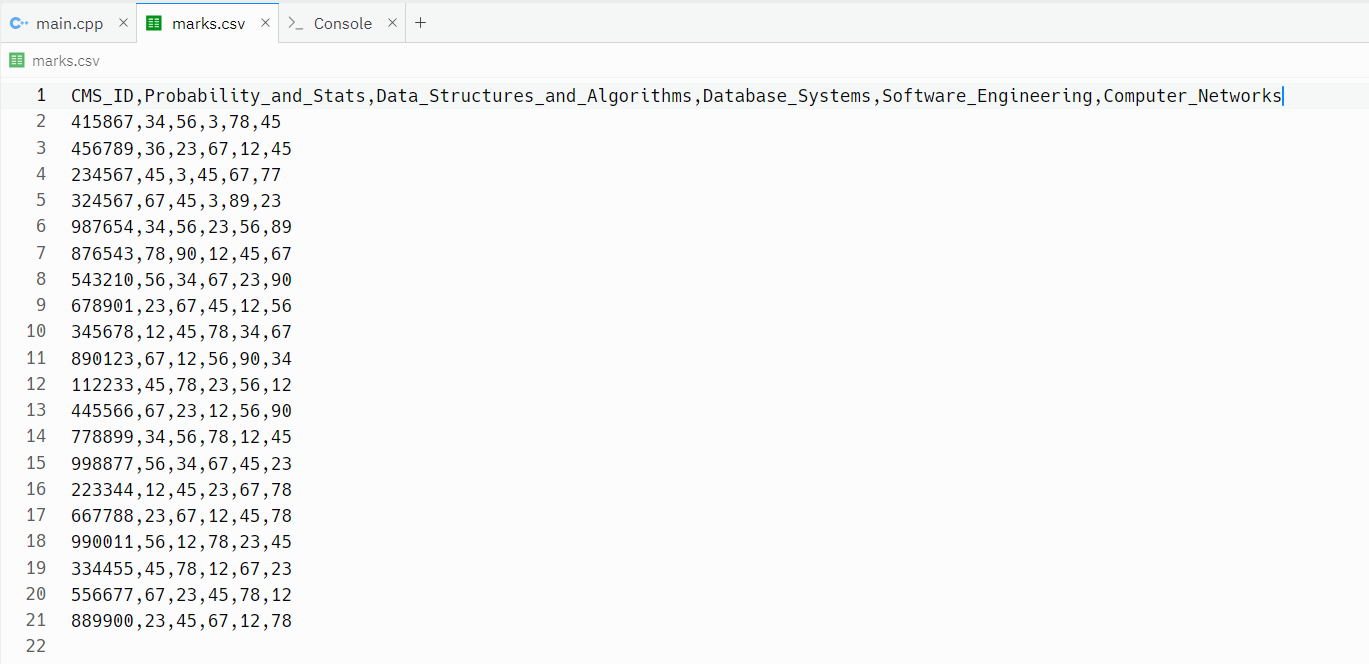
The basic idea of the task is that you have to take the grades of the courses you took in previous semester. The grades should be taken for a number of students, means for every individual course e.g., OOP, the list of grades will be created for a total of e.g., 40 students. Then you have to sort each individual list of grades, corresponding to each course. Once the individual lists are sorted, you have to merge all the lists.

**Lab Tasks**

Implement the problem described above. Make sure to use your own and unique dataset and sorting algorithm. An appropriate choice of data structure to store grades is required. Also, efficient and appropriate sorting algorithm is to be used for sorting purpose.

**Note:** Mention the reasoning for the choice of data structure and sorting algorithm used.

**Dataset:** For this implementation, a CSV file named marks.csv contains student CMS IDs and their corresponding marks in five subjects: Probability and Stats, Data Structure and Algorithm, Database Systems, Software Engineering, and Computer Networks.



**Code Implementation:**

#include <algorithm>

#include <fstream>

#include <iostream>

#include <sstream>

#include <vector>

using namespace std;

// Function to perform Quick Sort on a vector

void performQuickSort(vector<int> &data, int low, int high) {

if (low < high) {

int pivot = data[high];

int i = low - 1;

// Partitioning step

for (int j = low; j <= high - 1; j++) {

if (data[j] < pivot) {

i++;

swap(data[i], data[j]);

}

}

swap(data[i + 1], data[high]);

// Recursively sort the elements

performQuickSort(data, low, i);

performQuickSort(data, i + 2, high);

}

}

// Function to read data from CSV file, sort, and display the results

void processAndSortData() {

ifstream inputFile("marks.csv");

// Check if the file is opened successfully

if (!inputFile.is\_open()) {

cout << "Error opening file!" << endl;

return;

}

// Vector to store marks for each subject

vector<vector<int>> subjectWiseMarks(5);

string line, cmsID, marks;

// Read the first line to skip headers

getline(inputFile, line);

// Read data from CSV file

while (getline(inputFile, line)) {

stringstream ss(line);

getline(ss, cmsID, ',');

// Loop through each subject's marks

for (int i = 0; i < 5; i++) {

getline(ss, marks, ',');

try {

subjectWiseMarks[i].push\_back(stoi(marks));

} catch (const invalid\_argument &e) {

cerr << "Invalid argument: " << e.what() << endl;

// Handle invalid argument (non-integer value)

}

}

}

inputFile.close();

// Process and display sorted marks for each subject

for (int i = 0; i < 5; i++) {

cout << "Sorted Marks for Subject " << i + 1 << ": ";

performQuickSort(subjectWiseMarks[i], 0, subjectWiseMarks[i].size() - 1);

for (int mark : subjectWiseMarks[i]) {

cout << mark << " ";

}

cout << endl;

}

// Merge and sort marks into a single list

vector<int> mergedAndSortedMarks;

for (int i = 0; i < 5; i++) {

mergedAndSortedMarks.insert(mergedAndSortedMarks.end(),

subjectWiseMarks[i].begin(),

subjectWiseMarks[i].end());

}

// Sort the merged list

performQuickSort(mergedAndSortedMarks, 0, mergedAndSortedMarks.size() - 1);

// Display merged and sorted marks for all subjects

cout << "\nMerged and Sorted Marks for All Subjects: ";

for (int mark : mergedAndSortedMarks) {

cout << mark << " ";

}

cout << endl;

}

int main() {

// Call the function to process and sort data

processAndSortData();

return 0;

}

**A group of small black and white letters

Description automatically generated with medium confidenceOutput:**

**Reasoning for Data Structure and Sorting Algorithm:**

* **Data Structure:** A vector of vectors (vector<vector<int>>) is used to store the marks for each subject separately. This allows for easy organization and manipulation of data for each subject individually.
* **Sorting Algorithm:** Quick Sort is chosen as the sorting algorithm due to its efficiency and effectiveness for relatively small datasets. Quick Sort has an average-case time complexity of O(n log n), making it suitable for sorting the marks of students in each subject. The in-place sorting nature of Quick Sort also helps conserve memory.

**Deliverables**

Insert the solution/answer in this document. You must show the implementation of the tasks in the designing tool, along with your complete Word document to get your work graded. You must also submit this Word document on the LMS.

**Note:** Students are required to upload the lab on LMS before deadline.

Use proper indentation and comments. Lack of comments and indentation will result in deduction of marks.