Data Structures and Algorithms

Lab 3: Fundamentals

Q1): Write a program that:

Declare a structure named ‘student.’ Each student should contain a name and marks of three courses, namely, {Math, Chem, Physics}.

Make an array of pointers, where the size of the array is 10. Each pointer should be able to point to a ‘student.’

Calls out a function, that initializes the names and marks of these 10 students via the user. The function should have, as an argument, ‘n’ – the size of the array and ‘pointer_array’ – a pointer to the 1st location of the array of pointers.

Write a function named ‘sort’ which accepts (n, pointer_array) and sorts the list as per their names.

Call out a function named ‘print_list’ which accepts (n, pointer_array) and prints the sorted list as per their names.

Code:

```
#include <iostream>
#include <string>
#include <algorithm> // Added for sorting
#include <cmath>

using namespace std;

struct student {
    string name;
    double chem;
    double phy;
    double math;
};

int length = 0;

void initial(int n, struct student **data_arr) {
    cout << "----------- Loading ----------" << endl;
    for (int i = 0; i < n; i++) {
        cout << "[" << i + 1 << "]: " << "Enter the name of Student:
";
        cin >> data_arr[i]->name;
        cout << endl;
```

```
        cout << "Enter the chemistry marks: ";
        cin >> data_arr[i]->chem;
        cout << "Enter the physics marks: ";
        cin >> data_arr[i]->phy;
        cout << "Enter the math marks: ";
        cin >> data_arr[i]->math;
        cout << endl;
    }
    cout << "---------- Please Wait ----------" << endl;
    length = length + 1;
    cout << endl;
    return;
}

void calculate_average() {
    // Your implementation for calculating average goes here
    return;
}

bool compareByName(const student* a, const student* b) {
    return a->name < b->name;
}

void sort(int n, struct student **data_arr) {
    cout << "---------- Alphabetized order ----------" << endl;
    sort(data_arr, data_arr + n, compareByName);
    return;
}

void printing(int n, struct student **data_arr) {
    cout << "---------- Loading ----------" << endl;
    cout << endl << "---------- Recorded Data ----------" << endl;
    for (int i = 0; i < n; i++) {
        cout << "[" << i + 1 << "]: " << "Name: " << data_arr[i]-
>name
            << " Chemistry: " << data_arr[i]->chem
            << " Physics: " << data_arr[i]->phy
            << " Math: " << data_arr[i]->math << endl;
```

```
        double avg = (data_arr[i]->chem + data_arr[i]->phy +
data_arr[i]->math) / 3.0;

        cout << " Average: " << avg << endl;

    }

    cout << endl;

    length = length + 1;

}


int main() {

    int n;

    cout << "Enter the number of student's data that you want to
store in structure: ";

    cin >> n;

    cout << endl;

    student* arr[n]; // array of structure

    student* ptr_arr[n]; // pointer array points to structure.

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

        ptr_arr[i] = new student; // new pointer structure & new
multiples dynamic memory allocations.

    }

    cout << endl;

    initial(n, ptr_arr); // calling with original ptr_arr[] instead
of data_arr[].

    sort(n, ptr_arr);

    printing(n, ptr_arr);

    cout << "Length of Function: " << length << endl;

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

        delete ptr_arr[i];

    }

    return 0;

}
```

Methodology:

This C++ program manages student data by defining a `student` struct with attributes for name, chemistry marks, physics marks, and math marks. It implements functions to initialize data, calculate. averages, sort data alphabetically by name, and print recorded data with their averages. The `main` function prompts the user to input the number of students, then records and displays their data. Finally, it deallocates memory used by dynamically allocated structures.

Outputs:

```
Enter the number of student's data that you want to store in structure: 2


---------- Loading ----------
[1]: Enter the name of Student: Ahmed

Enter the chemistry marks: 99
Enter the physics marks: 98
Enter the math marks: 99
```

Fig: 01

```
[2]: Enter the name of Student: Haiqa

Enter the chemistry marks: 98
Enter the physics marks: 99
Enter the math marks: 98
```

Fig: 02

```
[1]: Name: Ahmed Chemistry: 99 Physics: 98 Math: 99
 Average: 98.6667
[2]: Name: Haiqa Chemistry: 98 Physics: 99 Math: 98
 Average: 98.3333
```

Fig: 03