Data Structures and Algorithms Lab(MCSE501P)

Assignment 2

Name :Mahesh Jagtap
Reg No. 24MCS1017

Somu is part of a software development team tasked with developing a simple student management system. The system should be able to store student records, including their registration numbers, names, and marks in five subjects. The records should be stored in a linked list, and the system should provide functionalities to insert new records and sort the students based on their average marks.

A) Implement a singly linked list to store student records. Each node should contain a student's registration number, name, five subject marks, and a pointer to the next node.

B) Demonstrate the insertion of at least five student records into the linked list.

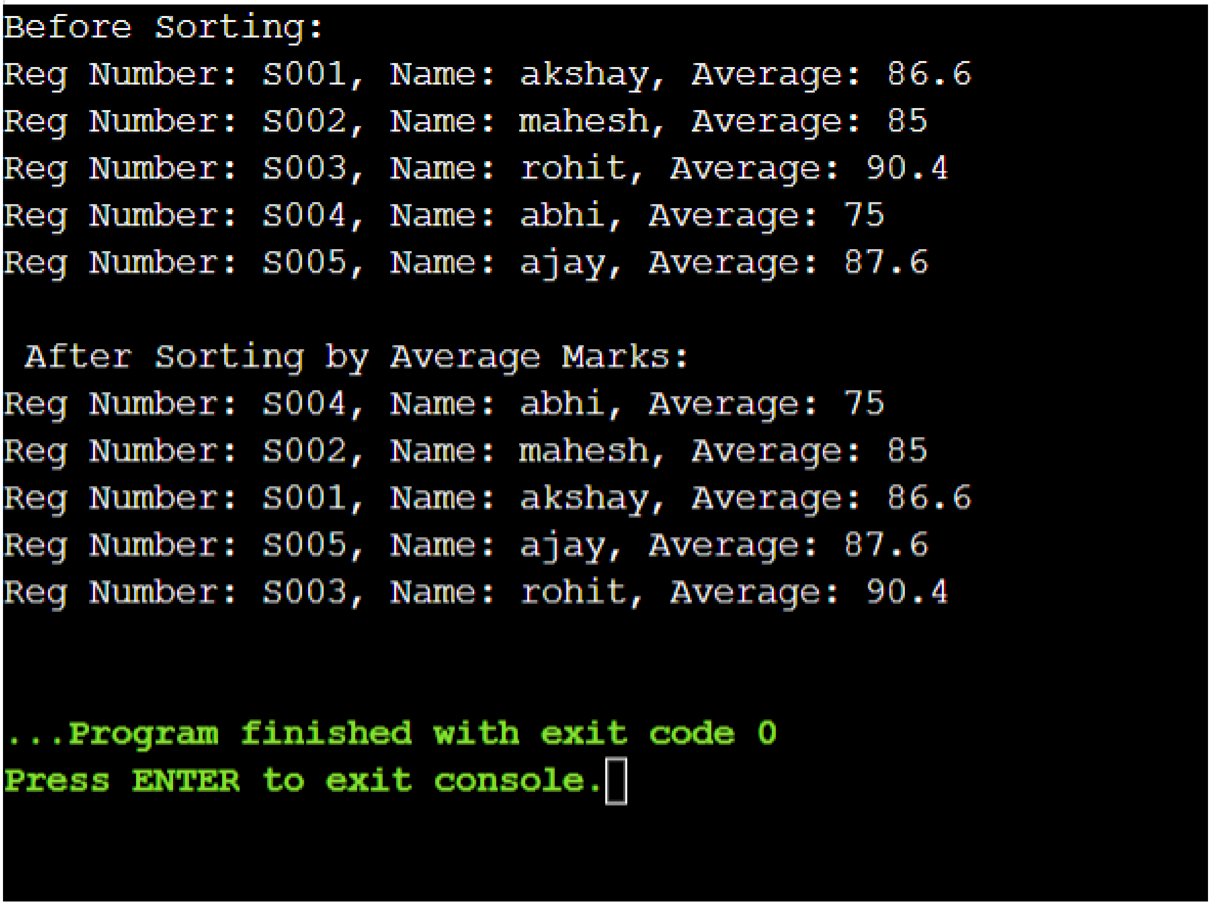
C) Implement a function to sort the linked list of students based on their average marks in ascending order. Choose an appropriate sorting algorithm (e.g., insertion sort or merge sort)

CODE:

```
#include <iostream>
#include <string>
using namespace std;

struct Student {
    string regNumber;
    string name;
    int marks[5];
    float average;

    Student(string regNum, string n, int m[5]) : regNumber(regNum), name(n) {
        int sum = 0;
        for (int i = 0; i < 5; ++i) {
            marks[i] = m[i];
            sum += m[i];
        }
        average = sum / 5.0;
    }
};
```

```
struct Node {
    Student data;
    Node* next;

    Node(Student student) : data(student), next(nullptr) {}
};

class LinkedList {
private:
    Node* head;

public:
    LinkedList() : head(nullptr) {}

    void insertStudent(Student student) {
        Node* newNode = new Node(student);
        if (!head) {
            head = newNode;
        } else {
            Node* temp = head;
            while (temp->next) {
                temp = temp->next;
            }
            temp->next = newNode;
        }
    }

    void displayStudents() {
        Node* temp = head;
        while (temp) {
            cout << "Reg Number: " << temp->data.regNumber
                 << ", Name: " << temp->data.name
                 << ", Average: " << temp->data.average << endl;
            temp = temp->next;
        }
    }

    void sortByAverage() {
        if (!head || !head->next) return;

        Node* sorted = nullptr;
        Node* current = head;
        while (current) {
            Node* next = current->next;
            if (!sorted || sorted->data.average >= current->data.average) {
                current->next = sorted;
                sorted = current;
            } else {
                Node* temp = sorted;
                while (temp->next && temp->next->data.average < current->data.average) {
                    temp = temp->next;
                }
                current->next = temp->next;
                temp->next = current;
            }
            current = next;
        }
        head = sorted;
    }
};

int main() {
    LinkedList list;
    int marks1[5] = {85, 90, 78, 92, 88};
    list.insertStudent(Student("S001", "akshay", marks1));

    int marks2[5] = {80, 85, 82, 88, 90};
    list.insertStudent(Student("S002", "mahesh", marks2));

    int marks3[5] = {95, 92, 89, 85, 91};
    list.insertStudent(Student("S003", "rohit", marks3));

    int marks4[5] = {70, 75, 80, 78, 72};
    list.insertStudent(Student("S004", "abhi", marks4));

    int marks5[5] = {88, 82, 91, 90, 87};
    list.insertStudent(Student("S005", "ajay", marks5));

    cout << "Before Sorting:" << endl;
    list.displayStudents();

    list.sortByAverage();

    cout << "\n After Sorting by Average Marks:" << endl;
    list.displayStudents();

    return 0;
}
```

OUTPUT:



```
Before Sorting:
Reg Number: S001, Name: akshay, Average: 86.6
Reg Number: S002, Name: mahesh, Average: 85
Reg Number: S003, Name: rohit, Average: 90.4
Reg Number: S004, Name: abhi, Average: 75
Reg Number: S005, Name: ajay, Average: 87.6

 After Sorting by Average Marks:
Reg Number: S004, Name: abhi, Average: 75
Reg Number: S002, Name: mahesh, Average: 85
Reg Number: S001, Name: akshay, Average: 86.6
Reg Number: S005, Name: ajay, Average: 87.6
Reg Number: S003, Name: rohit, Average: 90.4


...Program finished with exit code 0
Press ENTER to exit console.
```