# Practical 1 Source Code:-

#include <iostream>

#include <string.h>

#define SIZE 5 // Define the maximum size of the student list

struct Student { int rollNo; char name[50]; float SGPA;

};

// Function to display the student list void displayStudentList(const Student list[]) { int i;

std::cout << "\nStudent List:\n"; for (i = 0; i < SIZE; i++) {

std::cout << "Roll No: " << list[i].rollNo << ", Name: " << list[i].name << ", SGPA: " << list[i].SGPA << std::endl;

}

}

// Function to accept student data

void acceptStudentData(Student list[]) { int i;

for (i = 0; i < SIZE; i++) {

std::cout << "\nEnter details for student " << i + 1 << ":\n"; std::cout << "Roll No: "; std::cin >> list[i].rollNo; std::cout << "Name: "; std::cin.ignore(); // Clear input buffer before reading name std::cin.getline(list[i].name, 50); // Read the entire name std::cout << "SGPA: ";

std::cin >> list[i].SGPA;

}

}

// Function to perform Bubble Sort for SGPA (ascending order) void bubbleSort(Student list[], int size) { int i, j;

for (i = 0; i < size - 1; i++) { for (j = 0; j < size - i - 1; j++) {

if (list[j].SGPA > list[j + 1].SGPA) {

Student temp = list[j]; list[j] = list[j + 1];

list[j + 1] = temp;

}

}

}

}

// Function to search for a student by name (linear search)

void searchStudent(const Student list[], int size) { char searchName[50]; int found = 0;

std::cout << "\nEnter the name of the student you want to search: "; std::cin.ignore(); // Clear input buffer before reading name std::cin.getline(searchName, 50); // Read the entire name

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

if (strcmp(list[i].name, searchName) == 0) { std::cout << "\nStudent found!\n";

std::cout << "Roll No: " << list[i].rollNo << ", Name: " << list[i].name << ", SGPA: " << list[i].SGPA << std::endl;

found = 1;

break;

}

}

if (!found) {

std::cout << "\nStudent not found in the list.\n";

}

}

void binarySearch(Student list[], int size) { int lower, upper, mid, searchSGPA;

std::cout << "\nEnter the SGPA you want to search: "; std::cin >> searchSGPA;

// Ensure the list is sorted before binary search bubbleSort(list, size);

lower = 0;

upper = size - 1;

while (lower <= upper) { mid = (lower + upper) / 2;

if (list[mid].SGPA == searchSGPA) { std::cout << "\nStudent found!\n";

std::cout << "Roll No: " << list[mid].rollNo << ", Name: " << list[mid].name << ", SGPA: " << list[mid].SGPA

<< std::endl;

return; // Exit the function if found } else if (list[mid].SGPA < searchSGPA) { lower = mid + 1;

} else {

upper = mid - 1;

}

}

std::cout << "\nStudent not found in the list.\n";

}

int main() {

Student studentList[SIZE];

int choice;

std::cout << "\nStudent Management System\n";

while (1) {

std::cout << "\n1. Accept Student Data\n"; std::cout << "2. Display Student List\n"; std::cout << "3. Bubble Sort by SGPA\n"; std::cout << "4. Search Student by Name\n"; std::cout << "5. Binary Search by SGPA\n"; std::cout << "6. Exit\n"; std::cout << "Select your choice: ";

std::cin >> choice;

switch (choice) { case 1:

acceptStudentData(studentList);

break; case 2:

displayStudentList(studentList);

break; case 3:

bubbleSort(studentList, SIZE); std::cout << "\nStudent list sorted by SGPA.\n"; break; case 4:

searchStudent(studentList, SIZE);

break; case 5:

binarySearch(studentList, SIZE);

break; case 6: std::cout << "\nExiting...\n";

return 0; default:

std::cout << "\nInvalid choice!\n";

}

}

return 0;

}

# Output:-



