//Assignment 1 (a)

```
#include <iostream>
#include <cstring>   // For strcmp
#include <cstdlib>   // For malloc, realloc, free
using namespace std;
// Structure to store student details
struct Student {
   int id;
   char name[50];
   float cgpa;
};
// Function to input student details
void addStudent(Student*& students, int& size) {
   students = (Student*)realloc(students, (size + 1) * sizeof(Student));
   if (students == NULL) {
      cout << "Memory allocation failed!" << endl;
      exit(1);
   }
   cout << "Enter Student ID: ";
   cin >> students[size].id;
   cout << "Enter Student Name: ";
   cin.ignore();  // To clear newline
   cin.getline(students[size].name, 50);
   cout << "Enter Student CGPA: ";
   cin >> students[size].cgpa;
   size++;
```

```
}


// Function to display student details
void displayStudent(Student s) {
    cout << "ID: " << s.id << ", Name: " << s.name << ", CGPA: " << s.cgpa << endl;
}
// Linear Search
void linearSearch(Student* students, int size, int searchID) {
    for (int i = 0; i < size; i++) {
        if (students[i].id == searchID) {
            cout << "Student Found (Linear Search): ";
            displayStudent(students[i]);
            return;
        }
    }
    cout << "Student not found (Linear Search)." << endl;
}
// Comparison function for sorting (used in binary search)
int compareByID(const void* a, const void* b) {
    Student* s1 = (Student*)a;
    Student* s2 = (Student*)b;
    return s1->id - s2->id;
}
// Binary Search
void binarySearch(Student* students, int size, int searchID) {
    qsort(students, size, sizeof(Student), compareByID);  // Sort before binary search
    int low = 0, high = size - 1;
    while (low <= high) {
        int mid = (low + high) / 2;
        if (students[mid].id == searchID) {
            cout << "Student Found (Binary Search): ";
            displayStudent(students[mid]);
            return;
        }
        else if (students[mid].id < searchID) {
            low = mid + 1;
        }
        else {
            high = mid - 1;
        }
    }
    cout << "Student not found (Binary Search)." << endl;
}

int main() {
    Student* students = NULL;
    int size = 0;
    int choice, id;
    do {
        cout << "\n--- Student Database Menu ---\n";
        cout << "1. Add Student\n2. Linear Search by ID\n3. Binary Search by ID\n4. Exit\n";
        cout << "Enter your choice: ";
        cin >> choice;
        switch (choice) {
            case 1:
                addStudent(students, size);
                break;
            case 2:
                cout << "Enter ID to search (Linear): ";
                cin >> id;
                linearSearch(students, size, id);
                break;
            case 3:
                cout << "Enter ID to search (Binary): ";
                cin >> id;
                binarySearch(students, size, id);
                break;
            case 4:
                cout << "Exiting program..." << endl;
                break;
            default:
                cout << "Invalid choice. Try again." << endl;
        }
    } while (choice != 4);
    free(students);  // Free dynamically allocated memory
    return 0;
}
```

//OUTPUT

```
--- Student Database Menu ---
1. Add Student
2. Linear Search by ID
3. Binary Search by ID
4. Exit
Enter your choice: 1
Enter Student ID: 66
Enter Student Name: Vipul
Enter Student CGPA: 9.6

--- Student Database Menu ---
1. Add Student
2. Linear Search by ID
3. Binary Search by ID
4. Exit
Enter your choice: 1
Enter Student ID: 14
Enter Student Name: Parth
Enter Student CGPA: 8.9

--- Student Database Menu ---
1. Add Student
2. Linear Search by ID
3. Binary Search by ID
```

```
4. Exit
Enter your choice: 1
Enter Student ID: 88
Enter Student Name: Kiran
Enter Student CGPA: 9.9

--- Student Database Menu ---
1. Add Student
2. Linear Search by ID
3. Binary Search by ID
4. Exit
Enter your choice: 2
Enter ID to search (Linear): 88
Student Found (Linear Search): ID: 88, Name: Kiran, CGPA: 9.9

--- Student Database Menu ---
1. Add Student
2. Linear Search by ID
3. Binary Search by ID
4. Exit
Enter your choice: 3
Enter ID to search (Binary): 14
Student Found (Binary Search): ID: 14, Name: Parth, CGPA: 8.9

--- Student Database Menu ---
1. Add Student
```

```
2. Linear Search by ID

3. Binary Search by ID

4. Exit

Enter your choice: 4
```