Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

BÀI TẬP 1 :

1. Prompt gợi ý dùng trong Cursor AI

Bạn có thể nhập **nguyên văn** prompt này vào Cursor:

“Hãy viết chương trình C quản lý sinh viên.
Tạo struct Student gồm các trường: id (int), name (string), age (int), gpa (float).
Khởi tạo sẵn mảng gồm 3 sinh viên mẫu.
In danh sách sinh viên ra màn hình dưới dạng bảng có tiêu đề, các cột thẳng hàng, dễ đọc.”

2. Code C do AI sinh ra (đã chỉnh để in bảng đẹp)

```
#include <stdio.h>

// Khai báo struct Student
struct Student {
    int id;
    char name[50];
    int age;
    float gpa;
};

int main() {
    // Khởi tạo mảng 3 sinh viên
    struct Student students[3] = {
        {1, "Nguyen Van A", 20, 3.50},
        {2, "Tran Thi B", 19, 3.80},
        {3, "Le Van C", 21, 3.20}
    };

    // In tiêu đề bảng
    printf("+----+----------------+-----+-----+\n");
    printf("| ID | Name           | Age | GPA |\n");
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
    printf("+----+----------------+-----+-----+\n");

    // In danh sách sinh viên
    for (int i = 0; i < 3; i++) {
        printf("| %-2d | %-14s | %-3d | %.2f |\n",
               students[i].id,
               students[i].name,
               students[i].age,
               students[i].gpa);
    }

    // In dòng kết thúc bảng
    printf("+----+----------------+-----+-----+\n");

    return 0;
}
```

3. Kết quả khi chạy chương trình (Output)

```
+----+----------------+-----+-----+
| ID | Name           | Age | GPA |
+----+----------------+-----+-----+
| 1  | Nguyen Van A   | 20  | 3.50 |
| 2  | Tran Thi B     | 19  | 3.80 |
| 3  | Le Van C       | 21  | 3.20 |
+----+----------------+-----+-----+
```

Các cột **thẳng hàng**
GPA hiển thị **2 chữ số thập phân**
Dễ đọc, đúng yêu cầu “in dạng bảng”

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

4. Kiểm tra & cải tiến prompt (theo yêu cầu đề)

Vấn đề có thể gặp nếu prompt chưa rõ

- Tên bị lệch cột
- GPA in quá nhiều chữ số
- Không có đường viền bảng

Prompt cải tiến để AI sửa lỗi

Nếu bảng chưa đẹp, bạn có thể yêu cầu lại Cursor:

“Hãy chỉnh lại chương trình để in bảng sinh viên có viền, các cột thẳng hàng, GPA hiển thị đúng 2 chữ số thập phân, code dễ đọc và có chú thích.”

Sau khi cải tiến, Cursor sẽ tự điều chỉnh `printf` cho đúng định dạng như trên.

5. Gợi ý nâng cao

- Thêm hàm `printStudents()`
- Nhập sinh viên từ bàn phím
- Sắp xếp theo GPA
- Tìm sinh viên GPA cao nhất

BÀI TẬP 2 :

1. Chương trình C sau khi Cursor AI sinh và đã kiểm tra

```
#include <stdio.h>

#define MAX 5

// Khai báo struct Student
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
struct Student {
    int id;
    char name[50];
    int age;
    float gpa;
};

// Hàm nhập danh sách sinh viên
void inputStudents(struct Student students[], int *n) {
    printf("Nhap so luong sinh vien (toi da 5): ");
    scanf("%d", n);

    if (*n > MAX) {
        *n = MAX;
    }

    for (int i = 0; i < *n; i++) {
        printf("Nhap sinh vien thu %d:\n", i + 1);
        scanf("%d %s %d %f",
              &students[i].id,
              students[i].name,
              &students[i].age,
              &students[i].gpa);
    }
}

// Hàm in danh sách sinh viên
void printStudents(struct Student students[], int n) {
    printf("\nID   Name   Age   GPA\n");
    for (int i = 0; i < n; i++) {
        printf("%-4d %-6s %-5d %.1f\n",
               students[i].id,
               students[i].name,
               students[i].age,
               students[i].gpa);
    }
}
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
int main() {
    struct Student students[MAX];
    int n = 0;

    inputStudents(students, &n);
    printStudents(students, n);

    return 0;
}
```

2. Dữ liệu nhập (Input)

```
2
1 An 20 8.0
2 Bình 21 7.5
```

3. Kết quả khi chạy chương trình (Output)

```
ID   Name   Age   GPA
1    An     20    8.0
2    Bình   21    7.5
```

4. Kết luận

Chương trình đã được bổ sung thành công chức năng nhập tối đa 5 sinh viên từ bàn phím.
Dữ liệu sinh viên được lưu vào mảng struct và in ra màn hình đúng định dạng bảng.
Yêu cầu nhập và xuất danh sách sinh viên đã được hoàn thành.

BÀI TẬP 3 :

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

1. Prompt sinh viên sử dụng trong Cursor AI

Sinh viên nhập prompt sau vào Cursor AI:

“Hãy bổ sung vào chương trình C quản lý sinh viên đã có struct Student (id, name, age, gpa) một hàm `saveToFile` để lưu toàn bộ danh sách sinh viên vào file `students.txt`.
Mỗi sinh viên lưu trên một dòng theo định dạng: id name age gpa.
Sử dụng thao tác file trong C (fopen, fprintf, fclose).
Code rõ ràng, dễ đọc, phù hợp sinh viên mới học C.”

2. Chương trình C sau khi Cursor AI sinh và đã kiểm tra

```
#include <stdio.h>

#define MAX 5

// Khai báo struct Student
struct Student {
    int id;
    char name[50];
    int age;
    float gpa;
};

// Hàm nhập danh sách sinh viên
void inputStudents(struct Student students[], int *n) {
   printf("Nhap so luong sinh vien (toi da 5): ");
    scanf("%d", n);

    if (*n > MAX) {
        *n = MAX;
    }

    for (int i = 0; i < *n; i++) {
        printf("Nhap sinh vien thu %d:\n", i + 1);
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
        scanf("%d %s %d %f",
              &students[i].id,
              students[i].name,
              &students[i].age,
              &students[i].gpa);
    }
}

// Hàm in danh sách sinh viên
void printStudents(struct Student students[], int n) {
    printf("\nID   Name   Age   GPA\n");
    for (int i = 0; i < n; i++) {
        printf("%-4d %-6s %-5d %.1f\n",
               students[i].id,
               students[i].name,
               students[i].age,
               students[i].gpa);
    }
}

// Hàm lưu danh sách sinh viên vào file
void saveToFile(struct Student students[], int n) {
    FILE *f = fopen("students.txt", "w");

    if (f == NULL) {
        printf("Khong the mo file!\n");
        return;
    }

    for (int i = 0; i < n; i++) {
        fprintf(f, "%d %s %d %.1f\n",
                students[i].id,
                students[i].name,
                students[i].age,
                students[i].gpa);
    }

    fclose(f);
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
    printf("Da luu danh sach sinh vien vao file students.txt\n");
}

int main() {
    struct Student students[MAX];
    int n = 0;

    inputStudents(students, &n);
    printStudents(students, n);
    saveToFile(students, n);

    return 0;
}
```

3. Dữ liệu nhập (Input)

```
2
1 An 20 8.0
2 Bình 21 7.5
```

4. Nội dung file `students.txt` sau khi chạy chương trình

```
1 An 20 8.0
2 Bình 21 7.5
```

5. Kết luận

Chương trình đã bổ sung thành công hàm `saveToFile` để lưu danh sách sinh viên vào file văn bản.
File `students.txt` được tạo ra và chứa dữ liệu đúng định dạng yêu cầu.

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

Bài tập giúp sinh viên hiểu cách kết hợp struct, mảng và thao tác file trong ngôn ngữ C, đồng thời rèn kỹ năng sử dụng Cursor AI để hỗ trợ lập trình.

BÀI TẬP 4 :

1. Prompt sinh viên sử dụng trong Cursor AI

Sinh viên nhập prompt sau vào Cursor AI:

“Hãy bổ sung vào chương trình C quản lý sinh viên đã có struct Student (id, name, age, gpa) một hàm `readFromFile` để đọc dữ liệu từ file `students.txt`.
Mỗi dòng trong file có định dạng: id name age gpa.
Dữ liệu đọc được lưu vào mảng Student, sau đó in danh sách sinh viên ra màn hình.
Code rõ ràng, dễ đọc, phù hợp sinh viên mới học C.”

2. Chương trình C sau khi Cursor AI sinh và đã kiểm tra

```
#include <stdio.h>

#define MAX 5

// Khai báo struct Student
struct Student {
    int id;
    char name[50];
    int age;
    float gpa;
};

// Hàm đọc danh sách sinh viên từ file
void readFromFile(struct Student students[], int *n) {
    FILE *f = fopen("students.txt", "r");

    if (f == NULL) {
       printf("Khong the mo file students.txt\n");
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
        return;
    }

    *n = 0;
    while (fscanf(f, "%d %s %d %f",
                  &students[*n].id,
                  students[*n].name,
                  &students[*n].age,
                  &students[*n].gpa) == 4) {
        (*n)++;
        if (*n >= MAX) {
            break;
        }
    }

    fclose(f);
}

// Hàm in danh sách sinh viên
void printStudents(struct Student students[], int n) {
    printf("Danh sach doc tu file:\n");
    for (int i = 0; i < n; i++) {
        printf("%d %s %d %.1f\n",
               students[i].id,
               students[i].name,
               students[i].age,
               students[i].gpa);
    }
}

int main() {
    struct Student students[MAX];
    int n = 0;

    readFromFile(students, &n);
    printStudents(students, n);

    return 0;
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

}

3. Nội dung file `students.txt`

```
1 An 20 8.0
2 Bình 21 7.5
```

4. Kết quả khi chạy chương trình (Output)

```
Danh sach doc tu file:
1 An 20 8.0
2 Bình 21 7.5
```

5. Kết luận

Chương trình đã đọc thành công dữ liệu sinh viên từ file `students.txt` và lưu vào mảng struct Student.
Danh sách sinh viên được in ra đúng định dạng yêu cầu.
Bài tập giúp sinh viên hiểu cách kết hợp lưu trữ dữ liệu bằng file và sử dụng AI để hỗ trợ xây dựng hàm xử lý dữ liệu trong ngôn ngữ C.

BÀI TẬP 5 :

```
#include <stdio.h>


#define MAX 5


// Khai báo struct Student
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
struct Student {
   int id;
   char name[50];
   int age;
   float gpa;
};


/*
   searchStudentById:
   Hàm tìm kiếm sinh viên trong mảng theo ID.

   Tham số:
   - students: mảng chứa danh sách sinh viên
   - n: số lượng sinh viên hiện có
   - searchId: ID cần tìm

   Kết quả:
   - In ra thông tin sinh viên nếu tìm thấy
   - In ra thông báo "Khong tim thay" nếu không tồn tại
*/
void searchStudentById(struct Student students[], int n, int searchId) {
   int found = 0;

   for (int i = 0; i < n; i++) {
      if (students[i].id == searchId) {
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
            printf("Ket qua tim kiem:\n");
            printf("ID: %d, Name: %s, Age: %d, GPA: %.1f\n",
                students[i].id,
                students[i].name,
                students[i].age,
                students[i].gpa);
            found = 1;
            break;
        }
    }

    if (!found) {
        printf("Khong tim thay sinh vien co ID = %d\n", searchId);
    }
}

int main() {
    struct Student students[MAX] = {
        {1, "An", 20, 8.0},
        {2, "Binh", 21, 7.5},
        {3, "Cuong", 19, 6.8}
    };

    int n = 3;
    int searchId;
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
    printf("Nhap ID can tim: ");
    scanf("%d", &searchId);

    searchStudentById(students, n, searchId);

    return 0;
}
```

+. Dữ liệu nhập (Input)

yaml

Copy code

```
Nhap ID can tim: 2
```

+. Kết quả khi chạy chương trình (Output)

yaml

Copy code

```
Ket qua tim kiem:
ID: 2, Name: Binh, Age: 21, GPA: 7.5
```

6. Test case do AI sinh

Test case 1: ID tồn tại

Input: 2

Kết quả mong đợi:

yaml

Copy code

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

Ket qua tim kiem:

ID: 2, Name: Binh, Age: 21, GPA: 7.5

Test case 2: ID không tồn tại

Input: 10

Kết quả mong đợi:

java

Copy code

Khong tim thay sinh vien co ID = 10

Test case 3: ID âm

Input: -1

Kết quả mong đợi:

java

Copy code

Khong tim thay sinh vien co ID = -1

+. Kết luận (ghi trong báo cáo)

Chương trình đã bổ sung thành công chức năng tìm kiếm sinh viên theo ID.

Hàm tìm kiếm có docstring rõ ràng, mô tả đầy đủ chức năng và tham số.

Các test case giúp kiểm tra chương trình trong nhiều tình huống khác nhau, đảm bảo tính đúng đắn của chức năng tìm kiếm.

Việc sử dụng Cursor AI hỗ trợ sinh code, docstring và test case giúp quá trình phát triển chương trình nhanh và hiệu quả hơn.

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

BÀI TẬP 6 :

```
#include <stdio.h>


#define MAX 10


// Khai báo struct Student
struct Student {
   int id;
   char name[50];
   int age;
   float gpa;
};


/*
   calculateAverageGPA:
   Hàm tính GPA trung bình của lớp.

   Tham số:
   - students: mảng sinh viên
   - n: số lượng sinh viên

   Giá trị trả về:
   - GPA trung bình nếu n > 0
   - 0 nếu danh sách rỗng
*/
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
float calculateAverageGPA(struct Student students[], int n) {
   if (n == 0) {
      return 0;
   }

   float sum = 0;
   for (int i = 0; i < n; i++) {
      sum += students[i].gpa;
   }

   return sum / n;
}

int main() {
   // Dữ liệu giả lập 10 sinh viên
   struct Student students[MAX] = {
      {1, "SV1", 20, 8.0},
      {2, "SV2", 19, 7.5},
      {3, "SV3", 21, 8.5},
      {4, "SV4", 20, 9.0},
      {5, "SV5", 22, 8.0},
      {6, "SV6", 18, 7.0},
      {7, "SV7", 21, 8.5},
      {8, "SV8", 20, 9.0},
      {9, "SV9", 19, 8.0},
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
    {10, "SV10", 22, 8.0}
  };

  int n = 10;

  float avgGPA = calculateAverageGPA(students, n);
  printf("GPA trung binh cua lop la: %.2f\n", avgGPA);

  return 0;
}
```

4. Kết quả chạy chương trình (Output)

yaml

Copy code

GPA trung binh cua lop la: 8.25

+) Dữ liệu giả lập do AI sinh (phục vụ kiểm thử)

Sinh viên	GPA
SV1	8.0
SV2	7.5
SV3	8.5
SV4	9.0
SV5	8.0
SV6	7.0
SV7	8.5
SV8	9.0
SV9	8.0

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

SV10 8.0

Tổng GPA = 82.5

GPA trung bình = 82.5 / 10 = 8.25

+)Test case do AI sinh

Test case 1: Danh sách có sinh viên

Input: 10 sinh viên như trên

Kết quả mong đợi:

yaml

Copy code

GPA trung binh cua lop la: 8.25

Test case 2: Danh sách rỗng

Input: n = 0

Kết quả mong đợi:

yaml

Copy code

GPA trung binh cua lop la: 0.00

+) Kết luận (ghi trong báo cáo)

Chương trình đã tính đúng GPA trung bình của lớp dựa trên danh sách sinh viên.

Dữ liệu giả lập do AI sinh giúp kiểm thử chương trình một cách thuận tiện.

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

Test case danh sách rỗng đảm bảo chương trình hoạt động an toàn trong mọi tình huống.

Việc sử dụng Cursor AI hỗ trợ sinh hàm, dữ liệu giả lập và test case giúp quá trình phát triển và kiểm thử chương trình hiệu quả hơn.

BÀI TẬP 7 :

```
#include <stdio.h>


#define MAX 5


// Khai báo struct Student
struct Student {
   int id;
   char name[50];
   int age;
   float gpa;
};


/*
   sortByGPA:
   Hàm sắp xếp danh sách sinh viên theo GPA giảm dần
*/
void sortByGPA(struct Student students[], int n) {
   for (int i = 0; i < n - 1; i++) {
      for (int j = 0; j < n - i - 1; j++) {
         if (students[j].gpa < students[j + 1].gpa) {
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
            struct Student temp = students[j];
            students[j] = students[j + 1];
            students[j + 1] = temp;
        }
    }
  }
}

// Hàm in danh sách sinh viên
void printStudents(struct Student students[], int n) {
  printf("Danh sach sau khi sap xep:\n");
  for (int i = 0; i < n; i++) {
    printf("%d %s %d %.1f\n",
        students[i].id,
        students[i].name,
        students[i].age,
        students[i].gpa);
  }
}

int main() {
  struct Student students[MAX] = {
    {1, "An", 20, 8.0},
    {2, "Binh", 21, 7.5},
    {3, "Chi", 19, 9.0}
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
    };


    int n = 3;


    sortByGPA(students, n);

    printStudents(students, n);


    return 0;

}
```

Kết quả khi chạy chương trình (Output)

```
Danh sach sau khi sap xep:
3 Chi 19 9.0
1 An 20 8.0
2 Binh 21 7.5
```

Test case do AI sinh

Test case 1: GPA bằng nhau

- Input:

```
1 An 20 8.0
2 Binh 21 8.0
3 Chi 19 8.0
```

- Kết quả mong đợi: thứ tự có thể giữ nguyên hoặc thay đổi, nhưng GPA đều bằng nhau.

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

Test case 2: GPA tăng dần sẵn

- Input:

```
1 An 20 7.0
2 Binh 21 8.0
3 Chi 19 9.0
```

- Kết quả mong đợi:

```
3 Chi 19 9.0
2 Binh 21 8.0
1 An 20 7.0
```

Test case 3: GPA ngẫu nhiên

- Input:

```
1 An 20 8.2
2 Binh 21 6.9
3 Chi 19 9.1
4 Dung 22 7.5
```

- Kết quả mong đợi: danh sách được sắp xếp giảm dần theo GPA.

BÀI TẬP 8 :

1. Chương trình C sau khi Cursor AI sinh và đã kiểm tra

```
#include <stdio.h>

#define MAX 10

// Khai báo struct Student
struct Student {
    int id;
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
    char name[50];
    int age;
    float gpa;
};

/*    readClassFromFile:
    Hàm đọc danh sách sinh viên từ file lớp học.

    Các bước xử lý:
    1. Mở file theo tên người dùng nhập
    2. Kiểm tra file có mở được hay không
    3. Đọc từng dòng dữ liệu sinh viên
    4. Lưu vào mảng Student
    5. Đóng file sau khi đọc xong
*/
void readClassFromFile(char filename[], struct Student students[], int
*n) {
    FILE *f = fopen(filename, "r");

    // Kiểm tra mở file
    if (f == NULL) {
        printf("Khong the mo file %s\n", filename);
        *n = 0;
        return;
    }

    *n = 0;

    // Đọc dữ liệu sinh viên từ file
    while (fscanf(f, "%d %s %d %f",
                  &students[*n].id,
                  students[*n].name,
                  &students[*n].age,
                  &students[*n].gpa) == 4) {

        (*n)++;

        // Tránh vượt quá kích thước mảng
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
        if (*n >= MAX) {
            break;
        }
    }

    // Đóng file sau khi đọc xong
    fclose(f);
}

// Hàm in danh sách sinh viên
void printStudents(char className[], struct Student students[], int n)
{
    printf("Danh sach sinh vien %s:\n", className);
    for (int i = 0; i < n; i++) {
        printf("%d %s %d %.1f\n",
               students[i].id,
               students[i].name,
               students[i].age,
               students[i].gpa);
    }
}

int main() {
    struct Student students[MAX];
    int n = 0;
    char filename[50];

    // Nhập tên file lớp học
    printf("Nhap ten file: ");
    scanf("%s", filename);

    // Đọc dữ liệu từ file tương ứng
    readClassFromFile(filename, students, &n);

    // In danh sách sinh viên nếu đọc thành công
    if (n > 0) {
        printStudents(filename, students, n);
    }
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
    return 0;
}
```

3. Dữ liệu giả lập cho 3 lớp (do AI sinh)

File `classA.txt`

```
1 An 20 8.0
2 Binh 21 7.5
```

File `classB.txt`

```
1 Chi 19 9.0
2 Dung 22 8.2
3 Huy 20 7.8
```

File `classC.txt`

```
1 Lan 21 8.5
2 Minh 20 7.0
```

4. Dữ liệu nhập (Input)

```
Nhap ten file: classA.txt
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

5. Kết quả khi chạy chương trình (Output)

```
Danh sach sinh vien classA.txt:
1 An 20 8.0
2 Binh 21 7.5
```

BÀI TẬP 9 :

```
#include <stdio.h>


#define MAX 10


// Khai báo struct Student
struct Student {
   int id;
   char name[50];
   int age;
   float gpa;
};


/*
   addStudent:
   Hàm thêm một sinh viên mới vào danh sách.


   Tham số:
   - students: mảng sinh viên
   - n: con trỏ lưu số lượng sinh viên hiện tại
   - newStudent: sinh viên cần thêm
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
   Kết quả:
   - Thêm thành công nếu ID chưa tồn tại
   - Không thêm nếu ID bị trùng
*/
void addStudent(struct Student students[], int *n, struct Student newStudent) {
   // Kiểm tra trùng ID
   for (int i = 0; i < *n; i++) {
      if (students[i].id == newStudent.id) {
         printf("Khong the them: ID da ton tai\n");
         return;
      }
   }

   if (*n >= MAX) {
      printf("Danh sach da day\n");
      return;
   }

   students[*n] = newStudent;
   (*n)++;
}

/*
   deleteStudentById:
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
   Hàm xóa sinh viên khỏi danh sách theo ID.


   Tham số:
   - students: mảng sinh viên
   - n: con trỏ lưu số lượng sinh viên
   - deleteId: ID sinh viên cần xóa


   Kết quả:
   - Xóa sinh viên nếu tìm thấy ID
   - Thông báo nếu ID không tồn tại
*/
void deleteStudentById(struct Student students[], int *n, int deleteId) {
   int found = 0;


   for (int i = 0; i < *n; i++) {
      if (students[i].id == deleteId) {
         // Dồn các phần tử phía sau lên
         for (int j = i; j < *n - 1; j++) {
            students[j] = students[j + 1];
         }
         (*n)--;
         found = 1;
         break;
      }
   }
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
    if (!found) {
        printf("Khong tim thay sinh vien can xoa\n");
    }
}

// Hàm in danh sách sinh viên
void printStudents(struct Student students[], int n) {
    printf("Danh sach sau khi chinh sua:\n");
    for (int i = 0; i < n; i++) {
        printf("%d %s %d %.1f\n",
               students[i].id,
               students[i].name,
               students[i].age,
               students[i].gpa);
    }
}

int main() {
    struct Student students[MAX] = {
        {1, "An", 20, 8.0},
        {2, "Binh", 21, 7.5},
        {3, "Chi", 19, 9.0}
    };
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
    int n = 3;


    // Thêm sinh viên mới
    struct Student newStudent = {4, "Dung", 22, 6.5};
    addStudent(students, &n, newStudent);


    // Xóa sinh viên theo ID
    int deleteId = 2;
    deleteStudentById(students, &n, deleteId);


    // In danh sách sau chỉnh sửa
    printStudents(students, n);


    return 0;
}
```

Dữ liệu nhập (Input)

```
Them: 4 Dung 22 6.5
Xoa: 2
```

ết quả khi chạy chương trình (Output)

```
Danh sach sau khi chinh sua:
1 An 20 8.0
3 Chi 19 9.0
4 Dung 22 6.5
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

Test case do AI sinh

Test case 1: Thêm trùng ID

- Input: thêm sinh viên có ID = 1
- Kết quả mong đợi:

```
Khong the them: ID da ton tai
```

Test case 2: Xóa ID không tồn tại

- Input: xóa ID = 10
- Kết quả mong đợi:

```
Khong tim thay sinh vien can xoa
```

BÀI TẬP 10 :

```
#include <stdio.h> #include <string.h>

#define MAX 50

// Khai báo struct Student struct Student { int id; char name[50]; int
age; float gpa; };

/* ---------- CÁC HÀM CHỨC NĂNG ---------- */

// Nhập danh sách sinh viên void inputStudents(struct Student
students[], int *n) { printf("Nhap so luong sinh vien: "); scanf("%d",
n);

for (int i = 0; i < *n; i++) {
    scanf("%d %s %d %f",
          &students[i].id,
          students[i].name,
          &students[i].age,
          &students[i].gpa);
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
}


}

// In danh sách sinh viên void printStudents(struct Student
students[], int n) { printf("Danh sach sinh vien:\n"); for (int i = 0;
i < n; i++) { printf("%d %s %d %.1f\n", students[i].id,
students[i].name, students[i].age, students[i].gpa); } }

// Ghi danh sách sinh viên ra file void saveToFile(struct Student
students[], int n) { FILE *f = fopen("students.txt", "w"); for (int i
= 0; i < n; i++) { fprintf(f, "%d %s %d %.1f\n", students[i].id,
students[i].name, students[i].age, students[i].gpa); } fclose(f);
printf("Da ghi file students.txt\n"); }

// Đọc danh sách sinh viên từ file void readFromFile(struct Student
students[], int *n) { FILE *f = fopen("students.txt", "r"); *n = 0;
while (fscanf(f, "%d %s %d %f", &students[*n].id, students[*n].name,
&students[*n].age, &students[*n].gpa) == 4) { (*n)++; } fclose(f); }

// Tìm kiếm sinh viên theo ID void searchById(struct Student
students[], int n) { int id; printf("Nhap ID can tim: "); scanf("%d",
&id);

for (int i = 0; i < n; i++) {
    if (students[i].id == id) {
        printf("ID: %d, Name: %s, Age: %d, GPA: %.1f\n",
               students[i].id,
               students[i].name,
               students[i].age,
               students[i].gpa);
        return;
    }
}
printf("Khong tim thay\n");


}
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
// Sắp xếp sinh viên theo GPA giảm dần void sortByGPA(struct Student
students[], int n) { for (int i = 0; i < n - 1; i++) { for (int j = 0;
j < n - i - 1; j++) { if (students[j].gpa < students[j + 1].gpa)
{ struct Student temp = students[j]; students[j] = students[j + 1];
students[j + 1] = temp; } } } }

// Thêm sinh viên void addStudent(struct Student students[], int *n)
{ struct Student s; scanf("%d %s %d %f", &s.id, s.name, &s.age,
&s.gpa); students[*n] = s; (*n)++; }

// Xóa sinh viên theo ID void deleteStudent(struct Student students[],
int *n) { int id; printf("Nhap ID can xoa: "); scanf("%d", &id);

for (int i = 0; i < *n; i++) {
    if (students[i].id == id) {
        for (int j = i; j < *n - 1; j++) {
            students[j] = students[j + 1];
        }
        (*n)--;
        return;
    }
}
printf("Khong tim thay ID\n");


}

/* ---------- HÀM MAIN + MENU ---------- */

int main() { struct Student students[MAX] = { {1, "An", 20, 8.0}, {2,
"Binh", 21, 7.5}, {3, "Chi", 19, 9.0}, {4, "Dung", 22, 6.5}, {5,
"Hoa", 20, 8.2}, {6, "Khanh", 21, 7.8}, {7, "Lan", 19, 8.9}, {8,
"Minh", 22, 7.0}, {9, "Nam", 20, 8.4}, {10, "Phuc", 21, 6.9} };

int n = 10;
int choice;

do {
    printf("\n===== Student Manager =====\n");
    printf("1. Nhap sinh vien\n");
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
    printf("2. In danh sach\n");
    printf("3. Ghi file\n");
    printf("4. Doc file\n");
    printf("5. Tim kiem\n");
    printf("6. Sap xep\n");
    printf("7. Them sinh vien\n");
    printf("8. Xoa sinh vien\n");
    printf("0. Thoat\n");
    printf("==========================\n");
    printf("Chon chuc nang: ");
    scanf("%d", &choice);

    switch (choice) {
        case 1: inputStudents(students, &n); break;
        case 2: printStudents(students, n); break;
        case 3: saveToFile(students, n); break;
        case 4: readFromFile(students, &n); break;
        case 5: searchById(students, n); break;
        case 6: sortByGPA(students, n); break;
        case 7: addStudent(students, &n); break;
        case 8: deleteStudent(students, &n); break;
        case 0: printf("Thoat chuong trinh\n"); break;
        default: printf("Lua chon khong hop le\n");
    }
} while (choice != 0);

return 0;


}
```

Dữ liệu giả lập 10 sinh viên (do AI sinh)

```
1 An 20 8.0
2 Binh 21 7.5
3 Chi 19 9.0
4 Dung 22 6.5
5 Hoa 20 8.2
```

Họ và tên : Lý Gia Huy
Lớp : CNTT 5
BTVN : session 8

```
6 Khanh 21 7.8
7 Lan 19 8.9
8 Minh 22 7.0
9 Nam 20 8.4
10 Phuc 21 6.9
```

Giao diện menu (Output)

```
===== Student Manager =====
1. Nhap sinh vien
2. In danh sach
3. Ghi file
4. Doc file
5. Tim kiem
6. Sap xep
7. Them sinh vien
8. Xoa sinh vien
0. Thoat
============================
Chon chuc nang:
```

Giải thích luồng chạy chương trình (do AI sinh)

Chương trình bắt đầu bằng việc khởi tạo danh sách sinh viên giả lập.
 Menu được hiển thị liên tục trong vòng lặp do-while.
 Người dùng chọn chức năng bằng số tương ứng.
 Mỗi lựa chọn sẽ gọi một hàm xử lý riêng biệt như nhập, in, ghi file, đọc file, tìm kiếm, sắp xếp, thêm hoặc xóa sinh viên.
 Chương trình chỉ kết thúc khi người dùng chọn chức năng 0 (Thoát).