TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI

PHÂN HIỆU TẠI TP. HỒ CHÍ MINH

BỘ MÔN CÔNG NGHỆ THÔNG TIN

BÁO CÁO BÀI TẬP LỚN

ĐỀ TÀI: CHƯƠNG TRÌNH QUẢN LÝ SINH VIÊN

Giảng viên hướng dẫn:	Th.S TRẦN THỊ DUNG
Người thực hiện:	HÀ VĂN DŨNG
Lớp:	CÔNG NGHỆ THÔNG TIN
Khoá:	63

TP. Hồ Chí Minh, tháng 04 năm 2023

TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI

PHÂN HIỆU TẠI TP. HỒ CHÍ MINH

BỘ MÔN CÔNG NGHỆ THÔNG TIN

BÁO CÁO BÀI TẬP LỚN

ĐỀ TÀI: CHƯƠNG TRÌNH QUẢN LÝ SINH VIÊN

Giảng viên hướng dẫn: Th.S TRẦN THỊ DUNG

Nhóm sinh viên thực hiện: HÀ VĂN DŨNG

NGUYỄN NGUYÊN HUY

Lớp: CÔNG NGHỆ THÔNG TIN

Khoá: 63

TP. Hồ Chí Minh, tháng 04 năm 2023

TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

PHÂN HIỆU TẠI THÀNH PHỐ HỒ CHÍ MINH

Độc lập – Tự do – Hạnh phúc

BÀI TẬP LỚN

BỘ MÔN: **CÔNG NGHỆ THÔNG TIN**

***

Mã sinh viên: 6351071011 **Họ tên SV:** Hà Văn Dũng

Khóa: 63 **Lớp:** Công Nghệ Thông Tin

1. Tên đề tài bài tập lớn:

CHƯƠNG TRÌNH QUẢN LÝ SINH VIÊN

2. Mục đích, yêu cầu:

a. Mục đích: Tạo ra một chương trình quản lý thông tin sinh viên

b. Yêu cầu:

- **Tạo ra một chương trình quản lý thông tin sinh viên**
- **Áp dụng các kiến thức cấu trúc dữ liệu và giải thuật đã được học**

3. Nội dung và phạm vi đề tài:

a. Nội dung đề tài: Tạo ra một chương trình quản lý thông tin cơ bản của sinh viên

b. Phạm vi đề tài: Tập trung vào quản lý sinh viên

4. Công nghệ, công cụ và ngôn ngữ lập trình:

a. Công nghệ:

b. Công cụ: Visual Studio Code, Git, Github

c. Ngôn ngữ lập trình: C++

5. Các kết quả chính dự kiến sẽ đạt được và ứng dụng: Phát triển và triển khai chương trình quản lý sinh viên hiệu quả, có khả năng quản lý thông tin sinh viên, tên, tuổi, ngành học, điểm số,…. Ứng dụng của dự án có thể làm giảm công việc thủ công trong việc quản lý thông tin sinh viên tại các tổ chức giáo dục, giúp cải thiện hiệu suất và tính chính xác của quản lý dữ liệu.

6. Giáo viên và cán bộ hướng dẫn:

Họ tên: TRẦN THỊ DUNG.

Đơn vị công tác: Bộ môn Công Nghệ Thông Tin – Trường Đại học Giao thông Vận tải phân hiệu tại TP HCM

Điện thoại: Email:

Ngày 14 tháng 04 năm 2023	**Đã giao nhiệm vụ TKTN Giáo viên hướng dẫn**
BM Công Nghệ Thông Tin	
	ThS. Trần Thị Dung

Đã nhận nhiệm vụ Bài Tập Lớn

Sinh viên: HÀ VĂN DŨNG Ký tên: Dũng

Điện thoại: 0327250461 Email: brave2112love@gmail.com

LỜI CẢM ƠN

Qua thời gian học tập và rèn luyện tại trường Trường Đại học Giao thông Vận tải phân hiệu tại TP HCM, đến nay em đã được trang bị những kĩ năng, kiến thức cơ bản để có thể hoàn thành được bài tập lớn do giảng viên giao.

Cảm ơn tập thể các thầy cô giáo Bộ môn Công Nghệ Thông Tin và các thầy cô thỉnh giảng đã giảng dạy, luôn quan tâm và không ngần ngại dành thời gian để chỉ bày và giải đáp những thắc mắc của chúng em trong những tiết học và cả những lúc ngoài giờ.

Và cảm ơn thạc sĩ Trần Thị Dung đã luôn quan tâm nhiệt tình hướng dẫn, giúp đỡ chúng em trong quá trình triển khai và thực hiện bài tập lớn. Cô cũng luôn nhắc nhở, giúp đỡ mỗi khi chúng em gặp khó khăn, nhờ vậy mà em đã hoàn thành bài tập lớn của nhóm mình đúng thời hạn được giao. Nếu không có sự hướng dẫn nhiệt tình của cô thì có lẽ chúng em đã khó có thể thực hiện được bài tập đúng theo mong muốn của mình.

Nhóm chúng em đã bỏ ra nhiều thời gian để tìm hiểu và trang bị thêm kiến thức nhằm phục vụ cho việc thực hiện ý tưởng, nhưng chắc chắn rằng nhóm chúng em sẽ không thể tránh khỏi những sai sót không đáng có vì kiến thức còn hạn chế. Chúng em hi vọng rằng sẽ nhận được những lời góp ý quý báu của cô để chúng em có thể hoàn thiện ý tưởng của nhóm một cách tốt nhất có thể.

TP. Hồ Chí Minh, ngày 08 tháng 12 năm 2023

Sinh viên thực hiện

Hà Văn Dũng

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Tp. Hồ Chí Minh, ngày 8 tháng 11 năm 2023

Giáo viên hướng dẫn

ThS. Trần Thị Dung

MỤC LỤC

PHẦN 1: DỊCH SÁCH

CƠ BẢN THIẾT KẾ HƯỚNG ĐỐI TƯỢNG

1. Khái niệm cơ bản về đối tượng và lớp

- Lớp (Class):
  - Lớp là một mô hình, một bản thiết kế hoặc một khuôn mẫu để tạo ra các đối tượng.
  - Lớp định nghĩa các thuộc tính (biến thành viên) và phương thức (hành vi) mà các đối tượng thuộc lớp đó có thể có.
  - Lớp là một thực thể trừu tượng, không thể trực tiếp sử dụng mà cần tạo ra các đối tượng từ lớp để sử dụng.
- Đối tượng (Object):
  - Đối tượng là một phiên bản cụ thể được tạo ra từ lớp.
  - Đối tượng có các thuộc tính và phương thức mà lớp đã định nghĩa.
  - Đối tượng là một thực thể cụ thể, có thể được tạo ra, sử dụng và tương tác trong chương trình.
- Ví dụ: Giả sử có một lớp đơn giản là "Hình tròn" (Circle) định nghĩa các thuộc tính như bán kính và phương thức tính diện tích. Bằng cách sử dụng lớp này, chúng ta có thể tạo ra nhiều đối tượng hình tròn với các bán kính khác nhau và tính diện tích của chúng.

```
// Định nghĩa lớp
class Circle {
Private:
    // Thuộc tính
    double radius;
public:
    // Phương thức tính diện tích
    double calculateArea() {
        return 3.14 * radius * radius;
    }
};
```

2. Hàm tạo và hàm hủy

- Hàm tạo có tên giống với tên lớp và được sử dụng để khởi tạo các thành phần của đối tượng khi nó được tạo ra. Hàm tạo không có kiểu trả về và có thể có tham số hoặc không có tham số. Một lớp có thể có nhiều hàm tạo với các đặc điểm khác nhau (gọi là hàm tạo nạp chồng).
- Hàm hủy có tên giống với tên lớp, nhưng được đặt trước bởi ký tự ~. Hàm hủy được sử dụng để giải phóng bất kỳ tài nguyên nào mà đối tượng đã cấp phát trong quá trình sống. Hàm hủy không có tham số và không có giá trị trả về.
- Ví dụ, dưới đây là một lớp Person có một hàm tạo có tham số để khởi tạo tên của một người, chúng ta có thể thêm một hàm hủy vào lớp Person để giải phóng tài nguyên khi đối tượng bị hủy:

```
class Person{
public:
    string name;
    // Hàm tạo có tham số
    Person(string personName){
        name = personName;
        cout << "Ham tao duoc goi. \nTen: "<<name<<endl;
    }
    // Hàm hủy
    ~Person(){
        cout << "Ham huy duoc goi. \nTen: "<<name<<endl;
    }
};
int main(){
    Person obj("Dung Van Ha");
}

/**OUTPUT:
Ham tao duoc goi.
Ten: Dung Van Ha
Ham huy duoc goi.
Ten: Dung Van Ha */
```

Trong ví dụ trên, hàm dựng Person nhận một tham số personName và gán giá trị của tham số đó cho thuộc tính name của đối tượng Person. Mỗi khi một đối tượng Person được tạo ra, hàm dựng sẽ được gọi và hiển thị thông báo trên màn hình. Hàm hủy Person được thêm vào lớp và sẽ được gọi khi một đối tượng Person bị hủy. Trong hàm hủy, chúng ta có thể thực hiện bất kỳ việc giải phóng tài nguyên nào cần thiết cho đối tượng.
Lưu ý rằng hàm dựng và hàm hủy là hai thành phần tùy chọn trong một lớp. Nếu không định nghĩa chúng, C++ sẽ tạo ra các hàm dựng và hàm hủy mặc định.

MỘT SỐ CẤU TRÚC DỮ LIỆU

1. Mảng (Array)

Trong C++, mảng là một cấu trúc dữ liệu cho phép lưu trữ một tập hợp các phần tử có cùng kiểu dữ liệu, có thể được khai báo và sử dụng như sau:

- Khai báo mảng: Cần chỉ định kiểu dữ liệu của các phần tử trong mảng và kích thước của mảng. Ví dụ dưới đây khai báo một mảng nguyên có tên myArray với kích thước 5:

```
int myArray[5];
```

- Truy cập và gán giá trị cho phần tử trong mảng: Các phần tử trong mảng được đánh số từ 0 đến kích thước mảng trừ 1, có thể truy cập vào các phần tử và gán giá trị cho chúng như sau:

```
myArray[0] = 10;  // Gán giá trị 10 cho phần tử đầu tiên
myArray[1] = 20;  // Gán giá trị 20 cho phần tử thứ hai
int x = myArray[0];// Gán giá trị của phần tử đầu tiên vào biến x
```

- Khởi tạo mảng: Có thể khởi tạo mảng cùng lúc với khai báo bằng cách cung cấp giá trị cho từng phần tử. Ví dụ:

```
int myArray[] = {10, 20, 30, 40, 50};// Khởi tạo mảng với giá trị ban đầu
```

- Lặp qua các phần tử trong mảng: Để lặp qua từng phần tử trong mảng, có thể sử dụng vòng lặp for. Ví dụ:

```
for (int i = 0; i < 5; i++) {
    cout << myArray[i] << " ";
    // In giá trị của từng phần tử
}
```

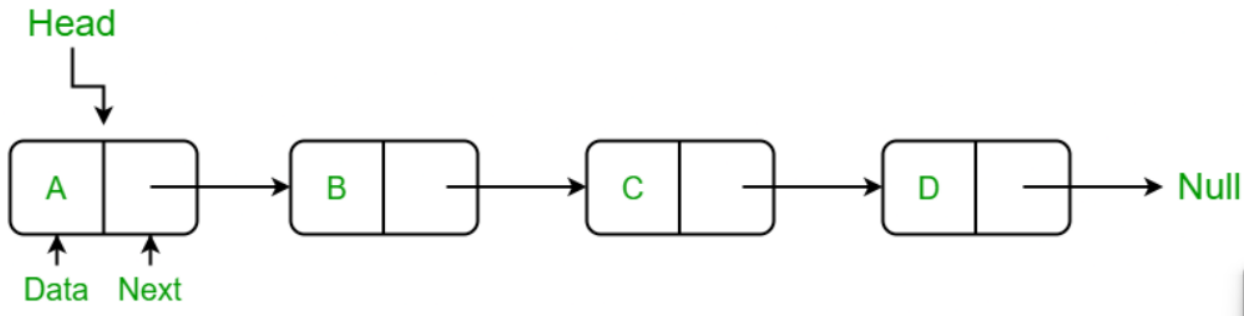
- Mảng đa chiều: Tức là mảng có nhiều chiều. Ví dụ dưới đây khai báo một mảng hai chiều:

```
int myArray[3][4]; // Mảng 2 chiều, có 3 hàng và 4 cột
```

2. Danh sách liên kết

Danh sách liên kết là một cấu trúc dữ liệu được sử dụng để lưu trữ các phần tử tương tự như mảng như có nhiều điểm khác biệt

- Tính chất:
    - DSLK có thể mở rộng và thu hẹp một cách linh hoạt.
    - Phần tử cuối cùng trong DSLK trỏ vào NULL
    - Không lãng phí bộ nhớ nhưng cần thêm bộ nhớ để lưu phần con trỏ
    - Các phần tử trong DSLK được gọi là Node, được cấp phát động
    - Đây là cấu trúc dữ liệu cấp phát động nên khi còn bộ nhớ thì sẽ còn thêm được phần tử vào DSLK



- So sánh mảng và DSLK:

- Mảng: Bộ nhớ được cấp phát cho mảng là 1 khối bộ nhớ liên tiếp nhau, các phần tử trong mảng có thể truy cập thông qua chỉ số với độ phức tạp O(1)

Ưu điểm	**Nhược điểm**
Đơn giản và dễ sử dụng	Có thể gây lãng phí bộ nhớ nếu không sử dụng hết bộ nhớ xin cấp phát cho mảng.
Truy cập mảng với độ phức tạp là hằng số	Kích thước mảng là cố định
	Bộ nhớ cấp phát theo khối
	Việc chèn và xóa phần tử khó khăn

- DSLK: Bộ nhớ cấp phát cho các node trong DSLK có thể nằm rải rác nhau trong bộ nhớ.

Ưu điểm	**Nhược điểm**
Có thể mở rộng với độ phức tạp là hằng số	Khó khăn trong việc truy cập 1 phần tử ở vị trí bất kỳ (O(n))
Dễ dàng mở rộng và thu hẹp kích thước	Khó khăn trong việc cài đặt
Có thể cấp phát số lượng lớn các node tùy vào bộ nhớ	Tốn thêm bộ nhớ cho phần tham chiếu bổ sung

- Độ phức tạp của các thao tác với mảng và DSLK:

Thao tác	**DSLK**	**Mảng**
Truy xuất phần tử	O(n)	O(1)
Chèn/Xóa ở đầu	O(1)	O(n) nếu mảng chưa full
Chèn ở cuối	O(n)	O(1) nếu mảng chưa

		full
Xóa ở cuối	O(n)	O(1)
Chèn giữa	O(n)	O(n) nếu mảng chưa full
Xóa giữa	O(n)	O(n)

Danh sách liên kết đơn:
Bao gồm 1 số các node trong đó mỗi node có con trỏ next tới node tiếp theo nó trong DSLK. Liên kết của node cuối cùng trong DSLK là con trỏ NULL.

- Cấu trúc một node của DSLK:

```
struct node{
    int data;
    node *next; // link
};
```

- Giải thích ý nghĩa của cấu trúc node:
    - Node ở đây có phần dữ liệu là một số nguyên lưu ở data, ngoài ra còn có 1 phần con trỏ tới chính struct node. Phần này chính là địa chỉ của node tiếp theo của nó trong DSLK.
    - Như vậy mỗi node sẽ có dữ liệu của nó và có địa chỉ của node tiếp sau nó. Đối với con trỏ cuối cùng trong DSLK thì phần địa chỉ này sẽ là con trỏ NULL.

- Các thao tác trên DSLK đơn:
    - Tạo node mới

```
Node *createNode(int x){
    Node *newNode = new Node();
    newNode->data = x;
```

```
    newNode->next = NULL;
    return newNode;
}
```

- Thêm phần tử vào đầu DSLK

```
void addFirst(int x){
    Node *newNode = createNode(x);
    if (head == NULL){
        head = newNode;
        return;
    }
    newNode->next = head;
    head = newNode;
}
```

- Thêm phần tử vào cuối DSLK

```
void addLast(int x){
    Node *newNode = createNode(x);
    if (head == NULL){
        head = newNode;
        return;
    }
    Node *tmp = head;
    while (tmp->next != NULL)
        tmp = tmp->next;
    tmp->next = newNode;
}
```

- Thêm phần tử vào bất kỳ trong DSLK

```
void insert(int x, int pos){
    int s = size();
    if (pos < 1 || pos > s + 1)
        return;
    if (pos == 1) {
        addFirst(x);
        return;
    }
    Node *tmp = head;
    for (int i = 1; i <= pos - 2; i++)
        tmp = tmp->next;
    // tmp: pos - 1
    Node *newNode = createNode(x);
    newNode->next = tmp->next;
    tmp->next = newNode;
}
```

- Xóa phần tử đầu

```
void removeFirst(){
    if (head == NULL)
        return;
    Node *tmp = head;
    head = head->next;
    delete tmp;
}
```

- Xóa phần tử cuối

```
void removeLast(){
    Node *tmp = head;
    if (tmp->next == NULL){
        head = NULL;
        delete tmp;
    }else{
        while (tmp->next->next != NULL)
            tmp = tmp->next;
        Node *last = tmp->next;
        tmp->next = NULL;
        delete last;
    }
}
```

- Xóa phần tử bất kỳ

```
void remove(int pos){
    if (pos < 1 || pos > size()) return;
    if (pos == 1)
        removeFirst();
    else {
        Node *tmp = head;
        for (int i = 1; i <= pos - 2; i++)
            tmp = tmp->next;
        Node *nodePos = tmp->next;
        tmp->next = nodePos->next;
        delete nodePos;
    }
}
```

3. Ngăn xếp (stack)

Là một cấu trúc dữ liệu mà các phần tử được thêm vào và lấy ra theo nguyên tắc LIFO (Last-In-First-Out), tức là phần tử cuối cùng được thêm vào ngăn xếp sẽ được lấy ra đầu tiên.

- Các thao tác được cài đặt bằng DSLK:

  - Cấu trúc

```
struct StackItem{
    char data;
    StackItem *next;
};
```

  - Tạo

```
StackItem *makeStackItem(int val){
    StackItem *newStackItem = new StackItem();
    newStackItem->data = val;
    newStackItem->next = NULL;
    return newStackItem;
}
```

  - Kiểm tra stack rỗng?

```
bool empty(){
    return top == NULL;
}
```

  - Thêm phần tử vào đỉnh stack

```
void push(int val){
    StackItem *newStackItem = makeStackItem(val);
    if (empty())
        top = newStackItem;
    else{
        newStackItem->next = top;
        top = newStackItem;
    }
}
```

- Trả về phần tử đỉnh stack

```
int peek(){
    return top->data;
}
```

- Xóa phần tử đỉnh stack

```
void pop(){
    if (!empty()){
        StackItem *first = top;
        top = top->next;
        delete first;
    }
}
```

4. Hàng đợi (queue)

Là một cấu trúc dữ liệu mà các phần tử được thêm vào ở cuối hàng và được lấy ra từ đầu hàng theo nguyên tắc FIFO (First-In-First-Out), tức là phần tử đầu tiên được thêm vào hàng sẽ được lấy ra đầu tiên.

- Các thao tác cài đặt bằng DSLK:
  - Cấu trúc

```
struct QueueItem{
    int data;
    QueueItem *next;
};
```

- Tạo phần tử

```
QueueItem *makeQueueItem(int val){
    QueueItem *newQueue = new QueueItem();
    newQueue->data = val;
    newQueue->next = NULL;
    return newQueue;
}
```

- Kiểm tra rỗng

```
bool empty(){
    return head == NULL;
}
```

- Thêm phần tử vào queue

```
void enqueue(int data){
   QueueItem *newQueueItem= makeQueueItem(data);
    if (head == NULL)
        head = newQueueItem;
    else if{
    QueueItem *tem = head;
      while (tem->next != NULL) {
         tem = tem->next;
      }
      tem->next = newQueueItem;
    }
}
```

- Lấy giá trị phần tử queue

```
int front(){ return head->data;}
```

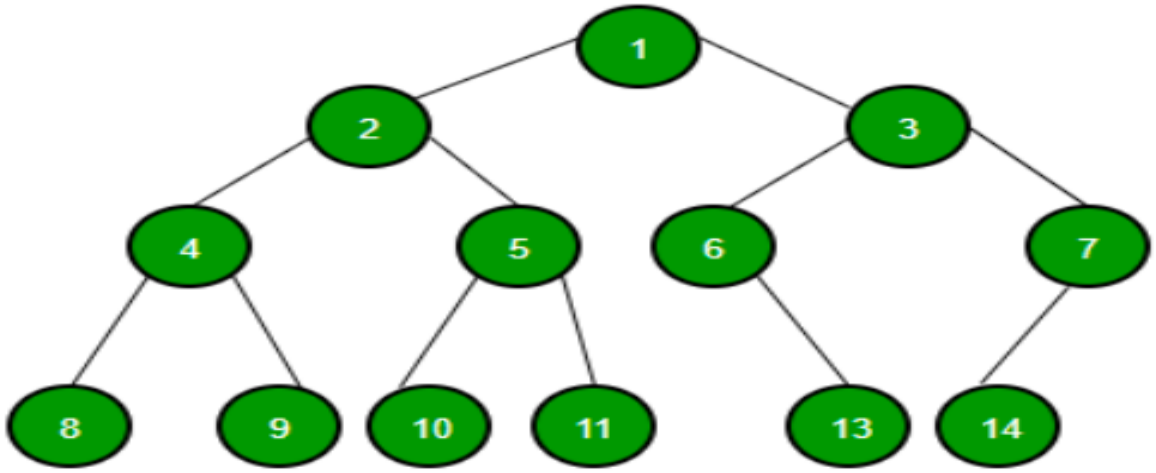
- Xóa 1 phần tử queue

```
void dequeue(){
    QueueItem *first = head;
    if (head->next == NULL) {
        head = NULL;
        delete first;
    }
    head = head->next;
    delete first;
}
```

5. Cây và cây nhị phân

5.1. Cây

Cấu trúc dữ liệu cây là một cấu trúc phân cấp được sử dụng để tổ chức dữ liệu theo mô hình cây. Nó bao gồm các nút và mối quan hệ cha-con giữa chúng.

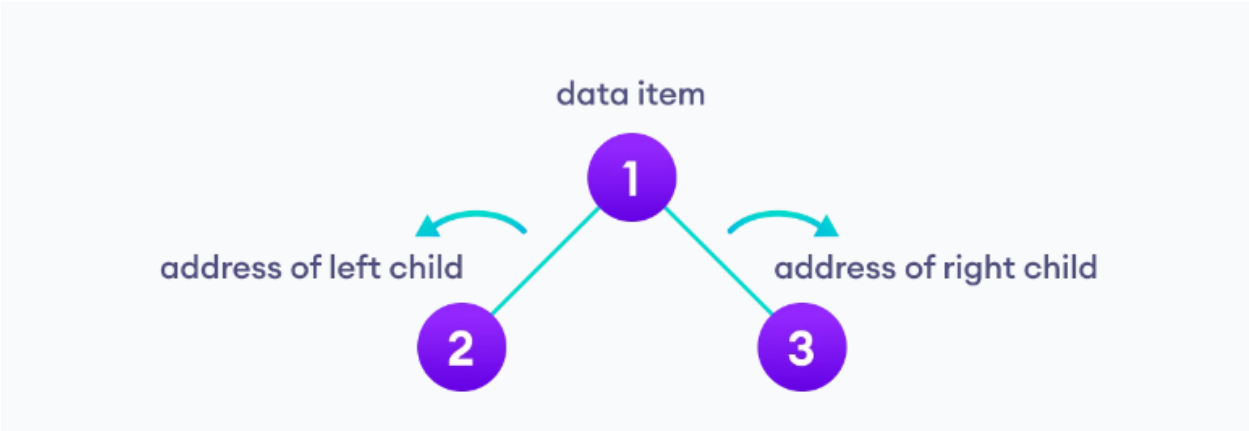


Một số thuật ngữ:

- Root - Nút gốc: Là nút trên cùng của cây, không có nút cha.
- Leaf - Nút lá: Là các nút không có nút con.
- Parent node - Nút cha: Là nút mà một nút con nằm dưới nó.
- Child node - Nút con: Là các nút nằm dưới một nút cha trong cây.
- Internal node - Nút trung gian / nút trong: Là các nút không phải là nút lá, có ít nhất một nút con.
- Sibling - Nút anh em: Là các nút có cùng nút cha.
- Level of node - Mức của nút: Là vị trí của một nút trong cây, tính từ nút gốc. Mức của nút gốc là 0, mức của các nút con tăng lên.
- Degree of node - Bậc của nút: Số lượng nút con mà một nút có.
- Degree of tree - Bậc của cây: Là bậc lớn nhất của các nút trong cây.
- Depth of tree - Chiều sâu/độ cao của cây: Là số lượng mức trong cây, tính từ nút gốc. Độ cao của cây rỗng (null tree) được xác định là -1 và độ cao của cây chỉ có một nút là 0.
- Path - Đường đi: Là một chuỗi các nút kết nối từ nút gốc đến một nút cụ thể trong cây.
- Path length - Chiều dài của đường đi: Là số lượng cạnh hoặc khoảng cách giữa các nút trên một đường đi trong cây.

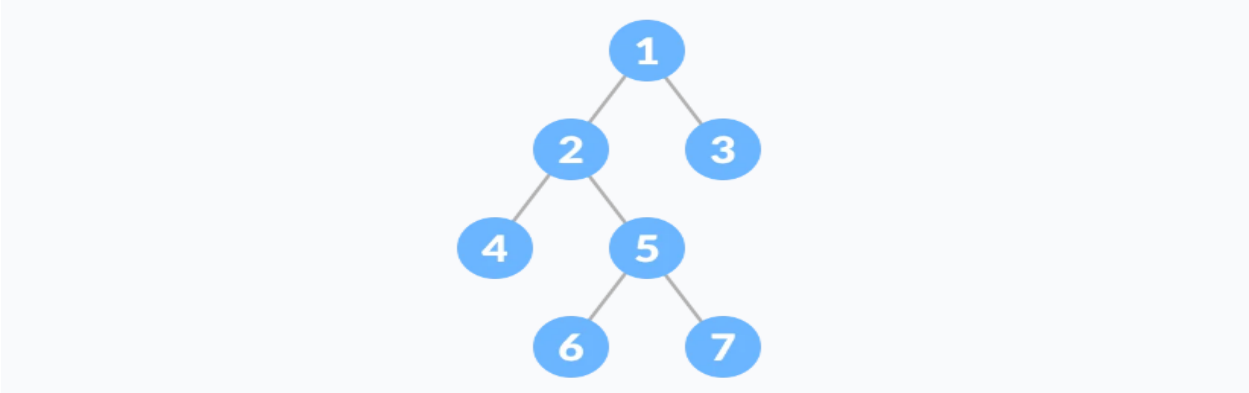
5.2. Cây nhị phân

Cây nhị phân trong đó mỗi nút cha có thể có tối đa hai nút con. Mỗi nút của cây nhị phân bao gồm ba thành phần: dữ liệu, địa chỉ của con bên trái và địa chỉ của con bên phải.

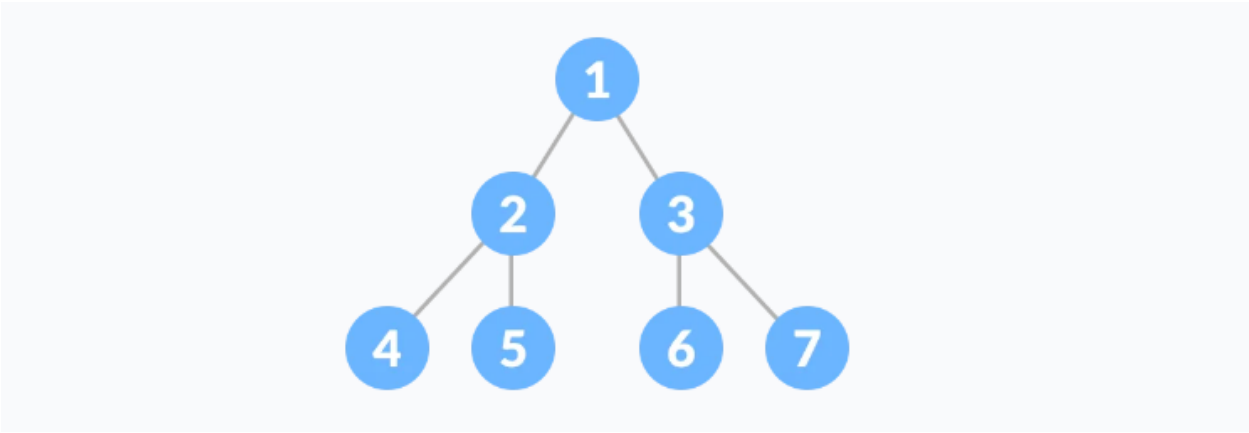


5.2.1. Các loại cây nhị phân:

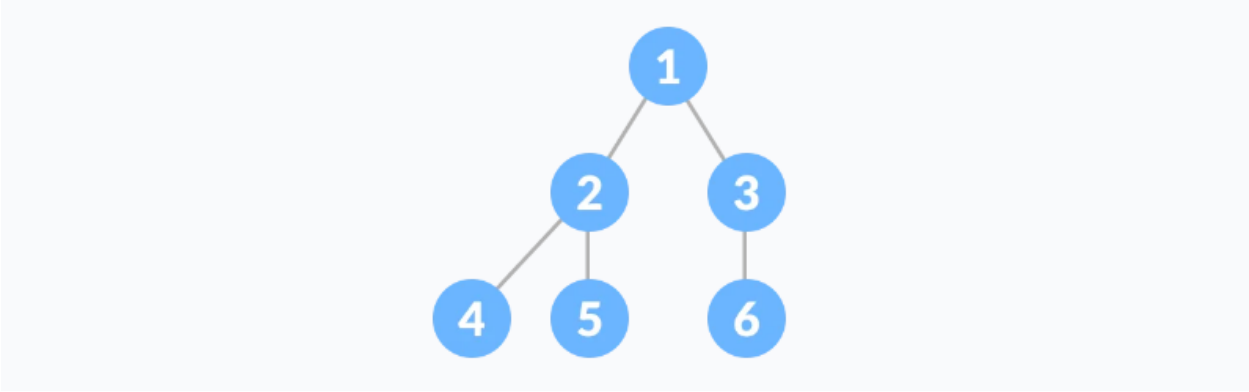
- Cây nhị phân đầy đủ: là một loại cây nhị phân đặc biệt trong đó mỗi nút cha có hai hoặc không có nút con.



- Cây nhị phân hoàn hảo: là một loại cây nhị phân trong đó mỗi nút bên trong có chính xác hai nút con và tất cả các nút lá đều có cùng cấp độ.



- Cây nhị phân hoàn chỉnh: cũng giống như cây nhị phân đầy đủ nhưng có điểm khác biệt chính:
    - Mọi cấp độ phải được điền đầy đủ
    - Tất cả các lá phải nghiêng về bên trái.
    - Phần tử lá cuối cùng có thể không có anh chị em bên phải, tức là cây nhị phân hoàn chỉnh không nhất thiết phải là cây nhị phân đầy đủ.



5.2.2. Cài đặt cây nhị phân:

- Tạo cấu trúc cây:

```
struct Node
{
    DataType data;
    Node *left;
    Node *right;
};
```

- Tạo node cho cây:

```
Node *createNode(DataType v)
{
    Node *newNode = new Node();
    newNode->data = v;
    newNode->left = NULL;
    newNode->right = NULL;
    return newNode;
}
```

- Thêm node vào bên trái:

```
void insertLeft(Node *p, Item v)
{
    if (p == NULL)
        cout << "Cannot insert to a NULL node. \n";
    else if (p->left != NULL)
        cout << "Left child existed. \n";
    else
        p->left = createNode(v);
}
```

- Thêm node vào bên phải:

```
void insertRight(Node *p, Item v)
{
    if (p == NULL)
        cout << "Cannot insert to a NULL node. \n";
```

```
    else if (p->right != NULL)
        cout << "Right child existed. \n";
    else
        p->right = createNode(v);
}
```

- Xóa node bên trái:

```
void deleteLeft(Node *p)
{
    if (p == NULL)
        cout << "Current node is NULL\n";
    else if (p->left == NULL)
        cout << "No left child to delete\n";
    else
    {
        Node *q = p->left;
        if (q->left != NULL || q->right != NULL)
            cout << "Cannot delete non-leaf node\n";
        else
        {
            Item value = q->data;
            p->left = NULL;
            delete q;
        }
    }
}
```

- Xóa node bên phải:

```
void deleteRight(Node *p)
{
    if (p == NULL)
        cout << "Current node is NULL\n";
    else if (p->right == NULL)
        cout << "No right child to delete\n";
    else
    {
```

```
        Node *q = p->right;
        if (q->right != NULL || q->left != NULL)
            cout << "Cannot delete non-leaf node\n";
        else
        {
            Item value = q->data;
            p->right = NULL;
            delete q;
        }
    }
}
```

- Xóa cây:

```
void deleteTree(Node *&root)
{
    if (root != NULL){
        deleteTree(root->left);
        deleteTree(root->right);
        delete root;
        root = NULL;
    }
}
```

- Duyệt cây NLR (Node - Left - Right)

```
void preOrder(Node *root)
{
    if (root != NULL){
        cout << root->data << "\t";
        preOrder(root->left);
        preOrder(root->right);
    }
}
```

- Duyệt cây LRN (Left - Right - Node)

```
void posOrder(Node *root)
{
    if (root != NULL){
        posOrder(root->left);
        posOrder(root->right);
        cout << root->data << "\t";
    }
}
```

- Duyệt cây LNR (Left - Node - Right)

```
void inOrder(Node *root)
{
    if (root != NULL){
        inOrder(root->left);
        cout << root->data << "\t";
        inOrder(root->right);
    }
}
```

6. Vung đống (Heap)

Heap là một cấu trúc dữ liệu được sử dụng để lưu trữ một tập hợp các phần tử sao cho các phần tử này luôn tuân theo một quy tắc sắp xếp được xác định trước. Cụ thể, trong một heap, mỗi phần tử có giá trị lớn hơn hoặc bằng (hoặc nhỏ hơn hoặc bằng) tất cả các phần tử con của nó.

Heap được sử dụng khi cần tìm kiếm phần tử lớn nhất (hoặc nhỏ nhất) trong một tập hợp phần tử và các thao tác thêm, xóa phần tử thường xuyên.

Heap được chia thành hai loại chính:

- Max Heap: giá trị của mỗi nút cha lớn hơn hoặc bằng giá trị của cả hai nút con. Nút gốc (nút trên cùng) luôn có giá trị lớn nhất trong Heap.

- Min Heap: giá trị của mỗi nút cha nhỏ hơn hoặc bằng giá trị của cả hai nút con. Nút gốc của luôn có giá trị nhỏ nhất trong Heap.

Các phép toán cơ bản trong Heap bao gồm:

- Insert: Chèn một phần tử mới vào Heap và duy trì thuộc tính Heap.
- Delete: Xóa phần tử đầu tiên (phần tử lớn nhất trong Max Heap hoặc phần tử nhỏ nhất trong Min Heap) và duy trì thuộc tính Heap.
- Peek: Truy cập phần tử đầu tiên mà không xóa nó khỏi Heap.
- Heapify: Xây dựng lại Heap từ một mảng không tuân theo thuộc tính Heap.

MỘT SỐ THUẬT TOÁN

1. Sắp xếp (Sorting)

1.1. Sắp xếp chọn (Selection sort)

Sắp xếp chọn là một thuật toán sắp xếp hoạt động bằng cách chọn liên tục phần tử nhỏ nhất (hoặc lớn nhất) từ phần chưa được sắp xếp của danh sách và di chuyển nó đến phần được sắp xếp của danh sách.
Pseudocode:

- Duyệt từ đầu đến trước phần tử cuối cùng của mảng [i = 0, i = n - 2]
- Khởi tạo min_pos = i để chọn ra phần tử đang xét là phần tử nhỏ nhất
- Với mỗi vòng lặp i thực hiện duyệt từ đầu đến cuối mảng [j = 0, j = n - 1], kiểm tra xem giá trị tại min_pos có lớn hơn giá trị tại j hay không?
    Đúng: min_pos = j

Sai: tiếp tục lặp j

- Kết thúc vòng lặp j thực hiện đổi chỗ 2 giá trị tại min_pos và tại j
- Quay lại bước 1

Cài đặt:

```
void selectionSort(int arr[], int n){
   for (int i = 0; i <= n - 2; i++){
      int min_pos = i;
      for (int j = i; j <= n - 1; j++){
         if (a[j] < a[min_pos])
            min_pos = j;
      }
      swap(a[min_pos], a[i]);
   }
}
// Độ phức tạp: O(n^2)
```

1.2. Sắp xếp nổi bọt (Bubble Sort)

Bubble Sort là thuật toán sắp xếp hoạt động bằng cách liên tục hoán đổi các phần tử liền kề nếu chúng sai thứ tự. Thuật toán này không phù hợp với các tập dữ liệu lớn vì độ phức tạp về thời gian trung bình và trường hợp xấu nhất của nó khá cao.

Pseudocode:

- Duyệt từ đầu đến trước phần tử cuối cùng của mảng [i = 0, i = n - 2]
- Mỗi vòng lặp i duyệt từ i đến trước phần tử cuối cùng của mảng [j = i, j = n - 2], kiểm tra xem arr[j] > arr[j + 1]?
  Đúng: Thực hiện đổi chỗ arr[j] và arr[j + 1]
  Sai: Tiếp tục vòng lặp j
- Quay lại B1

Cài đặt:

```
void bubbleSort(int arr[], int n)
{
   for (int i = 0; i <= n - 2; i++)
   {
      for (int j = i; j <= n - 2; j++){
         if (arr[j] > arr[j + 1])
            swap(arr[j], arr[j + 1]);
      }
   }
}
// Độ phức tạp: O(n^2)
```

1.3. Sắp xếp chèn (Insertion sort)

Sắp xếp chèn là một thuật toán sắp xếp hoạt động tương tự như cách sắp xếp các quân bài trên tay. Mảng hầu như được chia thành một phần được sắp xếp và một phần chưa được sắp xếp. Các giá trị từ phần chưa sắp xếp được chọn và đặt vào đúng vị trí trong phần đã sắp xếp.

Pseudocode:

- Duyệt từ phần tử thứ i = 2 đến phần tử cuối cùng của mảng [2, n - 1]
- Khởi tạo j = i - 1 và key = arr[i]
- Lặp j >= 0 và arr[j] > key
  arr[j + 1] = arr[j]
  j--
- Kết thúc vòng lặp bây giờ j = -1 hoặc arr[j] <= key. Thực hiện cho

arr[j + 1] = key

Cài đặt:

```
void insertionSort(int arr[], int n)
{
    for(int i = 1; i <= n - 1; i++){
        int j = i - 1, key = arr[i];
        while(j >= 0 && arr[j] > key)
        {
            arr[j + 1] = arr[j];
            j--;
        }
        arr[j + 1] = key;
    }
}
```

1.4. Sắp xếp nhanh (Quick Sort)

QuickSort là một thuật toán sắp xếp dựa trên thuật toán chia để trị, chọn một phần tử làm trục và phân chia mảng đã cho xung quanh trục đã chọn bằng cách đặt trục vào đúng vị trí của nó trong mảng đã sắp xếp.

Quá trình chính trong quickSort là phân hoạch. Mục tiêu của các phân hoạch là đặt trục (bất kỳ phần tử nào có thể được chọn làm trục) vào đúng vị trí của nó trong mảng đã sắp xếp và đặt tất cả các phần tử nhỏ hơn ở bên trái trục và tất cả các phần tử lớn hơn ở bên phải trục.

Hai cách phân hoạch:

- Phân hoạch Lomuto
  Pseudocode

```
1. Lomuto-partition(A, l, r)
```

```
2. x = A[r]
3. i = l - 1
4. for j = l đến r - 1
            if (A[l] <= x)
                    ++i
                    Hoán đổi A[i] và A[j]
5. ++i
6. Hoán đổi A[i] và A[r]
7. return i
```

Cài đặt

```
int partitionLomuto(int a[], int l, int r){
   int pivot = a[r];// Chọn pt cuối làm chốt
   int i = l - 1;
   for (int j = l; j < r; j++){
      if (a[j] <= pivot){
         ++i;
         swap(a[i], a[j]);
      }
      // a[j] > pivot -> bỏ qua
   }
   ++i;
   swap(a[i], a[r]); // Đưa chốt về chính giữa
   return i;
}
```

- Phân hoạch Hoare
  Pseudocode

```
1. Hoare-Partition(A, l, r)
2. i = l - 1, j = r - 1, x = A[r]
3. While TRUE
            do
                    j = j - 1
            while (A[j] > x)
            do
                    i = i + 1
```

```
while (A[i] < x)
if i < j
        Hoán đổi A[i] và A[j]
else return j
```

Cài đặt

```
int partitionHoare(int a[], int l, int r){
    int i = l - 1, j = r + 1, pivot = a[r];
    while (true){
        do
            ++i;
        while (a[i] < pivot);

        do
            --j;
        while (a[j] > pivot);

        if (i < j)
            swap(a[i], a[j]);
        else
            return j;
    }
}
```

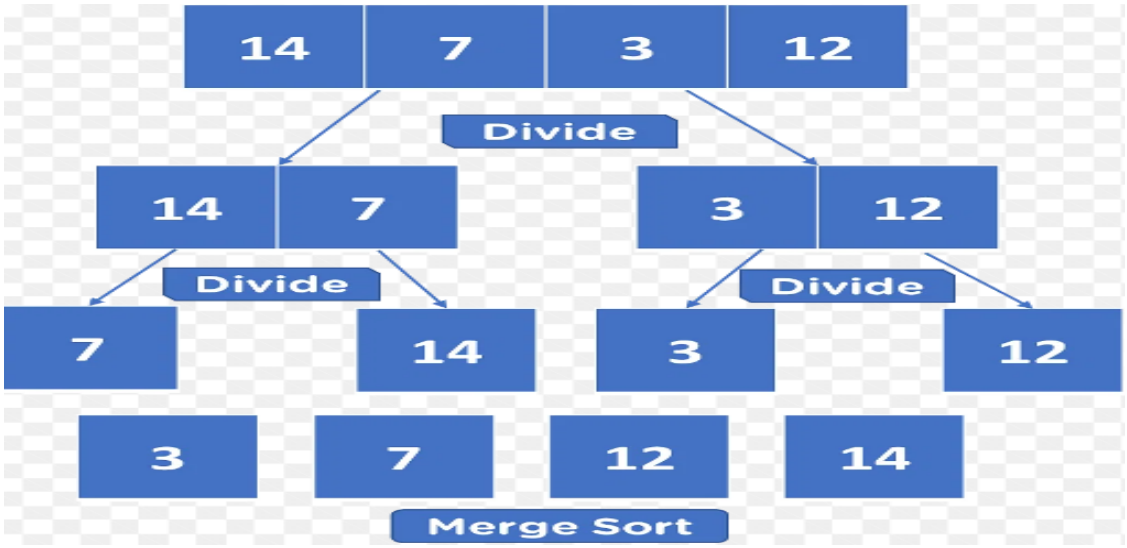
Giải thuật quick sort với 2 phân hoạch trên:

```
// Quick sort với phân hoạch Lomuto

void quickSortLomuto(int a[], int l, int r)
{
    if (l >= r)
        return;
    int m = partitionLomuto(a, l, r);
    quickSortLomuto(a, l, m - 1);
    quickSortLomuto(a, m + 1, r);
}
```

```
// Quick sort với phân hoạch Hoare

void quickSortHoare(int a[], int l, int r)
{
    if (l >= r)
        return;
    int m = partitionHoare(a, l, r);
    quickSortHoare(a, l, m - 1);
    quickSortHoare(a, m, r);
}
```

1.5. Sắp xếp trộn (Merge sort)

Sắp xếp trộn được định nghĩa là một thuật toán sắp xếp hoạt động bằng cách chia một mảng thành các mảng con nhỏ hơn, sắp xếp từng mảng con và sau đó hợp nhất các mảng con đã sắp xếp lại với nhau để tạo thành mảng được sắp xếp cuối cùng.




```
// Merge Sort
```

```
void mergeSort(int a[], int l, int r){
    if (l >= r)
        return;
    int m = (l + r) / 2;     // Lấy chỉ số giữa
    mergeSort(a, l, m);      // Sắp xếp bên trái
    mergeSort(a, m + 1, r);  // Sắp xếp bên phải
    // Trộn 2 bên thành được mảng sắp xếp
    merge(a, l, m, r);
}
```

```
// Thao tác merge của thuật toán merge Sort => merge mảng L[] và R[]
=> arr[]
void merge(int arr[], int l, int m, int r){
    int n1 = m - l + 1;
    int n2 = r - m;
    int L[n1], R[n2];

    for (int i = 0; i < n1; i++)
        L[i] = arr[l + i];
    for (int j = 0; j < n2; j++)
        R[j] = arr[m + 1 + j];

    int i = 0, j = 0;
    while (i < n1 && j < n2){
        if (L[i] <= R[j])
            arr[l++] = L[i++];
        else
            arr[l++] = R[j++];
    }

    while (i < n1)
        arr[l++] = L[i++];

    while (j < n2)
        arr[l++] = R[j++];
}
```

1.6. Sắp xếp vun đống (Heap Sort)

Heap Sort là một thuật toán sắp xếp dựa trên cấu trúc dữ liệu heap. Nó kết hợp cả tính chất của heap và quy tắc sắp xếp của thuật toán sắp xếp chọn (Selection Sort).

Quá trình Heap Sort gồm các bước:

- Xây dựng Max-Heap: Đầu tiên, chuyển đổi mảng ban đầu thành một Max-Heap bằng cách áp dụng phép heapify cho từng nút từ cuối cùng của mảng đến nút gốc (nút đầu tiên).
- Sắp xếp mảng: Sau khi xây dựng Max-Heap, phần tử lớn nhất sẽ nằm ở vị trí nút gốc (nút đầu tiên) của heap. Thực hiện các bước sau cho đến khi heap chỉ còn một phần tử:
    - Hoán đổi giữa nút gốc và phần tử cuối cùng của heap để đưa phần tử lớn nhất vào vị trí cuối cùng của mảng.
    - Giảm kích thước của heap đi 1 (loại bỏ phần tử cuối cùng) để không xét đến phần tử đã được sắp xếp.
    - Thực hiện phép heapify cho nút gốc để đảm bảo thuộc tính heap vẫn được duy trì.
- Kết quả: Sau khi hoàn thành quá trình sắp xếp, mảng sẽ chứa các phần tử theo thứ tự tăng dần.

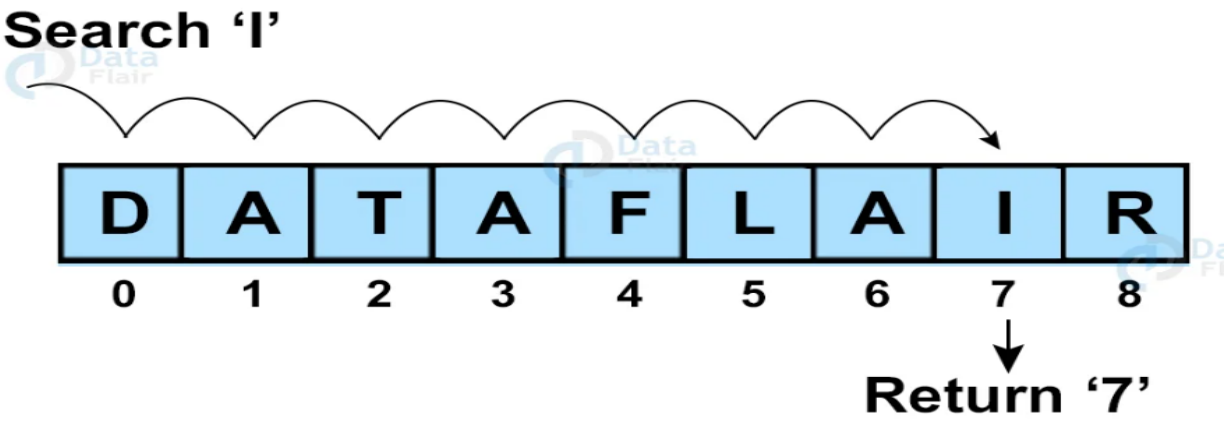
```
// Thao tác heapify để xây dựng max-heap
void heapify(int arr[], int n, int i)
{
    int largest = i;
    int left = 2 * i + 1;
    int right = 2 * i + 2;
    if (left < n && arr[left] > arr[largest])
        largest = left;
    if (right < n && arr[right] > arr[largest])
        largest = right;
    if (largest != i)
    {
        swap(arr[i], arr[largest]);
        heapify(arr, n, largest);
    }
}
```

```
// Thuật toán heap sort

void heapSort(int arr[], int n)
{
    // Xây dựng max - heap
    for (int i = n / 2 - 1; i >= 0; i--)
        heapify(arr, n, i);
    for (int i = n - 1; i >= 0; i--)
    {
        swap(arr[i], arr[0]);
        heapify(arr, i, 0);
    }
}
```

2. Tìm kiếm (Search)

2.1. Tìm kiếm tuyến tính (Linear Search)

Tìm kiếm tuyến tính được định nghĩa là thuật toán tìm kiếm tuần tự bắt đầu ở một đầu và đi qua từng phần tử của danh sách cho đến khi tìm thấy phần tử mong muốn, nếu không thì việc tìm kiếm sẽ tiếp tục cho đến hết tập dữ liệu.



Cách hoạt động

- Duyệt qua tất cả các phần tử
- Nếu tìm thấy bất kỳ phần tử nào bằng khóa thì tìm kiếm thành công.
- Nếu không tìm thấy phần tử nào bằng khóa, kết quả tìm kiếm sẽ là “Không tìm thấy kết quả khớp”.

```
// Thuật toán trả về true nếu key được tìm thấy trong mảng a[], ngược lại false

bool linearSearch(int arr[], int n, int key){
    for (int i = 0; i < n; i++){
        if (arr[i] == key)
            return true;
    }
    return false;
}
// Trường hợp tốt nhất: O(1)
// Trường hợp xấu nhất: O(N)
// Trường hợp trung bình: O(N)
```

2.2. Tìm kiếm nhị phân

Tìm kiếm nhị phân là một thuật toán tìm kiếm được sử dụng trong một mảng được sắp xếp bằng cách liên tục chia khoảng thời gian tìm kiếm thành một nửa.

Ý tưởng của tìm kiếm nhị phân là sử dụng mảng đã được sắp xếp và giảm độ phức tạp về thời gian xuống O(log N).

Pseudocode

- BinarySearch(arr, n, key)
- low = 0, high = n -1
- while low <= high
    mid = (low + high) / 2

```
        if arr[mid] = key
            return true
        else if(arr[mid] < key)
            low = mid + 1
        else
            high = mid - 1
    • return false
```

Cài đặt

```
// Thuật toán trả về true nếu key được tìm thấy trong mảng arr[],
ngược lại false
bool binarySearch(int arr[], int n, int key){
    int low = 0, high = n - 1;
    while (low <= high){
        int mid = (low + high) / 2;
        if (arr[mid] == key)
            return true;
        else if (arr[mid] < key)
            low = mid + 1;
        else
            high = mid - 1;
    }
    return false;
}
// Trường hợp tốt nhất: O(1)
// Trường hợp xấu nhất: O(logN)
// Trường hợp trung bình: O(logN)
```

PHẦN 2: CHƯƠNG TRÌNH

I. Lý do chọn đề tài quản lý

1. Nhu cầu thực tiễn: Trong hầu hết các tổ chức giáo dục, quản lý thông tin của sinh viên là một nhiệm vụ quan trọng. Các thông tin bao gồm danh tính, học phí, lịch học, điểm số, và nhiều thông tin khác. Một hệ thống quản lý sinh viên giúp tổ chức quản lý thông tin này hiệu quả hơn.
2. Tích hợp dữ liệu: Quản lý sinh viên liên quan đến nhiều dữ liệu khác nhau, từ thông tin cá nhân đến tình trạng học tập và tài chính. Cần một hệ thống để tổng hợp, lưu trữ và cung cấp dữ liệu này dễ dàng truy cập và quản lý.
3. Tăng cường hiệu suất: Một hệ thống quản lý sinh viên có thể giúp tổ chức giáo dục tăng cường hiệu suất. Nó có thể tự động hóa các quy trình quản lý, giảm thời gian và công sức của người làm việc và giảm thiểu sai sót.

II. Mô tả bài toán

Chương trình quản lý sinh viên là một chương trình dành để quản lý các sinh viên trong trong một trường đại học, cao đẳng,.. mục đích để tránh sự thống kê không chính xác và đưa ra các số liệu sinh viên theo nhiều tiêu chí khác nhau: thống kê điểm trung bình môn, thống kê số lượng nam/nữ, thống kê số lượng,..., muốn làm được chương trình này trước tiên chúng ta phải sử dụng cấu trúc class/struct để lưu trữ đối tượng nhiều thuộc tính, sau đó chúng ta sử dụng các cấu trúc dữ liệu, các thuật toán, cấu trúc lặp,... để quản lý chương trình.

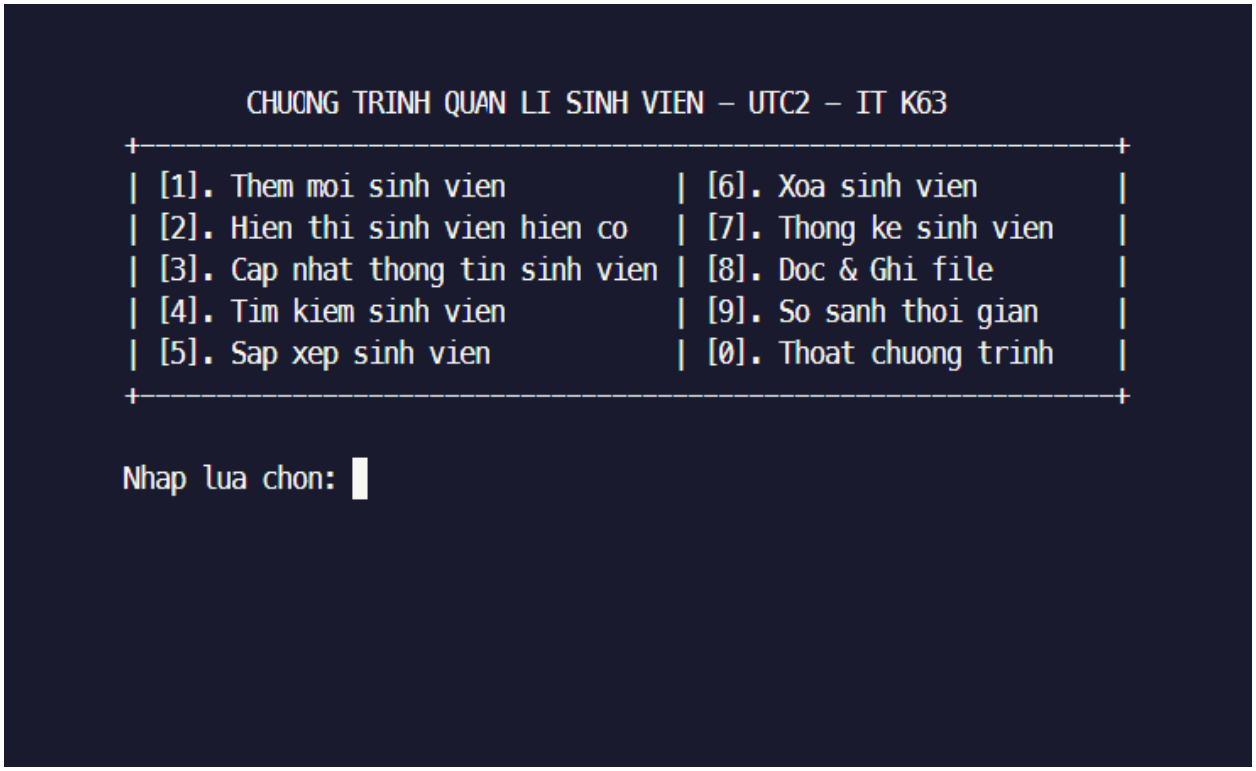
Chương trình bọn em gồm có 9 chức năng chính, trong đó có 1 số chức năng nổi bật như là: thêm, hiển thị, xóa, chỉnh sửa, tìm kiếm,

sắp xếp, so sánh thời gian thực hiện các thuật toán,... các cấu trúc dữ liệu được sử dụng là danh sách liên kết đôi cùng với 2 thuật toán nổi bật sắp xếp nhanh và sắp xếp trộn, tìm kiếm nhị phân,..

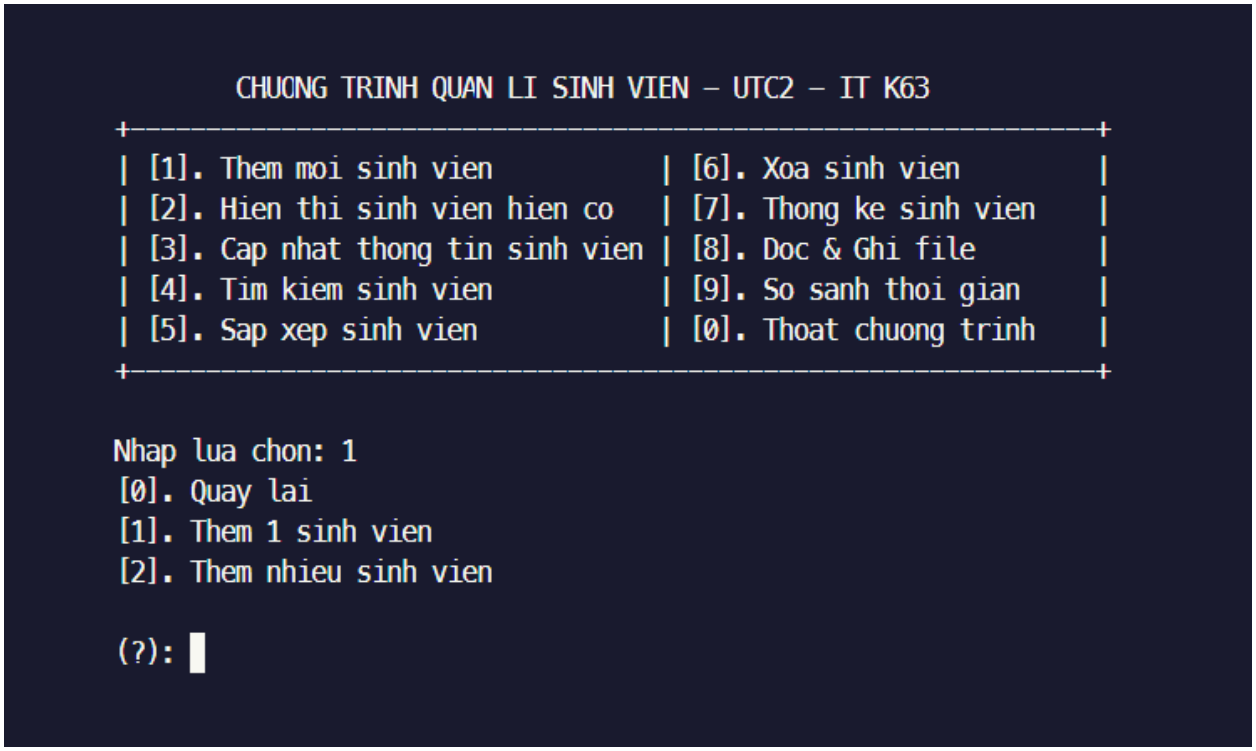
Lý do bọn em sử dụng cấu trúc dữ liệu danh sách liên kết đôi, các thuật toán sắp xếp nhanh, sắp xếp trộn, tìm kiếm nhị phân là thứ nhất là bọn em muốn ứng dụng từ những kiến thức đã học được ở môn cấu trúc dữ liệu và giải thuật, thứ hai là nó thích hợp với chương trình quản lý bọn em hướng tới, độ phức tạp thấp, dễ quản lý, dễ sử dụng.

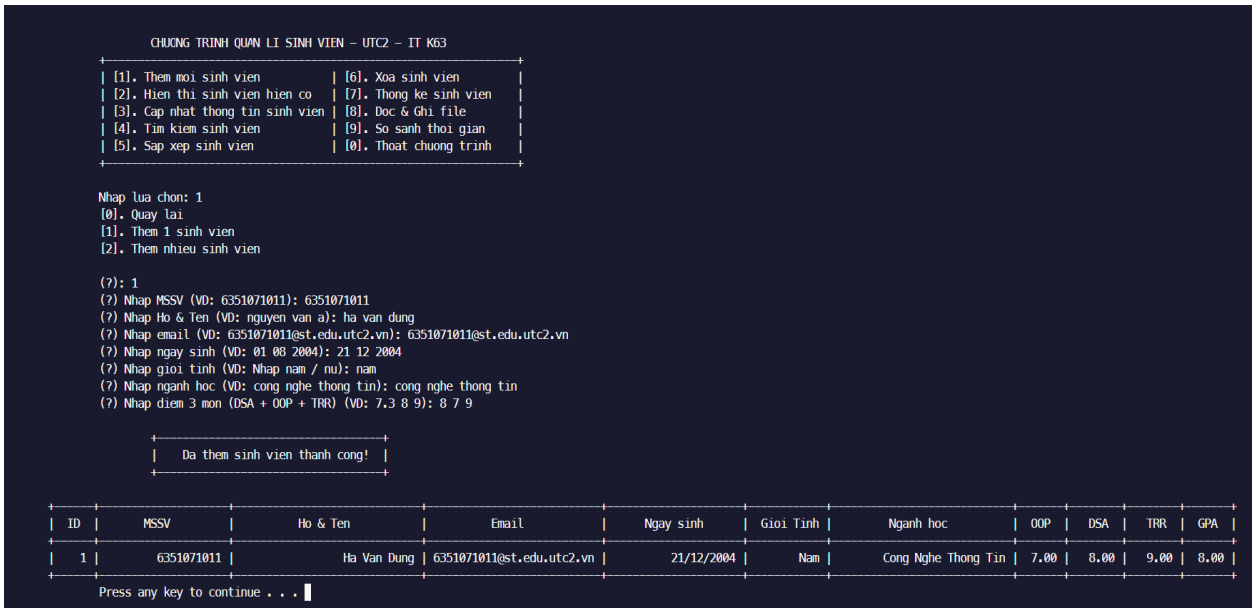
III. Giao diện chương trình

1. Giao diện menu chính của chương trình



2. Chức năng thêm sinh viên





Thêm 1 sinh viên

```
        [0]. Quay lai
        [1]. Them 1 sinh vien
        [2]. Them nhieu sinh vien

        (?): 2

        (?) So luong sinh vien: 2
        *Them sinh vien 1
        (?) Nhap MSSV (VD: 6351071011): 6351071012
        (?) Nhap Ho & Ten (VD: nguyen van a): nguyen huy
        (?) Nhap email (VD: 6351071011@st.edu.utc2.vn): 6351071012@st.edu.utc2.vn
        (?) Nhap ngay sinh (VD: 01 08 2004): 01 08 2004
        (?) Nhap gioi tinh (VD: Nhap nam / nu): nam
        (?) Nhap nganh hoc (VD: cong nghe thong tin): khoa hoc may tinh
        (?) Nhap diem 3 mon (DSA + OOP + TRR) (VD: 7.3 8 9): 9 8 7

        *Them sinh vien 2
        (?) Nhap MSSV (VD: 6351071011): 6351071011

                +----------------------------------------+
                |    Ma so sinh vien da ton tai roi !    |
                +----------------------------------------+


                +-----------------------------------+
                |  Da them 1 sinh vien thanh cong!  |
                +-----------------------------------+

+------+-------------+---------------------+-----------------------+---------------+-----------+-----------------------+-------+-------+-------+-------+
| ID   |    MSSV     |       Ho & Ten      |         Email         |   Ngay sinh   | Gioi Tinh |      Nganh hoc        |  OOP  |  DSA  |  TRR  |  GPA  |
+------+-------------+---------------------+-----------------------+---------------+-----------+-----------------------+-------+-------+-------+-------+
|    1 |  6351071011 |       Ha Van Dung   | 6351071011@st.edu.utc2.vn | 21/12/2004 |    Nam    | Cong Nghe Thong Tin   | 7.00  | 8.00  | 9.00  | 8.00  |
+------+-------------+---------------------+-----------------------+---------------+-----------+-----------------------+-------+-------+-------+-------+
|    3 |  6351071012 |       Nguyen Huy    | 6351071012@st.edu.utc2.vn | 01/08/2004 |    Nam    | Khoa Hoc May Tinh     | 8.00  | 9.00  | 7.00  | 8.00  |
+------+-------------+---------------------+-----------------------+---------------+-----------+-----------------------+-------+-------+-------+-------+
        Press any key to continue . . .
```

Thêm nhiều sinh viên

3. Chức năng hiển thị sinh viên hiện có

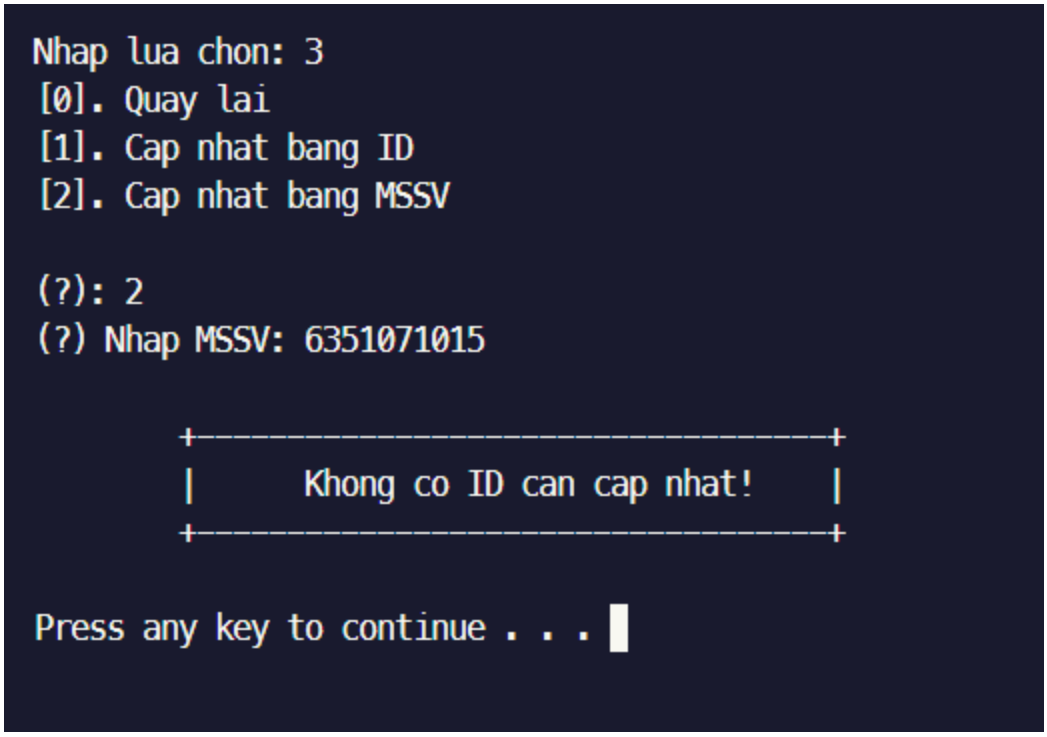
```
                CHUONG TRINH QUAN LI SINH VIEN – UTC2 – IT K63
        +-------------------------------------------------------------+
        | [1]. Them moi sinh vien          | [6]. Xoa sinh vien       |
        | [2]. Hien thi sinh vien hien co  | [7]. Thong ke sinh vien  |
        | [3]. Cap nhat thong tin sinh vien| [8]. Doc & Ghi file      |
        | [4]. Tim kiem sinh vien          | [9]. So sanh thoi gian   |
        | [5]. Sap xep sinh vien           | [0]. Thoat chuong trinh  |
        +-------------------------------------------------------------+

        Nhap lua chon: 2
+------+-------------+---------------------+-----------------------+---------------+-----------+-----------------------+-------+-------+-------+-------+
| ID   |    MSSV     |       Ho & Ten      |         Email         |   Ngay sinh   | Gioi Tinh |      Nganh hoc        |  OOP  |  DSA  |  TRR  |  GPA  |
+------+-------------+---------------------+-----------------------+---------------+-----------+-----------------------+-------+-------+-------+-------+
|    1 |  6351071011 |       Ha Van Dung   | 6351071011@st.edu.utc2.vn | 21/12/2004 |    Nam    | Cong Nghe Thong Tin   | 7.00  | 8.00  | 9.00  | 8.00  |
+------+-------------+---------------------+-----------------------+---------------+-----------+-----------------------+-------+-------+-------+-------+
|    3 |  6351071012 |       Nguyen Huy    | 6351071012@st.edu.utc2.vn | 01/08/2004 |    Nam    | Khoa Hoc May Tinh     | 8.00  | 9.00  | 7.00  | 8.00  |
+------+-------------+---------------------+-----------------------+---------------+-----------+-----------------------+-------+-------+-------+-------+
        Press any key to continue . . .
```

4. Chức năng cập nhật thông tin sinh viên

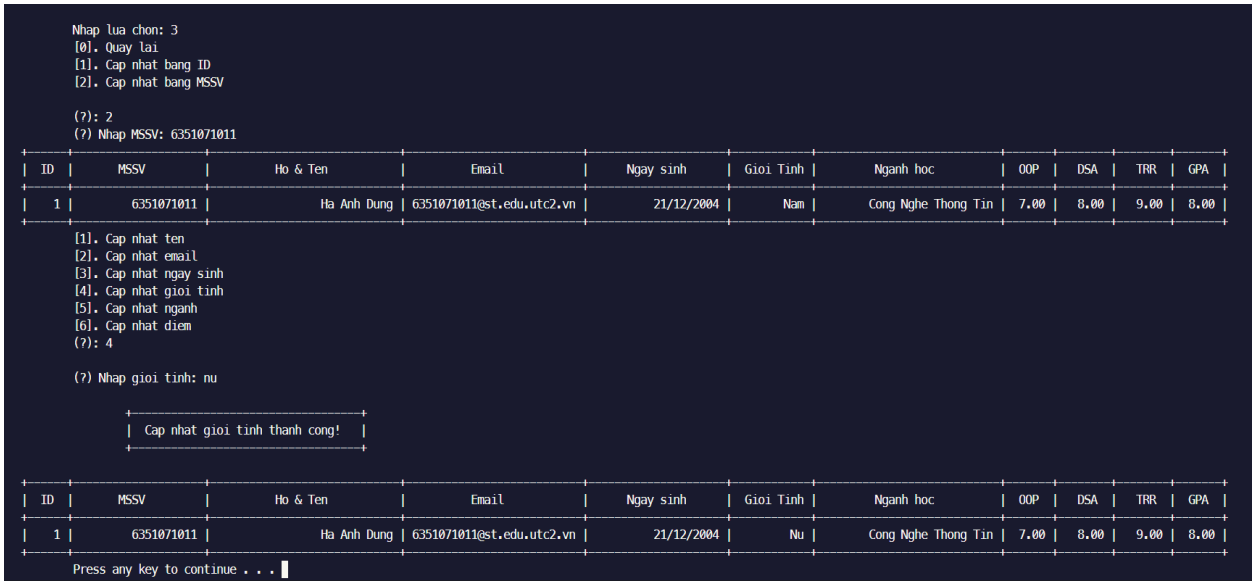
```
                CHUONG TRINH QUAN LI SINH VIEN – UTC2 – IT K63
+-------------------------------------------------------------------------+
| [1]. Them moi sinh vien           | [6]. Xoa sinh vien                  |
| [2]. Hien thi sinh vien hien co   | [7]. Thong ke sinh vien             |
| [3]. Cap nhat thong tin sinh vien | [8]. Doc & Ghi file                 |
| [4]. Tim kiem sinh vien           | [9]. So sanh thoi gian              |
| [5]. Sap xep sinh vien            | [0]. Thoat chuong trinh             |
+-------------------------------------------------------------------------+

Nhap lua chon: 3
[0]. Quay lai
[1]. Cap nhat bang ID
[2]. Cap nhat bang MSSV

(?):
```

```
Nhap lua chon: 3
[0]. Quay lai
[1]. Cap nhat bang ID
[2]. Cap nhat bang MSSV

(?): 1
(?) Nhap ID: 1
+------+------------+---------------+-------------------+------------+----------+-----------------------+------+------+------+------+
|  ID  |    MSSV    |     Ho & Ten  |       Email       |  Ngay sinh | Gioi Tinh|     Nganh hoc         | OOP  | DSA  | TRR  | GPA  |
+------+------------+---------------+-------------------+------------+----------+-----------------------+------+------+------+------+
|    1 | 6351071011 |  Ha Van Dung  | 6351071011@st.edu.utc2.vn | 21/12/2004 |    Nam   | Cong Nghe Thong Tin   | 7.00 | 8.00 | 9.00 | 8.00 |
+------+------------+---------------+-------------------+------------+----------+-----------------------+------+------+------+------+

[1]. Cap nhat ten
[2]. Cap nhat email
[3]. Cap nhat ngay sinh
[4]. Cap nhat gioi tinh
[5]. Cap nhat nganh
[6]. Cap nhat diem
(?): 1
(?) Nhap Ho & Ten: ha anh dung

+-------------------------------+
|    Cap nhat ten thanh cong!   |
+-------------------------------+

+------+------------+---------------+-------------------+------------+----------+-----------------------+------+------+------+------+
|  ID  |    MSSV    |     Ho & Ten  |       Email       |  Ngay sinh | Gioi Tinh|     Nganh hoc         | OOP  | DSA  | TRR  | GPA  |
+------+------------+---------------+-------------------+------------+----------+-----------------------+------+------+------+------+
|    1 | 6351071011 |  Ha Anh Dung  | 6351071011@st.edu.utc2.vn | 21/12/2004 |    Nam   | Cong Nghe Thong Tin   | 7.00 | 8.00 | 9.00 | 8.00 |
+------+------------+---------------+-------------------+------------+----------+-----------------------+------+------+------+------+
Press any key to continue . . .
```

Cập nhật tên cho sinh viên có ID là 1 thành công



Cập nhật MSSV không tồn tại trong hệ thống



Cập nhật giới tính cho sinh viên có MSSV là 6351071011 thành công

5. Chức năng tìm kiếm sinh viên

```
        CHUONG TRINH QUAN LI SINH VIEN – UTC2 – IT K63
+-------------------------------------------------------------------+
| [1]. Them moi sinh vien           | [6]. Xoa sinh vien            |
| [2]. Hien thi sinh vien hien co   | [7]. Thong ke sinh vien       |
| [3]. Cap nhat thong tin sinh vien | [8]. Doc & Ghi file           |
| [4]. Tim kiem sinh vien           | [9]. So sanh thoi gian        |
| [5]. Sap xep sinh vien            | [0]. Thoat chuong trinh       |
+-------------------------------------------------------------------+

Nhap lua chon: 4
[0]. Quay lai
[1]. Tim kiem bang ID
[2]. Tim kiem bang Ten

(?):
```

```
Nhap lua chon: 4
[0]. Quay lai
[1]. Tim kiem bang ID
[2]. Tim kiem bang Ten

(?): 1
(?) Nhap ID: 1
+------+--------------+-----------------+-----------------------+--------------+-----------+-----------------------+------+------+------+------+
| ID   |     MSSV     |      Ho & Ten   |         Email         |   Ngay sinh  | Gioi Tinh |      Nganh hoc        | OOP  | DSA  | TRR  | GPA  |
+------+--------------+-----------------+-----------------------+--------------+-----------+-----------------------+------+------+------+------+
|    1 |   6351071011 |      Ha Anh Dung | 6351071011@st.edu.utc2.vn |  21/12/2004 |        Nu | Cong Nghe Thong Tin | 7.00 | 8.00 | 9.00 | 8.00 |
+------+--------------+-----------------+-----------------------+--------------+-----------+-----------------------+------+------+------+------+
Press any key to continue . . .
```

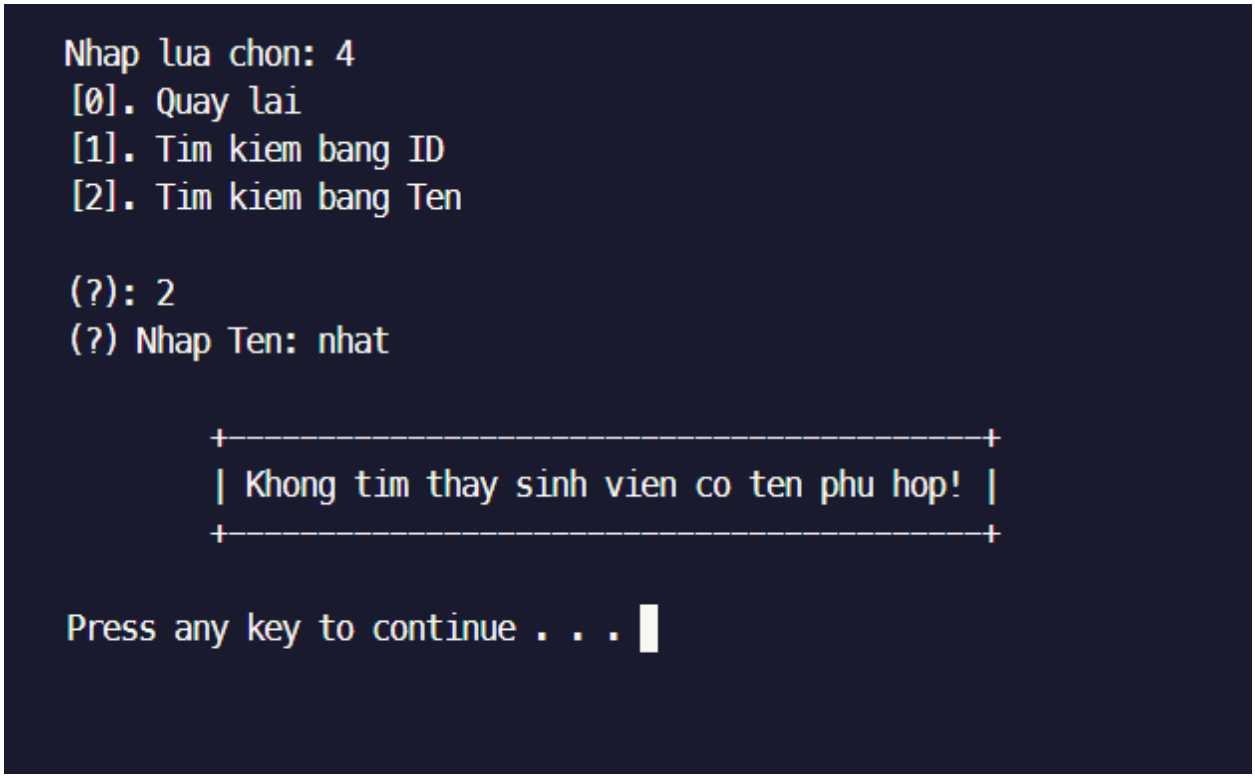
Tìm kiếm sinh viên có ID là 1 thành công

```
Nhap lua chon: 4
[0]. Quay lai
[1]. Tim kiem bang ID
[2]. Tim kiem bang Ten

(?): 1
(?) Nhap ID: 5

+-----------------------------------------+
| Khong tim thay sinh vien co ID phu hop! |
+-----------------------------------------+

Press any key to continue . . .
```

Tìm kiếm sinh viên có ID là 5 không thành công

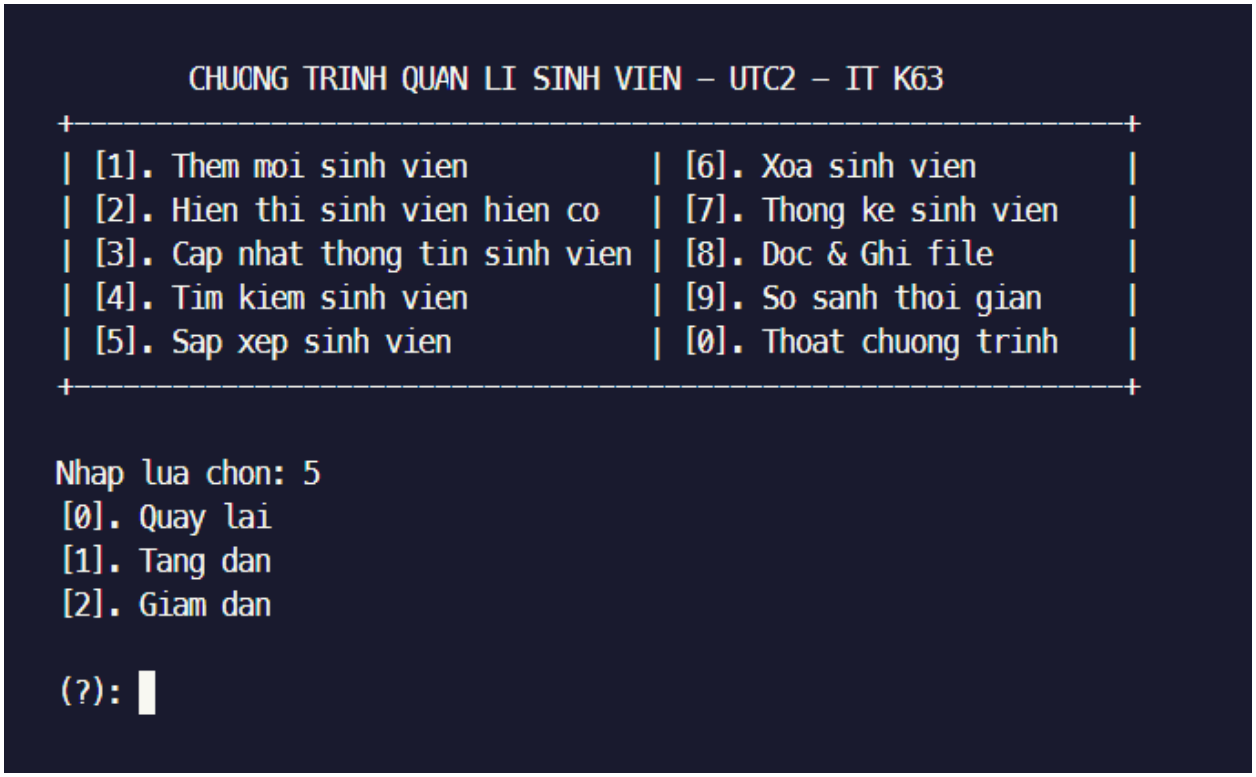
```
Nhap lua chon: 4
[0]. Quay lai
[1]. Tim kiem bang ID
[2]. Tim kiem bang Ten

(?): 2
(?) Nhap Ten: dung
+----+------------+---------------+--------------------+------------+-----------+--------------------+-------+-------+-------+-------+
| ID |    MSSV    |     Ho & Ten  |       Email        |  Ngay sinh | Gioi Tinh |     Nganh hoc      |  OOP  |  DSA  |  TRR  |  GPA  |
+----+------------+---------------+--------------------+------------+-----------+--------------------+-------+-------+-------+-------+
|  1 | 6351071011 |  Ha Anh Dung  | 6351071011@st.edu.utc2.vn | 21/12/2004 |    Nu     | Cong Nghe Thong Tin | 7.00  | 8.00  | 9.00  | 8.00  |
+----+------------+---------------+--------------------+------------+-----------+--------------------+-------+-------+-------+-------+
|  5 | 6351071015 | Nguyen Van Dung | 6351071015@st.edu.utc2.vn | 01/08/2004 |    Nam    |      Big Data      | 9.00  | 8.00  | 10.00 | 9.00  |
+----+------------+---------------+--------------------+------------+-----------+--------------------+-------+-------+-------+-------+
Press any key to continue . . .
```

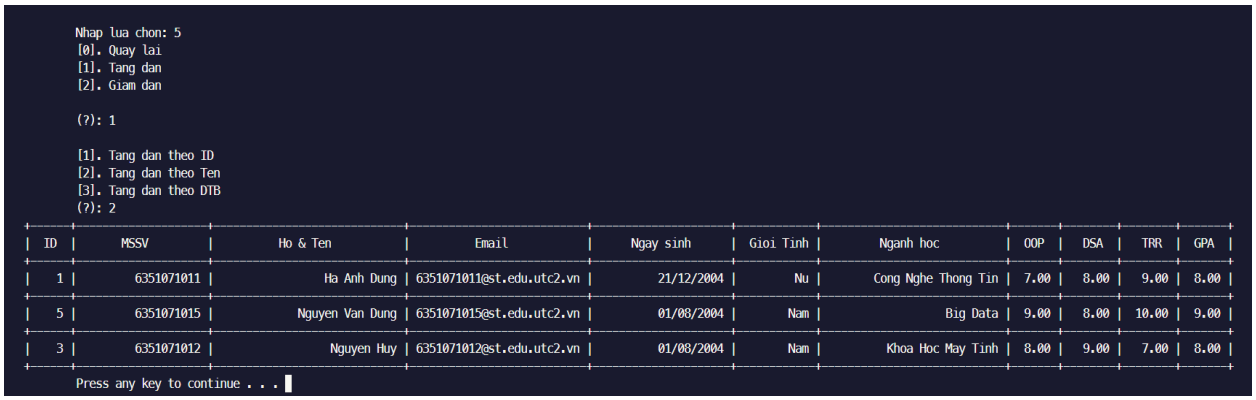
Tìm kiếm sinh viên có tên “dung” thành công



Tìm kiếm sinh viên có tên “nhat” không thành công

6. Chức năng sắp xếp sinh viên





ID	MSSV	Ho & Ten	Email	Ngay sinh	Gioi Tinh	Nganh hoc	OOP	DSA	TRR	GPA
1	6351071011	Ha Anh Dung	6351071011@st.edu.utc2.vn	21/12/2004	Nu	Cong Nghe Thong Tin	7.00	8.00	9.00	8.00
5	6351071015	Nguyen Van Dung	6351071015@st.edu.utc2.vn	01/08/2004	Nam	Big Data	9.00	8.00	10.00	9.00
3	6351071012	Nguyen Huy	6351071012@st.edu.utc2.vn	01/08/2004	Nam	Khoa Hoc May Tinh	8.00	9.00	7.00	8.00

Sắp xếp sinh viên tăng dần theo tên

```
Nhap lua chon: 5
[0]. Quay lai
[1]. Tang dan
[2]. Giam dan

(?): 1

[1]. Tang dan theo ID
[2]. Tang dan theo Ten
[3]. Tang dan theo DTB
(?): 1
```

ID	MSSV	Ho & Ten	Email	Ngay sinh	Gioi Tinh	Nganh hoc	OOP	DSA	TRR	GPA
1	6351071011	Ha Anh Dung	6351071011@st.edu.utc2.vn	21/12/2004	Nu	Cong Nghe Thong Tin	7.00	8.00	9.00	8.00
3	6351071012	Nguyen Huy	6351071012@st.edu.utc2.vn	01/08/2004	Nam	Khoa Hoc May Tinh	8.00	9.00	7.00	8.00
5	6351071015	Nguyen Van Dung	6351071015@st.edu.utc2.vn	01/08/2004	Nam	Big Data	9.00	8.00	10.00	9.00

```
Press any key to continue . . .
```

Sắp xếp sinh viên tăng dần theo ID

```
(?): 2

[1]. Giam dan theo ID
[2]. Giam dan theo Ten
[3]. Giam dan theo DTB
(?): 3
```

ID	MSSV	Ho & Ten	Email	Ngay sinh	Gioi Tinh	Nganh hoc	OOP	DSA	TRR	GPA
5	6351071015	Nguyen Van Dung	6351071015@st.edu.utc2.vn	01/08/2004	Nam	Big Data	9.00	8.00	10.00	9.00
3	6351071012	Nguyen Huy	6351071012@st.edu.utc2.vn	01/08/2004	Nam	Khoa Hoc May Tinh	8.00	9.00	7.00	8.00
1	6351071011	Ha Anh Dung	6351071011@st.edu.utc2.vn	21/12/2004	Nu	Cong Nghe Thong Tin	7.00	8.00	9.00	8.00

```
Press any key to continue . . .
```

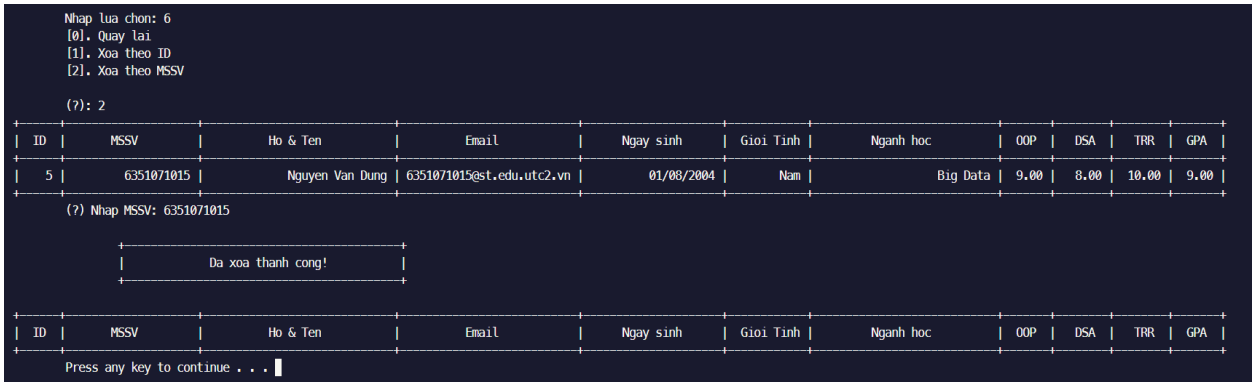
Sắp xếp sinh viên giảm dần theo điểm trung bình

7. Chức năng xóa sinh viên

```
        CHUONG TRINH QUAN LI SINH VIEN – UTC2 – IT K63
+-------------------------------------------------------------------+
| [1]. Them moi sinh vien           | [6]. Xoa sinh vien            |
| [2]. Hien thi sinh vien hien co   | [7]. Thong ke sinh vien       |
| [3]. Cap nhat thong tin sinh vien | [8]. Doc & Ghi file           |
| [4]. Tim kiem sinh vien           | [9]. So sanh thoi gian        |
| [5]. Sap xep sinh vien            | [0]. Thoat chuong trinh       |
+-------------------------------------------------------------------+

Nhap lua chon: 6
[0]. Quay lai
[1]. Xoa theo ID
[2]. Xoa theo MSSV

(?): 
```

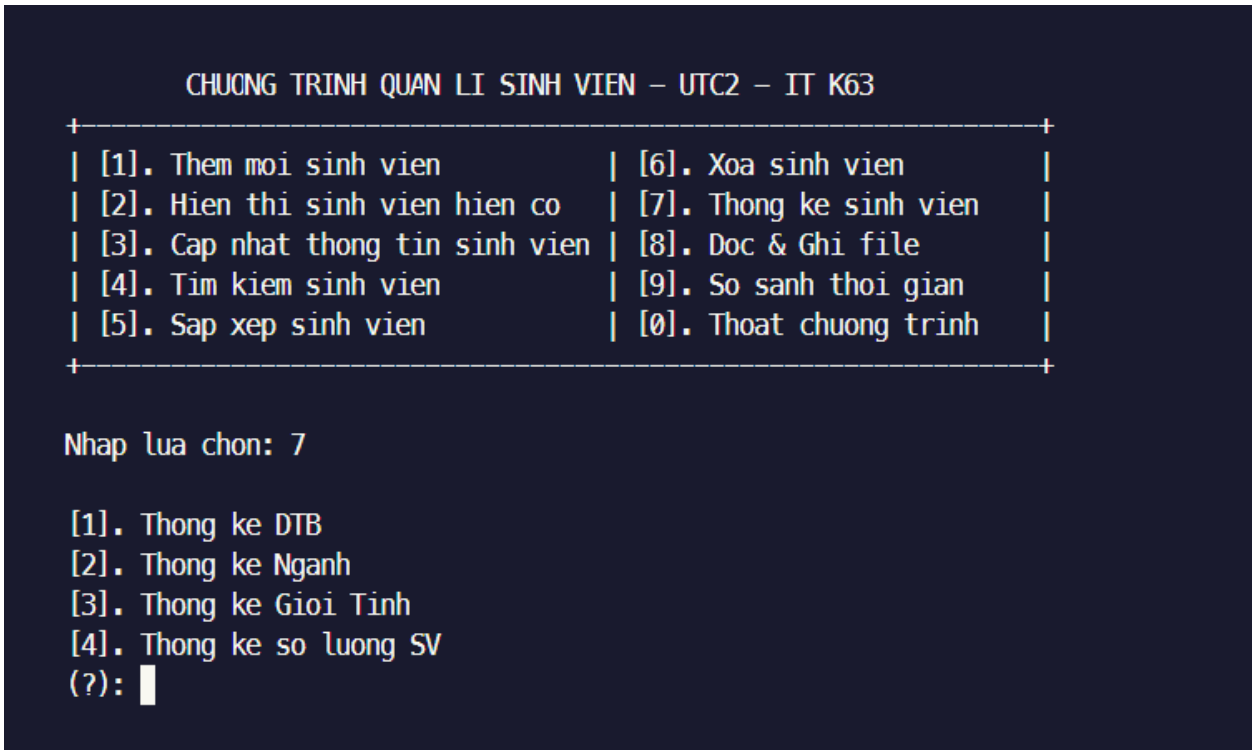
```
Nhap lua chon: 6
[0]. Quay lai
[1]. Xoa theo ID
[2]. Xoa theo MSSV

(?): 1
Nhap so luong SV can xoa: 2
+------+------------+---------------------+-----------------------+------------+-----------+---------------------+-------+-------+-------+-------+
|  ID  |    MSSV    |       Ho & Ten      |         Email         |  Ngay sinh | Gioi Tinh |     Nganh hoc       |  OOP  |  DSA  |  TRR  |  GPA  |
+------+------------+---------------------+-----------------------+------------+-----------+---------------------+-------+-------+-------+-------+
|    5 | 6351071015 |  Nguyen Van Dung    | 6351071015@st.edu.utc2.vn | 01/08/2004 |    Nam    |      Big Data       |  9.00 |  8.00 | 10.00 |  9.00 |
+------+------------+---------------------+-----------------------+------------+-----------+---------------------+-------+-------+-------+-------+
|    3 | 6351071012 |     Nguyen Huy      | 6351071012@st.edu.utc2.vn | 01/08/2004 |    Nam    |  Khoa Hoc May Tinh  |  8.00 |  9.00 |  7.00 |  8.00 |
+------+------------+---------------------+-----------------------+------------+-----------+---------------------+-------+-------+-------+-------+
|    1 | 6351071011 |     Ha Anh Dung     | 6351071011@st.edu.utc2.vn | 21/12/2004 |    Nu     | Cong Nghe Thong Tin |  7.00 |  8.00 |  9.00 |  8.00 |
+------+------------+---------------------+-----------------------+------------+-----------+---------------------+-------+-------+-------+-------+
Nhap danh sach ID can xoa: 1 3

        +-------------------------------+
        |       Da xoa thanh cong!      |
        +-------------------------------+

+------+------------+---------------------+-----------------------+------------+-----------+---------------------+-------+-------+-------+-------+
|  ID  |    MSSV    |       Ho & Ten      |         Email         |  Ngay sinh | Gioi Tinh |     Nganh hoc       |  OOP  |  DSA  |  TRR  |  GPA  |
+------+------------+---------------------+-----------------------+------------+-----------+---------------------+-------+-------+-------+-------+
|    5 | 6351071015 |  Nguyen Van Dung    | 6351071015@st.edu.utc2.vn | 01/08/2004 |    Nam    |      Big Data       |  9.00 |  8.00 | 10.00 |  9.00 |
+------+------------+---------------------+-----------------------+------------+-----------+---------------------+-------+-------+-------+-------+
Press any key to continue . . .
```

Xóa 2 sinh viên có ID là 1, 3 thành công



Xóa 1 sinh viên có MSSV là 6351071015 thành công

8. Chức năng thống kê sinh viên



```
Nhap lua chon: 7

[1]. Thong ke DTB
[2]. Thong ke Nganh
[3]. Thong ke Gioi Tinh
[4]. Thong ke so luong SV
(?): 1

[1]. Chinh xac muc diem
[2]. Cao hon muc diem
[3]. Thap hon muc diem
(?): 2
(?) Nhap DTB: 5

        So sinh vien co DTB cao hon 5.00 la: 2

+------+--------------+-----------------+-----------------------+--------------+------------+-----------------------+------+------+-------+------+
|  ID  |     MSSV     |      Ho & Ten   |         Email         |   Ngay sinh  | Gioi Tinh  |      Nganh hoc        | OOP  | DSA  |  TRR  | GPA  |
+------+--------------+-----------------+-----------------------+--------------+------------+-----------------------+------+------+-------+------+
|    6 |   6351071011 |        Dung Ha | 6351071011@st.edu.utc2.vn | 21/12/2004 |        Nu | Cong Nghe Thong Tin | 8.00 | 7.30 |  9.00 | 8.10 |
+------+--------------+-----------------+-----------------------+--------------+------------+-----------------------+------+------+-------+------+
|    7 |   6351071012 |   Nguyen Van A | 6351071012@st.edu.utc2.vn | 01/08/2005 |       Nam | Khoa Hoc May Tinh   | 8.00 | 9.00 | 10.00 | 9.00 |
+------+--------------+-----------------+-----------------------+--------------+------------+-----------------------+------+------+-------+------+
Press any key to continue . . .
```

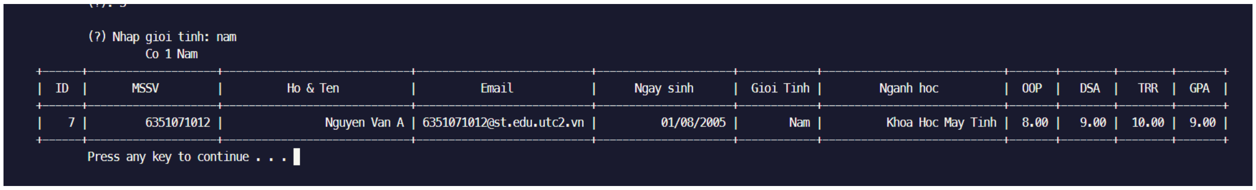
Thống kê các sinh viên có điểm trung bình lớn hơn 5

```
Nhap lua chon: 7

[1]. Thong ke DTB
[2]. Thong ke Nganh
[3]. Thong ke Gioi Tinh
[4]. Thong ke so luong SV
(?): 1

[1]. Chinh xac muc diem
[2]. Cao hon muc diem
[3]. Thap hon muc diem
(?): 3
(?) Nhap DTB: 9

        So sinh vien co DTB thap hon 9.00 la: 1

+------+--------------+-----------------+-----------------------+--------------+------------+-----------------------+------+------+-------+------+
|  ID  |     MSSV     |      Ho & Ten   |         Email         |   Ngay sinh  | Gioi Tinh  |      Nganh hoc        | OOP  | DSA  |  TRR  | GPA  |
+------+--------------+-----------------+-----------------------+--------------+------------+-----------------------+------+------+-------+------+
|    6 |   6351071011 |        Dung Ha | 6351071011@st.edu.utc2.vn | 21/12/2004 |        Nu | Cong Nghe Thong Tin | 8.00 | 7.30 |  9.00 | 8.10 |
+------+--------------+-----------------+-----------------------+--------------+------------+-----------------------+------+------+-------+------+
Press any key to continue . . .
```

Thống kê các sinh viên có điểm trung bình thấp hơn 9

```
Nhap lua chon: 7

[1]. Thong ke DTB
[2]. Thong ke Nganh
[3]. Thong ke Gioi Tinh
[4]. Thong ke so luong SV
(?): 2

(?) Nhap nganh hoc: cong nghe thong tin

        So sinh vien hoc nganh Cong Nghe Thong Tin la: 1

+------+--------------+-----------------+-----------------------+--------------+------------+-----------------------+------+------+-------+------+
|  ID  |     MSSV     |      Ho & Ten   |         Email         |   Ngay sinh  | Gioi Tinh  |      Nganh hoc        | OOP  | DSA  |  TRR  | GPA  |
+------+--------------+-----------------+-----------------------+--------------+------------+-----------------------+------+------+-------+------+
|    6 |   6351071011 |        Dung Ha | 6351071011@st.edu.utc2.vn | 21/12/2004 |        Nu | Cong Nghe Thong Tin | 8.00 | 7.30 |  9.00 | 8.10 |
+------+--------------+-----------------+-----------------------+--------------+------------+-----------------------+------+------+-------+------+
Press any key to continue . . .
```

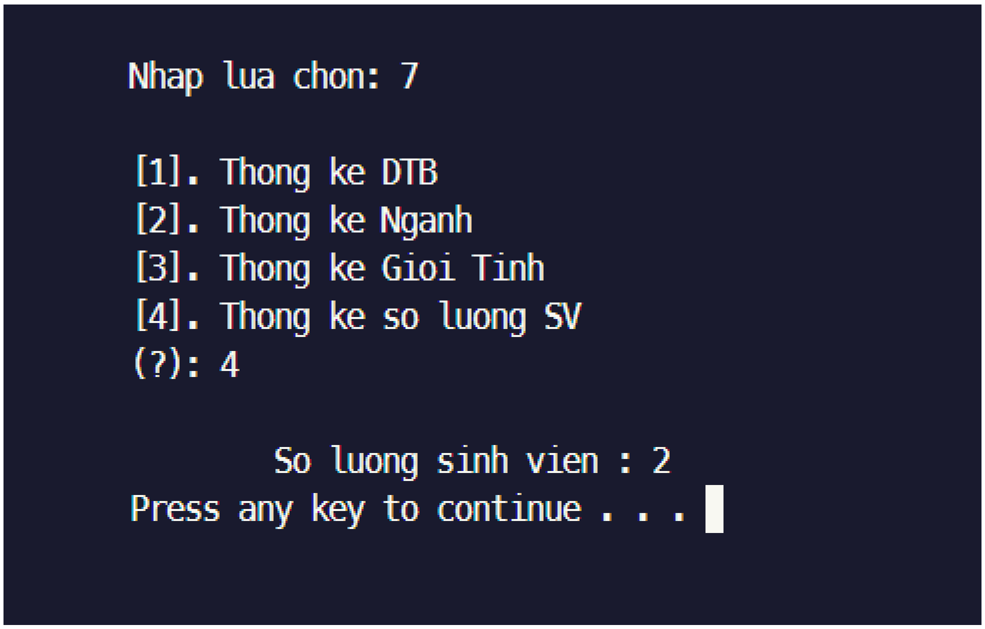
Thống kê các sinh viên học ngành “Cong Nghe Thong Tin”


(?) Nhap gioi tinh: nam
Co 1 Nam

ID	MSSV	Ho & Ten	Email	Ngay sinh	Gioi Tinh	Nganh hoc	OOP	DSA	TRR	GPA
7	6351071012	Nguyen Van A	6351071012@st.edu.utc2.vn	01/08/2005	Nam	Khoa Hoc May Tinh	8.00	9.00	10.00	9.00

Press any key to continue . . .

Thống kê sinh viên có giới tính là Nam


Nhap lua chon: 7

[1]. Thong ke DTB
[2]. Thong ke Nganh
[3]. Thong ke Gioi Tinh
[4]. Thong ke so luong SV
(?): 4

So luong sinh vien : 2
Press any key to continue . . .

Số lượng sinh viên có trong hệ thống

9. Chức năng đọc & ghi file

```
24;6351071034;Pham Van Z;pvz@gmail.com;19;9;2000;Nam;Lich Su;7;7.5;7;7.17;
39;6351071049;Pham Van Z;pvz@gmail.com;24;9;2002;Nam;Y Duoc;6.00;6.00;7.00;6.33;
23;6351071033;Ho Van Y;hvy@gmail.com;12;3;2001;Nam;Ngon Nganh;6.00;6.00;7.00;6.33;
38;6351071048;Ho Van Y;hvy@gmail.com;17;5;1998;Nam;Ngon Nganh;7.00;7.00;7.00;7.00;
37;6351071047;Le Van X;lvx@gmail.com;8;8;2003;Nam;Lich Su;8.00;8.00;7.00;7.67;
22;6351071032;Le Van X;lvx@gmail.com;25;11;2003;Nam;Y Te Cong Cong;7.00;7.00;7.00;7.00;
36;6351071046;Tran Van V;tvv@gmail.com;21;7;2000;Nam;Quoc Te Hoc;6.00;6.50;6.00;6.17;
21;6351071031;Tran Van V;tvv@gmail.com;1;1;2000;Nam;Cong Nghe Sinh Hoc;8.00;8.50;8.00;8.17;
20;6351071030;Nguyen Van U;nvu@gmail.com;7;6;1999;Nam;Quoc Te Hoc;6.00;6.50;6.00;6.17;
35;6351071045;Nguyen Van U;nvu@gmail.com;10;3;1999;Nam;Cong Nghe Thong Tin;7.00;7.00;7.00;7.00;
50;6351071060;Nguyen Van U;nvu@gmail.com;20;9;2002;Nam;Lich Su;7.00;7.50;7.00;7.33;
19;6351071029;Mai Van T;mvt@gmail.com;18;10;2002;Nam;Khoa Hoc May Tinh;8.00;8.00;8.00;8.00;
34;6351071044;Mai Van T;mvt@gmail.com;13;12;2001;Nam;Khoa Hoc Xa Hoi;8.00;8.50;8.00;8.33;
49;6351071059;Mai Van T;mvt@gmail.com;2;12;1998;Nam;Kinh Te;8.00;8.00;7.00;7.67;
18;6351071028;Le Van S;lvs@gmail.com;2;4;2001;Nam;Cong Nghe Thong Tin;7.00;7.50;7.00;7.33;
33;6351071043;Le Van S;lvs@gmail.com;16;10;1998;Nam;Cong Nghe May;6.00;6.50;7.00;6.83;
48;6351071058;Le Van S;lvs@gmail.com;13;7;1999;Nam;Y Te Cong Cong;6.00;6.50;6.00;6.17;
32;6351071042;Tran Van R;tvr@gmail.com;9;4;2003;Nam;Quan Tri Kinh Doanh;7.00;7.00;7.00;7.00;
```

File dữ liệu

```
Nhap lua chon: 8
[0]. Quay lai
[1]. Ghi file
[2]. Doc file

(?): 2

        +-----------------------------------------+
        |          Da doc file thanh cong!        |
        +-----------------------------------------+

Press any key to continue . . .
```

Đọc file thành công

```
Nhap lua chon: 2
```

ID	MSSV	Ho & Ten	Email	Ngay sinh	Gioi Tinh	Nganh hoc	OOP	DSA	TRR	GPA
24	6351071034	Pham Van Z	pvz@gmail.com	19/09/2000	Nam	Lich Su	7.50	7.00	7.00	7.17
39	6351071049	Pham Van Z	pvz@gmail.com	24/09/2002	Nam	Y Duoc	6.00	6.00	7.00	6.33
23	6351071033	Ho Van Y	hvy@gmail.com	12/03/2001	Nam	Ngon Nganh	6.00	6.00	7.00	6.33
38	6351071048	Ho Van Y	hvy@gmail.com	17/05/1998	Nam	Ngon Nganh	7.00	7.00	7.00	7.00
37	6351071047	Le Van X	lvx@gmail.com	08/08/2003	Nam	Lich Su	8.00	8.00	7.00	7.67
22	6351071032	Le Van X	lvx@gmail.com	25/11/2003	Nam	Y Te Cong Cong	7.00	7.00	7.00	7.00
36	6351071046	Tran Van V	tvv@gmail.com	21/07/2000	Nam	Quoc Te Hoc	6.50	6.00	6.00	6.17
21	6351071031	Tran Van V	tvv@gmail.com	01/01/2000	Nam	Cong Nghe Sinh Hoc	8.50	8.00	8.00	8.17
20	6351071030	Nguyen Van U	nvu@gmail.com	07/06/1999	Nam	Quoc Te Hoc	6.50	6.00	6.00	6.17
35	6351071045	Nguyen Van U	nvu@gmail.com	10/03/1999	Nam	Cong Nghe Thong Tin	7.00	7.00	7.00	7.00
50	6351071060	Nguyen Van U	nvu@gmail.com	20/09/2002	Nam	Lich Su	7.50	7.00	7.00	7.33
19	6351071029	Mai Van T	mvt@gmail.com	18/10/2002	Nam	Khoa Hoc May Tinh	8.00	8.00	8.00	8.00
34	6351071044	Mai Van T	mvt@gmail.com	13/12/2001	Nam	Khoa Hoc Xa Hoi	8.50	8.00	8.00	8.33
49	6351071059	Mai Van T	mvt@gmail.com	02/12/1998	Nam	Kinh Te	8.00	8.00	7.00	7.67
18	6351071028	Le Van S	lvs@gmail.com	02/04/2001	Nam	Cong Nghe Thong Tin	7.50	7.00	7.00	7.33

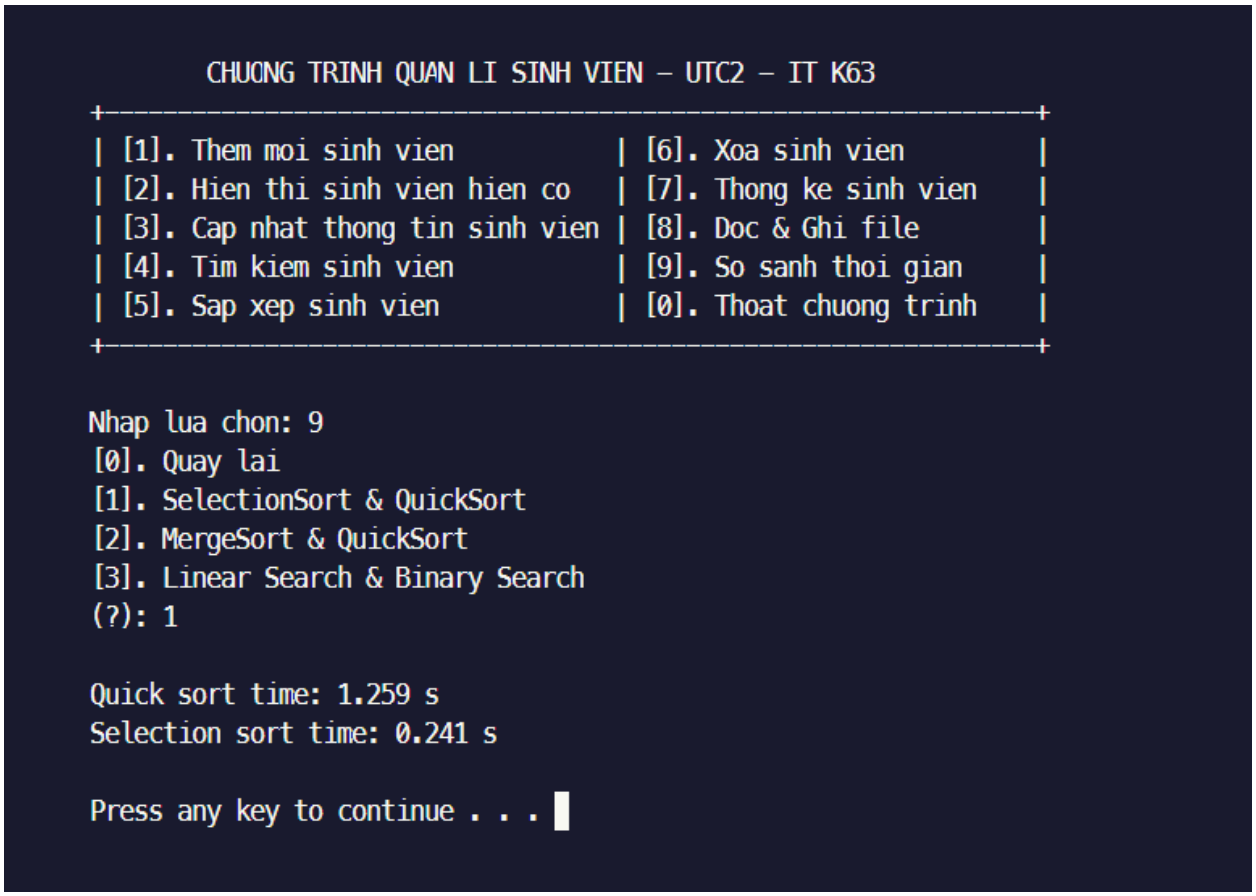
Sau khi đọc file thành công vào in ra màn hình kết quả

10. Chức năng so sánh thời gian chạy của các thuật toán

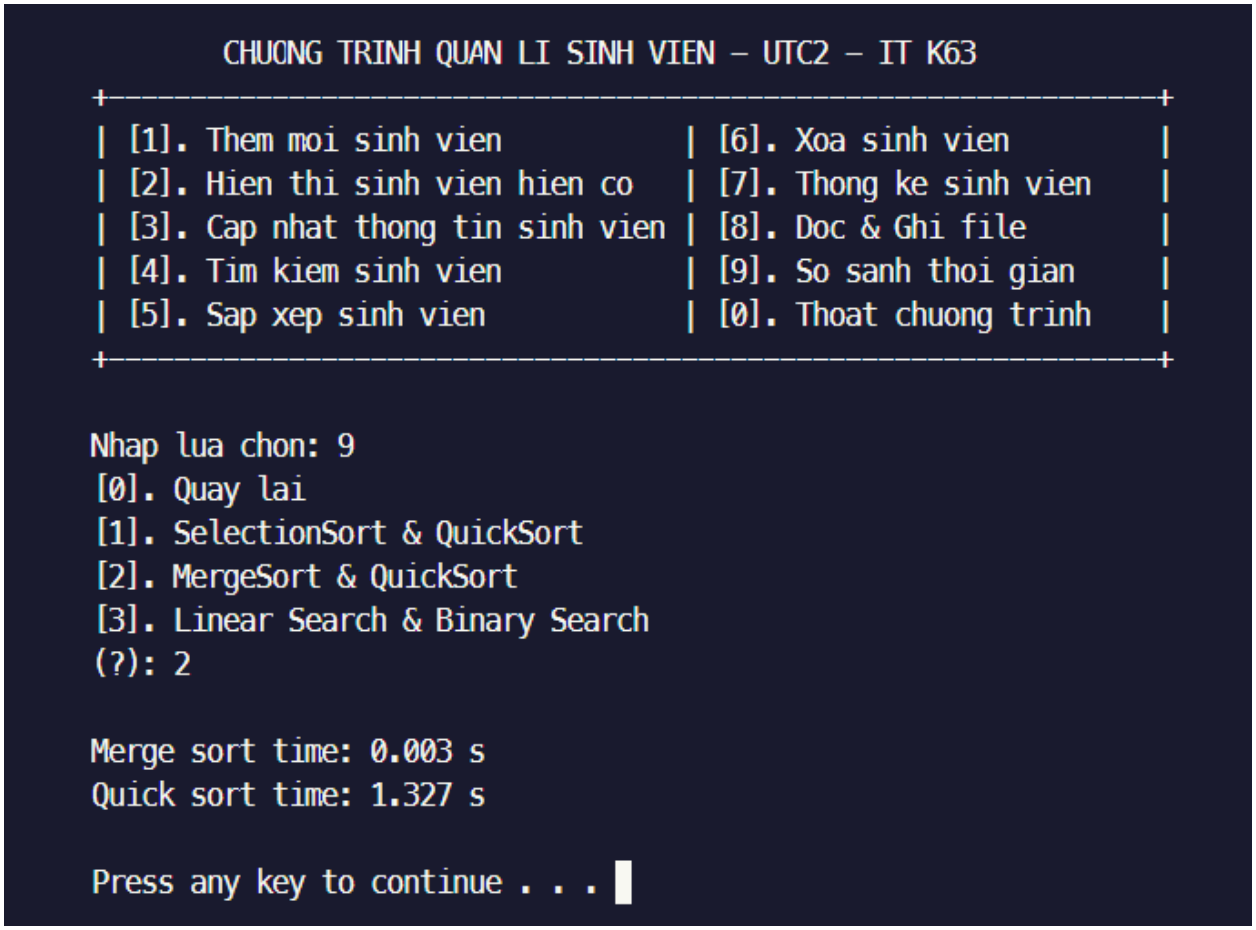
Link data

```
7993  8;6351071018;Do Van G;dvg@gmail.com;10;1;2001;Nam;Lich Su;6.00;6.00;7.00;6.33;
7994  7;6351071017;Mai Van F;mf@gmail.com;21;7;2000;Nam;Dien Tu Vien Thong;8.00;8.50;8.00;8.33;
7995  6;6351071016;Ho Thi E;ethe@gmail.com;2;11;2003;Nu;Quoc Te Hoc;7.00;7.00;7.00;7.00;
7996  1;6351071011;Ha Van Dung;dungha@gmail.com;1;1;2000;Nam;Cong Nghe Thong Tin;8.00;8.50;7.00;7.83;
7997  5;6351071015;Pham Van D;pvd@gmail.com;25;3;1998;Nam;Cong Nghe Sinh Hoc;8.00;8.00;8.00;8.00;
7998  4;6351071014;Le Van C;cle@gmail.com;20;8;2002;Nam;Y Duoc;6.00;6.50;7.00;6.50;
7999  3;6351071013;Tran Thi B;bitran@gmail.com;15;12;1999;Nu;Kinh Te;7.00;7.50;8.00;7.50;
8000  2;6351071012;Nguyen Van A;anv@gmail.com;10;5;2001;Nam;Ke Toan;7.00;8.00;7.00;7.33;
```

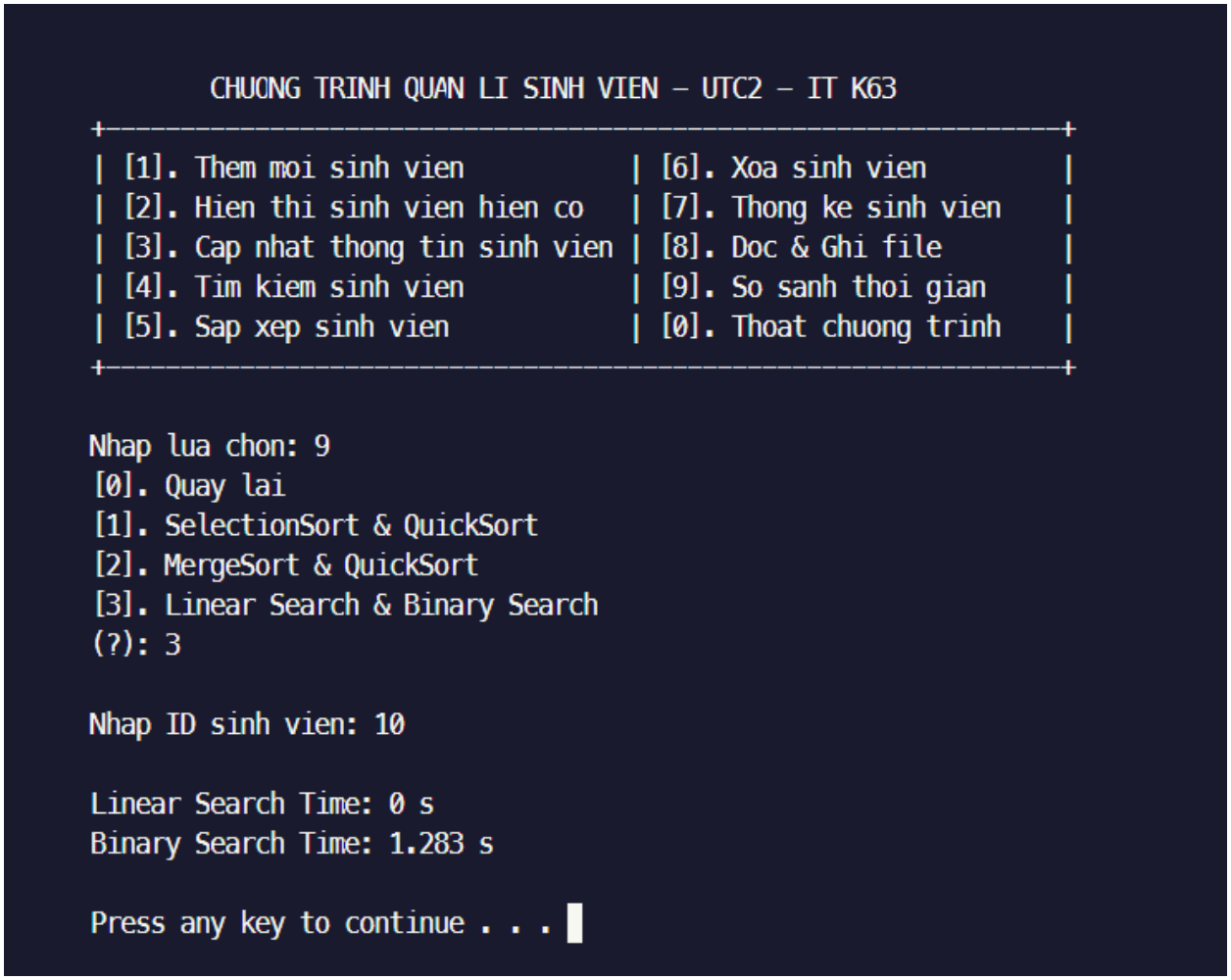
Dữ liệu ban đầu



Selection sort & Quick sort



Merge sort & Quicksort



Linear Search & Binary Search

- Quicksort:
    - Thời gian trung bình: O(n * log(n))
    - Thời gian tốt nhất (khi danh sách đã sắp xếp hoặc gần sắp xếp): O(n * log(n))
    - Thời gian xấu nhất (khi danh sách là một dãy tăng hoặc giảm): O(n^2)

Quicksort là một thuật toán sắp xếp hiệu quả với độ phức tạp trung bình là O(n * log(n)). Nó hoạt động tốt trong hầu hết các trường hợp, đặc biệt là khi danh sách lớn.

- Selection Sort:
    - Thời gian trung bình: O(n^2)
    - Thời gian tốt nhất: O(n^2)
    - Thời gian xấu nhất: O(n^2)

Selection Sort là một thuật toán sắp xếp đơn giản, nhưng có độ phức tạp thời gian là O(n^2) trong tất cả các trường hợp. Nó thường hiệu quả hơn khi danh sách nhỏ.

- Merge sort

Thời gian chạy của thuật toán merge sort là O(n log n). Merge sort là một thuật toán sắp xếp ổn định, chia để trị, và tách thành các phần con rồi ghép lại để sắp xếp dãy số. Thời gian chạy O(n log n) là tốt nhất có thể đạt được với một thuật toán sắp xếp so sánh, và nó không bị ảnh hưởng bởi sự sắp xếp ban đầu của dãy số.

Thuật toán merge sort làm việc theo cách chia dãy số thành hai phần bằng cách chia đôi dãy ban đầu, sau đó đệ quy sắp xếp từng phần con. Cuối cùng, nó ghép hai phần con đã sắp xếp lại thành một dãy số đã sắp xếp. Quá trình này lặp lại cho đến khi toàn bộ dãy số đã được sắp xếp.

Do đó, thời gian chạy của merge sort luôn là O(n log n) bất kể dãy số ban đầu là gì, và nó là một trong những thuật toán sắp xếp hiệu quả trong nhiều trường hợp.