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: XÂY DỰNG CÁC CHƯƠNG TRÌNH PHẦN MỀM BẰNG NGÔN NGỮ LẬP TRÌNH C

Giảng viên hướng dẫn: TRẦN THỊ DUNG

Sinh viên thực hiện: NGUYỄN VĂN DU

ĐẶNG QUANG TRƯỜNG NGUYÊN

Lớp: CQ.60.CNTT

Khoá: K60

LỜI CẢM ƠN

Sau quá trình học tập và rèn luyện tại bộ môn Công nghệ thông tin trường Đại học Giao thông Vận tải – Phân hiệu tại thành phố Hồ Chí Minh em đã được trang bị các kiến thức cơ bản, các kỹ năng thực tế để có thể hoàn thành đề tài xây dựng các chương trình phần mềm bằng ngôn ngữ lập trình C của tụi em.

Em xin gửi lời cảm ơn chân thành đến quý thầy, cô bộ môn Công nghệ thông tin trường Đại học Giao thông Vận tải – Phân hiệu tại thành phố Hồ Chí Minh đã quan tâm hướng dẫn truyền đạt học những kiến thức và kinh nghiệm cho em trong suốt thời gian học tập, và thực hiện bài tập lớn một cách tận tình và tâm huyết. Em xin chúc quý thầy cô thật nhiều sức khỏe và luôn đạt được thành công trong cuộc sống. Đặc biệt em xin cảm ơn cô Trần Thị Dung người đã trực tiếp hướng dẫn em và chỉ bảo em trong quá trình thực hiện đề tài xây dựng các chương trình phần mềm bằng ngôn ngữ lập trình C. Cô đã cùng tụi em góp ý và xây dựng đề tài "Xây dựng các chương trình phần mềm bằng ngôn ngữ lập trình C".

Sau một thời gian nỗ lực thực hiện thì đề tài cũng đã hoàn thành. Nhưng không sao tránh khỏi những sai sót do tụi em còn chưa có nhiều kinh nghiệm thực tế. Em kính mong nhận được sự góp ý và nhận xét từ quý thầy, cô để tụi em có thể hoàn thiện và hoàn thành tốt hơn cho đề tài của mình.

Lời sau cùng em một lần nữa kính chúc quý thầy, cô bộ môn Công nghệ thông tin Trường Đại học Giao thông Vận tải – Phân hiệu tại thành phố Hồ Chí Minh thật nhiều sức khỏe và thành công.

Tp. Hồ Chí Minh, ngày tháng 6 năm 2020

Sinh viên thực hiện

Nguyễn Văn Du

Đặng Quang Trường Nguyên

MỤC LỤC

Chương I: Sơ lược về đề tài

1.1. Lí do chọn đề tài

Quản lý sách là một chuỗi công việc rất vất vã và tốn nhiều công sức. Việc tin học hóa trong bài toán quản lí sách sẽ giúp cho viejc quản lý trở nên đơn giản và đặc biệt là tính chính xác cao. Đặc biệt tin hóa học trong bài toán quản lý sẽ giúp việc truy vấn thông tin được nhanh chóng theo yêu cầu khác nhau.

Và sự ra đời của môt hệ thống "*Quản lý sách*" phục vụ cho công tác nghiệp vụ của con người làm giảm thiểu tối đa những vất vả trong công việc.

Nhận thấy việc xây dựng chương trình quản lý cho thư viện có thể giúp khắc phục khó khan trên đồng thời tang khả năng tiếp cận của độc giả. Bởi vậy nhóm em quyết định tìm hiểu và thực hiện đề tài "Xây dựng các chương trình phần mềm bằng ngôn ngữ lập trình C ".

1.2. Mục đích đề tài

Hệ thống quản lý sách được xây dựng nhằm mục đích giải quyết các yêu cầu sau:

- ❖ Giúp sinh viên tra cứu sách.
- ❖ Tiết kiệm tối đa nguồn lực và thời gian.
- ❖ Dễ dàng chỉnh sửa phần mền khi quy định công việc thay đổi.
- ❖ Quản lý thông tin về các loại sách.
- ❖ Thông tin thống kê phải đảm bả tính chính xác, khách quan.

Phần mềm quản lý sách hỗ trợ đắc lực cho ván bộ quán lý nguồn tài nguyên hiện có trong thư viện nhanh chống và độ chính xác cao. Đồng thời giúp người dùng hiểu rõ hơn về lợi ích mà phần mềm mang lại.

Chương II: Làm việc với tệp (Work with the file):

2.1. Các kiểu file

2.1.1. Flie văn bản – text files

File văn bản là file thường có đuôi là .txt. Những file này bạn có thể dễ dàng tạo ra bằng cách dùng các text editer thông dụng như Notepad, Notepad++, Sublime Text, …

Khi bạn mở các file này bằng các text editer nói trên, bạn sẽ thấy được văn bản ngay và có thể dễ dàng thao tác sửa, xóa, thêm nội dung của file này.

Kiểu file này thuận tiện cho chúng ta trong việc sử dụng hàng ngày, nhưng nó sẽ kém bảo mật và cần nhiều bộ nhớ để lưu trữ hơn.

2.1.2. Flie nhị phân – Binary files

File nhị phân thường có đuôi mở rộng là **.bin**

Thay vì lưu trữ dưới dạng văn bản thuần thúy, các file này được lưu dưới dạng nhị phân, chỉ bao gồm các số 0 và 1. Bạn cũng sẽ thấy các con số này nếu cố mở nó bằng 1 text editer kể trên.

Loại file này giúp lưu trữ được dữ liệu với kích thước lớn hơn, không thể đọc bằng các text editer thông thường và thông tin lưu trữ ở loại file được bảo mật hơn so với file văn bản.

2.2. Các thao tác với file

Trong ngôn ngữ lập trình C, có một số thao tác chính khi làm việc với file, bao gồm cả file văn bản và file nhị phân:

- Tạo mới một file
- Mở một file đã có
- Đóng file đang mở
- Đọc thông tin từ file/ Ghi thông tin ra file

2.3.3. Thao tác với file trên ngôn ngữ C

Khi làm việc với file, bạn cần khai báo 1 con trỏ kiểu FILE. Việc khai báo này là cần thiết để có sự kết nối giữa chương trình của bạn và tập tin mà bạn cần thao tác.

```
1
2   FILE *fptr;
3
4
```

Hình ảnh: đoạn code khai báo một con trỏ

a. Thao tác mở flie

Bạn có thể sử dụng hàm **fopen()** để tạo file mới hoặc để mở các file đã tồn tại. Cách gọi này sẽ khởi tạo đối tượng loại **FILE**, mà bao gồm thông tin cần thiết để điều khiển luồng. Dưới đây là một cách gọi hàm:

```
FILE *fopen( const char * ten_file, const char * che_do );
```

Ở đây, **ten_file** là một chuỗi, được coi như tên file và giá trị **che_do** truy cập có thể là những giá trị dưới đây:

Mode	Miêu tả
r	Mở các file đã tồn tại với mục đích đọc
w	Mở các file với mục đích ghi. Nếu các file này chưa tồn tại thì file mới được tạo ra. Ở đây, chương trình bắt đầu ghi nội dung từ phần mở đầu của file
a	Mở file văn bản cho việc ghi ở chế độ ghi tiếp theo vào cuối, nếu nó chưa tồn tại thì file mới được tạo. Đây là chương trình ghi nội dung với phần cuối của file đã tồn tại.
r+	Mở file văn bản với mục đích đọc và ghi.
w+	Mở một file văn bản cho chế độ đọc và ghi. Nó làm trắng file đã tồn tại nếu file này có và tạo mới nếu file này chưa có.

a+	Mở file đã tồn tại với mục đích đọc và ghi. Nó tạo file mới nếu không tồn tại. Việc đọc file sẽ bắt đầu đọc từ đầu nhưng ghi file sẽ chỉ ghi vào cuối file.

Nếu bạn thao tác với các file nhị phân, bạn có thể có các cách truy xuất thay cho các trường hợp trên như sau:

```
"rb", "wb", "ab", "rb+", "r+b", "wb+", "w+b", "ab+", "a+b"
```

Trong đó `mode` là một tham số chúng ta cần chỉ định.

Ví dụ: code thao tác mở file

```
 9
10  fptr = fopen("E:\\cprogram\\newprogram.txt","w");
11
12  // hoac
13
14  fptr = fopen("E:\\cprogram\\oldprogram.bin","rb");
```

Hình ảnh: code thao tác mở file

- Giả sử tập tin `newprogram.txt` chưa có trong thư mục `E:\cprogram`. Ví dụ đầu tiên với `mode = "w"` sẽ cho phép chương trình tự động tạo ra file `newprogram.txt` nếu nó chưa có. Và sau đó mở file này lên nhưng chương trình chỉ có thể ghi dữ liệu vào mà không thể đọc.
- Mode là `w` chỉ cho phép chương trình ghi (nếu đã có dữ liệu thì ghi đè) nội dung của file.
- Với ví dụ thứ 2, mode là `rb` cho phép chương trình mở 1 file nhị phân đã có sẵn `oldprogram.bin`. Với trường hợp này, chương trình của bạn chỉ có thể đọc file và không thể ghi nội dung vào file.

b. Thao tác đóng file

Khi làm việc với tập tin hoàn tất, kể cả là file nhị phân hay file văn bản. Bạn cần đóng file sau khi làm việc với nó xong.

Việc đóng file đang mở có thể được thực hiện bằng cách dùng hàm fclose().

```
3
4   fclose(fptr); //Con tro FILE tro toi file can duoc dong.
5
```

Hình ảnh: đoạn code dùng hàm fclose()

2.3.4. Đọc/Ghi file văn bản trong C

Để làm việc với file văn bản, chúng ta sẽ sử dụng fprintf() và fscanf().

VD1: Ghi file sử dụng fprintf()

```
1
2   #include <stdio.h>
3   #include <stdlib.h>
4   int main()
5   {
6       int So;
7       FILE *fptr;
8       fptr = fopen("C:\\program.txt","w");
9       if(fptr == NULL)
10      {
11          printf("Error!");
12          exit(1);
13      }
14      printf("Nhap so: ");
15      scanf("%d",&So);
16      fprintf(fptr,"%d", So);
17      fclose(fptr);
18      return 0;
19  }
20
```

Hình ảnh: code ghi file sử dụng hàm fprintf()

Chương trình nhận số So từ bàn phím và ghi vào file văn bản program.txt.

Sau khi bạn chạy chương trình này, bạn sẽ thấy file văn bản program.txt được tạo mới trong ổ C trên máy tính bạn. Khi mở file này lên, bạn sẽ thấy số mà bạn vừa nhập cho biến So kia.

VD2: Đọc file sử dụng fscanf()

```c
#include <stdio.h>
#include <stdlib.h>
int main()
{
    int So;
    FILE *fptr;
    if ((fptr = fopen("C:\\program.txt","r")) == NULL){
        printf("Loi! mo tep tin");
        // Chuong trinh thoat neu con tro tep tra ve NULL.
        exit(1);
    }
    fscanf(fptr,"%d", &So);
    printf("Gia tri cua n = %d", So);
    fclose(fptr);
    return 0;
}
```

Hình ảnh: code đọc file sử dụng fscanf()

Chương trình này sẽ đọc giá trị số được lưu trong file program.txt mà chương trình ở VD1 vừa tạo ra và in lên màn hình.

2.3.5. Đọc/Ghi file nhị phân trong C

Các hàm fread() và fwrite() trong C được sử dụng để đọc và ghi file trong C ở dạng nhị phân.

a. Ghi file nhị phân trong C

Để ghi file nhị phân, bạn cần sử dụng hàm fwrite(). Hàm này cần 4 tham số: địa chỉ của biến lưu dữ liệu cần ghi, kích thước của biến lưu dữ liệu đó, số lượng kiểu dữ liệu của biến đó và con trỏ FILE trỏ tới file bạn muốn ghi.

```
3
4   fwrite(address_data,size_data,numbers_data,pointer_to_file);
5
```

Hình ảnh: đoạn code ghi file nhị phân trong C

VD3: Ghi file nhị phân sử dụng fwrite()

```
2   #include <stdio.h>
3   #include <stdlib.h>
4   struct threeNum
5   {
6      int n0, n1, n2;
7   };
8   int main(){
9      int n;
10     struct threeNum So;
11     FILE *fptr;
12     if ((fptr = fopen("C:\\program.bin","wb")) == NULL){
13        printf("Loi! Mo tep tin");
14        // Chuong trinh thoat neu con tro tep tra ve NULL.
15        exit(1);
16     }
17     for(n = 1; n < 5; ++n)
18     {
19        So.n0 = n;
20        So.n1 = 5*n;
21        So.n2 = 5*n + 1;
22        fwrite(&So, sizeof(struct threeNum), 1, fptr);
23     }
24     fclose(fptr);
25     return 0;
26  }
```

Hình ảnh: code ghi file nhị phân sử dụng hàm fwirte()

Trong VD3 này, chương trình sẽ tạo ra một file program.bin trên ổ đĩa C của bạn. Chương trình này đã khai báo 1 kiểu dữ liệu cấu trúc lưu 3 giá trị số n0, n1, n2; Và nó được sử dụng trong hàm main có tên biến là So.

Trong vòng lặp, các số được ghi vào file sử dụng hàm fwrite(). Các tham số gồm:

- Tham số đầu tiên là địa chỉ của biến So
- Tham số thứ 2 là kích thước của biến So
- Tham số thứ 3 là số lượng kiểu dữ liệu – ở đây là 1.
- Tham số thứ 4 là con trỏ FILE trỏ tới tệp tin program.bin

Cuối cùng, chúng ta đóng file sử dụng fclose().

b. Đọc file nhị phân trong C

Hàm fread() cũng có 4 tham số tương tự như hàm fwrite() phía trên.

```
2
3   fread(address_data,size_data,numbers_data,pointer_to_file);
4
```

Hình ảnh: đoạn code dùng hàm fread()

Ví dụ: đọc file nhị phân sử dụng fread()

```
3   #include <stdio.h>
4   #include <stdlib.h>
5   struct threeNum
6   {
7       int n1, n0, n2;
8   };
9   int main()
10  {
11      int n;
12      struct threeNum So;
13      FILE *fptr;
14      if ((fptr = fopen("C:\\program.bin","rb")) == NULL){
15          printf("Loi! Mo tep tin");
16          // Chuong trinh thoat neu con tro tep tra ve NULL.
17          exit(1);
18      }
19      for(n = 1; n < 5; ++n)
20      {
21          fread(&So, sizeof(struct threeNum), 1, fptr);
22          printf("n0: %d\tn1: %d\tn2: %d", So.n0, So.n1, So.n2);
23      }
24      fclose(fptr);
25      return 0;
26  }
```

Hình ảnh: code đọc flie nhị phân sử dụng hàm fread()

Trong ví dụ này, bạn đọc file program.bin và lặp qua từng dòng. Chúng ta sẽ nhận được các giá trị tương ứng khi bạn ghi vào trong VD3.

2.3.6. Một số ví dụ về đọc ghi file trong C

Một số đọc ghi file sau:

- Ghi văn bản vào file trong C
- Đọc dữ liệu văn bản từ file trong C

a. Ghi vào file một câu văn bản

```c
#include <stdio.h>
#include <stdlib.h>  /* Doi voi ha exit() */
int main()
{
   char BangChu[1000];
   FILE *fptr;
   fptr = fopen("program.txt", "w");
   if(fptr == NULL)
   {
      printf("Loi!");
      exit(1);
   }
   printf("Nhap mot cau:\n");
   gets(BangChu);
   fprintf(fptr,"%s", BangChu);
   fclose(fptr);
   return 0;
}
```

Hình ảnh: code ghi vào file một câu văn bản

Chạy thử:

```
Nhap mot cau:
I love you <3.
```

b. Đọc dữ liệu văn bản từ file

```c
#include <stdio.h>
#include <stdlib.h> // Doi voi ham exit()
int main()
{
    char c[1000];
    FILE *fptr;
    if ((fptr = fopen("program.txt", "r")) == NULL)
    {
        printf("Loi! Mo tep tin");
        // Chuong trinh thoat neu con tro tep tra ve NULL.
        exit(1);
    }
    // Doc van ban cho den dong moi
    fscanf(fptr,"%[^\n]", c);
    printf("Du lieu tu tep:\n%s", c);
    fclose(fptr);
    return 0;
}
```

Hình ảnh: code đọc dữ liệu văn bản từ file

Chạy thử:

```
Du lieu tu tep:
 I love you <3
```

Chương III: Danh sách liên kết đơn (Single linked list)

3.1. Danh sách liên kết

Về bản chất, danh sách liên kết có chức năng như một mảng, có thể thêm và xóa các phần tử ở bất kỳ vị trí nào khi cần thiết. Một số sự khác nhau giữa danh sách liên kết và mảng:

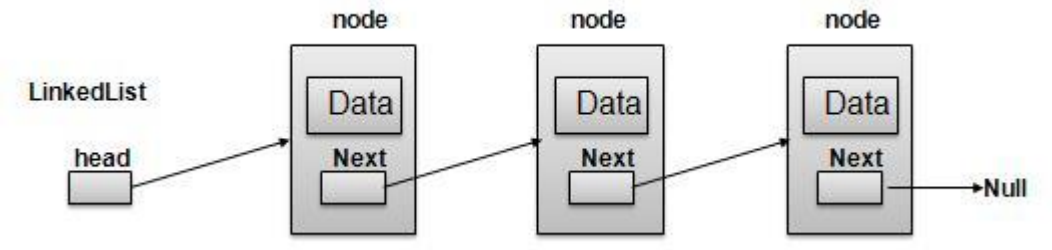
Nội dung	**Mảng**	**Danh sách liên kết**
Kích thước	• Kích thước cố định • Cần chỉ rõ kích thước trong khi khai báo	• *Kích thước thay đổi trong quá trình thêm/ xóa phần tử* • *Kích thước tối đa phụ thuộc vào bộ nhớ*
Cấp phát bộ nhớ	• Tĩnh: Bộ nhớ được cấp phát trong quá trình biên dịch	• *Động: Bộ nhớ được cấp phát trong quá trình chạy*
Thứ tự & sắp xếp	• Được lưu trữ trên một dãy ô nhớ liên tục	• *Được lưu trữ trên các ô nhớ ngẫu nhiên*
Truy cập	• *Truy cập tới phần tử ngẫu nhiên trực tiếp bằng cách sử dụng chỉ số mảng: O(1)*	• Truy cập tới phần tử ngẫu nhiên cần phải duyệt từ đầu/cuối đến phần tử đó: O(n)
Tìm kiếm	• *Tìm kiếm tuyến tính hoặc tìm kiếm nhị phân*	• Chỉ có thể tìm kiếm tuyến tính

3.2. Danh sách liên kết đơn là gì?

Danh sách liên kết đơn là một tập hợp các Node được phân bố động, được sắp xếp theo cách sao cho mỗi Node chứa "*một giá trị"(Data)* và "*một con trỏ"(Next)*. Con trỏ sẽ trỏ đến phần tử kế tiếp của danh sách liên kết đó. Nếu con trỏ mà trỏ tới NULL, nghĩa là đó là phần tử cuối cùng của danh sách liên kết.

Hình ảnh: mô tả cho một Node trong danh sách liên kết đơn:

Hình ảnh mô phỏng một danh sách liên đơn kết đầy đủ:



Mô phỏng của danh sách liên kết đơn.

Danh sách liên kết đơn (Single linked list): Chỉ có sự kết nối từ phần tử phía trước tới phần tử phía sau.

3.3. Cài đặt danh sách liên kết đơn

3.3.1. Tạo mới 1 Node

```
2
3
4   typedef struct LinkedList *node; //Tu gio dùng kieu du lieu LinkedList co the thay bang node cho ngan gon
5
6   node CreateNode(int value){
7       node temp; // declare a node
8       temp = (node)malloc(sizeof(struct LinkedList)); // Cap phat vung nho dung malloc()
9       temp->next = NULL;// Cho next tro toi NULL
10      temp->data = value; // Gan gia tri cho Node
11      return temp;//Tra ve node moi da co gia tri
12  }
```

Hình ảnh: tạo một kiểu dữ liệu của struct LinkedList để code clear

Mỗi một Node khi được khởi tạo, chúng ta cần cấp phát bộ nhớ cho nó, và mặc định cho con trỏ `next` trỏ tới NULL. Giá trị của Node sẽ được cung cấp khi thêm Node vào linked list.

- **typedef** được dùng để định nghĩa một kiểu dữ liệu trong C. VD: `typeder long long LL;`
- **malloc** là hàm cấp phát bộ nhớ của C. Với C++ chúng ta dùng `new`
- **sizeof** là hàm trả về kích thước của kiểu dữ liệu, dùng làm tham số cho hàm *malloc*

3.3.2. Thêm Node vào danh sách liên kết

Thêm vào đầu

Việc thêm vào đầu chính là việc cập nhật lại head. Ta gọi Node mới(temp), ta có:

- Nếu head đang trỏ tới NULL, nghĩa là linked list đang trống, Node mới thêm vào sẽ làm head.
- Ngược lại, ta phải thay thế head cũ bằng head mới. Việc này phải làm theo thứ tự như sau:
    - Cho next của temp trỏ tới head hiện hành
    - Đặt temp làm head mới

```
1
2
3  node ThemDau(node head, int value){
4      node temp = CreateNode(value); // Khoi tao node temp voi data = value
5      if(head == NULL){
6          head = temp; // //Neu linked list dang trong thi Node temp la head luon
7      }else{
8          temp->next = head; // Tro next cua temp = head hien tai
9          head = temp; // Doi head hien tai = temp(Vi temp bay gio la head moi )
10     }
11     return head;
12 }
13
14
```

Hình ảnh: đoạn code thêm vào đầu

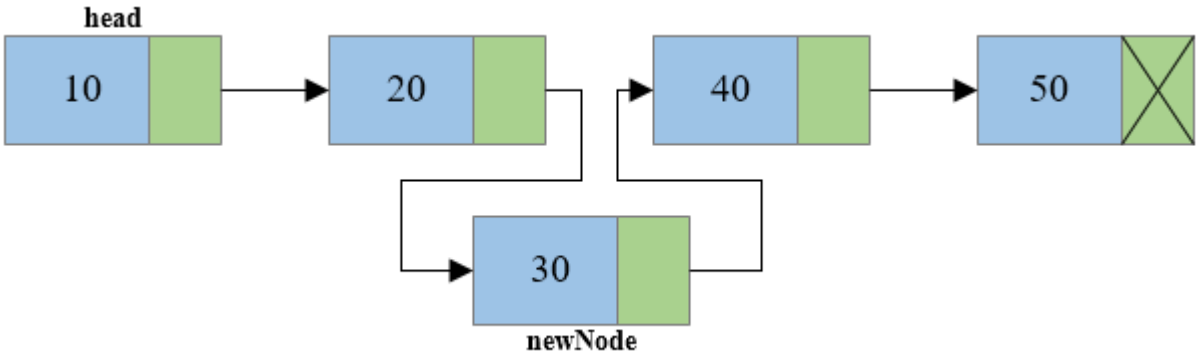
Thêm vào cuối
Chúng ta sẽ cần Node đầu tiên, và giá trị muốn thêm. Khi đó, ta sẽ:

1. Tạo một Node mới với giá trị value
2. Nếu head = NULL, tức là danh sách liên kết đang trống. Khi đó Node mới(temp) sẽ là head.
3. Ngược lại, ta sẽ duyệt tới Node cuối cùng (Node có next = NULL), và trỏ next cuối tới Node mới(temp).

```
2
3
4  node ThemCuoi(node head, int value){
5      node temp,p;// Khai bao 2 node temp va p
6      temp = CreateNode(value);//Goi ham createNode de khoi tao node temp co next tro toi NULL va gia tri la value
7      if(head == NULL){
8          head = temp;     //Neu linked list dang trong thi Node temp la head luon
9      }
10     else{
11         p  = head;// Khoi tao p tro toi head
12         while(p->next != NULL){
13             p = p->next;//Duyet danh sach lien ket don cuoi. Node cuoi la node có next = NULL
14         }
15         p->next = temp;//Gan next cua cai cuoi = temp. Khi do temp se la cai cuoi( temp->next = NULL )
16     }
17     return head;
18 }
19
```

Hình ảnh: đoạn code thêm vào cuối

Thêm vào vị trí bất kỳ



Hình ảnh: thêm node vào vị trị bất kì

Thêm Node vào giữa danh sách liên kết

Chúng ta cần phải duyệt từ đầu để tìm tới vị trí của Node cần chèn, giả sử là Node Q, khi đó chúng ta cần làm theo thứ tự sau:

- Cho next của Node mới trỏ tới Node mà Q đang trỏ tới
- Cho Node Q trỏ tới Node mới

3.3.3. Xóa Node khỏi danh sách liên kết

Xóa đầu

Bây giờ chỉ cần cho vị trí kế tiếp của head làm head là được. Mà vị trí kế tiếp của head chính là *head->next.*

```
node XoaDau(node head){
    if(head == NULL){
        printf("\nKhong co gi de xoa!");
    }else{
        head = head->next;
    }
    return head;
}
```

Hình ảnh: xóa ở vị trí đầu

Xóa cuối

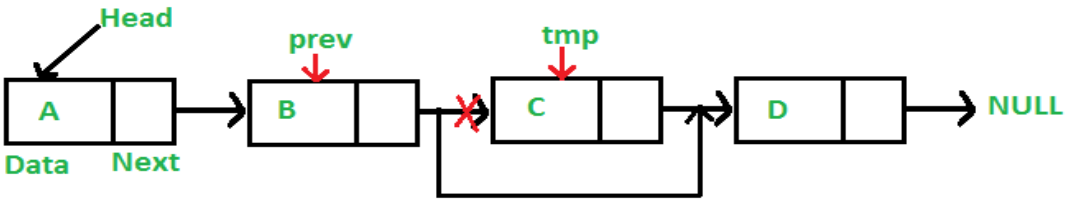
Phải duyệt đến vị trí cuối – 1, cho next của cuối – 1 đó bằng NULL.

```
node XoaCuoi(node head){
    if (head == NULL || head->next == NULL){
        return XoaCuoi(head);
    }
    node p = head;
    while(p->next->next != NULL){
        p = p->next;
    }
    p->next = p->next->next; // Cho next bang NULL
    return head;
}
```

Hình ảnh: code xóa cuối

Xóa ở vị trí bất kỳ

Việc xóa ở vị trí bất kỳ cũng khá giống xóa ở cuối kia. Chúng ta bỏ qua một phần tử, như ảnh sau:



Hình ảnh: xóa ở vị trí bất kì

Xóa Node trong danh sách liên kết

Lưu ý: Chỉ số xóa bắt đầu từ 0. Việc tìm vị trí càn xóa chỉ duyệt tới Node gần cuối (cuối – 1).

```
node XoaBatKy(node head, int position){
    if(position == 0 || head == NULL || head->next == NULL){
        head = DelHead(head);
    }else{

        int i = 1;
        node p = head;
        while(p->next->next != NULL && i != position){
            p = p->next;
            ++i;
        }

        if(i != position){
            head = DelTail(head);
        }else{
            p->next = p->next->next;
        }
    }
    return head;
}
```

Hình ảnh: code xóa vị trí bất kì

3.3.4. Lấy giá trị ở vị trí bất kỳ

Chúng ta sẽ viết một hàm để truy xuất giá trị ở chỉ số bất kỳ. Trong trường hợp chỉ số vượt quá chiều dài của linked list – 1, hàm này trả về vị trí cuối cùng. Do hạn chế là chúng ta không thể raise error khi chỉ số không hợp lệ. Mặc định chỉ số bạn truyền vào phải là số nguyên không âm. Nếu bạn muốn kiểm tra chỉ số hợp lệ thì nên kiểm tra trước khi gọi hàm này.

```
2
3  int Get(node head, int muc_luc){
4      int i = 0;
5      node p = head;
6      while(p->next != NULL && i != muc_uc){
7          ++i;
8          p = p->next;
9      }
10     return p->data;
11 }
```

Hình ảnh: đoạn code lấy giá trị ở vị trí bất kỳ

3.3.5. Tìm kiếm trong danh sách liên kết

Hàm tìm kiếm này sẽ trả về chỉ số của Node đầu tiên có giá trị bằng với giá trị cần tìm. Nếu không tìm thấy, chúng ta trả về -1.

```
3
4  int TimKiem(node head, int value){
5      int position = 0;
6      for(node p = head; p != NULL; p = p->next){
7          if(p->data == value){
8              return position;
9          }
10         ++position;
11     }
12     return -1;
13 }
14
```

Hình ảnh: đoạn code hàm tìm kiếm trong danh sách liên kết

Chúng ta có thể sử dụng hàm này để xóa tất cả các Node trong danh sách liên kết có giá trị chỉ định như sau:

```
2
3  node XoaLkChiDinh(node head, int value){
4      int position = TimKiem(head, value);
5      while(position != -1){
6          XoaBatKy(head, position);
7          position = Search(head, value);
8      }
9      return head;
10 }
11
```

Hình ảnh: đoạn code xóa liên kết chỉ định

3.3.6. Duyệt danh sách liên kết

Việc duyệt danh sách liên kết cực đơn giản. Khởi tạo từ Node head, bạn cứ thế đi theo con trỏ next cho tới trước khi Node đó NULL.

```
3  void Traverser(node head){
4      printf("\n");
5      for(node p = head; p != NULL; p = p->next){
6          printf("%d", p->data);
7      }
8  }
```

Hình ảnh: đoạn code duyệt dan sách liên kết

Chương IV: Các thuật toán sắp xếp (Sorting algorithms)

4.1. Sắp xếp chọn – Selection sort

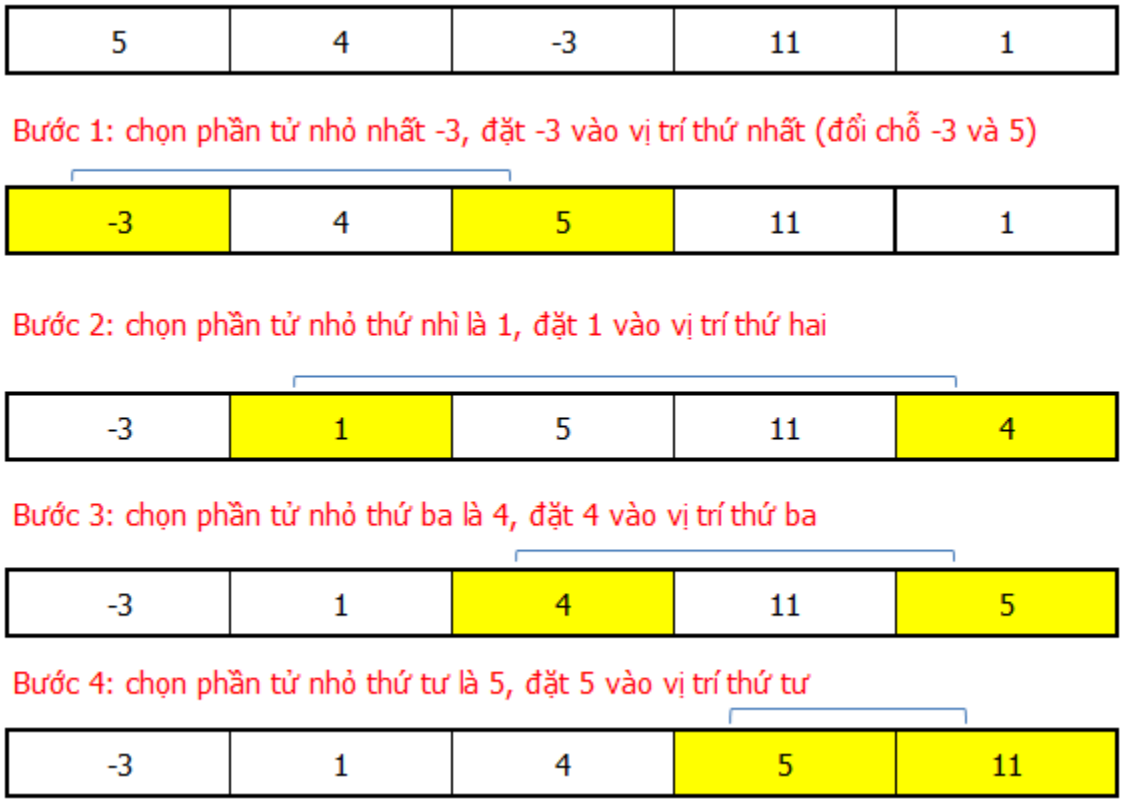
Sắp xếp chọn (*selection sort*) là phương pháp sắp xếp bằng cách chọn phần tử bé nhất xếp vào vị trí thứ nhất, tương tự với các phần tử nhỏ thứ hai, thứ ba, …

Thuật toán sắp xếp chọn sẽ sắp xếp một mảng bằng cách đi tìm phần tử có giá trị nhỏ nhất(giả sử với sắp xếp mảng tăng dần) trong đoạn đoạn chưa được sắp xếp và đổi cho phần tử nhỏ nhất đó với phần tử ở đầu đoạn chưa được sắp xếp(không phải đầu mảng). Thuật toán sẽ chia mảng làm 2 mảng con

- Một mảng con đã được sắp xếp
- Một mảng con chưa được sắp xếp

Tại mỗi bước lặp của thuật toán, phần tử nhỏ nhất ở mảng con chưa được sắp xếp sẽ được di chuyển về đoạn đã sắp xếp.

Ví dụ: Sắp xếp dãy 5, 4, -3, 11, 1 theo chiều tăng dần.



Hình ảnh: ví dụ sắp xếp chọn trong C

Hình ảnh code ví dụ sắp xếp chọn:

```c
#include <stdio.h>
#define N 5

int Mang[N] = {5, 4, -3, 11, 1};
void SapXepChon(){
    int k, i, j;
    // lap qua ta ca cac so
    for(i = 0; i < N-1; i++){
        // thiet lap phan tu hien tai la min
        k = i;
        // kiem tra phan tu hien tai co phai la nho nhat khong
        for(j = i+1; j<N; j++){
            if(Mang[j] < Mang[k]){
                k = j;
            }
        }
        if(k != i) {
            int temp;
            printf("Trao doi phan tu:  %d va %d \n" , Mang[i], Mang[k]);
            // Trao doi cac so
            temp = Mang[k];
            Mang[k] = Mang[i];
            Mang[i] = temp;
        }
        printf("\nVong lap thu %d: ",(i+1));
    }
}
void ChayMang() {
    int i;
    // Duyet qua tat ca phan tu
    for(i = 0; i < N; i++){
        printf("%d ", Mang[i]);
    }
    printf(" \n");
}
main() {
    printf("Mang du lieu dau vao: ");
    ChayMang();
    printf("================================\n");
    SapXepChon();
    printf("\n================================\n");
    printf("Mang sau khi da sap xep chon: ");
    ChayMang();
}
```

Chạy thử:

```
Mang du lieu dau vao: 5 4 -3 11 1
================================
Trao doi phan tu:  5 va -3
Vong lap thu 1: Trao doi phan tu:  4 va 1
Vong lap thu 2: Trao doi phan tu:  5 va 4
Vong lap thu 3: Trao doi phan tu:  11 va 5
Vong lap thu 4:
================================
Mang sau khi da sap xep chon: -3 1 4 5 11
--------------------------------
Process exited after 0.05676 seconds with return value 0
Press any key to continue . . .
```

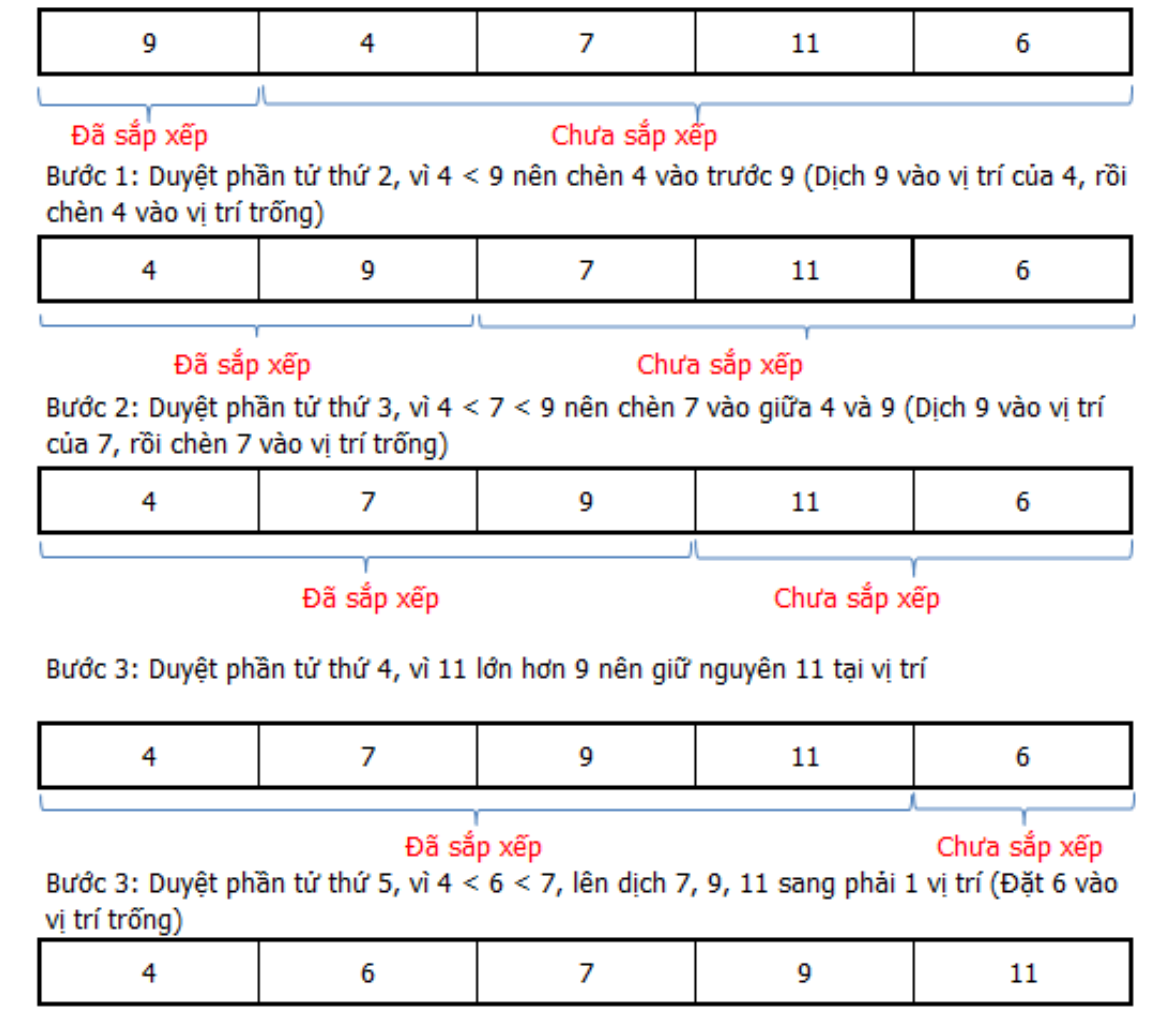
4.2. Sắp xếp chèn – Insertion sort

Sắp xếp chèn (*insertion sort*) là thuật toán sắp xếp cho một dãy đã có thứ tự. Chèn thêm một phần tử vào vị trí thích hợp của dãy số đã sắp xếp sao cho dãy số vẫn là dãy sắp xếp có thứ tự.

Thuật toán sắp xếp chèn thực hiện sắp xếp dãy số theo cách duyệt từng phần tử và chèn từng phần tử đó vào đúng vị trí trong mảng con(dãy số từ đầu đến phần tử phía trước nó) đã sắp xếp sao cho dãy số trong mảng sắp đã xếp đó vẫn đảm bảo tính chất của một dãy số tăng dần.

1. Khởi tạo mảng với dãy con đã sắp xếp có k = 1 phần tử(phần tử đầu tiên, phần tử có chỉ số 0)
2. Duyệt từng phần tử từ phần tử thứ 2, tại mỗi lần duyệt phần tử ở chỉ số i thì đặt phần tử đó vào một vị trí nào đó trong đoạn từ [0…i] sao cho dãy số từ [0…i] vẫn đảm bảo tính chất dãy số tăng dần. Sau mỗi lần duyệt, số phần tử đã được sắp xếp k trong mảng tăng thêm 1 phần tử.
3. Lặp cho tới khi duyệt hết tất cả các phần tử của mảng.

Ví dụ: Sắp xếp dãy 9, 4, 7, 11, 6 theo chiều tăng dần.



Hình ảnh: ví dụ sắp xếp chèn trong C

Hình ảnh code ví dụ sắp xếp chèn:

```c
#include <stdio.h>
#define N 5

    int Mang[N] = {9, 4, 7, 11, 6};
void SapXepChen(){
    int temp, i, j;
    // Thuc hien sap xep chen
    for (j = 1; j < N; j++){
        i = j - 1;
        temp = Mang[j];
        while (i >= 0 && temp < Mang[i]){
            Mang[i+1] = Mang[i];
            i= i - 1;
        }
        Mang[i+1] = temp;
    }
}
void ChayMang(){
    int i;
    // Duyet qua tat ca phan tu
    for(i = 0; i < N; i++){
        printf("%d ", Mang[i]);
    }
    printf("\n");
}
main() {
    SapXepChen();
    printf("Mang sau khi da sap xep chen la: ");
    ChayMang();
}
```

Chạy thử:

```
Mang sau khi da sap xep chen la: 4 6 7 9 11

--------------------------------
Process exited after 3.443 seconds with return value 0
Press any key to continue . . .
```

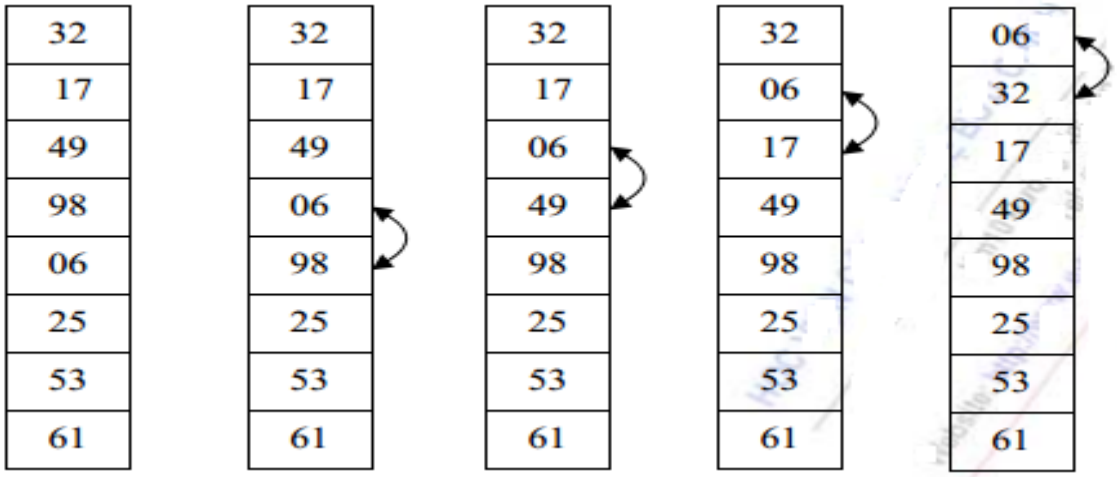
4.3. Sắp xếp nổi bọt – Bubble sort

- Sắp xếp nổi bọt (*bubble sort*) là phương pháp sắp xếp đơn giản, dễ hiểu thường được dạy trong khoa học máy tính. Nó so sánh hai phần tử cuối, nếu phần tử đứng trước lớn hơn phần tử đứng sau thì đổi chỗ chúng cho nhau. Tiếp tục làm như vậy với cặp phần tử tiếp theo cho đến cuối đầu dãy số. Sau đó nó quay lại với hai phần tử cuối cho đến khi không còn cần phải đổi chỗ nữa.

Thuật toán sắp xếp nổi bọt thực hiện sắp xếp dãy số bằng cách lặp lại công việc đổi chỗ 2 số liên tiếp nhau nếu chúng đứng sai thứ tự(số sau bé hơn số trước với trường hợp sắp xếp tăng dần) cho đến khi dãy số được sắp xếp.

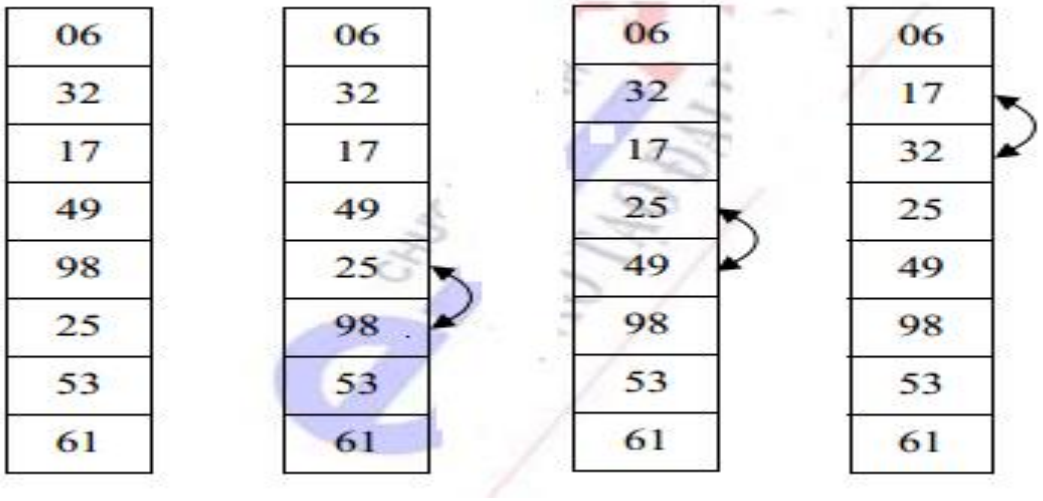
Ví dụ: Sắp xếp dãy 6, 32, 17, 49, 98, 25, 53, 61 theo chiều tăng dần.

Bước 1: tại bước này, khi duyệt từ cuối dãy lên, các cặp cần phải đổi chỗ (98, 6), (49, 6), (17, 6), (32, 6). Phần tử 6 nổi lên vị trí đầu tiên



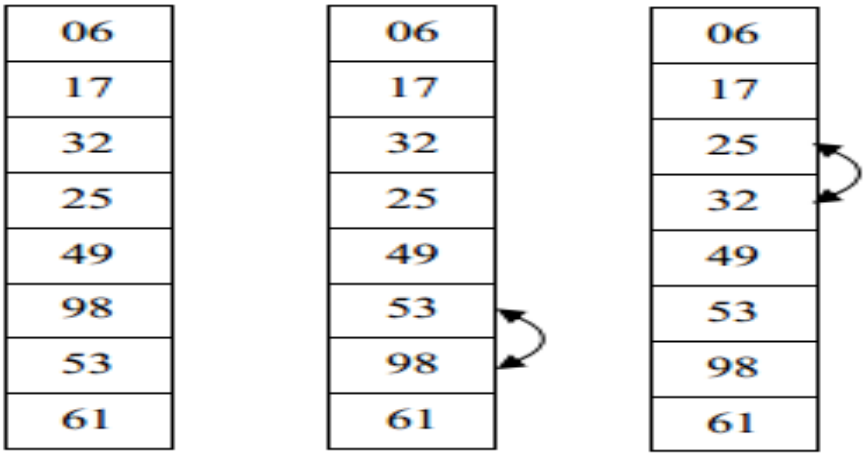
Hình ảnh 1.

Bước 2: tại bước này, khi duyệt từ cuối dãy lên, các cặp cần phải đổi chỗ (98, 25), (49, 25), (17, 32). Phần tử 17 nổi lên vị trí thứ 2



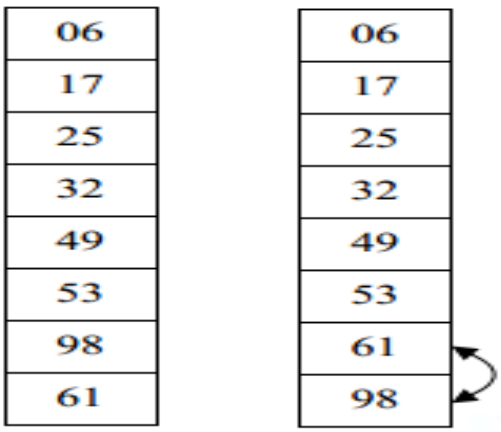
Hình ảnh 2.

Bước 3: tại bước này, khi duyệt từ cuối dãy lên, các cặp cần phải đổi chỗ (98, 53), (32, 25). Phần tử 25 nổi lên vị trí thứ 3



Hình ảnh 3.

Bước 4: tại bước này, khi duyệt từ cuối dãy lên, cặp cần phải đổi chỗ (98, 61).



Hình ảnh 4.

Hình ảnh code ví dụ sắp xếp nổi bọt:

```c
#include <stdio.h>
#define N 8
int main(){
    int i, j, temp, Mang[N]={6, 32, 17, 49, 98, 25, 53, 61};

    for (int i=0; i < N-1; i++){
        for(int j = 0; j < N-(i+1); j++){
            if (Mang[j] > Mang[j+1]){
                int temp = Mang[j];
                Mang[j] = Mang[j+1];
                Mang[j+1] = temp;
            }
        }
    }
    printf("\nMang duoc sap xep noi bot la: ");
    for (i=0; i < N; i++)
    printf(" %d ", Mang[i]);
}
```

Chạy thử:

```
2
3   Mang duoc sap xep noi bot la:  6  17  25  32  49  53  61  98
4   --------------------------------
5   Process exited after 3.422 seconds with return value 0
6   Press any key to continue . . .
7
```

Phần V: Tìm kiếm (S earch)

5.1. Tìm kiếm tuyến tính

Đây là cách đơn giản nhất để tìm kiếm một phần tử trong mảng, bằng cách duyệt tất cả các phần tử của mảng cho tới khi tìm thấy phần tử cần tìm.

Còn được gọi là tìm kiếm tuần tự (Sequential searching)

Hình ảnh code minh họa tìm kiếm tuyến tính**:**

```
1
2   #include <stdio.h>
3   #include <conio.h>
4   #define N 6
5   int main(){
6       int Mang[N] = {5, 2, 9, 15, 6, 10};
7       int i;
8       int a = 15;
9       for(i = 0; i < 6; i++){
10          if(a == Mang[i])
11          break;
12      }
13      printf("So %d tim thay tai vi tri %d", a, i);
14  }
15
```

Chạy thử:

```
3   So 15 tim thay tai vi tri 3
4   --------------------------------
5   Process exited after 0.06167 seconds with return value 0
6   Press any key to continue . . .
```

5.2. Tìm kiếm nhị phân:

Tìm kiếm nhị phân được thực hiện trên mảng đã được sắp xếp.

Thuật toán này sử dụng đệ quy để thực hiện tìm kiếm:

- Bước 1: So sánh phần thử tìm kiếm với phần tử ở vị trí giữa mảng. Nếu kết quả so sánh là bằng nhau, kết thúc tìm kiếm.
- Bước 2: Nếu kết quả so sánh là nhỏ hơn thì lặp lại bước 1 với phần bên trái của mảng.
- Bước 3: Nếu kết quả so sánh là lớn hơn lặp lại bước 1 với phần bên phải của mảng.

Hình ảnh code minh họa tìm kiếm nhị phân:

```c

#include <stdio.h>
#include <conio.h>

int main(){
    int Mang[] = {0,11,13,14,15,17,18};
    int Duoi = 0;
    int Tren = 6;
    int TimKiemGiaTri = 20;
    int flag = 0;

    while (Duoi <= Tren){
        int Giua = (Tren + Duoi)/2;
        if (TimKiemGiaTri == Mang[Giua]){
            flag = 1;
            printf("Phan tu %d xuat hien tai vi tri so: %d\n", TimKiemGiaTri, Giua);
            break;
        }
        else if (TimKiemGiaTri < Mang[Giua]){
            Tren = Giua -1; }
        else if (TimKiemGiaTri > Mang[Giua]){
            Duoi = Giua + 1; }
    }
    if(flag == 0){
        printf("Phan tu %d khong tim thay duoc trong mang", TimKiemGiaTri);
    }
}

```

Chạy thử:

```
Phan tu 20 khong tim thay duoc trong mang
--------------------------------
Process exited after 0.04943 seconds with return value 0
Press any key to continue . . .
```

Phần VI: Code Đề tài quản lí sách

6.1. Các hàm trong chương trình:

- Hàm xuất vào tệp: là hàm ghi thông tin sách vào file book.dat

```
book_st xuatvaotep( book_st booklist[],int n, char fileName[]){
    FILE *fp;
    fp = fopen (fileName,"w");
     fprintf(fp, "%20s%5s%5s\n", "Ten","theloai", "giatien");
    for(int i = 0;i < n;i++){
        fprintf(fp, "%20s%5s%5d\n", booklist[i].Ten,booklist[i].theloai, booklist[i].giatien);
    }
    fclose (fp);
}
```

Hình ảnh: hàm xuất vào tệp

- Hàm menu: Nhập dữ liệu, xuất chi tiết từng cuốn sách, sắp xếp tên theo thứ tự Z-A, tìm kiếm cuốn sách theo thể loại, xuất vào tệp tin nhị phân book.dat và phần thoát chương trình.

```
void menu(book_st booklist[], int n){
    char fileName[] = "book.dat";
    int luachon;
    do{
        system("cls");
        printf("\n\t\t*********************************************\n");
        printf("\t\t**    CHUONG TRINH QUAN LY SACH          **\n");
        printf("\t\t**     1. Nhap Thong Tin sach            **\n");
        printf("\t\t**     2. xuat chi tiet tung cuon sach   **\n");
        printf("\t\t*********************************************\n");
        printf("\t\t**     3. tim kiem sach theo the loai    **\n");
        printf("\t\t**     4.  xuat vao tep                  **\n");
        printf("\t\t**     5. Thoat                          **\n");
        printf("\t\t*********************************************\n");
        printf("\t\t**       Nhap lua chon cua ban           **\n");
         scanf("%d", &luachon);
```

Hình ảnh: hàm menu

- Hàm main:

```
22  int main()
23      {
24      int n=3;
25   book_st *bookList = (struct book_st *)malloc(n * sizeof(struct book_st));
26      char fileName[] = "book.dat";
27      book_st booklist[3];
28       system("color e");
29      menu(booklist,n);
30      return 0;
31  }
```

Hình ảnh: hàm main

- Hàm nhập: dùng để nhập thông tin quyển sách, tên quyển sách, thể loại và giá tiền của quyển sách.

```
87  void nhap(book_st booklist[] ,int n){
88      for(int i=0;i<n;i++){
89      printf("\n\n\t\tNHAP THONG TIN QUYEN SACH %d ",i);
90      fflush(stdin);
91      printf("\nNhap ten quyen sach : ");
92      gets(booklist[i].Ten);
93      fflush(stdin);
94      printf("\nnhap the loai: ");
95      gets(booklist[i].theloai);
96      fflush(stdin);
97      printf("\nNhap gia tien: ");
98      scanf("%d",&booklist[i].giatien);
99    }
100 }
```

Hình ảnh: hàm nhập

- Hàm hoán vị: hóa vị các cuốn sách.

```
126 void hoanvi( book_st &book1,book_st &book2){
127     book_st tempt;
128     tempt = book1;
129             book1=book2;
130             book2 = tempt;
131 }
```

Hình ảnh: hàm hoán vị

- Hàm sắp xếp: là hàm sắp xếp theo thể loại.

```
132 void sapxep(book_st booklist[],int n){
133     for(int i=1;i<n;i++){
134         for(int j=0;j<n-i;j++){
135             if(strcmp(booklist[j].theloai,booklist[j+1].theloai)<0){
136                 hoanvi(booklist[j],booklist[j+1]);
137
138
139             }
140         }
141     }
142 }
```

Hình ảnh: hàm sắp xếp

- Hàm tìm sách: là hàm tìm thông tin sách theo thể loại.

```
105 int timsach(book_st booklist[],int n)
106     {
107      char timsach[30];
108     printf("\n Nhap ten the loai : ");
109     fflush(stdin);
110     gets(timsach);
111     for(int i=0;i<n;i++)
112     {
113      if(strcmp(timsach,booklist[i].theloai) == 0)
114      {
115          printf("STT\t ten\t theloai\t giatien\n");
116          printf("%d\t %s\t %s\t   \t%d\n",i,booklist[i].Ten,booklist[i].theloai,booklist[i].giatien);
117        }
118       else if(strcmp(timsach,booklist[i].theloai)!=0)
119             {
120             printf("\nkhong co cuon sach thuoc the loai nay\n");
121             }
122             break;
123     }
124 }
```

Hình ảnh: hàm tìm sách

- Hàm xuất chi tiết cuốn sách:

```
147 void xuatChiTietCuonSach(book_st booklist[], int n){
148      sapxep(booklist,n);
149      printf("STT\t ten\t theloai\t giatien\n");
150     for(int i=0;i<n;i++){
151
152         printf("%d\t %s\t\t %s\t\t %d\n",i,booklist[i].Ten,booklist[i].theloai,booklist[i].giatien);
153     }
154
155 }
```

Hình ảnh: hàm xuất chi tiết cuốn sách

- Hàm thống kê: là thống kê sách theo thể loại.

```
156 int thongke(book_st booklist[],int n){
157     char theloai[30];
158     int dem=1;
159     for(int i=1;i<n;i++)
160  {
161     for(int j=0;j<n-i;j++)
162     {
163 
164      if(strcmp(booklist[j].theloai,booklist[j+1].theloai) == 0)
165         {
166            dem= dem + 1;
167         }
168 
169    if(strcmp(booklist[j].theloai,booklist[j+1].theloai) != 0)
170         {
171             dem=dem;
172         }
173     }
174  }
175    return dem;
176 }
```

Hình ảnh: hàm xuất thống kê sách theo thể loại

- Hàm xuất thống kê: là hiển thị thống kê số quyển sách theo thể loại.

```
177 void xuatthongke(book_st booklist[],int n)
178 {
179     int dem = thongke(booklist,n);
180     for(int i=0;i<n;i++){
181     printf("\nthe loai %s co so %d quyen",booklist[i].theloai,dem);
182 }
183 }
```

Hình ảnh: hàm xuất thống kê số quyển sách theo thể loại

- Hàm xuất vào tệp: Ghi vào tập tin nhị phân book.dat

```
184 book_st xuatvaotep( book_st booklist[],int n, char fileName[]){
185     FILE *fp;
186     fp = fopen (fileName,"w");
187      fprintf(fp, "%20s%5s%5s\n", "Ten","theloai", "giatien");
188     for(int i = 0;i < n;i++){
189         fprintf(fp, "%20s%5s%5d\n", booklist[i].Ten,booklist[i].theloai, booklist[i].giatien);
190     }
191     fclose (fp);
192 }
```

Hình ảnh: hàm xuất vào tệp

6.2. Hướng dẫn chạy code:

Khi chọn 1 từ Menu, thực hiện nhập dữ liệu của từng quyển sách theo yêu cầu dưới đây.

Ví dụ:

Bước 1: nhập lựa chọn thứ 1

```
        ******************************************
        **    CHUONG TRINH QUAN LY SACH         **
        **     1. Nhap Thong Tin sach           **
        **     2. xuat chi tiet tung cuon sach  **
        ******************************************
        **     3. tim kiem sach theo the loai   **
        **     4.  xuat vao tep                 **
        **     5. Thoat                         **
        ******************************************
        **       Nhap lua chon cua ban          **
1
```

Hình ảnh: nhập lựa chọn

Bước 2: Nhập thông tin quyển sách, thể loại, giá tiền

```
        ******************************************
        **    CHUONG TRINH QUAN LY SACH         **
        **     1. Nhap Thong Tin sach           **
        **     2. xuat chi tiet tung cuon sach  **
        ******************************************
        **     3. tim kiem sach theo the loai   **
        **     4.  xuat vao tep                 **
        **     5. Thoat                         **
        ******************************************
        **       Nhap lua chon cua ban          **
1


        NHAP THONG TIN QUYEN SACH 0
Nhap ten quyen sach : Conan

nhap the loai: phieu luu

Nhap gia tien: 50000
```

Hình ảnh: nhập thông tin, thể loại, giá tiền của quyển sách

Bước 3: Xuất chi tiết từng cuốn sách

```
        ****************************************
        **    CHUONG TRINH QUAN LY SACH       **
        **     1. Nhap Thong Tin sach         **
        **     2. xuat chi tiet tung cuon sach **
        ****************************************
        **     3. tim kiem sach theo the loai  **
        **     4.  xuat vao tep                **
        **     5. Thoat                        **
        ****************************************
        **       Nhap lua chon cua ban         **
2
STT     ten             theloai                 giatien
0       doremon         vui nhon                50000
1       conan           kinh di                 69000
2       haki            hanh dong               40000

the loai vui nhon co so 1 quyen
the loai kinh di co so 1 quyen
the loai hanh dong co so 1 quyen
Bam phim bat ky de tiep tuc!
```

Hình ảnh: xuất chi tiết từng cuốn sách

Bước 4: Tìm kiếm sách theo thể loại

```
        ****************************************
        **    CHUONG TRINH QUAN LY SACH       **
        **     1. Nhap Thong Tin sach         **
        **     2. xuat chi tiet tung cuon sach **
        ****************************************
        **     3. tim kiem sach theo the loai  **
        **     4.  xuat vao tep                **
        **     5. Thoat                        **
        ****************************************
        **       Nhap lua chon cua ban         **
3

 Nhap ten the loai : vui nhon
STT     ten         theloai         giatien
0       doremon     vui nhon        50000

Bam phim bat ky de tiep tuc!
```

Hình ảnh: tìm kiếm sách theo thể loại

Bước 5: Xuất và tệp

```
*****************************************
**    CHUONG TRINH QUAN LY SACH        **
**     1. Nhap Thong Tin sach          **
**     2. xuat chi tiet tung cuon sach  **
*****************************************
**     3. tim kiem sach theo the loai   **
**     4.  xuat vao tep                 **
**     5. Thoat                         **
*****************************************
**       Nhap lua chon cua ban          **
4

Xuat thong tin sach thanh cong vao file book.dat!

Bam phim bat ky de tiep tuc!
```

Hình ảnh: xuất file vào tệp

Bước 6: Thoát chương trình

```
*****************************************
**    CHUONG TRINH QUAN LY SACH        **
**     1. Nhap Thong Tin sach          **
**     2. xuat chi tiet tung cuon sach  **
*****************************************
**     3. tim kiem sach theo the loai   **
**     4.  xuat vao tep                 **
**     5. Thoat                         **
*****************************************
**       Nhap lua chon cua ban          **
5

Ban da chon thoat chuong trinh!
--------------------------------
Process exited after 25.38 seconds with return value 0
Press any key to continue . . .
```

Hình ảnh: thoát chương trình

Tài Liệu Tham Khảo

[1] “Nguyễn Văn Hiếu”,2019[online],Available: https://nguyenvanhieu.vn/doc- ghi- file-trong-c/ , [Accessed 14 6 2020]

[2]“Nguyễn Văn Hiếu”,2019[online],Available: https://nguyenvanhieu.vn/danh-sach-lien-ket-don/[Accessed 15 6 2020]

[3]“Tìm ở đây”, [online], Available: https://timoday.edu.vn/cac-thuat-toan-sap-xep-va-tim-kiem/ [Accessed 10 6 2020]

[4] “viettuts”,[online], Available: https://viettuts.vn/bai-tap-c/sap-xep-chon-selection-sort-trong-c [Accessed 10 6 2020]

[5] Phạm Văn Ất, “Kỹ thuật lập trình C”, Nhà Xuất Bản Bách Khoa Hà Nội, 2017.