MỤC LỤC

[LỜI NÓI ĐẦU 1](#_Toc43800362)

[CHƯƠNG 1: CƠ SỞ LÝ THUYẾT 2](#_Toc43800363)

[1.1 Danh sách liên kết đơn 2](#_Toc43800364)

[**1.1.1** **Khái niệm danh sách liên kết, danh sách liên kết đơn** 2](#_Toc43800365)

[**1.1.2** **Cài đặt danh sách liên kết đơn** 2](#_Toc43800366)

[1.2 Làm việc với tệp(File) 8](#_Toc43800367)

[**1.2.1** **Khái niệm** 8](#_Toc43800368)

[**1.2.2** **Đặc điểm** 9](#_Toc43800369)

[**1.2.3** **Các kiểu file cơ bản** 9](#_Toc43800370)

[**1.2.4** **Các thao tác với file** 9](#_Toc43800371)

[**1.2.5** **Ghi/đọc file văn bản trong C** 12](#_Toc43800372)

[1.3 Các thuật toán sắp xếp 13](#_Toc43800373)

[**1.3.1** **Khái niệm** 13](#_Toc43800374)

[**1.3.2** **Phân loại thuật toán sắp xếp** 13](#_Toc43800375)

[**1.3.3** **Một số thuật toán sắp xếp** 13](#_Toc43800376)

[1.4 Các thuật toán tìm kiếm 23](#_Toc43800377)

[**1.4.1** **Khái niệm** 23](#_Toc43800378)

[**1.4.2** **Một số thuật toán tìm kiếm** 23](#_Toc43800379)

[CHƯƠNG 2: CHƯƠNG TRÌNH QUẢN LÝ HÀNG HÓA 27](#_Toc43800380)

[2.1 Lí do chọn đề tài 27](#_Toc43800381)

[2.2 Mục đích của đề tài 27](#_Toc43800382)

[2.3 Giới thiệu 27](#_Toc43800383)

[2.4 Xây dựng chương trình quản lý hàng hóa 28](#_Toc43800384)

[**2.4.1 Các bước xây dựng** 28](#_Toc43800385)

[**2.4.2** **Chương trình** 33](#_Toc43800386)

[**2.4.3** **Kết luận** 35](#_Toc43800387)

[TÀI LIỆU THAM KHẢO 36](#_Toc43800388)

# LỜI NÓI ĐẦU

Trong quá trình làm bài tập lớn và báo cáo sẽ không tránh khỏi những sai sót vì đây là lần đầu tiên nên mong cô có thể góp ý và chia sẻ cho nhóm chúng em để rút kinh nghiệm lần sau.

# CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

* 1. **Danh sách liên kết đơn**
     1. **Khái niệm danh sách liên kết, danh sách liên kết đơn**
* Danh sách liên kết: Danh sách liên kết là tập các phần tử liên kết móc nối liên tiếp với nhau, có kiểu truy cập tuần tự. Mỗi một phần tử là một nút (Node).

Mỗi nút gồm hai phần:

- Dữ liệu (Data): là các thành phần dữ liệu mà một nút đó lưu trữ

- Liên kết (Linked): là con trỏ kiểu nút đang định nghĩa được dùng để liên kết với các nút khác.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.

* Danh sách liên kết đơn: Là một danh sách liên kết chỉ có sự liên kết móc nối từ phần tử phía trước tới phần tử phía sau.



Hình 1.1. Mô tả cho một Node trong danh sách liên kết đơn



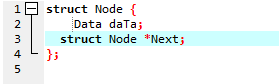
Hình 1.2. Mô phỏng một danh sách liên đơn kết đầy đủ

* + 1. **Cài đặt danh sách liên kết đơn**
       1. **Khai báo**

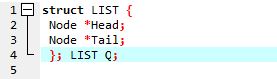
Để khai báo cấu trúc dữ liệu dạng danh sách liên kết đơn ta thực hiện:

- Khai báo cấu trúc dữ liệu của thông tin được lưu trữ - Data

- Khai báo cấu trúc của một nút – Node



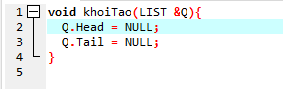
- Khai báo danh sách: Mỗi danh sách được xác định bởi phần tử đầu (Head) và phần tử đuôi (Tail)



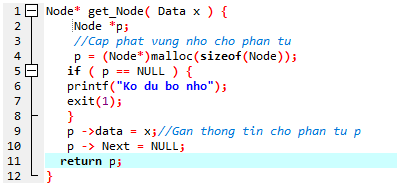
* + - 1. **Các thao tác cơ bản trên danh sách liên kết đơn**

1. Khởi tạo danh sách rỗng

Danh sách rỗng là danh sách chưa có chứa bất kỳ phần tử nào, có nghĩa là phần tử đầu và phần tử cuối của danh sách trỏ đến địa chỉ NULL.

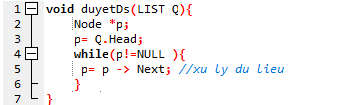


1. Tạo một phần tử với thông tin chứa trong x



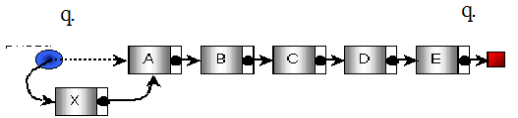
1. Duyệt danh sách

Khi duyệt danh sách chúng ta xuất phát từ đầu danh sách và duyệt qua từng phần tử của danh sách và thực hiện xử lý dữ liệu.



1. Thêm một nút vào danh sách

* **Thêm vào đầu danh sách**



Hình 1.3. Minh họa thêm phần tử mới vào đầu danh sách liên kết đơn

Giả sử ta có 1 danh sách mà đầu của danh sách được trỏ tới bởi con trỏ q.Head, cuối danh sách được trỏ bởi con trỏ q.Tail.

* Thuật toán: -Nếu danh sách rỗng:

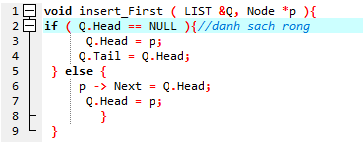
+ Phần tử đầu là phần tử mới chèn vào

+ Phần tử cuối cũng là phần tử đầu

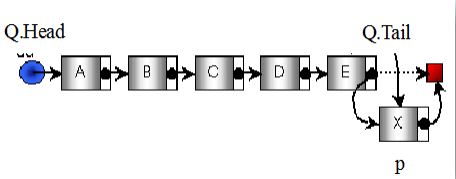
-Ngược lại (danh sách khác rỗng):

+ Phần tử mới trỏ tới phần tử đầu

+ Phần tử đầu là phần tử mới chèn vào



* **Thêm vào cuối danh sách**

****

Hình 1.4. Minh họa chèn thêm phần tử mới vào cuối danh sách

Để con trỏ tiếp của nút cuối q.Tail trỏ đến nút mới tạo là p, và cho con trỏ tiếp của p trỏ tới NULL.

* Thuật toán: -Nếu danh sách rỗng:

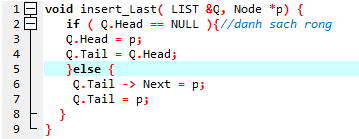
+Phần tử đầu là phần tử mới chèn vào

+Phần tử cuối cũng là phần tử đầu

-Ngược lại (danh sách khác rỗng):

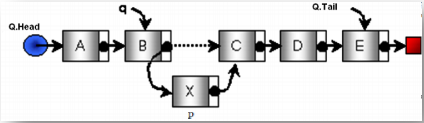
+ Phần tử cuối trỏ tới phần tử mới chèn vào

+ Phần tử cuối là phần tử mới chèn vào



* **Thêm vào sau phần tử cho trước**

Thêm một phần tử vào sau phần tử q của danh sách. Cho con trỏ tiếp của p trỏ tới phần tử đứng sau q, con trỏ tiếp của q trỏ tới p.

****

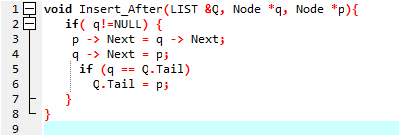
Hình 1.5. Minh họa chèn thêm phần tử mới vào sau phần tử q xác định

* Thuật toán: -Nếu phần tử q rỗng ⬄ không thêm được

-Nếu phần tử q khác rỗng ⬄ có tồn tại phần tử q

+ Phần tử mới trỏ tới phần tử đứng sau phần tử q

+ Phần tử q trỏ tới phần tử mới thêm vào



1. Xóa một nút trong danh sách

* **Xóa phần tử đầu danh sách**

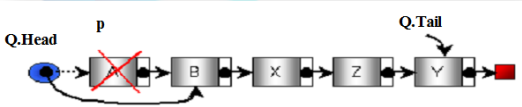
- Kiểm tra xem danh sách rỗng hay là không rỗng.

- Lưu phần tử đầu tạm thời vào p.

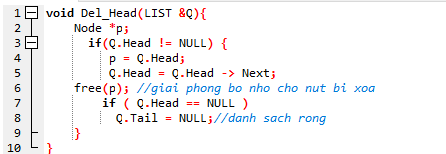
- Chuyển phần tử đầu tới phần tử tiếp theo.

- Xóa phần tử đầu đã được lưu tạm thời (xóa p).

- Kiểm tra: Nếu danh sách chỉ có một phần tử, khi xóa phần tử đi thì danh sách rỗng.



Hình 1.6. Minh họa xóa phần tử đầu danh sách

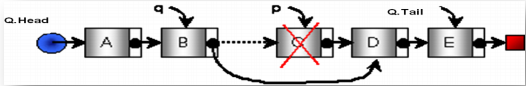


* **Xóa phần tử đứng sau phần tử q của danh sách**

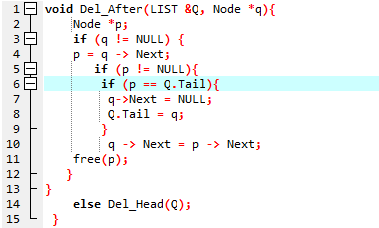
- Nếu có phần tử q: Lưu phần tử đứng sau phần tử q (lưu vào p)

- Nếu có phần tử p (q không phải là phần tử cuối)

- Tách phần tử p ra khỏi danh sách



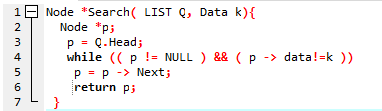
Hình 1.7. Minh họa xóa phần tử đứng sau phần tử q xác định



1. Tìm kiếm trong danh sách

Danh sách liên kết đơn chỉ cho phép truy cập tuần tự đến từng phần tử nên chỉ có thể áp dụng thuật toán tìm kiếm tuần tự để tìm xem một phần tử có giá trị k có tồn tại trong danh sách hay không.

Sử dụng một con trỏ p, trỏ đến đầu danh sách rồi sau đó lần lượt duyệt qua từng phần tử của danh sách. Trong quá trình duyệt so sánh giá trị của mỗi phần tử với giá trị cần tìm k, nếu không bằng thì đi tiếp đến hết ngược lại trả về địa chỉ nút tìm thấy.

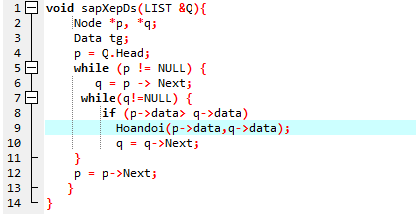


1. Sắp xếp danh sách

Sử dụng 2 con trỏ p, q để duyệt và so sánh với nhau. Ban đầu con trỏ p trỏ đến đầu danh

sách. Con trỏ q trỏ đến phần tử sau p. So sánh giá trị của phần tử p và q nếu không

đúng trật tự thì hoán đổi giá trị của p và q cho nhau. Quá trình này được thực hiện lặp lại cho đến khi hết toàn bộ dãy.



* 1. **Làm việc với tệp(File)**
     1. **Khái niệm**

File (còn gọi là tập tin, tệp) là một tập hợp của thông tin được đặt tên, và được lưu trữ trong bộ nhớ máy tính. Các thông tin này là một hay nhiều chuỗi kí tự, kí hiệu giống hoặc khác nhau.

* + 1. **Đặc điểm**

-Một file thường được lưu trữ ở secondary storage(đĩa cứng, đĩa mềm, các thiết bị lưu trữ dữ liệu kỹ thuật số khác…).

-Nội dung file có thể là rỗng và file này còn gọi là file rỗng hay file trống.

-Độ dài của file thường tùy thuộc vào kiến trúc phần cứng cũng như khả năng phần mềm.

-Đơn vị độ lớn của file là byte.

-Tên file thường bao gồm 2 phần: phần tên và phần mở rộng (không nhất thiết phải có).

* + 1. **Các kiểu file cơ bản**
* File văn bản – text files: là loại tập tin dùng để ghi các ký tự lên đĩa.

- File văn bản là file thường có đuôi là .txt.

- Điểm đặc biệt là dữ liệu của tập tin được lưu trữ thành các dòng, mỗi dòng được kết thúc bằng ký tự xuống dòng (new line), ký hiệu “\n”.

- Truy xuất tập tin theo kiểu văn bản chỉ có thể truy xuất theo kiểu tuần tự.

* File nhị phân – Binary files: Dữ liệu được xem như là một dãy byte liên tục, việc xử lý với tập tin dựa trên việc đọc ghi dãy byte.

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

- Khi một tập tin được mở ra để làm việc, tại mỗi thời điểm, sẽ có một vị trí của tập tin mà tại đó việc đọc/ghi thông tin sẽ xảy ra. Người ta hình dung có một con trỏ đang chỉ đến vị trí đó và đặt tên nó là con trỏ tập tin.

- Sau khi đọc/ghi xong dữ liệu, con trỏ sẽ chuyển dịch thêm một phần tử về phía cuối tập tin.

* + 1. **Các thao tác với file**
* Khai báo
* Khi làm việc với file, bạn cần khai báo một 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.
* Cú pháp: FILE < Danh sách các biến con trỏ, đại diện cho tập tin >;
* VD: FILE \*f ;
* Mở file
* Cú pháp: FILE \*fopen(char \*Path, const char \*Mode) ;

Trong đó:

* Path: chuỗi chỉ đường dẫn đến tập tin trên đĩa.
* Mode: chuỗi xác định cách thức mà tập tin sẽ mở.

Bảng1.1. Các mode trong file

|  |  |
| --- | --- |
| Mode | Ý nghĩa |
| r | Mở tập tin văn bản để đọc |
| w | Tạo ra tập tin văn bản mới để ghi |
| a | Nối vào tập tin văn bản |
| rb | Mở tập tin nhị phân để đọc |
| wb | Tạo ra tập tin nhị phân để ghi |
| ab | Nối vào tập tin nhị phân |
| r+ | Mở một tập tin văn bản để đọc/ghi |
| w+ | Tạo ra tập tin văn bản để đọc ghi |
| a+ | Nối vào hay tạo mới tập tin văn bản để đọc/ghi |
| r+b | Mở ra tập tin nhị phân để đọc/ghi |
| w+b | Tạo ra tập tin nhị phân để đọc/ghi |
| a+b | Nối vào hay tạo mới tập tin nhị phân |

* Hàm fopen trả về một con trỏ tập tin. Chương trình của ta không thể thay đổi giá trị của con trỏ này. Nếu có một lỗi xuất hiện trong khi mở tập tin thì hàm này trả về con trỏ NULL.
* VD:

f = fopen(" card.txt ", " w "); // Mở một tập tin tên card.txt để ghi.

* Đóng file
* Khi làm việc với tập tin hoàn tất, kể cả là file nhị phân hay file văn bản chúng ta cần phải đóng file sau khi làm việc với nó xong.
* Việc đóng file mở bởi hàm fopen() được thực hiện bằng cách dùng hàm fclose().
* Cú pháp: int fclose(FILE \*f);
* Trong đó f là con trỏ tập tin được mở bởi hàm fopen(). Giá trị trả về của hàm là 0 báo rằng việc đóng tập tin thành công. Hàm trả về EOF nếu có xuất hiện lỗi.
* VD: fclose( f );
* Kiểm tra đến cuối tập tin hay chưa ?
* Cú pháp: int feof(FILE \*f);
* Kiểm tra xem đã chạm tới cuối tập tin hay chưa và trả về EOF nếu cuối tập tin được chạm tới, ngược lại trả về 0.

Trả về:   -1  (Đã hết file)

0  (Chưa hết file)

Số nguyên khác (Error!)

* Di chuyển con trỏ tập tin về đầu tập tin
* Sử dụng hàm rewind().
* Cú pháp:  void rewind(FILE \*f);
* f: con trỏ tập tin đang thao tác.
* Chuyển đến vị trí bất kỳ
* Sử dụng hàm fseek().
* Cú pháp: int fseek( FILE \*f, long sb, int xp);

Trong đó:

+ f: con trỏ tập tin đang thao tác.

+ sb: số byte cần dịch chuyển con trỏ tập tin.

+ xp: vị trí bắt đầu để tính khoảng cách dịch chuyển cho sb.

xp có thể nhận các giá trị:

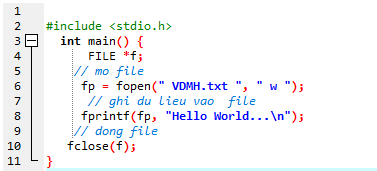
xp = SEEK\_SET hay 0: Xuất phát từ đầu tệp.

xp = SEEK\_CUR hay 1: Xuất phát từ vị trí hiện tại của con trỏ chỉ vị.

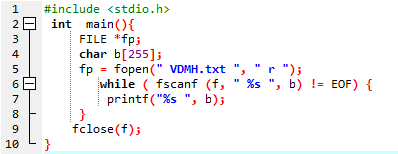
xp= SEEK\_END hay 2: Xuất phát từ cuối tệp.

+ Kết quả trả về của hàm là 0 nếu việc di chuyển thành công. Nếu không thành công, một giá trị khác 0 (error) được trả về.

* + 1. **Ghi/đọc file văn bản trong C**
* Ghi file
* Sử dụng hàm fprintf().
* VD:



* Đọc file
* Sử dụng hàm fsanf().
* VD:



* 1. **Các thuật toán sắp xếp**
     1. **Khái niệm**

Thuật toán sắp xếp là một [thuật toán](https://vi.wikipedia.org/wiki/Thu%E1%BA%ADt_to%C3%A1n) sắp xếp các phần tử của một [danh sách](https://vi.wikipedia.org/wiki/Danh_s%C3%A1ch) (hoặc một [mảng](https://vi.wikipedia.org/wiki/M%E1%BA%A3ng)) theo thứ tự (tăng hoặc giảm).

* + 1. **Phân loại thuật toán sắp xếp**
* Sắp xếp ổn định:

Một thuật toán sắp xếp được gọi là sắp xếp ổn định nếu sau khi tiến hành sắp xếp vị trí tương đối giữa các phần tử bằng nhau không bị thay đổi.

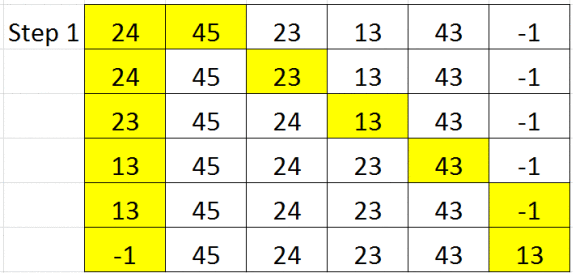
* Sắp xếp so sánh:

Một thuật toán sắp xếp được gọi là sắp xếp so sánh nếu trong quá trình thực hiện thuật toán ta tiến hành so sánh các khoá và đổi chỗ các phần tử cho nhau.

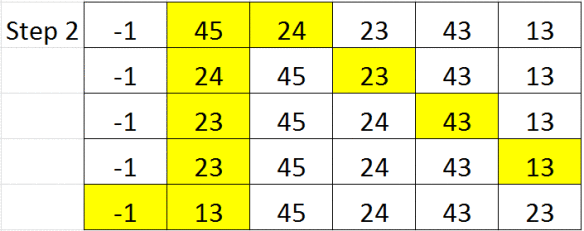
* + 1. **Một số thuật toán sắp xếp**
* Sắp xếp đổi chỗ (interchange sort):
* Xuất phát từ đầu dãy, lần lượt so sánh phần tử đầu dãy với các phần tử còn lại, nếu thấy lớn hơn thì đổi chỗ cho nhau, mục đích là để sau khi quét một lượt, phần tử bé nhất sẽ về đầu dãy.

ex

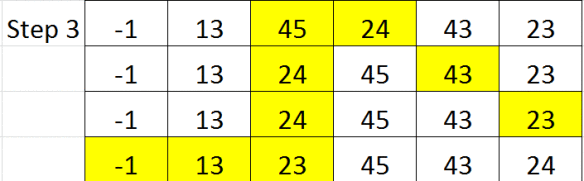
Hình 1.8. VD minh họa sắp xếp đổi chỗ



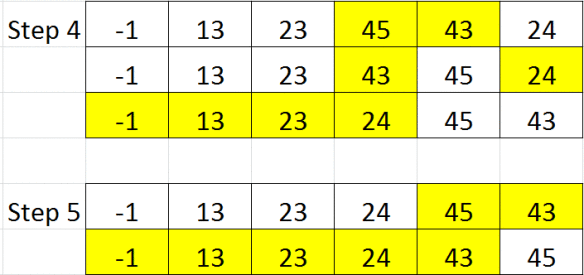
Hình 1.8.1. Bước 1 sắp xếp đổi chỗ



Hình 1.8.2. Bước 2 sắp xếp đổi chỗ

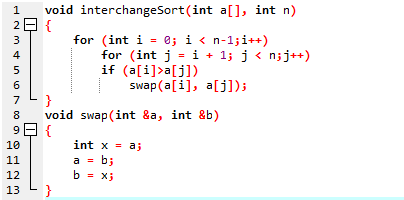


Hình 1.8.3. Bước 3 sắp xếp đổi chỗ



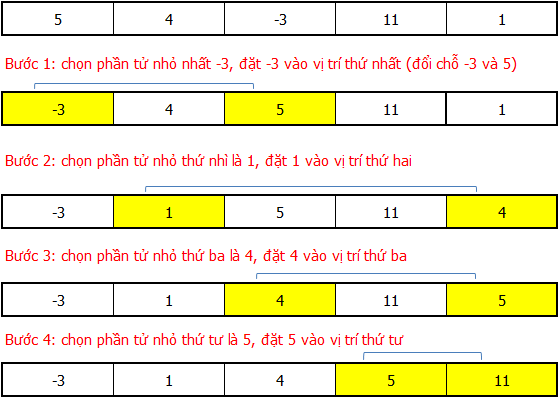
Hình 1.8.4. Bước 4 và Bước 5 sắp xếp đổi chỗ

Code:



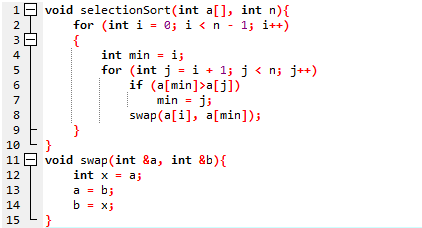
* Sắp xếp chọn (selection sort):  là phương pháp sắp xếp bằng cách chọn phần tử bé nhất xếp vào vị trí thứ nhất, tương tự với các phần tử nhỏ thứ hai, thứ ba,...

VD:  Sắp xếp dãy 5, 4, -3, 11, 1 theo chiều tăng dần

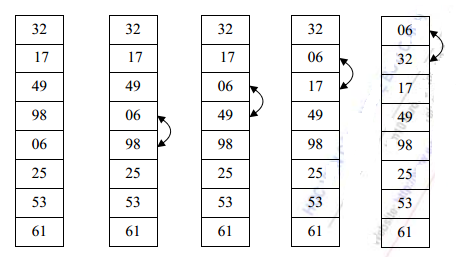


Hình 1.9. VD minh họa sắp xếp chọn

Code:



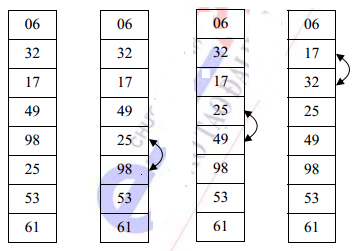
* Sắp xếp nổi bọt (bubble sort):  là phương pháp sắp xếp đơn giản, dễ hiểu thường được dạy trong khoa học máy tính. Nó so sánh hai phần tử cuối, nếu phần tử đứng trước lớn hơn phần tử đứng sau thì đổi chỗ chúng cho nhau. Tiếp tục làm như vậy với cặp phần tử tiếp theo cho đến cuối đầu dãy số. Sau đó nó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



Hình 1.10. VD minh họa, bước 1 của sắp xếp nổi bọt

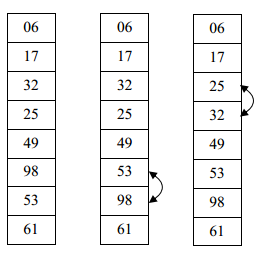
+ Bước 1: Tại bước này, khi duyệt từ cuối dãy lên, các cặp cần phải đổi chỗ (98, 6), (49, 6), (17, 6), (32, 6). Phần tử 6 nổi lên vị trí đầu tiên( hình 1.10).

+ 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 1.10.1).



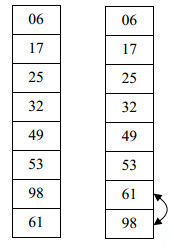
Hình 1.10.1. Bước 2 của sắp xếp nổi bọt

+ 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 1.10.2).



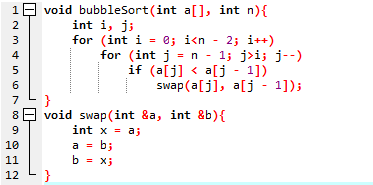
Hình 1.10.2. Bước 3 của sắp xếp nổi bọt

Bước 4: Tại bước này, khi duyệt từ cuối dãy lên, cặp cần phải đổi chỗ (98, 61)( hình 1.10.3)

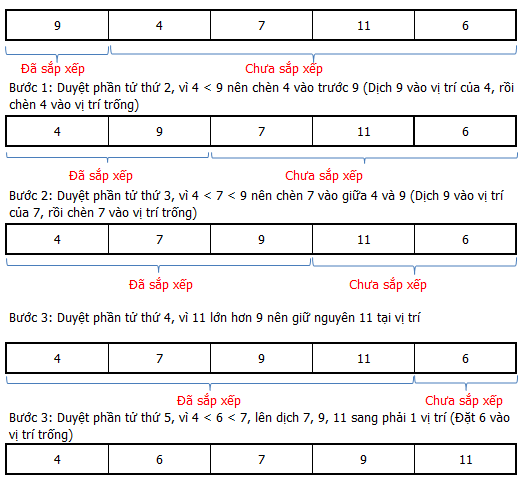


Hình 1.10.3. Bước 4 của sắp xếp nổi bọt

Code:

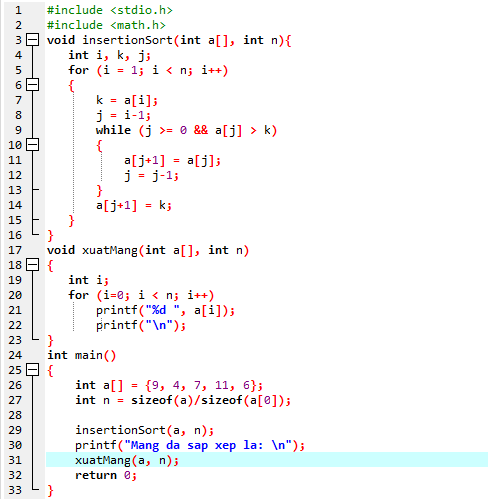


* Sắp xếp chèn (insertion sort): là thuật toán sắp xếp xử lý chèn phần tử đang xét vào vị trí thích hợp của dãy số đã sắp xếp phía trước sao cho dãy số vẫn là dãy sắp xếp có thứ tự.
* VD:  Sắp xếp dãy  theo chiều tăng dần

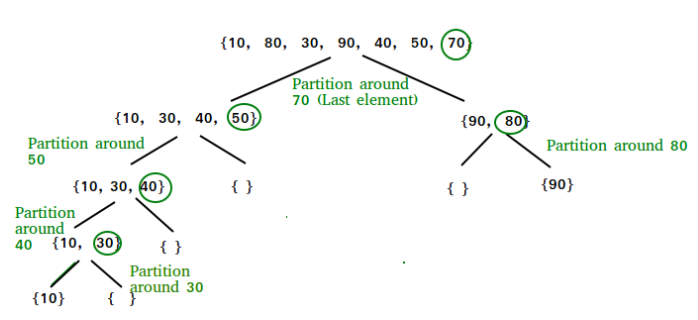


Hình 1.11. VD minh họa sắp xếp chọn

Code:

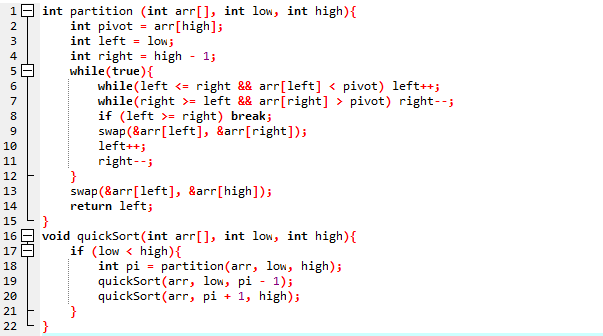


* Sắp xếp nhanh (quick sort):
* Thuật toán sắp xếp quick sort là một thuật toán chia để trị( Divide and Conquer algorithm). Nó chọn một phần tử trong mảng làm điểm đánh dấu(pivot). Thuật toán sẽ thực hiện chia mảng thành các mảng con dựa vào pivot đã chọn. Việc lựa chọn pivot ảnh hưởng rất nhiều tới tốc độ sắp xếp.  Nhưng máy tính lại không thể biết khi nào thì nên chọn theo cách nào. Dưới đây là cách chọn pivot: Luôn chọn phần tử cuối cùng của mảng.
* VD:

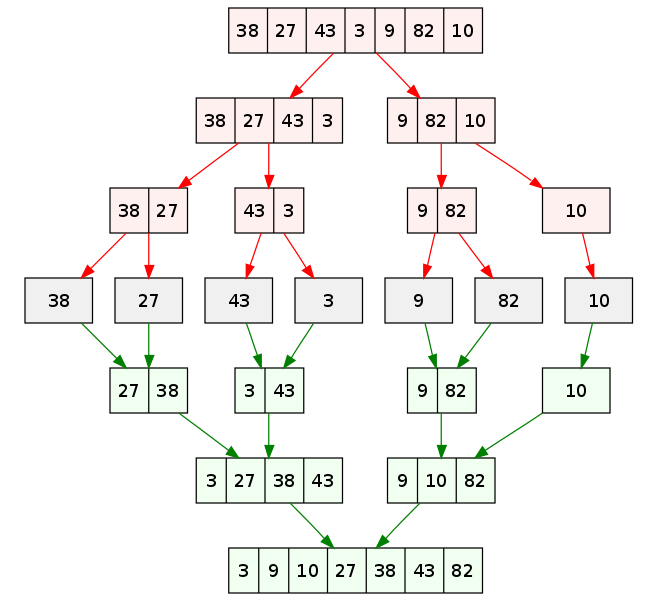


Hình 1.12. VD minh họa sắp xếp nhanh chọn pivot phần tử cuối

Code:

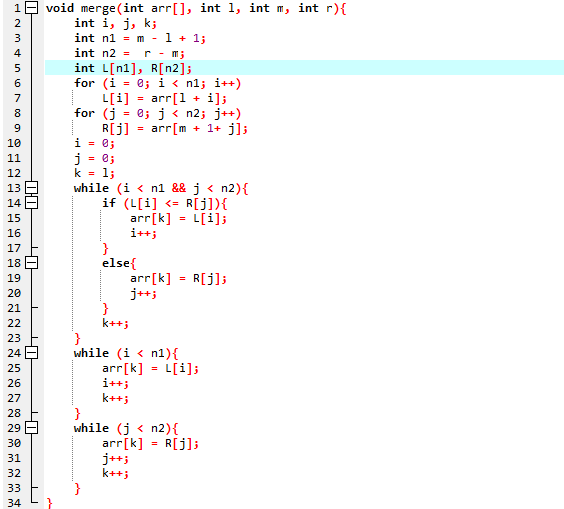


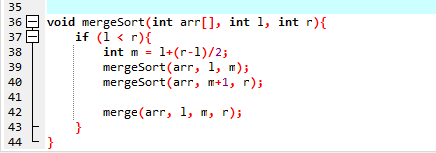
* Sắp xếp trộn (merge sort):
* Thuật toán này chia mảng cần sắp xếp thành 2 nửa. Tiếp tục lặp lại việc này ở các nửa mảng đã chia. Sau cùng gộp các nửa đó thành mảng đã sắp xếp. Hàm merge() được sử dụng để gộp hai nửa mảng. Hàm merge(arr, l, m, r) là tiến trình quan trọng nhất sẽ gộp hai nửa mảng thành 1 mảng sắp xếp, các nửa mảng là arr[l…m] và arr[m+1…r] sau khi gộp sẽ thành một mảng duy nhất đã sắp xếp.
* VD:



Hình 1.13. VD minh họa sắp xếp trộn

Code:

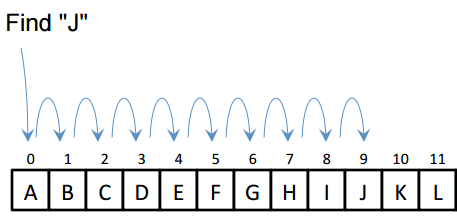




* 1. **Các thuật toán tìm kiếm**
     1. **Khái niệm**

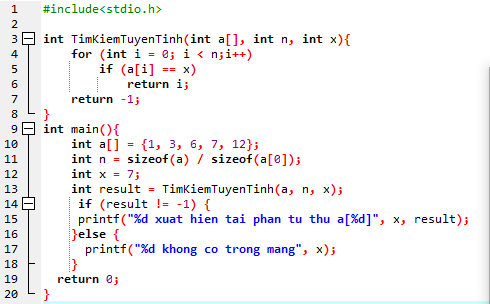
Thuật toán tìm kiếm là thuật toán dùng để tìm kiếm phần tử trong một danh sách cho trước theo một tiêu chí nhất định.

* + 1. **Một số thuật toán tìm kiếm**
* Tìm kiếm tuyến tính(Tìm kiếm tuần tự)
* Là kiểm tra tuần tự từng phần tử của mảng, đến khi nào giống thì thôi.
* VD:



Hình 1.14. VD minh họa tìm kiếm tuyến tính

Code:

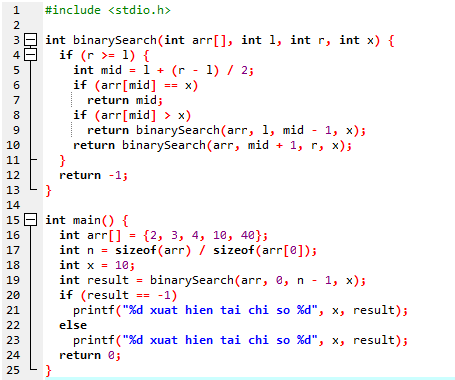


* Tìm kiếm nhị phân:
* Điều kiện của thuật toán này là mảng đã được sắp xếp tăng dần. So sánh x với giá trị của phần tử nằm ở giữa mảng. Nếu x nhỏ thì nó chỉ có thể nằm ở nửa bên trái, ngược lại x lớn hơn thì x nằm ở nửa bên phải. Xác định x nằm ở nửa nào thì ta lặp lại thuật toán với nửa đó.
* VD:



Hình 1.15. VD minh họa tìm kiếm nhị phân

Code:

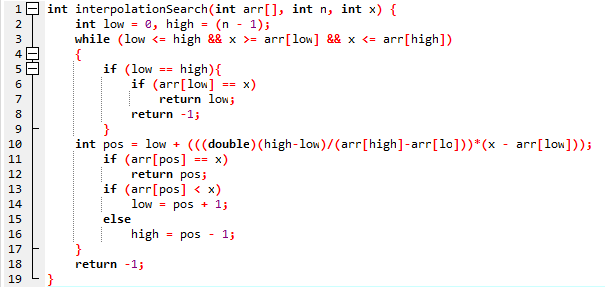


* Tìm kiếm nội suy:
* Là cải tiến của thuật toán tìm kiếm nhị phân. Thay vì chia đôi, thuật toán này chia theo phép tính giúp thu gọn khoảng tìm kiếm hơn.



Hình 1.16. VD minh họa tìm kiếm nội suy

Code:



# CHƯƠNG 2: CHƯƠNG TRÌNH QUẢN LÝ HÀNG HÓA

1. **Lí do chọn đề tài**

* Công việc kinh doanh buôn bán và quản lý hàng hóa đã xuất hiện từ rất lâu, trải qua từng giai đoạn lịch sử nó lại mang một hình thức đặc thù riêng. Trước kia khi các công cụ hỗ trợ công việc mua bán và quản lý thông tin hàng hóa chưa phát triển, hoạt động mua bán diễn ra trực tiếp, việc quản lý hàng hàng hóa được ghi chép lại thủ công bằng giấy bút, các dụng cụ lưu trữ thô sơ. Việc làm mất thông tin hàng hóa, thông tin khách hàng là khó tránh khỏi cho tới khi khoa học công nghệ phát triển. Với sự phát triển của khoa học công nghệ, nó đã tạo ra các loại hình thức mua bán mới cùng với phương thức quản lý thông tin hàng hóa mới, điển hình là mua bán trực tuyến và quản lý hàng hóa trên thiết bị điện tử.
* Việc quản lý hàng hóa trên thiết bị điện tử khá đơn giản và đem lại nhiều lợi ích cho người mua và người bán hàng: tiết kiệm thời gian, tránh khỏi được một vài sai sót, lưu trữ thông tin dễ dàng, có thể khôi phục thông tin qua tệp lưu trữ,...
* Trên cơ sở những kiến thức đã học được trong nhà trường và quá trình tìm hiểu trên mạng internet về ngôn ngữ lập trình C, em đã quyết định chọn đề tài “Chương trình quản lý hàng hóa”.

1. **Mục đích của đề tài**

* Nghiên cứu về các chương trình phần mềm bằng ngôn ngữ lập trình căn bản như: ngôn ngữ lập trình C.
* Phát triển chương trình ứng dụng đơn giản trong thực tế, có tính thực tiễn cao, giúp con người có thể thao tác dễ dàng và tiện lợi.

1. **Giới thiệu**

Khi đến mua hàng tại các siêu thị, mọi người cần lấy cho mình một giỏ hàng hoặc một chiếc xe đẩy hàng để chứa những thứ mà mình muốn mua, trong quá trình mua sắm, có thể chọn hàng cho vào giỏ, tăng giảm số lượng, thay đổi bằng loại hàng hóa khác hoặc bỏ lại một sản phẩm không ưng ý. Những sản phẩm trong giỏ hàng sẽ được lưu trữ tạm thời và được bỏ ra khi thanh toán.

Giỏ hàng trong chương trình quản lý hàng hóa cũng vậy. Nó cho phép thực hiện các chức năng:

+ Tạo giỏ hàng: tên khách hàng, địa chỉ.

+ Chọn hàng: tên sản phẩm, số lượng, giá.

+ Xem chi tiết giỏ hàng: sắp xếp theo tên sản phẩm từ A -> Z, hiển thị thông tin chi tiết.

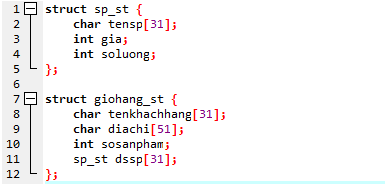
+ Loại bỏ một mặt hàng.

+ Ghi tất cả thông tin của giỏ hàng vào file.

1. **Xây dựng chương trình quản lý hàng hóa**

### **2.4.1 Các bước xây dựng**

* Định nghĩa 1 cấu trúc



* Xây dựng main và menu
* Khai báo 1 biến lưu có kiểu struct “giohang\_st” với tên “mybag”.
* Khai báo tên và kiểu dữ liệu của file.
* Tạo ra menu gồm 6 lựa chọn: 1. Tạo giỏ hàng

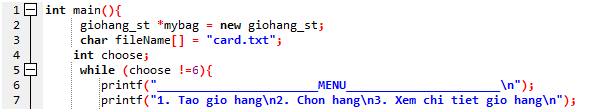
2. Chọn hàng

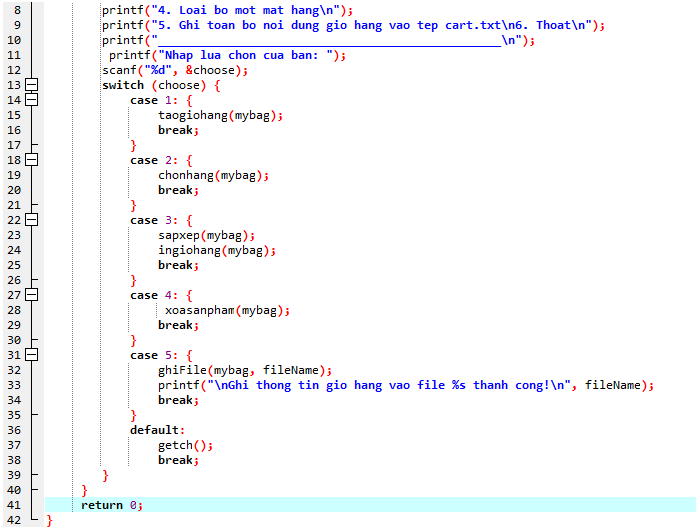
3. Xem chi tiết giỏ hàng

4. Loại bỏ một mặt hàng

5. Ghi tất cả thông tin của giỏ hàng vào file

6. Thoát

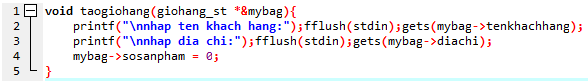




* Xây dựng hàm taogiohang
* Sử dụng tham biến là con trỏ cấu trúc giohang\_st.
* Lưu các giá trị nhập từ người sử dụng vào biến mybag trong hàm main().

Khi chọn 1 từ menu, thực hiện các yêu cầu:

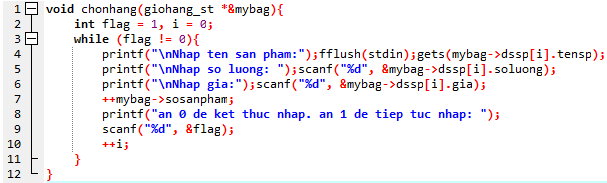
* Hiển thị “nhap ten khach hang:”
* Sau đó, hiển thị “nhap dia chi:”
* Gán số lượng sản phẩm là 0 (giỏ hàng hiện đang trống)



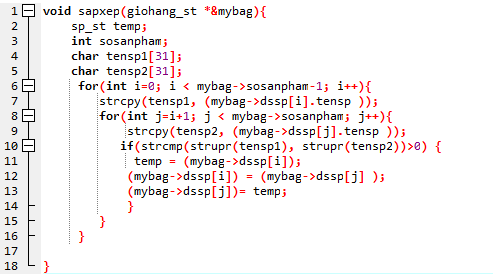
* Xây dựng hàm chonhang
* Sử dụng tham biến là con trỏ cấu trúc giohang\_st
* Tăng số lượng sản phẩm lên 1 sau mỗi lần chọn hàng

Khi chọn 2 từ menu, thực hiện các yêu cầu:

* Hiển thị “Nhap ten san pham:”
* Sau đó hiển thị “Nhap so luong:”
* Sau đó hiển thị “Nhap gia:”
* Cuối cùng hiển thị “an 0 de ket thuc nhap. an 1 de tiep tuc nhap:”



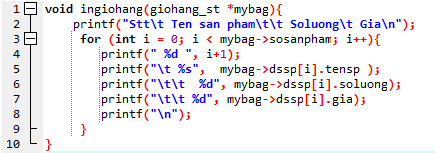
* Xây dựng hàm sắp xếp
* Sử dụng tham biến là con trỏ cấu trúc giohang\_st
* Hàm này được sử dụng để sắp xếp sản phẩm theo tên theo thứ tự tăng dần A->Z
* Chọn ra 2 sản phẩm rồi lần lượt so sánh với nhau
* Sử dụng:
* Hàm strcmp – hàm so sánh 2 chuỗi
* Hàm strcpy – hàm copy chuỗi
* Hàm strupr – đưa chuỗi về dạng uppercase



* Xây dựng hàm ingiohang

- Sử dụng tham biến là con trỏ cấu trúc giohang\_st

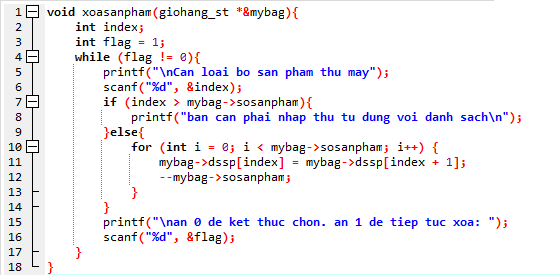
- Khi chọn 3 từ menu, sắp xếp các sản phẩm theo tên từ A->Z sau đó hiển thị thông tin chi tiết



* Xây dựng hàm xoasanpham
* Sử dụng tham biến là con trỏ cấu trúc giohang\_st tham chiếu đến biến mybag trong hàm main().

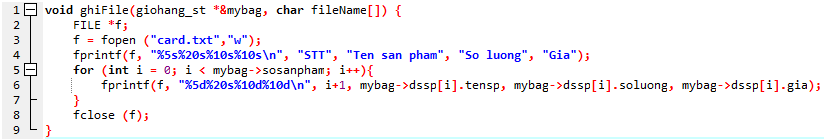
Khi chọn 4 từ menu, thực hiện các yêu cầu:

* Hiển thị “Can loai bo san pham thu may:”
* Tìm và loại bỏ sản phẩm tương ứng với thứ tự được chọn. Trong trường hợp người dùng nhập con số không đúng với số lượng sản phẩm đang có thì hiển thị thong báo “Ban phai nhap thu tu dung voi danh sach”



* Xây dựng hàm ghiFile
* Sử dụng tham biến là con trỏ cấu trúc giohang\_st
* Tạo file
* Lưu trữ tất cả thông tin của giỏ hàng vào trong file này.
* Đóng file

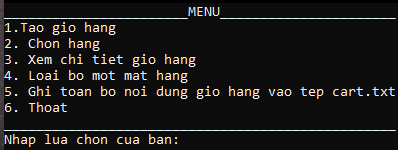
Khi chọn 5 từ menu, ghi tất cả thông tin của giỏ hàng vào file“cart.txt”



* Các thư viện cần khai báo

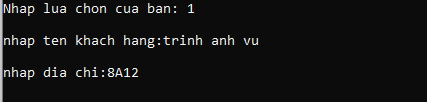


* + 1. **Chương trình**



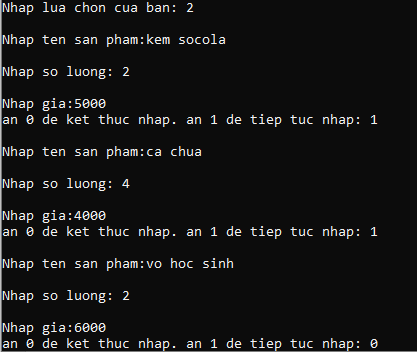
Hình 2.1. Trang chủ

* Tạo giỏ hàng



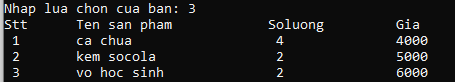
Hình 2.2. Chức năng tạo giỏ hàng

* Chọn hàng



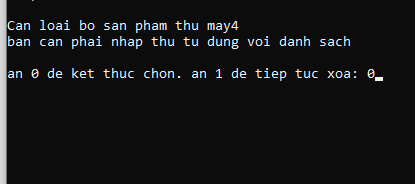
Hình 2.3. Chức năng chọn hàng

* Xem chi tiết giỏ hàng



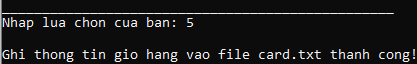
Hình 2.4. Chức năng xem giỏ hàng

* Loại bỏ một mặt hàng

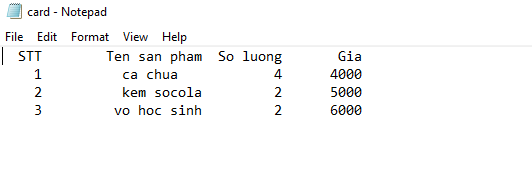


Hình 2.5. Chức năng loại bỏ một mặt hàng

* Ghi toàn bộ nội dung giỏ hàng vào tệp card.txt

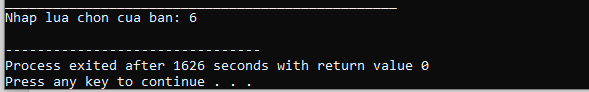


Hình 2.6. Chức năng ghi nội dung giỏ hàng vào file



Hình 2.7. Mở nội dung file vừa được ghi

* Thoát



Hình 2.8. Chức năng thoát

* + 1. **Kết luận**
* Kết quả đạt được
* Hiểu biết hơn về các cấu trúc, hàm trong ngôn ngữ lập trình C.
* Hiểu biết hơn về chương trình ứng dụng đơn giản trong thực tế.
* Nâng cao hơn 1 phần ý thức tự học, tự nghiên cứu.
* Hạn chế
* Chương trình còn một số chức năng chưa hoàn thiện và chính xác.
* Chưa có giao diện đồ họa.
* Kiến thức cơ bản chưa tốt, có nhiều vấn đề chưa giải thích được.
* **Hướng phát triển**
* Thêm 1 vài chức năng mới: tính tổng tiền, tổng số lượng sản phẩm, xuất hóa đơn, thanh toán qua mạng, ...
* Thiết kế giao diện đồ họa

# TÀI LIỆU THAM KHẢO

[1]. Đỗ Xuân Lôi, Cấu trúc dữ liệu và giải thuật, Nhà xuất bản Đại học Quốc gia Hà Nội, năm 2006.

[2]. GS. Phạm Văn Ất, Kỹ thuật lập trình C cơ sở và nâng cao, Nhà xuất bản Giao thông vận tải, năm 2006.

[3]. <https://nguyenvanhieu.vn/>.