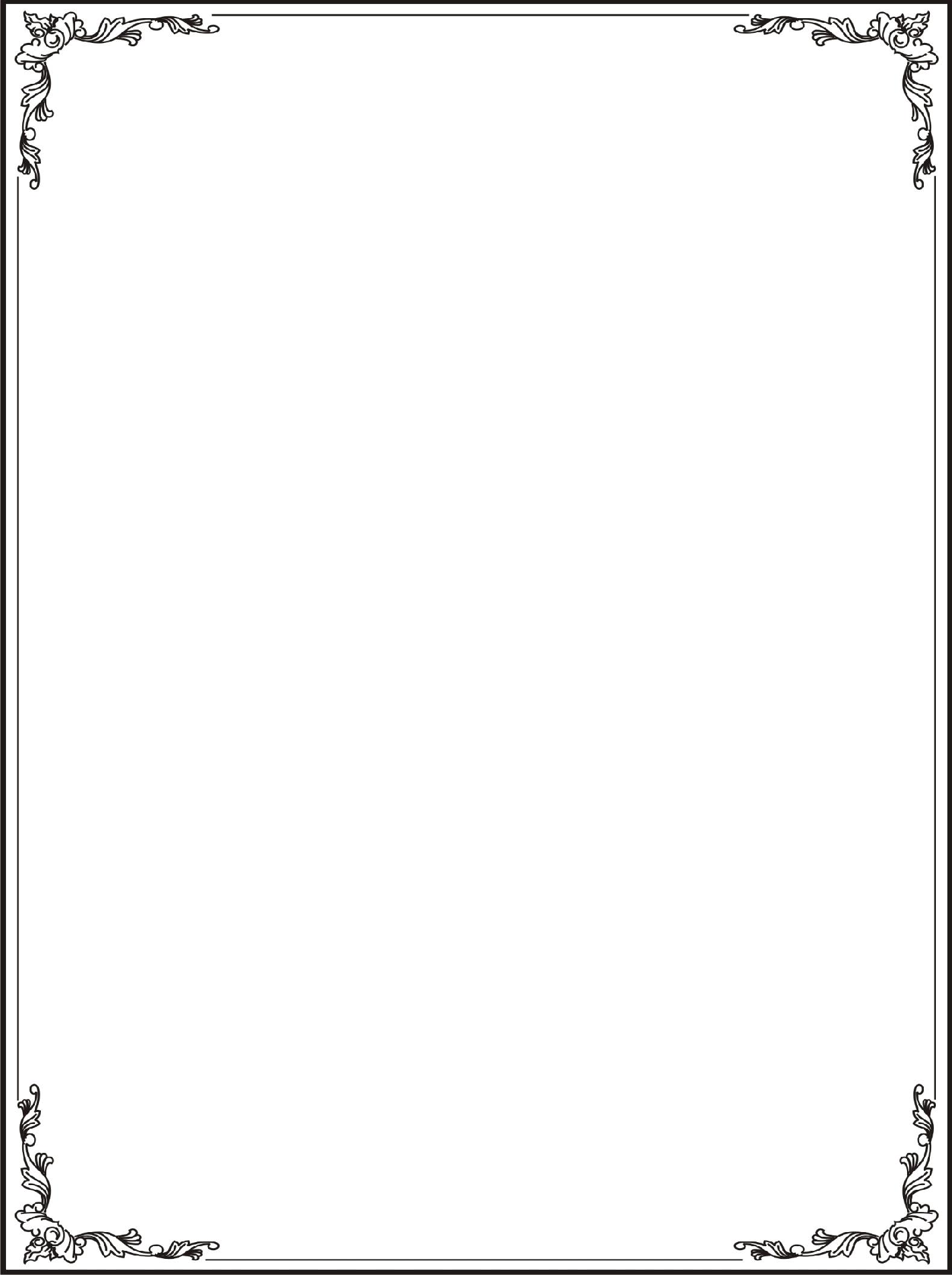
**BỘ GIÁO DỤC VÀ ĐÀO TẠO**



**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT THÀNH PHỐ HỒ CHÍ MINH**

**ĐỀ TÀI**

**MÔ PHỎNG CÁC THUẬT TOÁN SẮP XẾP**

**BẰNG GRAPHICS.H**

**GVHD: TS. Huỳnh Xuân Phụng**

**Mã học phần: DASA230179\_22\_1\_08**

|  |  |
| --- | --- |
| **Họ và tên** | **Mã số sinh viên** |
| **Ngô Quốc Đạt** | **21110417** |
| **Nguyễn Văn Tín** | **21110931** |
| **Nguyễn Đại Phúc** | **21110603** |

**TP. Hồ Chí Minh - Tháng 12/2022**

**DANH SÁCH THÀNH VIÊN**

Mã học phần: DASA230179\_22\_1\_08

Nhóm: 13

Tên đề tài: *Mô phỏng các thuật toán sắp xếp bằng graphics.h*

|  |  |  |  |
| --- | --- | --- | --- |
| STT | HỌ VÀ TÊN SINH VIÊN | MÃ SỐ SINH VIÊN | % HOÀN THÀNH |
| 1 | Ngô Quốc Đạt | 21110417 | 100% |
| 2 | Nguyễn Văn Tín | 21110931 | 100% |
| 3 | Nguyễn Đại Phúc | 21110603 | 100% |

**Ghi chú:**

Tỉ lệ %: Mức độ phần trăm hoàn thành của từng sinh viên tham gia.

Trưởng nhóm: Ngô Quốc Đạt

Nhận xét của giảng viên

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

….………………………………………………………………………………...

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

TP. Hồ Chí Minh - Tháng 12 năm 2022

MỤC LỤC

**[PHẦN 1: MỞ ĐẦU 1](#_Toc24028)**

**[1. Lý do chọn đề tài 1](#_Toc31563)**

**[2. Mục tiêu 1](#_Toc8095)**

**[3. Phương pháp nghiên cứu 2](#_Toc8880)**

**[PHẦN 2: NỘI DUNG 3](#_Toc22403)**

**[1. Thuật toán Comparison Counting Sort 3](#_Toc5222)**

**[2. Thuật toán Merge-exchange Sort 3](#_Toc19288)**

**[3. Thuật toán Straight Selection Sort 4](#_Toc7841)**

**[4. Thuật toán Quicksort 5](#_Toc14749)**

**[5. Thuật toán Shell Sort 7](#_Toc29331)**

**[PHẦN 3: KẾT LUẬN 9](#_Toc4257)**

PHẦN 1: MỞ ĐẦU

1. Lý do chọn đề tài

Trong lĩnh vực công nghệ thông tin nói riêng hay bất kỳ công việc xử lý thông tin sử dụng thiết bị công nghệ nào nói chung cũng đều yêu cầu một khối lượng kiến thức nhất định để có thể thực thi công việc. Tuy nhiên không phải khối kiến thức nào cũng dễ dàng tiếp cận đối với mọi cá nhân, sẽ có những vùng kiến thức yêu cầu kỹ năng có liên quan để có thể lĩnh hội, do đó việc tiến lên theo quá trình từ căn bản đến nâng cao là vô cùng cần thiết.

Là những người đang ở giai đoạn đầu chặng đường chinh phục tri thức, chúng tôi được tiếp cận những kiến thức cơ bản vô cùng quan trọng, là nền tảng của cả sự nghiệp chúng tôi sau này, Cấu trúc dữ liệu và Giải thuật là một trong số đó. Đây là khối kiến thức liên quan đến cách tổ chức giữa dữ liệu và các giải thuật trên từng tổ chức đó. Và 1 chương trình máy tính muốn chạy được phải có dữ liệu và các thuật toán, việc đảm bảo được dữ liệu tổ chức theo cấu trúc phù hợp và được thực thi với một thuật toán hợp lý sẽ tăng được hiệu suất lên rất nhiều. Trong đó, một trong các loại thuật toán quan trọng và cơ bản nhất đó là thuật toán sắp xếp. Qua bộ môn cấu trúc dữ liệu và giải thuật, chúng ta được học rất nhiều thuật toán liên quan tới sắp xếp một cách trực quan, và qua đề tài này mà chúng ta cũng hiểu rõ được thuật toán sắp xếp đó được chạy như thế nào, thông qua đó mà chúng ta có thể rút ra cho chính mình kinh nghiệm trong lập trình rằng lúc nào cần dùng thuật toán nào cho phù hợp với vấn đề cần giải quyết để đạt được hiệu suất cao nhất có thể.

1. Mục tiêu

Ngoài việc chúng ta biết thêm về các thuật toán sắp xếp, chúng ta có thể hiểu sâu hơn về cách mà thuật toán sắp xếp đó chạy thông qua các chuyển động dễ hiểu, nhằm mục đích nâng cao sự thú vị trong việc học, nhưng chủ yếu là sẽ giúp chúng ta hiểu được rằng thuật toán sẽ được chạy như thế nào để áp dụng vào từng bài toán cho phù hợp, đồng thời rút ra được ưu và nhược điểm của thuật toán sắp xếp đó.

1. Phương pháp nghiên cứu

- Làm việc theo nhóm.

- Đưa ra kịch bản, hình dung ra cách mà bài toán sẽ vận hành.

- Viết code sườn cho thuật toán để biết rằng thuật toán đó có chạy đúng hay không.

- Bổ sung thêm màu sắc, chuyển động cho thuật toán để người xem có thể hiểu thuật toán 1 cách trực quan hơn.

PHẦN 2: NỘI DUNG

1. Thuật toán Comparison Counting Sort

Thuật toán này sắp xếp mảng chính dựa trên 1 mảng phụ, mảng phụ này có chức năng lưu lại vị trí xuất hiện của các phần tử trong mảng gốc dựa theo giá trị của chúng so với nhau.

***Thuật toán:***

- **Bước 1**: Tạo 1 mảng có liểu dữ liệu là Record có n phần tử được đặt như sau ( Record R[n] ), mỗi phần tử Record chứa 2 kiểu dữ liệu (char data; int key).

- **Bước 2**: Tạo mảng phụ Count có kiểu dữ liệu int có n phần tử, đặt Count[0] = Count[1] = … = Count[n-1] = 0.

- **Bước 3**: Tạo 1 vòng lặp đối với biến int i. Vòng lặp được bắt đầu áp dụng từ bước 4 cho tới bước 5, biến i chạy từ n-1 cho tới i=1.

- **Bước 4**: Tạo 1 vòng lặp cho biến int j bên trong vòng lặp cho biến i. j đi từ i-1 cho tới j=0.

- **Bước 5**: So sánh R[i].key và R[j].key, nếu R[i].key < R[j].key thì Count[j]++. Ngược lại, nếu R[i].key R[j].key thì Count[i]++.

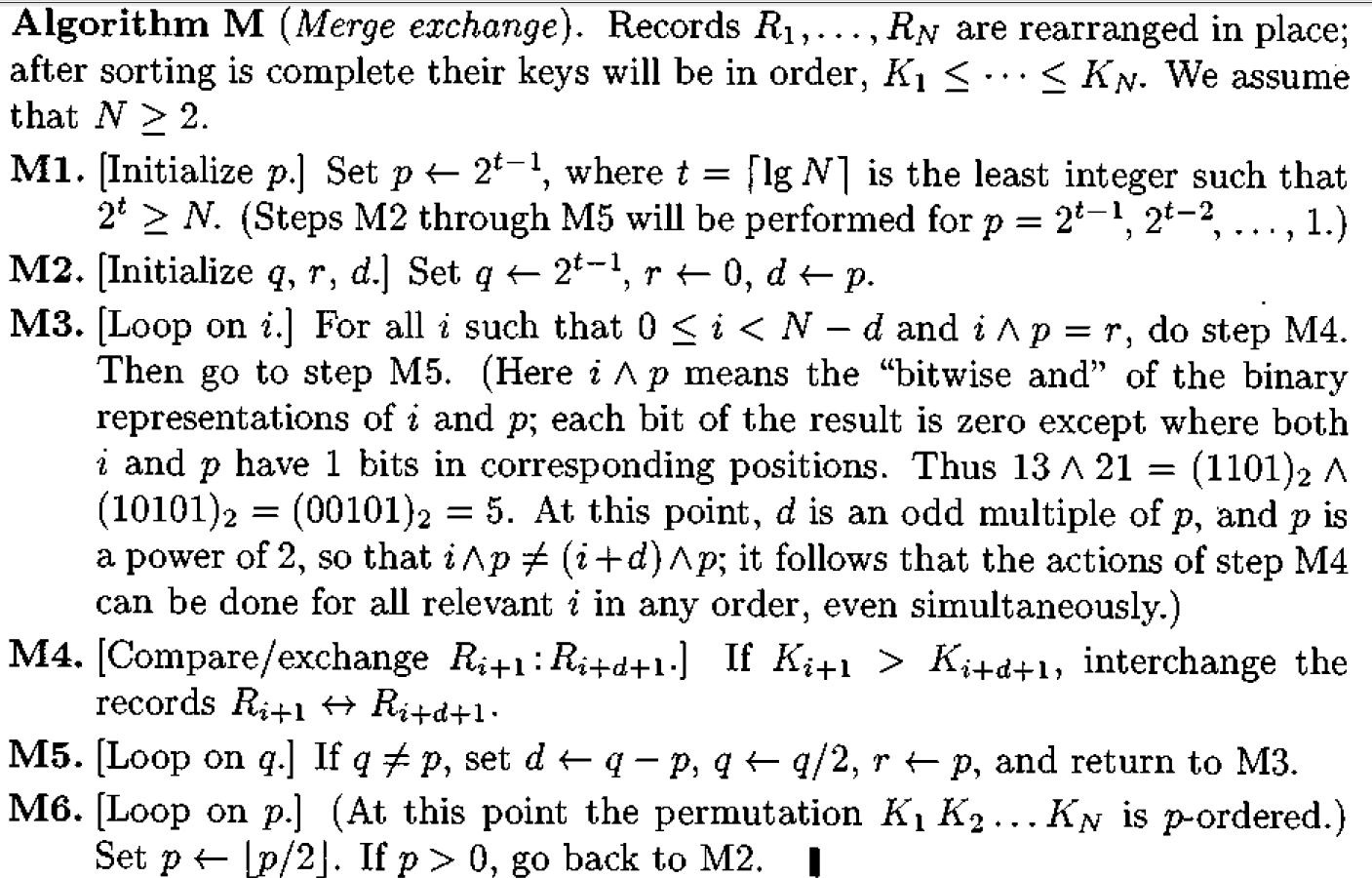
- **Bước 6**: Sau khi vòng lặp biến i ở trên kết thúc. Chúng ta đã thu được một mảng Count[n] có các giá trị đã được thay đổi. Cuối cùng, tạo 1 Record Output[n] có n phần tử. Tạo 1 vòng lặp cho k từ 0 đến n. Ta đặt Output[Count[k]] = R[k].

- **Bước 7**: Kết thúc thuật toán, ta thu được 1 mảng Output[n] đã được sắp xếp từ mảng R[n] ban đầu.

1. Thuật toán Merge-exchange Sort

Thuật toán này dựa trên lý thuyết phương pháp song song của Kenneth Edward Batcher.

***Thuật toán:***



1. Thuật toán Straight Selection Sort

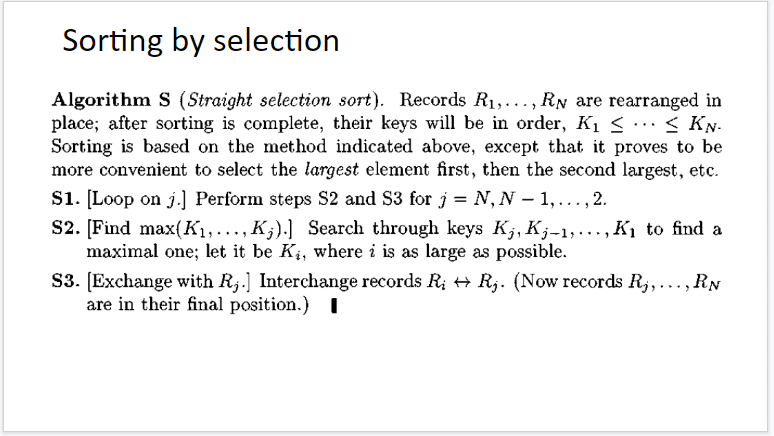
Thuật toán gồm N phần tử trong đó cần sắp xếp các Records kí hiệu R1, R2,..., RN trong đó chứa các key sao cho K1<=...<=KN.Ta sẽ sắp xếp dựa vào các key này

Thuật toán sẽ chọn phần tử lớn nhất trong N phần tử của danh sách ban đầu. Tìm và đổi vị trí **phần tử lớn nhất** với phần tử cuối cùng trong danh sách. Lúc này, **RN** sẽ chứa key có giá trị lớn nhất trong danh sách.

Sau đó, xem danh sách cần sắp xếp hiện tại chỉ gồm N-1 phần tử, bắt đầu từ phần tử thứ N-1 trong danh sách ban đầu. Tức là danh sách hiện tại chỉ gồm **R1, R2,…, RN-1**

Lặp lại quá trình trên cho danh sách hiện tại đến khi danh sách hiện tại chỉ còn 1 phần tử.

Các bước của thuật toán:



Các bước thực hiện thuật toán:

B1: Cho chạy vòng for j từ N, N-1, ...2

B2: ứng với mỗi j thì ta sẽ cho chạy vòng for i từ j tới 1

Mục đích là để tìm Ki là lớn nhất so với con Kj

B3: Nếu như ta tìm được thì đổi Ri và Rj

1. Thuật toán Quicksort

Quick sort là thuật toán sắp xếp, hoạt động theo cách sau: Chọn một phần tử trong mảng làm điểm đánh dấu và sau đó chia mảng thành hai mảng con bằng cách so sánh các phần tử trong mảng với điểm đánh dấu. Mảng 1 sẽ chứ các phần tử nhỏ hơn hoặc bằng điểm đánh dấu và mảng 2 sẽ gồm các phần tử lớn hơn hoặc bằng điểm đánh dấu.

Tốc độ sắp xếp của thuật toán tùy thuộc vào việc lựa chọn điểm đánh dấu, tùy từng trường hợp sẽ có một số cách chọn như sau:

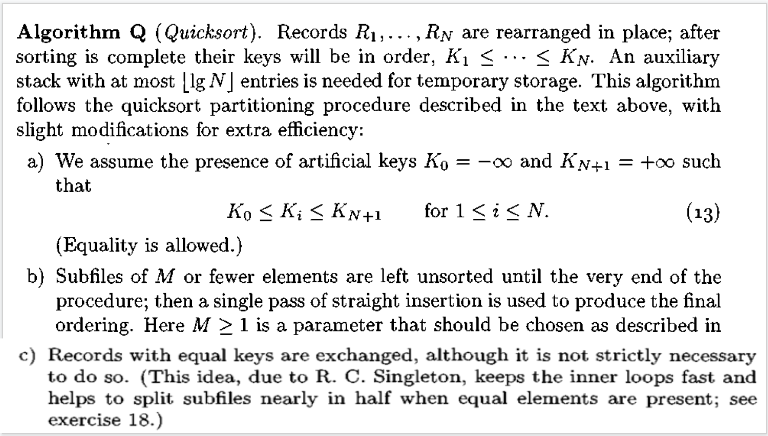
+Chọn phần tử đầu tiên của mảng.

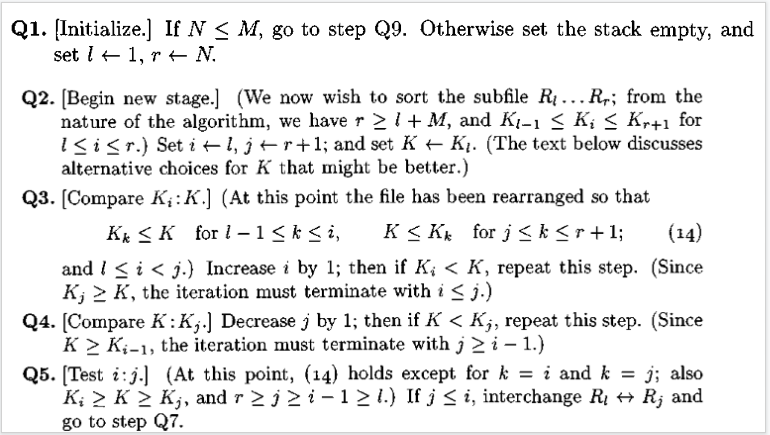
+Chọn phần tử cuối cùng của mảng.

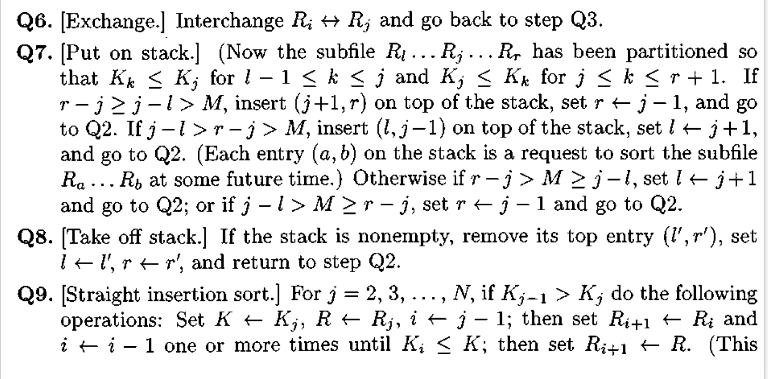
+Chọn phần tử có giá trị nằm giữa mảng.

+Chọn Random một phần tử của mảng.

Các bước của thuật toán:







Các bước của thuật toán:

B1: Khởi tạo l,r trong đó l là vị trí bắt đầu của mảng đang xét còn r là vị trí cuối cùng của mảng đang xét, khởi tạo stack sau đó thêm (l,r) vào stack

B2: Lấy từ stack ra khởi tạo i = l, j = r+1, K=Kl và xoá phần từ vừa lấy từ stack ra

B3: tăng i lên để kiếm phần tử lớn hơn K

B4: Giảm j xuống để kiếm phần tử nhỏ hơn K

B5: Nếu j<=i thì đổi Rl và Rj

B6 Không thì đổi Ri và Rj quay lại bước 3 tiếp tục tăng i

B7: Nếu r>j thì thêm vào stack (j+1,r).Nếu j>l thì thêm vào stack (l,j-1)

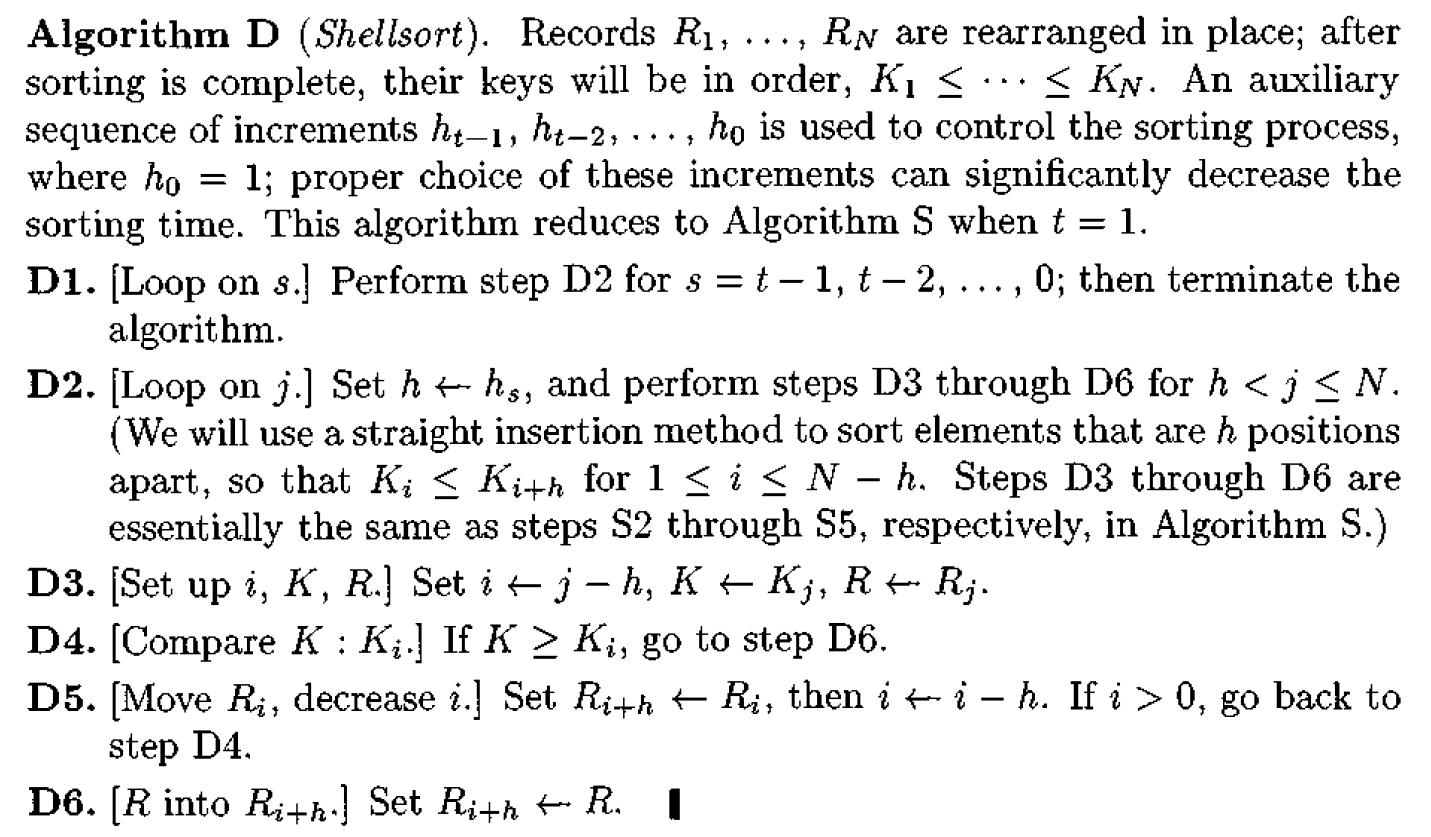
B8: Nếu stack không rỗng thì lấy stack ra và set lại giá trị l,r và quay lại B2

B9: In mảng ra

1. Thuật toán Shell Sort

Shell Sort là một thuật toán sắp xếp các phần tử cách xa nhau ban đầu và liên tiếp giảm khoảng cách giữa các phần tử được sắp xếp. Nó là một phiên bản hiệu quả cao hơn của sắp xếp chèn. Các phần tử tại một khoảng cụ thể sẽ được sắp xếp. Khoảng cách giữa các phần tử được giảm dần dựa trên trình tự được sử dụng. Hiệu suất của kiểu sắp xếp Shell phụ thuộc vào kiểu trình tự được sử dụng cho một mảng đầu vào nhất định.

Thuật toán:



PHẦN 3: KẾT LUẬN

Qua bộ môn Cấu trúc dữ liệu và giải thuật nói chung, đồng thời qua việc làm đồ án về các thuật toán sắp xếp sử dụng header file graphics.h nói riêng. Nhóm chúng em cũng học được cách làm việc theo nhóm sao cho hiệu quả, thêm vào đó là học được các thuật toán mới và cũng học được các hướng giải quyết khác cho các thuật toán đó, giúp mở mang thêm kiến thức. Ngoài ra, nhóm còn học được cách tự học, và chọn lọc kiến thức thông qua việc tìm kiếm các thông tin mà mình cần để bắt tay vào thực hiện.