BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC NHA TRANG

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO THỰC TẬP CƠ SỞ

MÔ PHỎNG QUÁ TRÌNH SẮP XẾP MẢNG BẰNG GIẢI THUẬT SELECTION SORT

Giảng viên hướng dẫn: ThS. Đoàn Vũ Thịnh

Sinh viên thực hiện: Nguyễn Đức Thịnh

Mã số sinh viên: 62134265

KHÁNH HÒA-2022

TRƯỜNG ĐẠI HỌC NHA TRANG

KHOA CÔNG NGHỆ THÔNG TIN

BỘ MÔN HỆ THỐNG THÔNG TIN

BÁO CÁO THỰC TẬP CƠ SỞ

MÔ PHỎNG QUÁ TRÌNH SẮP XẾP MẢNG BẰNG GIẢI THUẬT SELECTION SORT

Giảng viên hướng dẫn: ThS. Đoàn Vũ Thịnh

Sinh viên thực hiện: Nguyễn Đức Thịnh

Mã số sinh viên: 62134265

Khánh Hòa, tháng 12/2022

TRƯỜNG ĐẠI HỌC NHA TRANG
Khoa: Công nghệ Thông tin

PHIẾU THEO DÕI TIẾN ĐỘ VÀ ĐÁNH GIÁ BÁO CÁO THỰC TẬP CƠ SỞ

Tên đề tài: MÔ PHỎNG QUÁ TRÌNH SẮP XẾP MẢNG BẰNG GIẢI THUẬT SELECTION SORT

Giảng viên hướng dẫn: ThS. Đoàn Vũ Thịnh

Sinh viên được hướng dẫn: Nguyễn Đức Thịnh

MSSV: 62134265

Khóa: 62 **Ngành:** Công nghệ Thông tin

Lần	Ngày	Nội dung	Nhận xét của GVHD
1	11/12/2022	Thông báo cho GVHD về đề tài đã chọn. Sinh viên trình bày sơ bộ chương trình demo.	Về mặt thuật toán thì sinh viên cần xác định rõ số lượng phần tử cụ thể để sắp xếp, mặt giao diện thì nên thêm vào màu nền đằng sau, chỉnh lại sao cho gọn gàng
2	18/12/2022	Sinh viên trình bày việc mô phỏng thuật toán chính dựa trên kiến thức đã được học ở môn kỹ thuật đồ họa và các kiến thức thu nhận được từ Internet để mô phỏng thuật toán.	Chương trình demo có cải thiện hơn so với lần 1, đã xác định rõ số lượng phần tử và mô phỏng được thuật toán, tuy nhiên tốc độ sắp xếp còn chậm, có thể thêm một số nút chức năng phía dưới giao diện như nút tăng giảm tốc độ hoặc tạm dừng và cũng như có thể sáng tạo thêm bằng cách thêm hiệu ứng khi các con số đang chạy hay thêm âm thanh vào. Sinh viên cần thiết kế thêm giao diện trang bìa, menu và giao diện sắp xếp từ những giá trị được người dùng nhập từ bàn phím.
3	28/12/2022	Sinh viên hoàn thiện mô phỏng thuật toán đã đề ra với dữ liệu đầu vào được nhập từ bàn phím.	Sinh viên đã cải thiện chương trình demo theo một vài góp ý của GVHD. Việc lập trình cũng hoàn thành ở mức độ nhập dữ liệu từ bàn phím.

4	4/01/2023	Sinh viên nộp bản thảo của báo cáo thực tập.	Sinh viên nghiêm túc thiết kế báo cáo theo định hướng của GVHD.

Nhận xét chung (sau khi sinh viên hoàn thành ĐA/KL):

Điểm hình thức: /10 Điểm nội dung: /10 **Điểm tổng kết: /10**

Đồng ý cho sinh viên: Được bảo vệ: ☑ Không được bảo vệ: ☐

Khánh Hòa, ngày 29 tháng 12 năm 2022

Cán bộ hướng dẫn

(Ký và ghi rõ họ tên)

LỜI CẢM ƠN

Để có thể hoàn thành đợt thực tập lần này, em xin chân thành cảm ơn đến quý thầy cô khoa Công nghệ Thông tin đã tạo điều kiện hỗ trợ và giúp đỡ em trong quá trình học tập và nghiên cứu đề tài này.

Qua đây, em xin chân thành cảm ơn thầy Đoàn Vũ Thịnh, người đã trực tiếp quan tâm và hướng dẫn em hoàn thành tốt đợt thực tập trong thời gian qua.

Do kiến thức còn hạn chế và thời gian thực hiện còn ngắn nên bài báo cáo của em còn nhiều thiếu sót, kính mong sự góp ý của quý thầy cô.

MỤC LỤC

TÓM TẮT

Trong khoa học máy tính, thuật toán sắp xếp là một thuật toán đặt các phần tử của danh sách thành một thứ tự. Những loại thứ tự được sử dụng thường xuyên nhất là thứ tự số và thứ tự từ điển, và tăng dần hoặc giảm dần. Sắp xếp hiệu quả rất quan trọng để tối ưu hóa hiệu quả của các thuật toán khác (như thuật toán tìm kiếm và hợp nhất) yêu cầu dữ liệu đầu vào phải nằm trong danh sách được sắp xếp. Sắp xếp cũng thường hữu ích cho việc kinh điển hóa dữ liệu và để tạo ra đầu ra có thể đọc được của con người. Quy trình thực hiện được trải qua các bước từ cài đặt thuật toán, hiển thị kết quả đầu ra trên màn hình đều được thực hiện trên môi trường C/C++ thông qua phần mềm Visual Studio có kết hợp với thư viện graphics.h.

Sản phẩm đã minh họa được từng bước sắp xếp mảng sử dụng giải thuật sắp xếp Selection Sort. Đồng thời cũng chỉ ra các trường hợp hạn chế của mỗi thuật toán và cách khắc phục các nhược điểm đó. Sản phẩm chạy tốt với dữ liệu được nhập từ bàn phím. Về yêu cầu với các thao tác từ chuột chưa được triển khai. Đây cũng là thiếu sót của đề tài ngay từ khi đặt ra.

Toàn bộ mã nguồn của báo cáo được tải lên theo địa chỉ:

https://github.com/thinhdoanvu/DOAN_TTCS_TTCN/tree/main/NguyenDucThinh_62134265

CHƯƠNG 1. GIỚI THIỆU

Trong khoa học máy tính và trong toán học, thuật toán sắp xếp là một thuật toán mà sắp xếp các phần tử của một danh sách (hoặc một mảng) theo thứ tự (tăng hoặc giảm). Người ta thường xét trường hợp các phần tử cần sắp xếp là các số. Bài toán sắp xếp đã được nhiều nhà khoa học quan tâm. Từ thuật toán sắp xếp chúng ta có nhiều ứng dụng trong thực tế như: Danh sách lớp với các học sinh được sắp xếp theo thứ tự bảng chữ cái, danh bạ điện thoại, danh sách các truy vấn được tìm kiếm nhiều nhất trên Google…

Sắp xếp xuất hiện trong bất kỳ lĩnh vực nào của tin học, từ những ứng dụng ẩn bên trong của hệ điều hành như bài toán điều khiển quá trình, bài toán lập lịch cho CPU, bài toán quản lý bộ nhớ… cho đến những ứng dụng thông thường như sắp xếp dãy các từ, các câu, các bản ghi theo thứ tự nào đó.

1.1. Thuật toán sắp xếp

Sắp xếp là quá trình xử lý một danh sách các phần tử (hoặc các phần tử trong mảng) để đặt chúng theo một thứ tự, thỏa mãn một tiêu chuẩn nào đó dựa trên nội dung thông tin lưu giữ tại mỗi phần tử.

Cho trước một dãy số a_1, a_2,…, a_N được lưu trữ trong cấu trúc dữ liệu mảng. Sắp xếp dãy số a_1, a_2,…, a_N là thực hiện việc bố trí lại các phần tử sao cho được dãy mới a_{k1}, a_{k2},…, a_{kN} được hình thành có thứ tự (giả sử xét thứ tự tăng) nghĩa là $a_{ki} > a_{ki-1}$... Hai thao tác so sánh và gán là các thao tác cơ bản của hầu hết các thuật toán sắp xếp.

Khi xây dựng một thuật toán sắp xếp, cần tìm cách giảm thiểu những phép so sánh và đổi chỗ không cần thiết để tăng hiệu quả của thuật toán. Đối với các dãy số được lưu trữ trong bộ nhớ chính, nhu cầu tiết kiệm bộ nhớ được đặt nặng, do vậy những thuật toán sắp xếp đòi hỏi cấp phát thêm vùng nhớ để lưu trữ dãy kết quả ngoài vùng nhớ lưu trữ dãy số ban đầu thường ít được quan tâm. Thay vào đó, các thuật toán sắp xếp trực tiếp trên dãy số ban đầu – gọi là các thuật toán sắp xếp tại chỗ – lại được đầu tư phát triển.

1.2 Một số phương pháp sắp xếp thông dụng

1.2.1 Bubble Sort

Đi từ cuối mảng về đầu mảng, trong quá trình đi, nếu phần tử ở dưới (đứng phía sau) nhỏ hơn phần tử đứng ngay trên (trước) nó thì theo nguyên tắc của bọt khí, phần tử nhẹ hơn sẽ bị "trồi" lên trên phần tử nặng (hai phần tử này sẽ được đổi chỗ cho nhau). Kết quả là phần tử nhỏ nhất (nhẹ nhất) sẽ được đưa về đầu dãy rất nhanh. Sau đó sẽ không xét đến nó ở bước tiếp theo, do vậy ở lần xử lý thứ i sẽ có vị trí đầu dãy là i. Lặp lại xử lý trên cho đến khi không còn cặp phần tử nào để xét và ta có dãy sắp xếp theo một thứ tự.

1.2.2 Quick Sort

Tính từ khóa chốt, các phần tử nhỏ hơn khóa chốt phải được xếp trước chốt; mọi phần tử lớn hơn khóa chốt được xếp vào sau chốt. Để làm được điều đó, các phần tử trong danh sách sẽ được so sánh với khóa chốt và tráo vị trí cho nhau hoặc cho khóa chốt nếu phần tử đó lớn hơn chốt mà lại nằm trước chốt hoặc nhỏ hơn chốt nhưng lại nằm sau chốt. Khi việc đổi chỗ lần đầu tiên đã được thực hiện xong thì danh sách tạo thành hai đoạn: một đoạn gồm các phần tử nhỏ hơn chốt, một đoạn gồm các phần tử lớn

hơn chốt còn khóa chốt chính là vị trí của phần tử trong danh sách được sắp xếp. Áp dụng kỹ thuật như trên cho mỗi đoạn trước chốt và sau chốt cho tới khi các đoạn còn lại hai phần tử thì dừng lại. Khi đó, danh sách đã được sắp xếp.

Cụ thể giả sử để sắp xếp dãy số a gồm n phần tử: a_1, a_2,…, a_n; giải thuật Quick Sort dựa trên việc phân hoạch dãy ban đầu được chia thành hai phần:

- Dãy con 1 gồm các phần tử: a_1,…, a_{i-1} có giá trị nhỏ hoặc bằng a_i;
- Dãy con 2 gồm các phần tử: a_{i+1},…, a_n có giá trị lớn hơn a_i;

  Với ai là một phần tử bất kỳ trong dãy.

Để sắp xếp dãy con 1 và 2, lần lượt tiến hành việc phân hoạch từng dãy con theo cùng phương pháp phân hoạch dãy ban đầu.

1.3. Thuật toán sắp xếp Selection Sort

1.3.1. Ý tưởng chính của thuật toán

Ban đầu, dãy gồm n phần tử không có thứ tự. Ta chọn phần tử nhỏ nhất trong n phần tử của dãy cần sắp xếp và đổi giá trị của nó với a[1], khi đó a[1] có giá trị nhỏ nhất trong dãy. Lần thứ 2, chọn phần tử nhỏ nhất trong n – 1 phần tử còn lại a[2], a[3], ..., a[n] và đổi giá trị của nó với a[2]. Ở lượt thứ i ta chọn trong dãy a[i ... n] ra phần tử nhỏ nhất và đổi chỗ giá trị của nó với a[i]. Tới lượt thứ n – 1. Chọn trong 2 phần tử a[n – 1] và a[n] phần tử có giá trị nhỏ hơn và đổi giá trị của nó với a[n – 1]. Khi đó, dãy thu được có thứ tự không giảm.

Ta có thể tóm tắt các bước chính của thuật toán như sau:

Bước 1: Cho i = 0

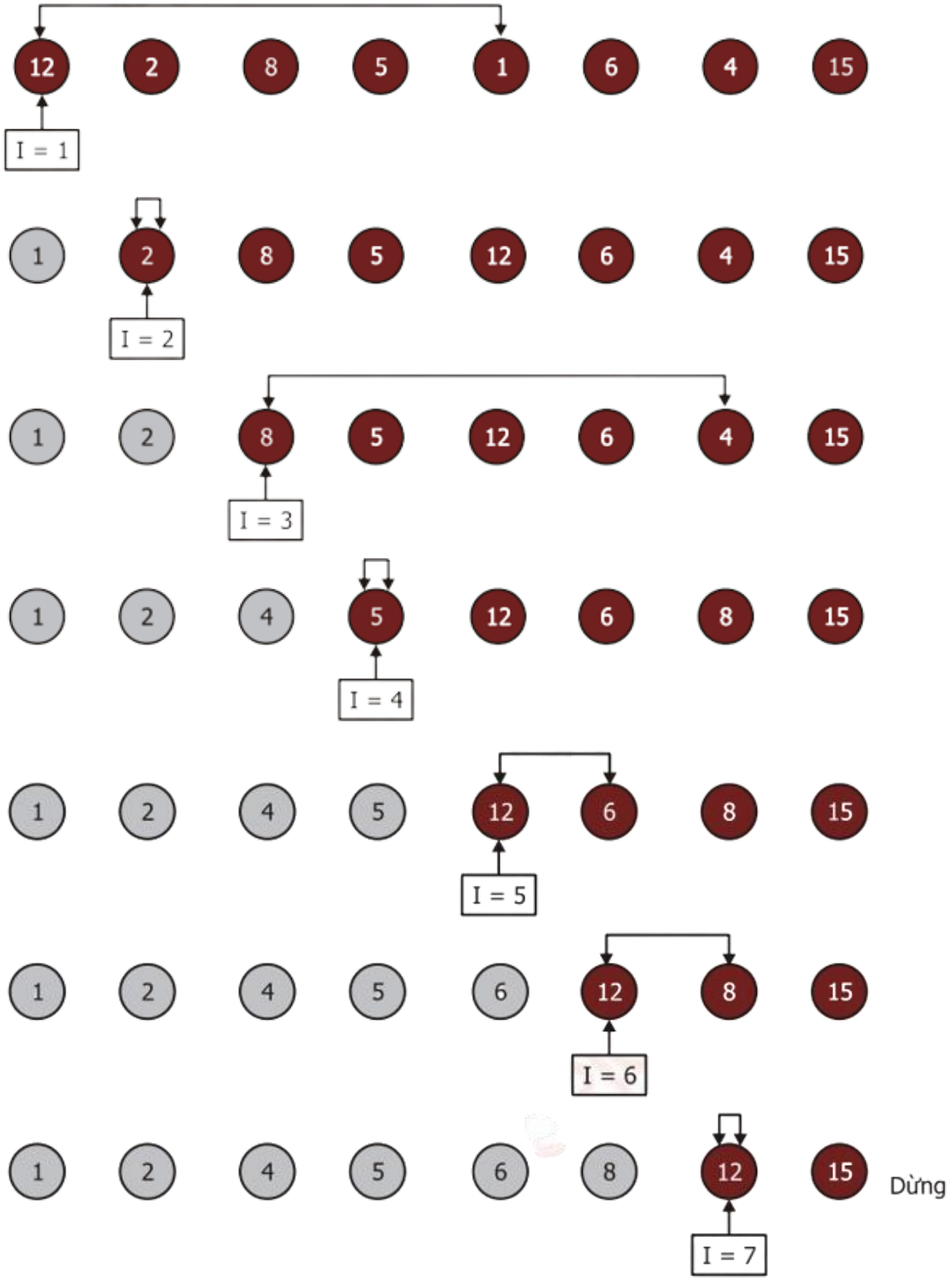
Bước 2: Tìm phần tử a[min] nhỏ nhất trong dãy từ a[i] tới a[n – 1]

Bước 3: Hoán vị a[min] và a[i]

Bước 4: Nếu i <= n – 2 thì i = i + 1, lặp lại bước 2, ngược lại thì dừng

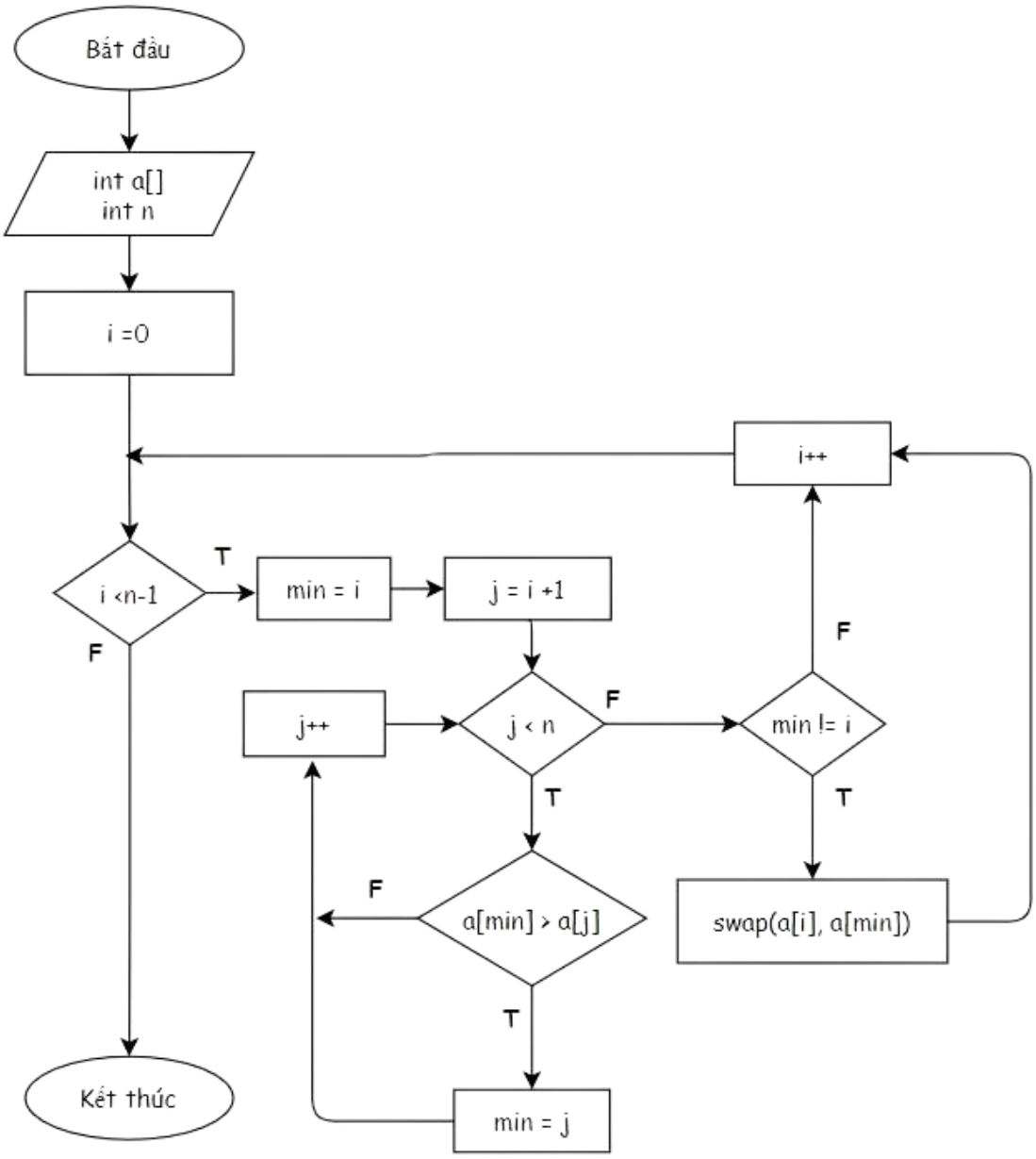
Ví dụ minh hoạ:

Cho dãy số a: 12, 2, 8, 5, 1, 6, 4, 15. Sắp xếp dãy số này theo thứ tự tăng dần bằng thuật toán Selection Sort được minh họa như sau:



Hình 1.1 Minh hoạ từng bước thuật toán Selection Sort

1.3.2 Lưu đồ thuật toán



Hình 1.2. Lưu đồ thuật toán Selection Sort

1.3.3 Đánh giá thuật toán

1.3.3.1 Độ phức tạp

Ta nhận thấy ở lượt thứ i, để chọn ra phần tử nhỏ nhất bao giờ cũng cần n – i phép so sánh. Số lượng phép so sánh trong thuật toán này không phụ thuộc vào tình trạng ban đầu của mảng, do đó, tổng số phép toán so sánh là:

$S_{so\ sánh} = (n – 1) + (n – 2) + ... + 2 + 1 = n(n – 1) / 2$

- Trong trường hợp tốt nhất, khi dãy đã có thứ tự tăng dần: Số phép gán $S_{gán} = 0$
- Trong trường hợp tồi nhất khi dãy có thứ tự giảm dần:

Số phép gán $S_{gán} = 3 * (n – 1) + n * (n – 1) / 2 = (n – 1)(n + 6) / 2$

Như vậy, độ phức tạp của thuật toán sắp xếp lựa chọn này là $O(n^2)$.

1.3.3.2 Ưu điểm và nhược điểm

Ưu điểm: Thực hiện rất nhanh đối với mảng có số lượng phần tử ít. Và vì là thuật toán sắp xếp tại chỗ, chỉ cần hoán đổi vị trí cho nhau cho nên tiết kiệm được phần nào bộ nhớ cần thiết để đáp ứng. Ngoài ra, thứ tự ban đầu của các phần tử trước khi sắp xếp cũng tác động đáng kể đến độ hiệu quả của thuật toán khi thực hiện sắp xếp.

Nhược điểm: Thực hiện chậm đối với mảng có số lượng phần tử lớn, luôn luôn chạy với n bước lặp dẫn đến việc không thể tự dừng sắp xếp đối với cả mảng đã hoàn toàn được sắp xếp.

1.3.3.3 Tốc độ thực hiện

Khi thực hiện sắp xếp trên mảng lớn (gồm 100000 phần tử ngẫu nhiên không theo trật tự), thuật toán Selection Sort tỏ ra vượt trội hơn so với Bubble Sort khi thời gian sắp xếp của Selection Sort là 20.425 giây, còn của Bubble Sort là 58.540 giây. Tuy nhiên khi so với với Quick Sort chỉ với 0.314 giây, ta mới thấy được sự phát triển của các thuật toán sắp xếp theo thời gian. Điều này cho ta hiểu vì sao hiện nay Quick Sort luôn được ưa thích hơn các thuật toán khác về thời gian sắp xếp.

1.4. Visual Studio

Microsoft Visual Studio (https://visualstudio.microsoft.com/) là môi trường phát triển tích hợp (IDE) từ Microsoft. Đây là công cụ lập trình được sử dụng phổ biến trong việc phát triển website, các phần mềm của Microsoft như Windows API, Windows Forms,...Visual Studio hỗ trợ lập trình trên nhiều ngôn ngữ giúp lập trình viên có thể lập trình được hệ thống một cách thuận tiện và nhanh chóng nhất.

4 tính năng chính của Visual Studio gồm:

- ✓ Hỗ trợ đa ngôn ngữ (HTML, CSS, C#, F#, C/C++, JSON, JavaScript...)
- ✓ Hỗ trợ việc Debug một cách dễ dàng, mạnh mẽ nhờ vào Breakpoint
- ✓ Hỗ trợ kết nối với GitHub để để đồng bộ các dự án về máy, tích hợp đầy đủ các chức năng làm việc, quản lý dự án với GitHub ngay trên giao diện của Visual Studio.
- ✓ Kho tiện ích mở rộng phong phú để người dùng có thể tùy biến thoải mái theo nhu cầu và mục đích sử dụng của mình.

1.5. Thư viện Graphics.h

Vì sử dụng Visual Studio làm trình biên dịch cho việc cài đặt thuật toán nên không thể thực hiện trên môi trường Windows. Vì vậy, một môi trường giả lập graphic của Borland C được một nhóm người gồm Grant Macklem, Gregory Schmelter, Alan Schmidt, Ivan Stashak, Michael Main cùng nhau tạo ra thư viện có tên là Graphics.h. Để có thể làm được điều đó, họ đã cùng nhau thay đổi BGI library (thư viện BGI) thành thư viện có tên graphics.h để có thể sử dụng tốt trên Windows. Và bây giờ chúng ta đã có thể sử dụng tốt các hàm đặc biệt của Borland trên Visual Studio ở đường dẫn dưới (https://github.com/thinh26/Borland-Graphics-Interface-for-Visual-Studio)

CHƯƠNG 2. PHƯƠNG PHÁP NGHIÊN CỨU

2.1. Cài đặt Visual Studio và thư viện graphics.h

Tải file cài đặt phần mềm Visual Studio theo đường dẫn trong mục 1.4. Sau đó mở file vừa tải, và tiến hành cài đặt. Thư viện graphics.h được tiến hành cài đặt theo các bước:

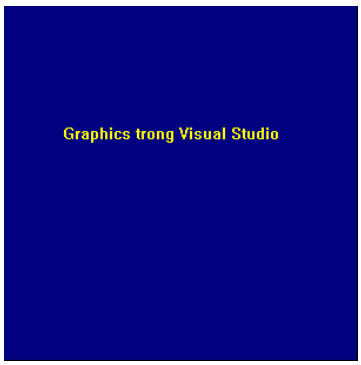
Bước 1: Tạo project mới => Chọn Empty Project => Chọn nơi lưu trữ project => Tích vào « Place solution and project in the same directory » => Chọn Create

Bước 2: Sao chép tệp graphics.h và dán vào nơi lưu project đã chọn ở bước 1

Bước 3: Sao chép tệp graphics.lib và dán vào nơi lưu project đã chọn ở bước 1

Bước 4: Sử dụng đoạn code mẫu bên dưới để test thư viện graphics.h

```
#include <stdio.h>
#include "graphics.h"
#pragma comment(lib, "graphics.lib")

int main()
{
    initwindow(300, 300); //Thiết lập chiều dài và chiều cao của màn hình đồ hoạ
    setbkcolor(BLUE);     //Chọn màu nền
    cleardevice();
    setcolor(YELLOW);   //Chọn màu chữ
    outtextxy(50, 100, "Graphics trong Visual Studio");
    getch();
    closegraph();
}
```



Hình 2.1. Ví dụ minh hoạ thư viện graphics.h

2.2. Cài đặt thuật toán Selection Sort

Khai báo thư viện

```
#include <iostream>
#include <cstring>
#include <conio.h>
#include <Windows.h>
using namespace std;
#include "graphics.h"
#pragma comment(lib, "graphics.lib")
```

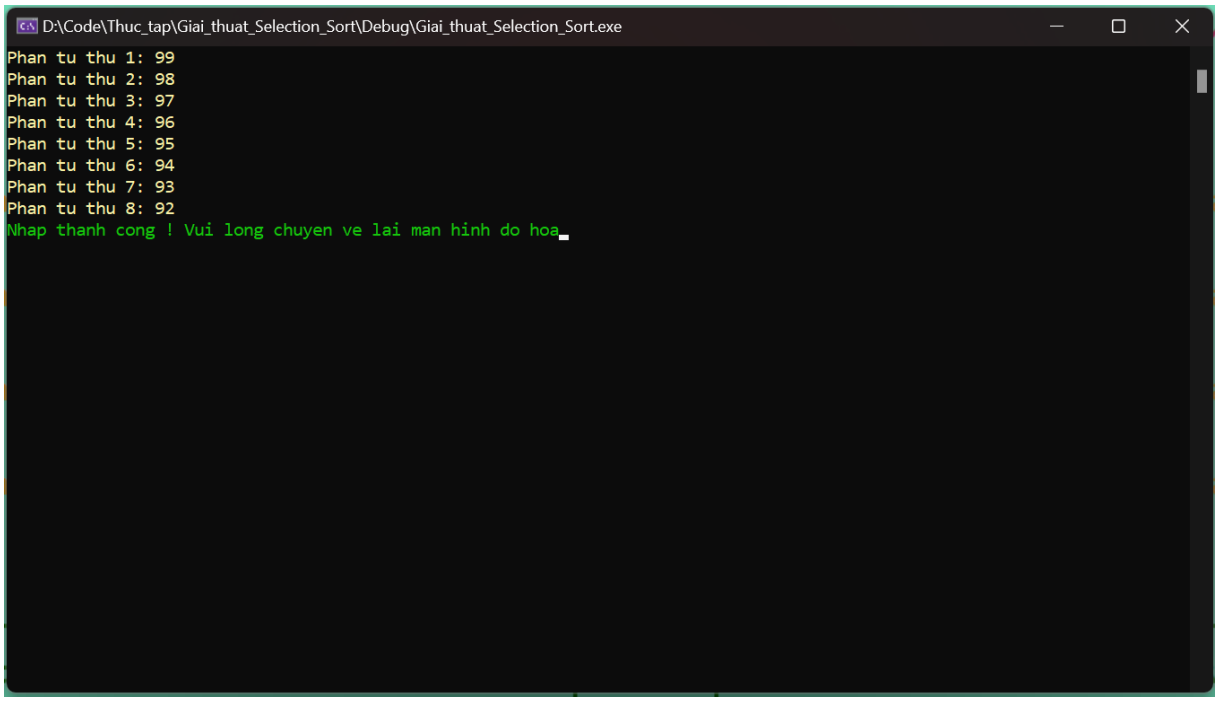
Khai báo hàm và biến

```
void moPhongTuNhap(); void MauTuNhap(); void swap(int* a, int* b);
void Trao_Doi_2_So(int xCircle01, int yCircle01,
                   int xCircle02, int yCircle02,
                   int xText01, int yText01,
                   const char* Text01,
                   int xText02, int yText02,
                   const char* Text02);

/* Màu chữ cửa sổ dòng lệnh */
#define BOLD_YELLOW_CMD_TEXT "\033[1m\033[33m"
#define BOLD_GREEN_CMD_TEXT "\033[1m\033[32m"

/* Biến toàn cục dành cho phần dữ liệu tự nhập */
int int_arr[8];
char const* array_element[8];
```

Nhập dữ liệu từ bàn phím vào mảng: Chúng ta sẽ cố định thuật toán Selection Sort sắp xếp 8 phần tử. Mỗi phần tử được lưu trữ trong mảng một chiều. Mỗi phần tử là kiểu số nguyên

```
/* Nhập các phần tử */
for (int i = 0;i < 8;i++)
{
    cout << BOLD_YELLOW_CMD_TEXT << "Phan tu thu " << i + 1 << ": ";
    cin >> int_arr[i];
}
cout << BOLD_GREEN_CMD_TEXT << "Nhap thanh cong ! Vui long chuyen ve lai man hinh do hoa";
```

Hình 2.2. Code nhập dữ liệu từ bàn phím



Hình 2.3. Màn hình nhập dữ liệu từ bàn phím

Kết thúc nhập dữ liệu sẽ chuyển đổi kiểu số nguyên thành kiểu ký tự

```
//Chuyển kiểu int sang kiểu char
string t[8];
for (int i = 0;i < 8;i++)
{
    t[i] = to_string(int_arr[i]); //Từ int sang string
    array_element[i] = t[i].c_str(); //Từ string sang char
}
```

Hình 2.4. Code chuyển đổi

và hiển thị lên màn hình đồ hoạ

```
//In các phần tử ra màn hình đồ hoạ
settextstyle(BOLD_FONT, HORIZ_DIR, 4);
int Bronze = COLOR(205, 127, 50);
setcolor(Bronze);
outtextxy(460, 80, "DANH SACH PHAN TU");
int Bright_Orange = COLOR(255, 172, 28);
setcolor(Bright_Orange);
int xText[8] = { 100,100,100,100,800,800,800,800 },
    yText[8] = { 150,220,290,360,150,220,290,360 };
char const* chuThich[8] = { "Phan tu thu nhat:",
                            "Phan tu thu hai:",
                            "Phan tu thu ba:",
                            "Phan tu thu tu:",
                            "Phan tu thu nam:",
                            "Phan tu thu sau:",
                            "Phan tu thu bay:",
                            "Phan tu thu tam:" };
for (int i = 0;i < 8;i++)
    outtextxy(xText[i], yText[i], chuThich[i]);
```

```
int xElement[8] = { 430,410,390,390,1110,1110,1110,1110 },
    yElement[8] = { 150,220,290,360,150,220,290,360 };
for (int i = 0;i < 8;i++)
    outtextxy(xElement[i], yElement[i], array_element[i]);

delay(1000);
/* Phím chức năng */
setcolor(GREEN);
setlinestyle(SOLID_LINE, 0, THICK_WIDTH);
settextstyle(BOLD_FONT, HORIZ_DIR, 4);
rectangle(80, 480, 575, 550);
outtextxy(100, 500, "Enter de bat dau sap xep");
rectangle(680, 480, 1195, 550);
outtextxy(700, 500, "ESC de huy va tro ve menu");

int ch;
if (!kbhit())
{
    ch = getch();
    if (ch == 27) //Nếu ấn Escape
        menu(); //Về menu chính
    if (ch == 13) // Nếu ấn Enter
        MauTuNhap(); //Bắt đầu thuật toán
}
```

Hình 2.5. Code hiển thị các phần tử

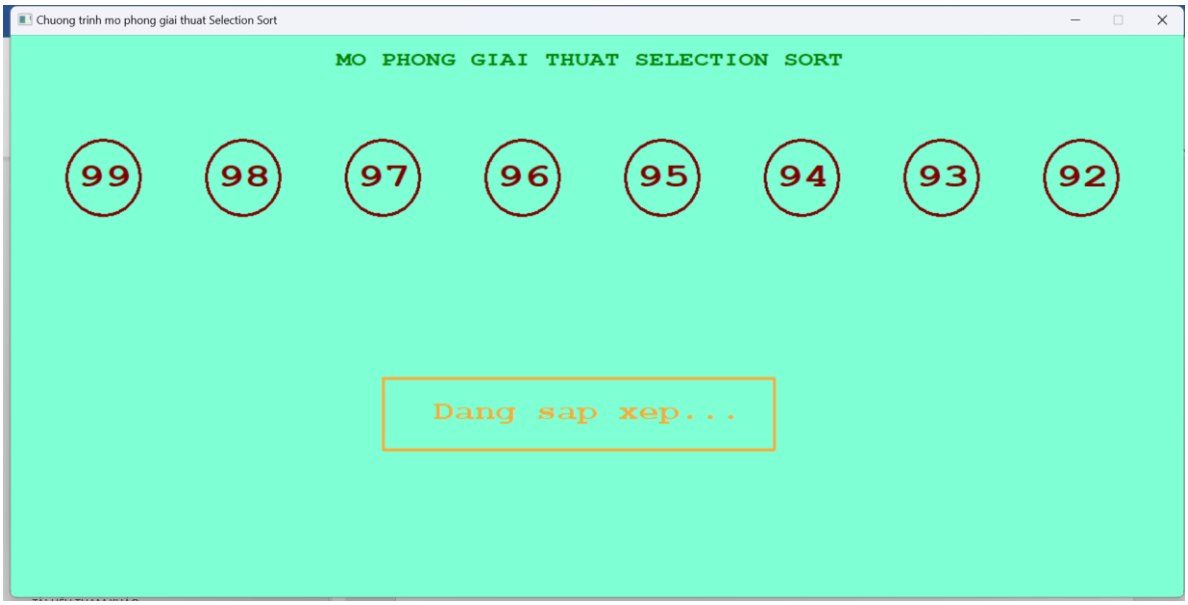
Hình 2.6. Màn hình hiển thị các phần tử

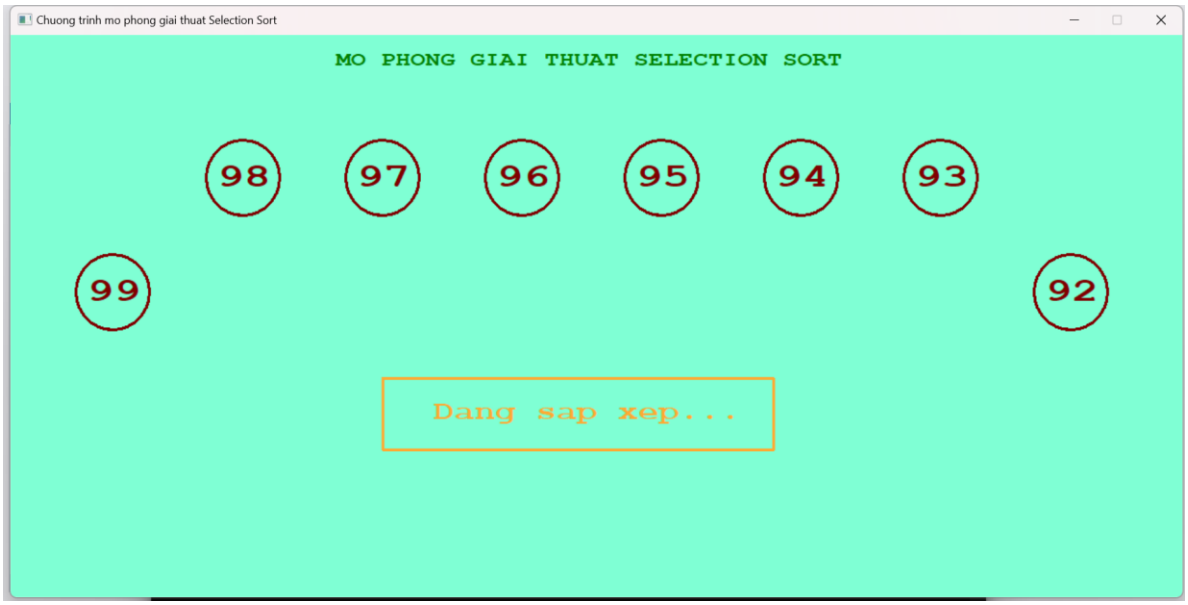
Thuật toán Selection Sort:

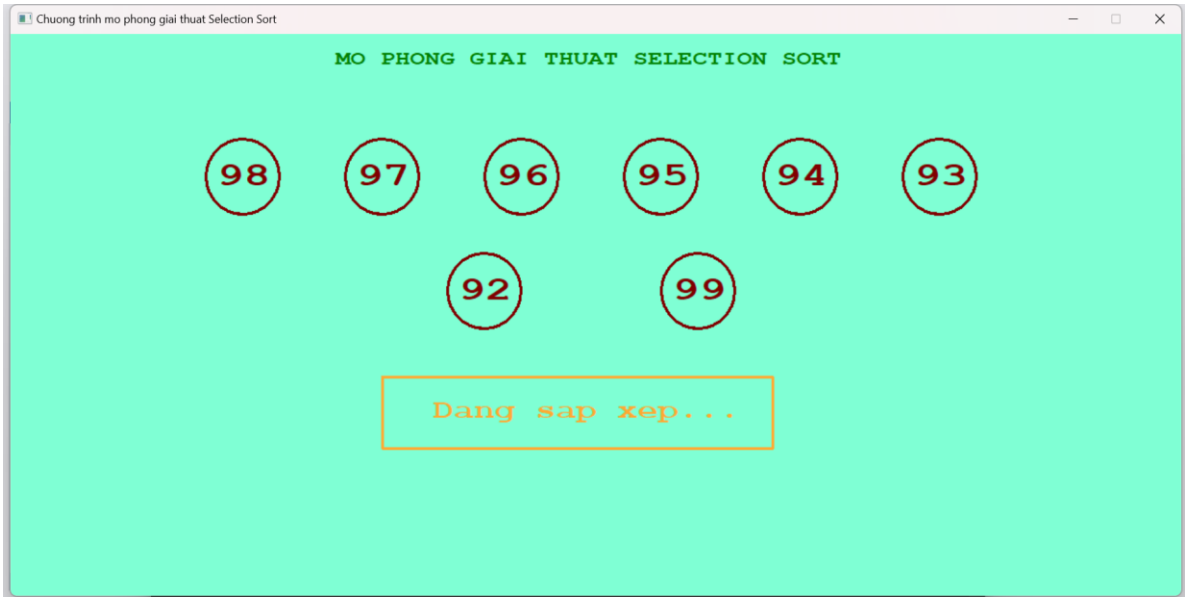
```
/* Bắt đầu sắp xếp */
int i, j, pos_min,
    posX_Circle1, posY_Circle1, posX_Text1, posY_Text1, pos_Context1,
    posX_Circle2, posY_Circle2, posX_Text2, posY_Text2, pos_Context2;
for (i = 0;i < size(int_arr);i++)
{
    pos_min = i; //Ban đầu cho phần tử đầu tiên là nhỏ nhất
    /* Lấy toạ độ từng phần tử đang ở đâu trên màn hình đồ hoạ */
    posX_Circle2 = xCircle[i], posY_Circle2 = yCircle[i];
    posX_Text2 = xNumber[i]; posY_Text2 = yNumber[i];
    pos_Context2 = i;
    for (j = i + 1;j < size(int_arr);j++) //Cho j chạy từ i+1 đến n-1
    {
        if (int_arr[j] < int_arr[pos_min]) //Nếu phát hiện phần tử nhỏ nhất
        {
            /* Lấy vị trí phần tử đó trong mảng và toạ độ của nó trên màn hình đồ hoạ */
            pos_min = j;
            cout << int_arr[i] << " " << int_arr[j] << endl;
            posX_Circle1 = xCircle[j]; posY_Circle1 = yCircle[j];
            posX_Text1 = xNumber[j]; posY_Text1 = yNumber[j];
            pos_Context1 = j;
        }
    }

    /* Nếu không tìm thấy phần tử nhỏ nhất, không tráo đổi, ngược lại tráo đổi vị trí cho nhau */
    if (pos_min != i)
    {
        /* Tráo đổi trên màn hình đồ hoạ */
        Trao_Doi_2_So(posX_Circle1, posY_Circle1,
                      posX_Circle2, posY_Circle2,
                      posX_Text1, posY_Text1, array_element[pos_Context1],
                      posX_Text2, posY_Text2, array_element[pos_Context2]);
        /* Tráo đổi trong mảng */
        swap(&int_arr[pos_min], &int_arr[i]);
    }
}
```

Hình 2.7. Code Selection Sort

CHƯƠNG 3. KẾT QUẢ THỰC HIỆN



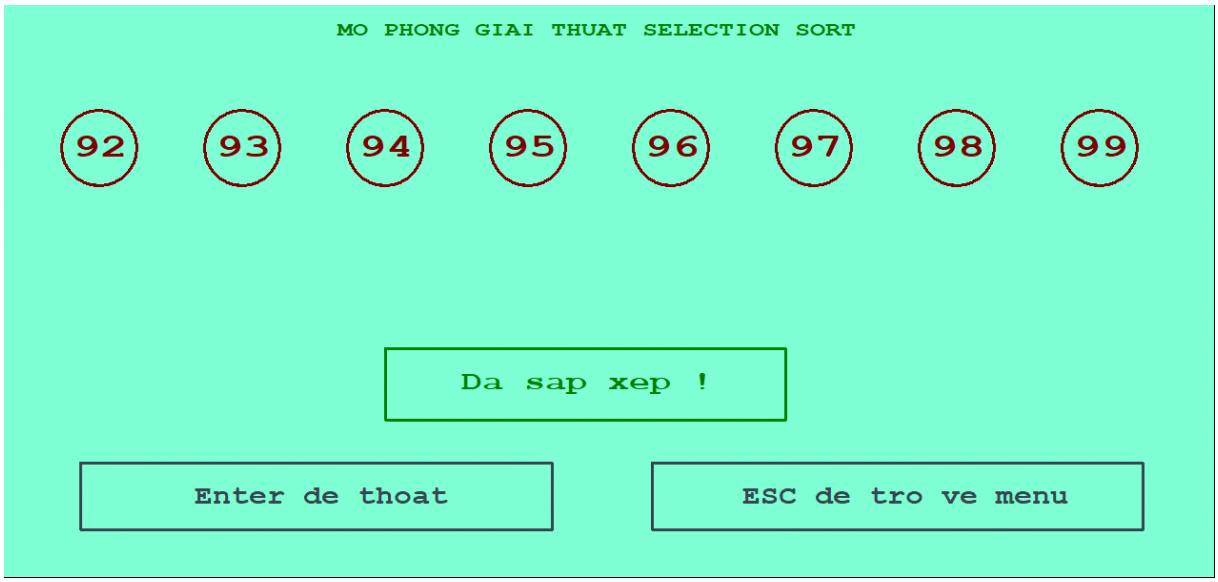




Hình 3.1. Selection Sort thực hiện sắp xếp

Từ hình 3.1 ta thấy, thuật toán bắt đầu chọn i = 0 (số 99) làm mốc, và tìm trong đoạn từ i+1 (số 98) cho đến n (số 92) số nhỏ nhất trong đoạn đó. Sau khi tìm ra số nhỏ nhất (số 92), thuật toán lưu lại vị trí số nhỏ nhất (i = 7) và kiểm tra xem liệu số 92 có nhỏ hơn số 99, sau khi thấy nhỏ hơn thì hoán đổi vị trí hai số cho nhau. Lúc này số 92 sẽ nằm ở vị trí i = 0, số 99 nằm ở vị trí i = 7.

Tiếp tục thực hiện như vậy cho đến khi mảng được sắp xếp hoàn chỉnh như bên dưới:



Hình 3.2. Selection Sort hoàn thành việc sắp xếp phần tử

THẢO LUẬN

Đề tài đã mô phỏng quá trình sắp xếp mảng bằng thuật toán Selection Sort với dữ liệu nhập từ bàn phím. Ngoài ra còn tìm hiểu thêm về thuật toán Bubble Sort và Quick Sort, thấy được thuật toán đã có cải tiến hơn so với Bubble Sort, tuy nhiên thời gian thực hiện sắp xếp của cả hai thuật toán trên đều chậm hơn và không được sử dụng nhiều như thuật toán Quick Sort, một thuật toán sắp xếp hiện đại và được sử dụng rất phổ biến hiện nay.

Đề tài vẫn chưa cài đặt được thuật toán với việc nhập dữ liệu từ chuột thay vì bàn phím. Trong tương lai, em sẽ tiếp tục tìm hiểu thêm và cài đặt thuật toán nhập dữ liệu từ chuột và sẽ cố gắng tìm hiểu và cải tiến thuật toán Selection Sort.

TÀI LIỆU THAM KHẢO

1. “Thuật toán sắp xếp”, Wikipedia Tiếng Việt
2. “Sorting algorithm”, English Wikipedia
3. “Bài 7: Sắp xếp”, Tổ hợp Công nghệ Giáo dục Topica