BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC NHA TRANG**

**KHOA CÔNG NGHỆ THÔNG TIN**

---🙢🕮🙠---

Logo, icon

Description automatically generated

**BÁO CÁO THỰC TẬP CƠ SỞ**

**Đề bài:** **Sử dụng công cụ đồ hoạ graphics.h để mô phỏng quá trình sắp xếp dãy số bằng giải thuật Selection Sort.**

**Giảng viên hướng dẫn: Ths. Đoàn Vũ Thịnh**

**Sinh viên thực hiện: Trần Hàng Tống Đạt**

**Mã số sinh viên: 63130193**

**Lớp: 63.CNTT-4**

**Khánh Hoà – tháng 12/2023**

**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:** Sử dụng công cụ đồ hoạ graphics.h để mô phỏng quá trình sắp xếp dãy số bằng giải thuật Selection Sort.

**Giảng viên hướng dẫn:** ThS. Đoàn Vũ Thịnh

**Sinh viên được hướng dẫn:** Trần Hàng Tống Đạt

**MSSV:** 63130193

**Khóa:** 63 **Ngành:** Công nghệ Thông tin

|  |  |  |  |
| --- | --- | --- | --- |
| **Lần** | **Ngày** | **Nội dung** | **Nhận xét của GVHD** |
| 1 | 27/11/2023 | Nhận đề tài hướng dẫn và định hướng giải quyết vấn đề. Sinh viên trình bày kế hoạch thực hiện. |  |
| 2 | 6/12/2023 | Sinh viên thực hiện mô phỏng giao diện thông qua việc tạo và đổi màu các button khi click, xử lý sự kiện chuột, các circle quanh mỗi phần tử và khung giao diện chứa dãy số cần sắp xếp. |  |
| 3 | 12/12/2023 | Sinh viên tiến hành thực hiện thuật toán đã đề ra với việc tạo ra các cách nhập dữ liệu đầu vào đa dạng như bàn phím, đọc file, random. |  |
| 4 | 21/12/2023 | Sinh viên tiếp tục thực hiện phân tích thuật toán, nắm rõ cách sắp xếp và di chuyển của các phần tử để tạo các chức năng sắp xếp tăng dần, giảm dần, tạo thêm hai nút chức năng là làm mới và thoát chương trình. |  |
| 5 | 28/12/2023 | Sinh viên thực hiện demo và sửa những lỗi phát sinh trong quá trình sắp xếp như lỗi dư thừa điểm ảnh và tô màu trong lúc di chuyển các phần tử. |  |
| 6 |  | Sinh viên nộp bản thảo lần cuối sau khi đã chỉnh sửa các yêu cầu như đã đề ra. |  |

**MỤC LỤC**

[**LỜI CẢM ƠN** 1](#_Toc155474147)

[**TÓM TẮT** 2](#_Toc155474148)

[**CHƯƠNG 1. TỔNG QUAN** 3](#_Toc155474149)

[**1.1 Dev C++** 3](#_Toc155474150)

[**1.2. Thư viện Graphics.h** 3](#_Toc155474151)

[**1.3 Thuật toán sắp xếp Selection Sort (sắp xếp chọn)** 4](#_Toc155474152)

[**1.3.1 Tổng quan và ý tưởng về thuật toán** 4](#_Toc155474153)

[**1.3.2 Lưu đồ thuật toán** 7](#_Toc155474154)

[**1.3.3 Đánh giá thuật toán** 7](#_Toc155474155)

[**CHƯƠNG 2. PHƯƠNG PHÁP NGHIÊN CỨU** 10](#_Toc155474156)

[**2.1. Tổng quan giao diện** 10](#_Toc155474157)

[**2.1.1. Cài đặt Dev-C++ và thư viện graphics.h** 10](#_Toc155474158)

[**2.1.2. Xử lý sự kiện chuột với thư viện BGI** 10](#_Toc155474159)

[**2.1.3. Xây dựng giao diện** 14](#_Toc155474160)

[**2.2. Phân tích thuật toán Selection Sort** 21](#_Toc155474161)

[**2.2.1. Khởi tạo mảng** 22](#_Toc155474162)

[**2.2.2. Thuật toán Selection Sort** 29](#_Toc155474163)

[**2.2.3. Button chức năng** 33](#_Toc155474164)

[**CHƯƠNG 3. KẾT QUẢ THỰC HIỆN** 35](#_Toc155474165)

[**CHƯƠNG 4. KẾT LUẬN** 39](#_Toc155474166)

[**4.1 Ưu điểm** 39](#_Toc155474167)

[**4.2 Nhược điểm** 39](#_Toc155474168)

[**TÀI LIỆU THAM KHẢO** 40](#_Toc155474169)

**DANH MỤC HÌNH**

[Hình 1.1 Minh hoạ từng bước thuật toán Selection Sort. 6](#_Toc155473939)

[Hình 1.2. Lưu đồ thuật toán Selection Sort. 7](#_Toc155473940)

[Hình 2.1. Ví dụ minh hoạ thư viện graphics.h 10](#_Toc155473941)

[Hình 2.2. Ví dụ code xử lý sự kiện chuột. 11](#_Toc155473942)

[Hình 2.3. Ví dụ về lấy tọa độ bằng cách sử dụng chuột. 12](#_Toc155473943)

[Hình 2.4. Code xử lý khi click chuột vào các nút. 14](#_Toc155473944)

[Hình 2.5. Hình ảnh giao diện. 15](#_Toc155473945)

[Hình 2.6. Code tạo khung. 15](#_Toc155473946)

[Hình 2.7. Code tạo nút. 16](#_Toc155473947)

[Hình 2.8. Code tạo màu nút. 19](#_Toc155473948)

[Hình 2.9. Hình giao diện tạo màu nút. 20](#_Toc155473949)

[Hình 2.10. Code tạo văn bản. 20](#_Toc155473950)

[Hình 2.11. Code tạo Circle. 21](#_Toc155473951)

[Hình 2.12. Hình ảnh Circle. 21](#_Toc155473952)

[Hình 2.13. Code nhập số lượng phần tử từ bàn phím. 23](#_Toc155473953)

[Hình 2.14. Hình ảnh nhập số lượng phần tử từ bàn phím. 23](#_Toc155473954)

[Hình 2.15. Hình ảnh giao diện nhập số lượng phần tử từ bàn phím. 24](#_Toc155473955)

[Hình 2.16. Hình ảnh code nhập phần tử từ bàn phím với số lượng mặc định. 25](#_Toc155473956)

[Hình 2.17. Hình ảnh nhập phần tử từ bàn phím với số lượng mặc định. 25](#_Toc155473957)

[Hình 2.18. Hình ảnh giao diện nhập phần tử từ bàn phím với số lượng mặc định. 26](#_Toc155473958)

[Hình 2.19. Hình ảnh code tạo phần tử ngẫu nhiên. 27](#_Toc155473959)

[Hình 2.20. Hình ảnh tạo phần tử ngẫu nhiên. 27](#_Toc155473960)

[Hình 2.21. Hình ảnh giao diện tạo phần tử ngẫu nhiên. 27](#_Toc155473961)

[Hình 2.22. Hình ảnh code tạo các phần tử bằng cách đọc file. 28](#_Toc155473962)

[Hình 2.23. Hình ảnh tạo các phần tử bằng cách đọc file. 28](#_Toc155473963)

[Hình 2.24. Hình ảnh giao diện tạo các phần tử bằng cách đọc file. 29](#_Toc155473964)

[Hình 2.25. Hình ảnh code thuật toán Selection Sort sắp xếp tăng dần. 30](#_Toc155473965)

[Hình 2.26. Hình ảnh sau khi sắp xếp tăng dần. 31](#_Toc155473966)

[Hình 2.27. Hình ảnh code thuật toán Selection Sort sắp xếp giảm dần. 32](#_Toc155473967)

[Hình 2.28. Hình ảnh sau khi sắp xếp giảm dần. 33](#_Toc155473968)

[Hình 2.29. Hình ảnh code Reset. 33](#_Toc155473969)

[Hình 2.30. Hình ảnh code Exit. 34](#_Toc155473970)

[Hình 3.1. Hình ảnh kết quả khởi tạo mảng. 35](#_Toc155473971)

[Hình 3.2. Hình ảnh kết quả đổi màu đỏ cho cặp phần tử đang so sánh. 35](#_Toc155473972)

[Hình 3.3. Hình ảnh kết quả đổi màu xanh cho phần tử nhỏ nhất được chọn 36](#_Toc155473973)

[Hình 3.4. Hình ảnh kết quả tiếp tục so sánh với phần tử được chọn cho đến hết mảng 36](#_Toc155473974)

[Hình 3.5. Hình ảnh đổi màu xanh cho phần tử cần di chuyển vị trí. 37](#_Toc155473975)

[Hình 3.6. Hình ảnh đổi màu đỏ cho cặp phần tử đang so sánh 37](#_Toc155473976)

[Hình 3.7. Hình ảnh kết quả Selection Sort sắp xếp tăng dần. 38](#_Toc155473977)

# **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 trính 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 chúng 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.

Em xin chân thành cảm ơn!

# **TÓM TẮT**

Đồ họa máy tính là một lĩnh vực của Công nghệ Thông tin, ở đó việc nghiên cứu xây dựng và tập hợp các công cụ (mô hình lý thuyết và phần mềm) khác nhau để kiến tạo, xây dựng, lưu trữ và xử lý các mô hình và hình ảnh của các đối tượng, sự vật, hiện tượng trong cuộc sống, sản xuất. Thuật toán tô màu bằng dòng quét và tô màu bằng đường biên trong đồ họa máy tính có tầm quan trọng rất lớn và được sử dụng rộng rãi trong các phần mềm đồ họa phổ biến hiện nay như Adobe Photoshop, Corel Draw, Microsoft Paint.

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 ứng dụng Dev C/C++ có kết hợp với thư viện graphics.h.

Sản phẩm đã minh họa được từng bước giải thuật Selection sort (sắp xếp chọn). 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 lấy từ file, nhập từ bàn phím và tưởng tác thuận lợi với sự hỗ trợ của chuột.

# **CHƯƠNG 1. TỔNG QUAN**

## **1.1 Dev C++**

Bloodshed Dev-C ++ là môi trường phát triển tích hợp (IDE) đầy đủ tính năng cho ngôn ngữ lập trình C/C ++ sử dụng Mingw của GCC (Bộ sưu tập trình biên dịch GNU) làm trình biên dịch.

Dev-C ++ cũng có thể kết hợp với Cygwin hoặc bất kỳ trình biên dịch dựa trên GCC nào khác. Các tính năng của Dev-C++:

* Hỗ trợ trình biên dịch dựa trên GCC.
* Gỡ lỗi tích hợp (sử dụng GDB- General DeBug).
* Quản lý dự án.
* Trình chỉnh sửa cú pháp.
* Trình duyệt lớp.
* Hoàn thành mã.
* Danh sách chức năng.
* Hồ sơ hỗ trợ.
* Nhanh chóng tạo Windows, console, thư viện tĩnh và DLL13.
* Hỗ trợ các mẫu để tạo các loại dự án của riêng bạn.
* Tạo Makefile.
* Chỉnh sửa và biên dịch các tệp Tài nguyên.
* Quản lý công cụ.
* Hỗ trợ in.
* Tìm và thay thế mã lệnh.
* Hỗ trợ CVS.

## **1.2. Thư viện Graphics.h**

Vì sử dụng Dev C++ 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 Michael tạo ra thư viên có tên là Graphics.h. để có thể làm được điều đó. Micheal đã thay đổi BGI library (thư viện BGI) thành thư viện có tên WinBGIm để có thể sử dụng tốt trên windows. Và bây giờ đã có thể sử dụng tốt các hàm đặc biệt của borland bằng Dev C++.

## **1.3 Thuật toán sắp xếp Selection Sort (sắp xếp chọn)**

### **1.3.1 Tổng quan và ý tưởng về thuật toán**

Selection sort là một thuật toán sắp xếp đơn giản thuộc loại thuật toán so sánh. Thuật toán này hoạt động bằng cách tìm kiếm phần tử nhỏ nhất trong danh sách và đưa nó về vị trí đầu tiên. Sau đó, nó tìm kiếm phần tử nhỏ nhất trong danh sách còn lại và đưa nó về vị trí thứ hai. Quá trình này lặp lại cho đến khi toàn bộ danh sách được sắp xếp.

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:

A diagram of a number system

Description automatically generated

A diagram of a number diagram

Description automatically generated with medium confidence

A diagram of numbers and circles

Description automatically generated

Hình .1 Minh hoạ từng bước thuật toán Selection Sort

### **1.3.2 Lưu đồ thuật toán**

A diagram of a flowchart

Description automatically generated

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:

Sso 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 Sgán = 0

• Trong trường hợp xấu nhất khi dãy có thứ tự giảm dần:

Số phép gán Sgán = 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(n2).

1.3.3.2 Ưu điểm và nhược điểm

Ưu điểm của Selection Sort:

* Đơn giản và dễ hiểu: Selection Sort có cấu trúc rất đơn giản, dễ hiểu và dễ triển khai.
* Không yêu cầu bộ nhớ phụ: Selection Sort thực hiện trực tiếp trên dãy số đã cho và không yêu cầu bộ nhớ phụ, nó chỉ cần một hằng số bộ nhớ để lưu vị trí của phần tử nhỏ nhất hoặc lớn nhất.
* Hiệu suất ổn định với dãy số nhỏ hoặc đã sắp xếp tốt: Trong một số trường hợp, Selection Sort có thể hiệu quả khi dãy số đã gần sắp xếp.

Nhược điểm của Selection Sort:

* Độ phức tạp thời gian lớn: Độ phức tạp thời gian của Selection Sort là O(n2), nơi n là số lượng phần tử cần sắp xếp. Điều này làm cho nó không phù hợp cho các dãy số lớn.
* Không ổn định khi xử lý các phần tử giống nhau: Selection Sort không giữ nguyên thứ tự của các phần tử giống nhau. Điều này có thể tạo ra sự không ổn định trong kết quả sắp xếp.
* Không phải là thuật toán ổn định: Selection Sort không bảo đảm rằng các phần tử có giá trị giống nhau sẽ được duyệt và giữ nguyên thứ tự của chúng, làm cho nó không phải là thuật toán ổn định.

So sánh với các thuật toán khác:

* So sánh với Bubble Sort: Selection Sort và Bubble Sort đều có độ phức tạp thời gian là O(n2), nhưng Bubble Sort có thể thích hợp hơn khi cần giữ nguyên thứ tự của các phần tử giống nhau.
* So sánh với Quick Sort và Merge Sort: Quick Sort và Merge Sort có độ phức tạp thời gian tốt hơn là O(n log n) và thích hợp cho các dãy số lớn. Tuy nhiên, chúng yêu cầu thêm bộ nhớ và phức tạp hơn trong việc triển khai.
* So sánh với Insertion Sort: Insertion Sort cũng có độ phức tạp thời gian O(n2) nhưng thích hợp hơn khi dãy số gần sắp xếp và có thể thực hiện tốt trên các dãy số nhỏ.

1.3.3.3 So sánh độ phức tạp về thời gian của các thuật toán

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuật toán sắp xếp** | **Thời gian thực hiện** | | |
| **Nhanh nhất** | **Trung bình** | **Chậm nhất** |
| Bubble Sort | O(n) | O(n2) | O(n2) |
| Selection Sort | O(n2) | O(n2) | O(n2) |
| Insertion Sort | O(n) | O(n2) | O(n2) |
| Merge Sort | O(nlogn) | O(nlogn) | O(nlogn) |
| Quick Sort | O(nlogn) | O(nlogn) | O(n2) |

# **CHƯƠNG 2. PHƯƠNG PHÁP NGHIÊN CỨU**

## **2.1. Tổng quan giao diện**

### **2.1.1. Cài đặt Dev-C++ và thư viện graphics.h**

Tải file cài đặt phần mềm Visual Studio theo đường dẫn ( <https://s.net.vn/f2Cq> ). 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

A screenshot of a computer

Description automatically generated

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

### **2.1.2. Xử lý sự kiện chuột với thư viện BGI**

Thư viện BGI nhận các loại sự kiện chuột như:

* WM\_MOUSEMOVE: bắt sự kiện khi di chuyển chuột trong màn hình đồ họa.
* WM\_LBUTTONDBLCLK: bắt sự kiện khi nhấn đúp chuột trái.
* WM\_LBUTTONDOWN: bắt sự kiện khi nhấn chuột trái.
* WM\_LBUTTONUP: bắt sự kiện khi thả sau khi nhấn chuột trái.

Tương tự với chuột giữa và phải.

* WM\_MBUTTONDBLCLK
* WM\_MBUTTONDOWN
* WM\_MBUTTONUP
* WM\_RBUTTONDBLCLK
* WM\_RBUTTONDOWN
* WM\_RBUTTONUP

Các hàm bắt sự kiện click chuột:

void getmouseclick( int kind, int& x, int& y );

-> lấy tọa độ x,y ( theo pixel) mà ở đó có click

bool ismouseclick( int kind );

-> trả về true nếu xảy ra click

Thư viện BGI cung cấp một số phương thức để tương tác với chuột và bàn phím trong ứng dụng đồ họa. Trong trường hợp bắt sự kiện khi người dùng nhấn chuột trái, chúng ta thường sử dụng sự kiện WM\_LBUTTONDOWN. Bộ đôi biến x\_mouse và y\_mouse sẽ lưu trữ tọa độ của điểm được nhấn chuột trái trên khung cửa sổ ứng dụng tại thời điểm hàm này được gọi.

Khi người dùng nhấn chuột trái, hàm xử lý sự kiện sẽ được gọi và cập nhật giá trị của x\_mouse và y\_mouse với tọa độ của điểm được nhấn.

Đoạn chương trình sau mô tả cách lấy tọa độ khi nhấn chuột trái, “Exiting…” khi nhấn chuột phải và in ra màn hình:

A computer screen shot of a computer code

Description automatically generated

Hình 2.2. Ví dụ code xử lý sự kiện chuột

A computer screen shot of a black screen

Description automatically generated

Hình 2.3. Ví dụ về lấy tọa độ bằng cách sử dụng chuột.

Sau khi lấy tọa độ, nếu nút nào có tọa độ nằm trong vùng giới hạn của x\_mouse và y\_mouse thì sẽ thực hiện gọi hàm đó. Ví dụ đối với nút “Increase” có tọa độ góc trái trên là (1290,50) và tọa độ góc phải dưới là (1465,200) thì ta giới hạn x\_mouse > *1*2*9*0 && x\_mouse < 1465 && y\_mouse > 50 && y\_mouse < 200, khi click chuột vào vùng tọa độ đó thì gọi hàm increase() và tiến hành sắp xếp. Các nút khác thực hiện tương tự.

*A screenshot of a computer program

Description automatically generated*

*A screenshot of a computer program

Description automatically generated*

Hình 2.4. Code xử lý khi click chuột vào các nút

### **2.1.3. Xây dựng giao diện**

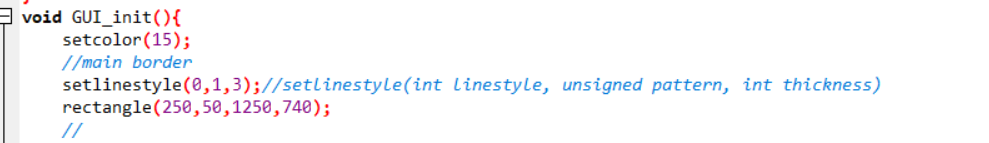
Giao diện được dùng để mô tả các thông tin của đề tài như: tên giáo viên hướng dẫn, tên sinh viên thực tập, mảng trước khi được sắp xếp, mảng sau khi sắp xếp. Các nút dùng để tạo mảng, nhập số liệu mảng, đọc dữ liệu từ file, tạo mảng ngẫu nhiên, chọn cách sắp xếp tăng hay giảm dần, nút làm mới lại mảng và nút thoát chương trình.

**A screenshot of a computer

Description automatically generated**

Hình 2.5. Hình ảnh giao diện

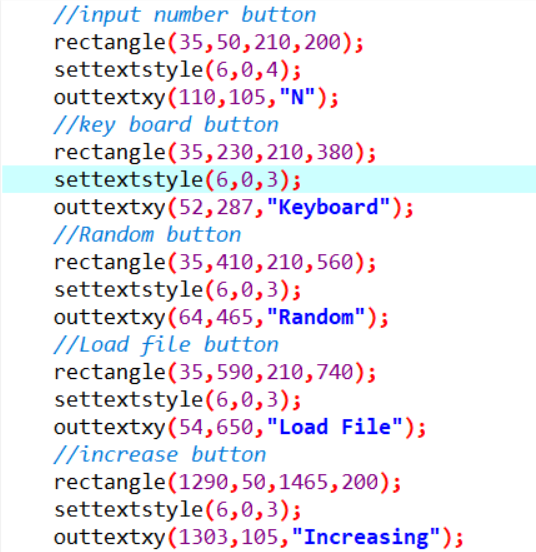
2.1.3.1. Xây dựng khung



Hình 2.6. Code tạo khung

Để xây dựng được khung như trên giao diện, đầu tiên ta sử dụng setcolor(15) để thiết lập màu vẽ trong thư viện đồ họa cụ thể là màu trắng. Tiếp theo ta thiết lập kiểu đường vẽ setlinestyle(0, 1, 3). Cụ thể, đối số đầu tiên (0) là kiểu đường vẽ nét liền, đối số thứ hai là một biểu diễn nhị phân của mẫu đường vẽ và sử dụng số 1 để vẽ đường đơn liền, và đối số cuối cùng dùng để vẽ độ dày của đường vẽ là 3. Cuối cùng để có được một hình chữ nhật hoàn chỉnh ta sử dụng hàm rectangle(250, 50,1250,740). Hình chữ nhật từ điểm có tọa độ góc trái trên là (250, 50) và tọa độ góc phải dưới là (1250, 740). Hình chữ nhật có chiều dài là 1000 pixel và chiều cao là 690 pixel.

2.1.3.1. Xây dựng nút



A screen shot of a computer code

Description automatically generated

Hình 2.7. Code tạo nút

* Nút “N”:

Sử dụng rectangle(35,50,210,200) để vẽ hình chữ nhật từ điểm có tọa độ góc trái trên là (35, 50) và tọa độ góc phải dưới là (210, 200). Nút “N” có chiều dài 175 pixel và chiều cao là 150 pixel.

Sử dụng settextstyle(6,0,4); outtextxy(110,105,"N") để tạo chữ "N" tại tọa độ (110, 105) với kiểu chữ số 6, không in đậm (0), và kích thước là 4.

* Nút “Keyboard”:

Sử dụng rectangle(35,230,210,380) để vẽ hình chữ nhật từ điểm có tọa độ góc trái trên là (35, 230) và tọa độ góc phải dưới là (210, 380). Nút “Keyboard” có chiều dài 175 pixel và chiều cao là 150 pixel.

Sử dụng settextstyle(6,0,3); outtextxy(52,287,"Keyboard") để tạo chữ "Keyboard" tại tọa độ (52, 287) với kiểu chữ số 6, không in đậm (0), và kích thước là 3.

* Nút “Random”:

Sử dụng rectangle(35,410,210,560) để vẽ hình chữ nhật từ điểm có tọa độ góc trái trên là (35, 410) và tọa độ góc phải dưới là (210, 560). Nút “Random” có chiều dài 175 pixel và chiều cao là 150 pixel.

Sử dụng settextstyle(6,0,3); outtextxy(64,465,"Random") để tạo chữ "Random" tại tọa độ (64,465) với kiểu chữ số 6, không in đậm (0), và kích thước là 3.

* Nút “Load File”:

Sử dụng rectangle(35,590,210,740) để vẽ hình chữ nhật từ điểm có tọa độ góc trái trên là (35, 590) và tọa độ góc phải dưới là (210, 740). Nút “Load File” có chiều dài 175 pixel và chiều cao là 150 pixel.

Sử dụng settextstyle(6,0,3); outtextxy(54,650,"Load File") để tạo chữ "Load File" tại tọa độ (54,650) với kiểu chữ số 6, không in đậm (0), và kích thước là 3.

* Nút “Increasing”:

Sử dụng rectangle(1290,50,1465,200) để vẽ hình chữ nhật từ điểm có tọa độ góc trái trên là (1290,50) và tọa độ góc phải dưới là (1465,200). Nút “Increasing” có chiều dài 175 pixel và chiều cao là 150 pixel.

Sử dụng settextstyle(6,0,3); outtextxy(1303,105,"Increasing") để tạo chữ "Increasing" tại tọa độ (1303,105) với kiểu chữ số 6, không in đậm (0), và kích thước là 3.

* Nút “Decreasing”:

Sử dụng rectangle(1290,230,1465,380) để vẽ hình chữ nhật từ điểm có tọa độ góc trái trên là (1290,230) và tọa độ góc phải dưới là (1465,380). Nút “Decreasing” có chiều dài 175 pixel và chiều cao là 150 pixel.

Sử dụng settextstyle(6,0,3); outtextxy(1300,287,"Decreasing") để tạo chữ "Decreasing" tại tọa độ (1300,287) với kiểu chữ số 6, không in đậm (0), và kích thước là 3.

* Nút “Reset”:

Sử dụng rectangle(1290,410,1465,560) để vẽ hình chữ nhật từ điểm có tọa độ góc trái trên là (1290,410) và tọa độ góc phải dưới là (1465,560). Nút “Reset” có chiều dài 175 pixel và chiều cao là 150 pixel.

Sử dụng settextstyle(6,0,3); outtextxy(1338,467,"Reset") để tạo chữ "Reset" tại tọa độ (1338,467) với kiểu chữ số 6, không in đậm (0), và kích thước là 3.

* Nút “Exit”:

Sử dụng rectangle(1290,590,1465,740) để vẽ hình chữ nhật từ điểm có tọa độ góc trái trên là (1290,590) và tọa độ góc phải dưới là (1465,740). Nút “Exit” có chiều dài 175 pixel và chiều cao là 150 pixel.

Sử dụng settextstyle(6,0,3); outtextxy(1350,650,"Exit") để tạo chữ "Exit" tại tọa độ (1350,650) với kiểu chữ số 6, không in đậm (0), và kích thước là 3.

2.1.3.3. Xây dựng màu nút

A computer screen shot of a computer code

Description automatically generated

A screen shot of a computer program

Description automatically generated

Hình 2.8. Code tạo màu nút

Sử dụng setcolor(4) để chuyển thành màu đỏ cho nút được click và setcolor(15) để chuyển các nút còn lại thành màu trắng. Ví dụ khi click vào nút “N” để nhập số lượng phần tử thì nút “N” sẽ chuyển sang màu đỏ và các nút còn lại sẽ có màu trắng. Tương tự với khi click với các nút khác.

A computer screen with a black screen

Description automatically generated

Hình 2.9. Hình giao diện tạo màu nút.

2.1.3.4. Xây dựng văn bản

A close-up of a text

Description automatically generated

Hình 2.10. Code tạo văn bản

Sử dụng settextstyle(6,0,4) để tạo văn bản với kiểu chữ số 6, không in đậm (0), và kích thước là 4.

* outtextxy(200,750," |GVHD: Doan Vu Thinh |") để tạo “|GVHD: Doan Vu Thinh |” tại tọa độ (200, 750).
* outtextxy(740,750," |Ten sv: Tran Hang Tong Dat|") để tạo “|Ten sv: Tran Hang Tong Dat|” tại tọa độ (740, 750).

2.1.3.5. Xây dựng Circle

A computer code with text

Description automatically generated

Hình 2.11. Code tạo Circle

* Sử dụng hàm circle(point.x, point.y, newRadius) để vẽ một hình tròn tại tọa độ (point.x, point.y) với bán kính là 40.
* Sử dụng settextstyle(1,0,4) để tạo văn bản với kiểu chữ số 1, không in đậm (0), và kích thước là 4.
* Để hiển thị số tại tọa độ (point.x - 11, point.y – 15), và giá trị của số là point.value được chuyển đổi thành chuỗi và lưu vào buffer (có thể là một mảng ký tự), sau đó được hiển thị.

A number in a circle

Description automatically generated

Hình 2.12. Hình ảnh Circle

## **2.2. Phân tích thuật toán Selection Sort**

**Khai báo thư viện**

**A screenshot of a computer code

Description automatically generated**

**Khai báo hàm và biến toàn cục**

**A screen shot of a computer program

Description automatically generated**

**A white background with blue text

Description automatically generated**

### **2.2.1. Khởi tạo mảng**

2.2.1.1Nhập số lượng phần tử từ bàn phím

Người dùng sẽ nhập số lượng phần tử trong khoảng từ 2 đến 9 và giá trị phần tử mà họ muốn sắp xếp. Tọa độ của các phần tử sẽ được cố định sẵn trong code. Sau khi nhập xong tọa độ và giá trị của các phần tử sẽ được in ra màn hình.

A computer code with numbers and symbols

Description automatically generated

Hình 2.13. Code nhập số lượng phần tử từ bàn phím

Ví dụ: Nhập số lượng phần tử là 5 và giá trị lần lượt là 3,5,7,9,1

A screenshot of a computer

Description automatically generated

Hình 2.14. Hình ảnh nhập số lượng phần tử từ bàn phím

*A screenshot of a computer

Description automatically generated*

Hình 2.15. Hình ảnh giao diện nhập số lượng phần tử từ bàn phím

2.2.1.2Nhập phần tử từ bàn phím với số lượng mặc định

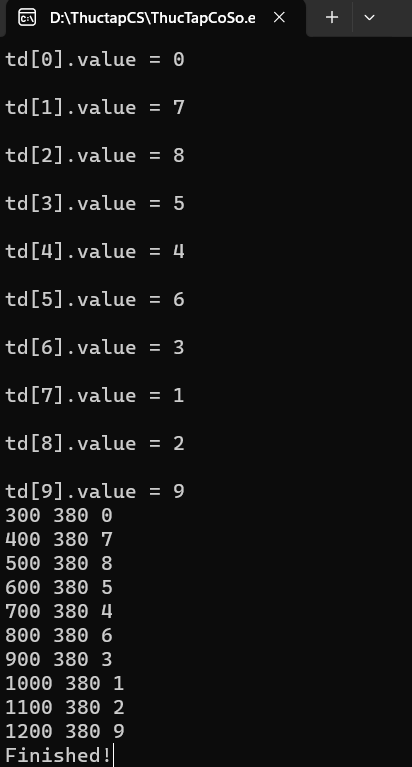
Chúng ta sẽ cố định thuật toán Selection Sort sắp xếp 10 phần tử. Tọa độ của các phần tử được lấy từ file và giá trị của 10 phần tử sẽ được nhập từ bàn phím. Sau khi nhập xong tọa độ và giá trị của các phần tử sẽ được in ra màn hình.

*A computer code with red and blue text

Description automatically generated*

Hình 2.16. Hình ảnh code nhập phần tử từ bàn phím với số lượng mặc định

Ví dụ: Nhập giá trị cho 10 phần tử lần lượt là 0,7,8,5,4,6,3,1,2,9.



Hình 2.17. Hình ảnh nhập phần tử từ bàn phím với số lượng mặc định

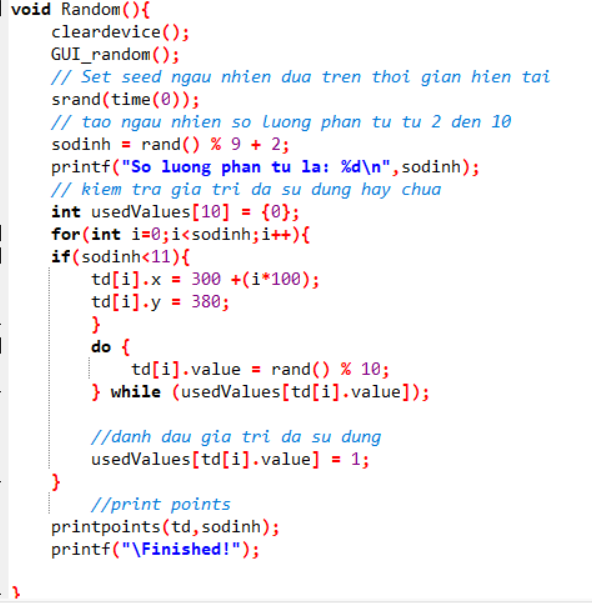
*A screenshot of a computer

Description automatically generated*

Hình 2.18. Hình ảnh giao diện nhập phần tử từ bàn phím với số lượng mặc định

2.2.1.3Tạo các phần tử ngẫu nhiên

Ta sử dụng hai hàm rand(), một hàm để tạo số lượng ngẫu nhiên và một hàm để tạo ra các giá trị ngẫu nhiên không trùng nhau. Sau mỗi lần click nút “Random”, giao diện sẽ được xóa và vẽ lại để tránh trường hợp các số bị chồng lên nhau. Tọa độ của các số được cài đặt trong code. Sau khi tạo ngẫu nhiên, số lượng phần tử, tọa độ và giá trị sẽ được in ra.

****

Hình 2.19. Hình ảnh code tạo phần tử ngẫu nhiên

Ví dụ: Khi click vào nút Random các phần tử ngẫu nhiên sẽ được tạo ra.

*A screenshot of a computer

Description automatically generated*

Hình 2.20. Hình ảnh tạo phần tử ngẫu nhiên.

*A screenshot of a computer

Description automatically generated*

Hình 2.21. Hình ảnh giao diện tạo phần tử ngẫu nhiên.

2.2.1.4Tạo các phần tử bằng cách đọc file

Chúng ta sẽ cố định thuật toán Insertion Sort sắp xếp 10 phần tử. Tọa độ của các phần tử và giá trị của chúng sẽ được lấy từ file. Sau khi lấy từ file tọa độ và giá trị của các phần tử sẽ được in ra màn hình.

**A computer code with many text

Description automatically generated with medium confidence**

Hình 2.22. Hình ảnh code tạo các phần tử bằng cách đọc file.

Ví dụ: Tạo file chứa 10 phần tử có giá trị là 0, 3, 2, 6, 4, 9, 7, 5, 8.

*A screenshot of a computer

Description automatically generated*

Hình 2.23. Hình ảnh tạo các phần tử bằng cách đọc file.

*A screenshot of a computer

Description automatically generated*

Hình 2.24. Hình ảnh giao diện tạo các phần tử bằng cách đọc file.

### **2.2.2. Thuật toán Selection Sort**

* + - 1. Sắp xếp tăng dần

Tạo hàm increase() để thực hiện quá trình sắp xếp . Để ngăn chặn sự kiện click chuột trong quá trình thực hiện sắp xếp ta gán giá trị 0 cho biến enable\_click. Đầu tiên, sử dụng vòng lặp ngoài cùng để duyệt qua từng phần tử của mảng td.

Khởi tạo min\_index là chỉ số của phần tử nhỏ nhất (i) trong mảng chưa được sắp xếp. Trong vòng lặp thứ hai, duyệt qua các phần tử chưa được sắp xếp và tìm ra min\_index - chỉ số của phần tử nhỏ nhất.

Trong quá trình tìm kiếm, các phần tử được đánh dấu màu đỏ để thể hiện quá trình

so sánh. Nếu min\_index khác với i, thực hiện hoán đổi giữa phần tử nhỏ nhất và phần tử đầu tiên chưa được sắp xếp. Các phần tử tham gia hoán đổi được đánh dấu màu xanh lá cây để thể hiện quá trình hoán đổi. Sau khi sắp xếp xong, vòng lặp cuối cùng được sử dụng để vẽ lại tất cả các điểm trên giao diện với màu cam (drawpoint(td[i], 14)) để thể hiện rằng mảng đã được sắp xếp. Gán giá trị 1 cho enable\_click để sau khi sắp xếp hoàn tất, chương trình cho phép thực hiện các hành động liên quan đến click chuột.

A screenshot of a computer program

Description automatically generated

A computer code with text

Description automatically generated

Hình 2.25. Hình ảnh code thuật toán Selection Sort sắp xếp tăng dần.

A screenshot of a computer

Description automatically generated

Hình 2.26. Hình ảnh sau khi sắp xếp tăng dần.

2.2.2.2 Sắp xếp giảm dần

Tạo hàm decrease() để thực hiện quá trình sắp xếp . Để ngăn chặn sự kiện click chuột trong quá trình thực hiện sắp xếp ta gán giá trị 0 cho biến enable\_click. Đầu tiên, sử dụng vòng lặp ngoài cùng để duyệt qua từng phần tử của mảng td.

Khởi tạo max\_index là chỉ số của phần tử lớn nhất (i) trong mảng chưa được sắp xếp. Trong vòng lặp thứ hai, duyệt qua các phần tử chưa được sắp xếp và tìm ra max\_index - chỉ số của phần tử lớn nhất.

Trong quá trình tìm kiếm, các phần tử được đánh dấu màu đỏ để thể hiện quá trình so sánh. Nếu max\_index khác với i, thực hiện hoán đổi giữa phần tử lớn nhất và phần tử đầu tiên chưa được sắp xếp. Các phần tử tham gia hoán đổi được đánh dấu màu xanh lá cây để thể hiện quá trình hoán đổi. Sau khi sắp xếp xong, vòng lặp cuối cùng được sử dụng để vẽ lại tất cả các điểm trên giao diện với màu cam (drawpoint(td[i], 14)) để thể hiện rằng mảng đã được sắp xếp. Gán giá trị 1 cho enable\_click để sau khi sắp xếp hoàn tất, chương trình cho phép thực hiện các hành động liên quan đến click chuột.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

Hình 2.27. Hình ảnh code thuật toán Selection Sort sắp xếp giảm dần.

*A screenshot of a computer

Description automatically generated*

Hình 2.28. Hình ảnh sau khi sắp xếp giảm dần.

### **2.2.3. Button chức năng**

2.2.3.1 Button Reset

Nút “Reset” được sử dụng để đặt lại trạng thái của giao diện người dùng về trạng thái ban đầu sau khi đã thực hiện một số thay đổi như khởi tạo mảng, sắp xếp mảng, và để chuẩn bị cho các tương tác tiếp theo.

A screen shot of a computer code

Description automatically generated

Hình 2.29. Hình ảnh code Reset.

Để xóa nội dung hiện tại trên thiết bị đồ họa ta sử dụng hàm cleardevice(). Điều này có thể đưa giao diện về trạng thái mặc định. Sau đó gọi hàm GUI\_init() để khởi tạo lại giao diện người dùng về trạng thái ban đầu. Gán giá trị 1 cho biến enable\_click được sử dụng để kiểm soát việc kích hoạt chức năng click chuột, việc gán giá trị 1 đại diện cho việc cho phép thực hiện các hành động liên quan đến click chuột sau khi đã thực hiện đặt lại.

2.2.3.2 Button Exit

Nút “Exit” được sử dụng để đảm bảo việc giải phóng tất cả các tài nguyên đồ họa và chấm dứt chương trình một cách sạch sẽ khi người dùng quyết định thoát khỏi ứng dụng.

A computer code with black text

Description automatically generated with medium confidence

Hình 2.30. Hình ảnh code Exit.

Để đóng lại hoặc giải phóng tất cả các tài nguyên đồ họa đã được sử dụng trong chương trình ta sử dụng một hàm từ thư viện đồ họa (closegraph()). Hàm này có thể giải phóng bộ nhớ đồ họa, đóng cửa sổ đồ họa và các tài nguyên khác liên quan đến đồ họa. Hàm exit (0) dùng để kết thúc chương trình và thoát khỏi quá trình thực thi. Tham số truyền vào là 0 biểu thị rằng chương trình đã kết thúc mà không có lỗi.

# **CHƯƠNG 3. KẾT QUẢ THỰC HIỆN**

**A screenshot of a computer

Description automatically generated**

Hình 3.1. Hình ảnh kết quả khởi tạo mảng.

**A screenshot of a computer

Description automatically generated**

Hình 3.2. Hình ảnh kết quả đổi màu đỏ cho cặp phần tử đang so sánh.

A screenshot of a computer

Description automatically generated

Hình 3.3. Hình ảnh kết quả đổi màu xanh cho phần tử nhỏ nhất được chọn

A screenshot of a computer

Description automatically generated

Hình 3.4. Hình ảnh kết quả tiếp tục so sánh với phần tử được chọn cho đến hết mảng

A screenshot of a computer

Description automatically generated

Hình 3.5. Hình ảnh đổi màu xanh cho phần tử cần di chuyển vị trí.

A screenshot of a computer

Description automatically generated

Hình 3.6. Hình ảnh đổi màu đỏ cho cặp phần tử đang so sánh

A screenshot of a computer

Description automatically generated

Hình 3.7. Hình ảnh kết quả Selection Sort sắp xếp tăng dần.

# **CHƯƠNG 4. KẾT LUẬN**

**4.1 Ưu điểm**

Sau một thời gian tìm hiểu, nghiên cứu, lập trình thì đề tài cũng đã được mô phỏng trên giao diện Dev-C++ quá trình sắp xếp dãy số bằng giải thuật Selection Sort sử dụng công cụ đồ hoạ graphics.h. Giao diện trình bày được đầy đủ các chức năng, tiến trình của việc sắp xếp chọn, sử dụng chuột để nhập dữ liệu, có nhiều sự lựa chọn khởi tạo mảng trong việc sắp xếp tăng dần và giảm dần như nhập từ bàn phím, sử dụng chuột hay tạo ngẫu nhiên. Các công đoạn được trình bày cụ thể, dễ hiểu, dễ nắm bắt, thông tin được đưa ra rõ ràng, dễ nhìn. Khi quy trình sắp xếp chọn được kích hoạt, ta có thể sắp xếp được các mảng theo một trật tự nhất định, có thể là tăng dần hoặc giảm dần.

**4.2 Nhược điểm**

Tuy nhiên, đề tài vẫn vẫn còn một số nhược điểm như giao diện chưa được bắt mắt, code chưa được xử lý tối ưu và thời gian thực hiện sắp xếp của thuật toán Selection Sort vẫn chậm hơn và không được sử dụng nhiều như thuật toán Quick Sort. Em sẽ cố gắng tìm hiểu và cải tiến nhược điểm trên trong quá trình lập trình.

**TÀI LIỆU THAM KHẢO**

1. Jadoon, S., Solehria, S. F., Rehman, S. U., & Jan, H. (2011). Design and analysis of optimized selection sort algorithm. *International Journal of Electric & Computer Sciences (IJECS-IJENS)*, *11*(01), 16-22.
2. Thạc Bình Cường (2023), *Cấu trúc dữ liệu và giải thuật,* NXB Bách Khoa Hà Nội, Hà Nội.
3. Nguyễn Văn Hiếu (2019). Thuật toán sắp xếp chọn (Selection Sort). Truy cập ngày 25/12/2023,

từ < https://blog.luyencode.net/thuat-toan-sap-xep-chon/>.

1. Hải Hà (2019). Các thuật toán sắp xếp cơ bản. Truy cập ngày 30/12/2023, từ <<https://viblo.asia/p/cac-thuat-toan-sap-xep-co-ban-Eb85ooNO52G>>.
2. Đỗ Xuân Lôi, *Cấu trúc dữ liệu và Giải thuật*, NXB GD 2005.
3. Lương Chi Mai (2000), *Nhập môn đồ họa máy tính*, NXB Khoa học và Kỹ thuật, Hà Nội.
4. TS. Nguyễn Hữu Tài (2017), *Đồ họa máy tính,* NXB Đại học Huế, Huế.
5. Đoàn Vũ Thịnh, *Bài giảng Kỹ thuật đồ họa*, 2019, Đại học Nha Trang.
6. Shirley, P., Ashikhmin, M., & Marschner, S. (2009). *Fundamentals of computer graphics,* AK Peters/CRC Press.
7. Willms, A. (2008*). C++-der interaktive Programmierkurs: CD-ROM mit 8 Stunden Lernprogramm und Entwicklungsumgebung Dev-C++;* *Begleitbuch (Vol. 1).* Pearson Deutschland GmbH.