BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC NHA TRANG**

**KHOA CÔNG NGHỆ THÔNG TIN**

🙡•🙣



**BÁO CÁO THỰC TẬP CƠ SỞ (CÔNG NGHỆ THÔNG TIN)**

**ỨNG DỤNG GIẢI THUẬT SẮP XẾP ĐỂ SẮP XẾP DỮ LIỆU THỜI TIẾT (HEAP SORT VÀ INSERTION SORT)**

**Giảng viên hướng dẫn: TS. Cấn Thị Phượng**

**Sinh viên thực hiện: Nguyễn Nhật Thành**

**Mã số sinh viên: 64132192**

**Lớp: 64CNTT-4**

**Khánh Hòa, tháng 12/2024**

**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 thực hiện đề tài này.

Qua đây, em xin chân thành cảm ơn cô Cấn Thị Phượng, 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.

Em xin chân thành cảm ơn!

**TÓM TẮT**

Ứng dụng giải thuật sắp xếp để sắp xếp dữ liệu thời tiết có thể giúp người dùng trực quan hóa và hiểu rõ hơn về cách các thuật toán sắp xếp hoạt động. Trong ứng dụng này, hai thuật toán sắp xếp phổ biến là **Heap Sort** và **Insertion Sort** được áp dụng để sắp xếp các thông tin dữ liệu thời tiết như nhiệt độ (temperature), độ ẩm (humidity) và tốc độ gió (wind speed). Việc sử dụng các thuật toán sắp xếp không chỉ giúp tối ưu hóa việc xử lý dữ liệu mà còn tạo cơ hội cho người dùng theo dõi quá trình sắp xếp dữ liệu một cách trực quan.

Ứng dụng này sử dụng **PyQt5** để xây dựng giao diện người dùng, cho phép người dùng nhập dữ liệu thời tiết hoặc tải dữ liệu từ các nguồn khác nhau, chẳng hạn như các tệp dữ liệu thời tiết hoặc API thời tiết. Sau khi nhập dữ liệu, người dùng có thể chọn thuật toán sắp xếp mà họ muốn sử dụng. **Heap Sort** được chọn để sắp xếp dữ liệu với độ phức tạp thời gian O(n log n), giúp xử lý hiệu quả với các tập dữ liệu lớn. Ngược lại, **Insertion Sort** có độ phức tạp thời gian O(n²), phù hợp với các dữ liệu nhỏ hoặc giúp người dùng hiểu rõ hơn về cách hoạt động của thuật toán sắp xếp cơ bản.

Kết quả sắp xếp được hiển thị dưới dạng bảng dữ liệu hoặc đồ họa trực quan, giúp người dùng dễ dàng theo dõi và so sánh các thuật toán sắp xếp. Với ứng dụng này, người dùng không chỉ có thể sắp xếp dữ liệu thời tiết mà còn học được cách thức các thuật toán sắp xếp ảnh hưởng đến hiệu suất và kết quả xử lý dữ liệu.

**MỤC LỤC**

[TÓM TẮT 2](#_Toc10870)

[DANH MỤC HÌNH ẢNH 4](#_Toc29592)

[Chương 1. TỔNG QUAN VỀ ĐỀ TÀI 5](#_Toc3197)

[1.1.GIỚI THIỆU ĐỀ TÀI 5](#_Toc2655)

[1.2.MỤC TIÊU ĐỀ TÀI 5](#_Toc1720)

[1.3. PYTHON VÀ THƯ VIỆN PYQT5 5](#_Toc158)

[1.3.1. Giới thiệu về Python 5](#_Toc4935)

[1.3.2. Giới thiệu về thư viện PyQt5 6](#_Toc12971)

[Chương 2. CƠ SỞ LÝ THUYẾT 7](#_Toc1301)

[2.1. GIỚI THIỆU CHUNG VỀ THUẬT TOÁN 7](#_Toc6150)

[2.1.1. Thuật toán sắp xếp 7](#_Toc11481)

[2.1.2. Một số thuật toán sắp xếp phổ biến 7](#_Toc216)

[2.1.3. So sánh các thuật toán sắp xếp 7](#_Toc6949)

[2.2. GIỚI THIỆU VỀ THUẬT TOÁN HEAP SORT 8](#_Toc24659)

[2.2.1. Giới thiệu về thuật toán 8](#_Toc4026)

[2.2.2. Ý tưởng về thuật toán 8](#_Toc4898)

[2.2.3. Ví dụ về Max Heap và Min Heap 9](#_Toc26817)

[2.2.4. Ví dụ minh họa 9](#_Toc19021)

[2.3. GIỚI THIỆU THUẬT TOÁN INSRTION 10](#_Toc14495)

[2.3.1. Giới thiệu về thuật toán 10](#_Toc17975)

[2.3.2. Ý tưởng về thuật toán 10](#_Toc8051)

[2.3.3. Lưu đồ thuật toán 11](#_Toc10918)

[2.2.4. Ví dụ minh họa 12](#_Toc11653)

[Chương 3. KẾT QUẢ NGHIÊN CỨU 13](#_Toc15350)

[3.1. SƠ ĐỒ PHÂN RÃ CHỨC NĂNG (FDD) 13](#_Toc32300)

[- Nhận dữ liệu và yêu cầu từ người dùng 13](#_Toc5853)

[- Xử lý và sắp xếp dữ liệu 13](#_Toc30460)

[- Hiển thị kết quả và lưu trữ 13](#_Toc16246)

[3.2. SƠ ĐỒ LUÔNG DỮ LIỆU (DFD) 14](#_Toc13621)

[3.2.1. Sơ đồ mức ngữ cảnh 14](#_Toc2403)

[3.2.2. Sơ đồ mức 0 15](#_Toc17350)

[3.3. SƠ ĐỒ UML 16](#_Toc10826)

[3.3.1. Sơ đồ Use Case Diagram (Biểu đồ trường hợp sử dụng) 16](#_Toc18024)

[3.3.2. Sơ đồ Class Diagram (Biểu đồ Lớp) 17](#_Toc4349)

[3.3.3. Sơ đồ Sequence Diagram 18](#_Toc27296)

[3.4 SƠ ĐỒ THỰC THỂ KẾT HỢP (ERD) 19](#_Toc12035)

[3.4.1. Sơ đồ ERD mức quan niệm 19](#_Toc13728)

[3.4.2. Sơ đồ ERD mức vật lý 19](#_Toc3945)

[3.4.3. Sơ đồ ERD mức logic 20](#_Toc15968)

[TÀI LIỆU THAM KHẢO (GeeksforGeeks, 2024) 21](#_Toc11511)

[TÀI LIỆU TIẾNG ANH 21](#_Toc7534)

**DANH MỤC HÌNH ẢNH**

[Hình 1. 1. Giới thiệu về Python 9](#_Toc184370620)

[Hình 1. 2.Giới thiệu về thư viện PyQt5 9](#_Toc184370621)

[Hình 2. 1.Tổng quan về so sánh của một số thuật toán sắp xếp 11](#_Toc184370622)

[Hình 2. 2.Ví dụ về Max Heap và Min Heap 12](#_Toc184370623)

[Hình 2. 3.Ví dụ minh họa cho thuật toán Heap Sort 12](#_Toc184370624)

[Hình 2. 4. Lưu đồ thuật toán 14](#_Toc184370625)

[Hình 2. 5.Ví dụ minh họa 14](#_Toc184370626)

[Hình 3. 1.Sơ đồ phân rã FDD 15](#_Toc184370627)

[Hình 3. 2.Sơ đồ mức ngữ cảnh 16](#_Toc184370628)

[Hình 3. 3.Sơ đồ DFD mức 0 17](#_Toc184370629)

[Hình 3. 4.Sơ đồ Use Case Diagram 18](#_Toc184370630)

[Hình 3. 5.Sơ đồ Class Diagram 19](#_Toc184370631)

[Hình 3. 6. Sơ đồ Sequence Diagram 20](#_Toc184370632)

[Hình 3. 7.Sơ đồ ERD mức quan niệm 21](#_Toc184370633)

[Hình 3. 8.Sơ đồ ERD mức vật lý 21](#_Toc184370634)

[Hình 3. 9.Sơ đồ ERD mức logic 22](#_Toc184370635)

**Chương 1. TỔNG QUAN VỀ ĐỀ TÀI**

**1.1.GIỚI THIỆU ĐỀ TÀI**

Trong bối cảnh công nghệ thông tin ngày càng phát triển, việc xử lý và phân tích dữ liệu ngày càng trở nên quan trọng. Một trong những lĩnh vực ứng dụng rộng rãi của việc xử lý dữ liệu là phân tích dữ liệu thời tiết. Dữ liệu thời tiết bao gồm các thông số như nhiệt độ, độ ẩm, tốc độ gió và áp suất, được thu thập từ nhiều nguồn khác nhau, như các trạm khí tượng hoặc API thời tiết. Tuy nhiên, khi có một lượng lớn dữ liệu, việc sắp xếp và phân loại dữ liệu trở thành một vấn đề quan trọng, giúp người dùng dễ dàng truy xuất và phân tích thông tin.

Trong đề tài này, chúng tôi phát triển một ứng dụng sử dụng hai thuật toán sắp xếp phổ biến: Heap Sort và Insertion Sort để sắp xếp các dữ liệu thời tiết theo các tiêu chí như nhiệt độ, độ ẩm, hoặc tốc độ gió. Ứng dụng không chỉ giúp sắp xếp dữ liệu mà còn cung cấp một giao diện người dùng trực quan, giúp người dùng dễ dàng theo dõi và hiểu cách thức các thuật toán sắp xếp hoạt động.

**1.2.MỤC TIÊU ĐỀ TÀI**

Mục tiêu của đề tài là phát triển một ứng dụng giúp người dùng thu thập, xử lý và sắp xếp dữ liệu thời tiết từ các nguồn khác nhau, sử dụng hai thuật toán sắp xếp phổ biến là Heap Sort và Insertion Sort. Ứng dụng này không chỉ cung cấp khả năng sắp xếp dữ liệu thời tiết theo các tiêu chí như nhiệt độ, độ ẩm, tốc độ gió mà còn hiển thị kết quả sắp xếp một cách trực quan, giúp người dùng dễ dàng theo dõi quá trình sắp xếp và đánh giá hiệu quả của từng thuật toán. Mục tiêu cuối cùng là giúp người dùng hiểu rõ hơn về cách thức hoạt động của các thuật toán sắp xếp thông qua minh họa trực quan, đồng thời nâng cao khả năng phân tích và xử lý dữ liệu trong các ứng dụng thực tế.

**1.3. PYTHON VÀ THƯ VIỆN PYQT6**

**1.3.1. Giới thiệu về Python**

Python là một ngôn ngữ lập trình mạnh mẽ và linh hoạt, được phát triển bởi Guido van Rossum và lần đầu tiên phát hành vào năm 1991. Python được thiết kế với mục tiêu đơn giản hóa quá trình lập trình, giúp lập trình viên dễ dàng đọc và viết mã nguồn. Với cú pháp rõ ràng và dễ hiểu, Python trở thành lựa chọn phổ biến cho người mới bắt đầu cũng như những lập trình viên có kinh nghiệm. Ngôn ngữ này hỗ trợ nhiều mô hình lập trình như lập trình hướng đối tượng, lập trình thủ tục và lập trình hàm, giúp phát triển ứng dụng nhanh chóng và hiệu quả.

Ảnh có chứa văn bản, phần mềm, ảnh chụp màn hình

Mô tả được tạo tự động

Hình 1. 1. Giới thiệu về Python

**1.3.2. Giới thiệu về thư viện PyQt6**

PyQt5 là một thư viện phần mềm mạnh mẽ dùng để phát triển các ứng dụng đồ họa trên nền tảng Python. PyQt6 là một bộ công cụ giao diện người dùng (UI) cho phép xây dựng ứng dụng với các cửa sổ, widget và các thành phần giao diện đồ họa khác. Thư viện này được xây dựng dựa trên Qt, một framework phát triển phần mềm phổ biến trong việc xây dựng ứng dụng đồ họa, và đã trở thành lựa chọn phổ biến cho việc phát triển ứng dụng giao diện người dùng trong Python.

Ảnh có chứa phần mềm, văn bản, Phần mềm đa phương tiện, Biểu tượng máy tính

Mô tả được tạo tự động

Hình 1. 2.Giới thiệu về thư viện PyQt6

**Chương 2. CƠ SỞ LÝ THUYẾT**

**2.1. GIỚI THIỆU CHUNG VỀ THUẬT TOÁN**

**2.1.1. Thuật toán sắp xếp**

Sắp xếp là một quá trình quan trọng trong lập trình và khoa học máy tính, nhằm sắp xếp các phần tử trong một dãy số theo một thứ tự nhất định. Mục tiêu chính của việc sắp xếp là tạo ra một trình tự có thứ tự từ dữ liệu không có trình tự hoặc có thứ tự không đúng.

**2.1.2. Một số thuật toán sắp xếp phổ biến**

* Sắp xếp vun đống (Heap Sort): là thuật toán sắp xếp sử dụng cấu trúc dữ liệu heap (cây nhị phân). Thuật toán xây dựng heap từ dãy dữ liệu và sau đó sắp xếp bằng cách lấy phần tử lớn nhất (hoặc nhỏ nhất) ra khỏi heap và đưa vào đúng vị trí trong dãy.
* Sắp xếp chèn (Insertion Sort): Thuật toán so sánh các phần tử để quyết định vị trí của một phần tử trong mảng đã được sắp xếp một phần.
* Sắp xếp lựa chọn (Selection Sort): Thuật toán so sánh các phần tử để đặt các phần tử nhỏ nhất vào vị trí phía trước.
* Sắp xếp nhanh (Quick Sort): Thuật toán so sánh các phần tử của phân vùng mảng chưa được sắp xếp thành hai nửa khác nhau xung quanh giá trị pivot.
* Sắp xếp trộn (Merge Sort): Thuật toán so sánh các phần tử của hai phần tử đã sắp xếp để hợp nhất chúng thành mảng được sắp xếp cuối cùng.
* Sắp xếp nổi bọt (Bubble Sort): Thuật toán so sánh các phần tử để đặt các phần tử lớn nhất vào vị trí cuối cùng

**2.1.3. So sánh các thuật toán sắp xếp**

**Ảnh có chứa văn bản, số, Phông chữ, ảnh chụp màn hình

Mô tả được tạo tự động**

Hình 2. 1.Tổng quan về so sánh của một số thuật toán sắp xếp

**2.2. GIỚI THIỆU VỀ THUẬT TOÁN HEAP SORT**

**2.2.1. Giới thiệu về thuật toán**

Heap Sort là một thuật toán sắp xếp được phát triển bởi J. W. J. Williams vào năm 1964. Thuật toán này sử dụng cấu trúc dữ liệu cây nhị phân heap để thực hiện sắp xếp.

Heap Sort cũng dựa trên nguyên lý **“**chia để trị**”** nhưng thực hiện theo cách khác với Merge Sort. Thuật toán này bắt đầu bằng việc xây dựng một heap (có thể là max-heap hoặc min-heap) từ dãy dữ liệu. Sau đó, phần tử lớn nhất (hoặc nhỏ nhất) trong heap sẽ được di chuyển vào cuối dãy và quá trình điều chỉnh lại heap (heapify) tiếp tục cho đến khi toàn bộ dãy được sắp xếp.

Heap Sort có ưu điểm là không yêu cầu bộ nhớ phụ (sắp xếp tại chỗ) và có độ phức tạp thời gian O(n log n) trong cả trường hợp xấu nhất. Tuy nhiên, thuật toán này không đảm bảo tính ổn định, có nghĩa là nếu có hai phần tử có giá trị bằng nhau, thứ tự của chúng trong dãy ban đầu có thể thay đổi sau khi sắp xếp.

**2.2.2. Ý tưởng về thuật toán**

Thuật toán Heap Sort sử dụng cấu trúc dữ liệu heap (cây nhị phân) để sắp xếp dãy số. Ý tưởng chính của thuật toán này bao gồm các bước sau:

* Xây dựng heap (Build heap): Ban đầu, thuật toán xây dựng một cây heap từ dãy số. Cây heap có thể là một max-heap (cây nhị phân với phần tử lớn nhất ở gốc) hoặc min-heap (cây nhị phân với phần tử nhỏ nhất ở gốc). Với max-heap, phần tử lớn nhất sẽ được đặt ở đầu dãy số.
* Lấy phần tử lớn nhất (Extract maximum): Sau khi xây dựng heap, thuật toán lấy phần tử lớn nhất (đối với max-heap) từ gốc của heap và di chuyển nó về cuối dãy. Sau đó, thuật toán điều chỉnh lại heap (thực hiện thao tác heapify) để đảm bảo cấu trúc heap vẫn đúng.
* Lặp lại (Repeat): Quy trình trên sẽ được lặp lại cho đến khi tất cả các phần tử được di chuyển về cuối dãy, và dãy số được sắp xếp theo thứ tự mong muốn.

Ảnh có chứa biểu đồ, vòng tròn, màu cam, ảnh chụp màn hình

Mô tả được tạo tự động

Hình 2. 2.Ví dụ về Max Heap và Min Heap

**Ảnh có chứa màu cam, thiết kế

Mô tả được tạo tự động**

Hình 2. 3.Ví dụ minh họa cho thuật toán Heap Sort

**2.3. GIỚI THIỆU THUẬT TOÁN INSRTION**

**2.3.1. Giới thiệu về thuật toán**

Insertion Sort là một thuật toán sắp xếp đơn giản và hiệu quả đối với các danh sách nhỏ. Thuật toán này hoạt động theo cách mô phỏng quá trình sắp xếp bài: mỗi phần tử trong dãy sẽ được so sánh và chèn vào vị trí thích hợp trong dãy đã được sắp xếp trước đó.

Ý tưởng chính của thuật toán là duy trì một phần của dãy (phần đầu) luôn ở trạng thái đã sắp xếp. Mỗi phần tử mới sẽ được chèn vào đúng vị trí trong phần đã sắp xếp, mở rộng vùng sắp xếp dần đến khi toàn bộ dãy được xử lý.

Insertion Sort có độ phức tạp thời gian O(n²) trong trường hợp xấu nhất (dãy ngược thứ tự) và O(n) trong trường hợp tốt nhất (dãy đã sắp xếp). Thuật toán có ưu điểm là dễ triển khai, phù hợp cho dãy có kích thước nhỏ, và đảm bảo tính ổn định: thứ tự tương đối của các phần tử có giá trị bằng nhau trong dãy gốc được giữ nguyên sau khi sắp xếp. Tuy nhiên, với dãy lớn, hiệu suất của thuật toán không cao so với các thuật toán sắp xếp tiên tiến hơn như Merge Sort hay Heap Sort.

**2.3.2. Ý tưởng về thuật toán**

Thuật toán Insertion Sort dựa trên việc sắp xếp từng phần tử của dãy số giống như cách con người sắp xếp các lá bài trên tay. Ý tưởng chính bao gồm các bước sau:

* Duyệt qua từng phần tử trong dãy: Bắt đầu từ phần tử thứ hai (vì phần tử đầu tiên mặc định đã được xem là "đã sắp xếp").
* Chọn và chèn phần tử: Với mỗi phần tử chưa sắp xếp, thuật toán sẽ tìm vị trí thích hợp trong phần đã sắp xếp để chèn phần tử đó vào đúng vị trí.
* Dịch chuyển phần tử: Trong quá trình tìm đúng vị trí, các phần tử lớn hơn phần tử hiện tại sẽ được dịch chuyển sang bên phải để tạo khoảng trống cho phần tử cần chèn.

Quy trình lặp lại: Thuật toán tiếp tục duyệt qua tất cả các phần tử chưa sắp xếp và lặp lại thao tác trên cho đến khi toàn bộ dãy đã được sắp xếp theo thứ tự mong muốn.

**2.3.3. Lưu đồ thuật toán**

**Ảnh có chứa biểu đồ, bản phác thảo, Bản vẽ kỹ thuật, Kế hoạch

Mô tả được tạo tự động**

Hình 2. 4. Lưu đồ thuật toán

**Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động**

Hình 2. 5.Ví dụ minh họa

**Chương 3. KẾT QUẢ NGHIÊN CỨU**

**3.1. SƠ ĐỒ PHÂN RÃ CHỨC NĂNG (FDD)**

**Ảnh có chứa văn bản, hàng, biểu đồ, biên lai

Mô tả được tạo tự động**

Hình 3. 1.Sơ đồ phân rã FDD

* **Nhận dữ liệu và yêu cầu từ người dùng**
* Chức năng đầu tiên của ứng dụng là thu thập dữ liệu đầu vào từ người dùng. Ứng dụng cung cấp ba cách để người dùng nhập dữ liệu thời tiết. Người dùng có thể nhập dữ liệu thủ công bằng cách cung cấp từng giá trị trực tiếp qua giao diện, tải lên file chứa thông tin dữ liệu thời tiết, hoặc lấy dữ liệu thời gian thực từ API thời tiết như OpenWeatherMap. Mỗi phương pháp đều đảm bảo rằng ứng dụng có thể tiếp nhận dữ liệu phù hợp để xử lý.
* **Xử lý và sắp xếp dữ liệu**
* Sau khi nhận được dữ liệu, ứng dụng tiến hành xử lý và sắp xếp. Đầu tiên, dữ liệu được chuẩn hóa để đảm bảo tính hợp lệ, loại bỏ các giá trị không phù hợp. Tiếp theo, ứng dụng sử dụng các thuật toán sắp xếp như Heap Sort hoặc Insertion Sort để sắp xếp dữ liệu theo thứ tự mong muốn. Sau khi sắp xếp, ứng dụng kiểm tra lại kết quả để đảm bảo rằng dữ liệu đã được xử lý chính xác.
* **Hiển thị kết quả và lưu trữ**
* Kết quả sắp xếp được hiển thị trực quan trên giao diện để người dùng dễ dàng theo dõi. Ứng dụng cũng cung cấp tùy chọn xuất kết quả chi tiết dưới dạng báo cáo hoặc trực tiếp hiển thị trên giao diện. Ngoài ra, ứng dụng lưu trữ lịch sử sắp xếp để phục vụ nhu cầu truy xuất dữ liệu hoặc phân tích trong tương lai. Việc này giúp người dùng quản lý dữ liệu dễ dàng và hiệu quả hơn.

Chú thích: vì hình họa có thể nhỏ và khó nhìn, cô có thể xem trực tiếp trên ứng dụng web draw.io bằng file: FDD.drawiođính kèm

**3.2. SƠ ĐỒ LUÔNG DỮ LIỆU (DFD)**

**3.2.1. Sơ đồ mức ngữ cảnh**

**Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, hàng

Mô tả được tạo tự động**

Hình 3. 2.Sơ đồ mức ngữ cảnh

Hình ảnh trên là sơ đồ mức ngữ cảnh, thể hiện tổng quan hoạt động của hệ thống sắp xếp dữ liệu thời tiết. Sơ đồ này mô tả cách hệ thống tương tác với các thực thể bên ngoài và luồng dữ liệu chính giữa chúng.

Người dùng: Đây là một trong những thực thể bên ngoài của hệ thống. Người dùng gửi yêu cầu thông qua giao diện để hệ thống cung cấp thông tin thời tiết cụ thể. Sau khi xử lý, hệ thống trả về kết quả dưới dạng dữ liệu đã được sắp xếp và tổ chức theo nhu cầu của người dùng.

**API thời tiết**: Là nguồn dữ liệu đầu vào cho hệ thống. API thời tiết cung cấp dữ liệu thô về điều kiện thời tiết, đáp ứng các yêu cầu từ hệ thống. Dữ liệu này sau đó được truyền vào để xử lý và sắp xếp.

**Hệ thống giải thuật sắp xếp dữ liệu:** Đây là trung tâm xử lý của sơ đồ. Hệ thống nhận dữ liệu đầu vào từ hai nguồn chính:

**Từ người dùng**: Yêu cầu cụ thể liên quan đến thông tin thời tiết.

**Từ API thời tiết**: Dữ liệu thời tiết thô để xử lý. Hệ thống sử dụng các thuật toán để sắp xếp, tổ chức dữ liệu phù hợp với yêu cầu của người dùng. Kết quả được trả lại dưới dạng dữ liệu đã được sắp xếp và tối ưu.

**Luồng dữ liệu**: Các mũi tên trong sơ đồ biểu thị luồng dữ liệu giữa các thành phần. Dữ liệu yêu cầu và dữ liệu thời tiết thô được hệ thống xử lý, sau đó tạo ra kết quả cuối cùng để trả về người dùng.

Sơ đồ mức ngữ cảnh này cung cấp cái nhìn tổng quan, giúp xác định rõ ràng các thành phần chính, chức năng của hệ thống, và mối liên hệ giữa hệ thống với các tác nhân bên ngoài. Điều này là nền tảng để thiết kế chi tiết hơn ở các mức tiếp theo trong quá trình phân tích hệ thống.  
Chú thích: vì hình họa có thể nhỏ và khó nhìn, cô có thể xem trực tiếp trên ứng dụng powerdesinger bằng file: dfd\_ngu\_canh.bpm đính kèm.

**3.2.2. Sơ đồ mức 0**

**Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, Phông chữ

Mô tả được tạo tự động**

Hình 3. 3.Sơ đồ DFD mức 0

Hình ảnh này là sơ đồ luồng dữ liệu (DFD) mức 0, biểu diễn cách một hệ thống xử lý dữ liệu. Trong sơ đồ:

Người dùng và API thời tiết là các nguồn cung cấp và nhận dữ liệu từ hệ thống.

Quy trình bắt đầu với việc nhận dữ liệu và yêu cầu từ người dùng (quy trình số 1).

Sau đó, dữ liệu được xử lý và sắp xếp trong quy trình 2 để tạo ra dữ liệu thời tiết đã xử lý.

Cuối cùng, kết quả được hiển thị và lưu trữ trong cơ sở dữ liệu thông qua quy trình 3.

Cơ sở dữ liệu đóng vai trò là nơi lưu trữ dữ liệu phục vụ cho các lần truy cập sau.

Luồng thông tin giữa các thực thể, quy trình, và cơ sở dữ liệu được thể hiện rõ ràng qua các mũi tên.

Chú thích: vì hình họa có thể nhỏ và khó nhìn, cô có thể xem trực tiếp trên ứng dụng powerdesinger bằng file: Data Flow Diagram\_1.bpm đính kèm

**3.3. SƠ ĐỒ UML**

**3.3.1. Sơ đồ Use Case Diagram (Biểu đồ trường hợp sử dụng)**

**Ảnh có chứa biểu đồ, ảnh chụp màn hình, vòng tròn, thiết kế

Mô tả được tạo tự động**

Hình 3. 4.Sơ đồ Use Case Diagram

Mô tả: Đây là sơ đồ trường hợp sử dụng, thể hiện các chức năng chính mà hệ thống cung cấp cho người dùng hoặc API. Các thành phần chính: Người dùng: Tương tác với hệ thống để thực hiện các chức năng như nhập dữ liệu, xem kết quả, kiểm tra thời tiết. API: Là một tác nhân ngoài, đóng vai trò lấy dữ liệu thời tiết từ nguồn bên ngoài. Các trường hợp sử dụng (Use Cases): Bao gồm kiểm tra dữ liệu, xử lý dữ liệu, lấy thông tin lịch sử, và hiển thị dữ liệu. Quan hệ: • Quan hệ bao gồm <include> (thực thi một chức năng khác như một phần của chức năng hiện tại) giữa các chức năng.

Chú thích: vì hình họa có thể nhỏ và khó nhìn, cô có thể xem trực tiếp trên ứng dụng powerdesinger bằng file: Use Case Diagram đính kèm

**3.3.2. Sơ đồ Class Diagram (Biểu đồ Lớp)**

**Ảnh có chứa văn bản, biểu đồ, bản phác thảo, Bản vẽ kỹ thuật

Mô tả được tạo tự động**

Hình 3. 5.Sơ đồ Class Diagram

Đây là sơ đồ lớp (Class Diagram), mô tả cấu trúc logic của các lớp trong hệ thống xử lý và sắp xếp dữ liệu thời tiết. Sơ đồ gồm nhiều lớp chính, mỗi lớp đảm nhiệm một vai trò cụ thể để đảm bảo hoạt động của hệ thống.

**Lớp User:** lưu trữ thông tin người dùng, bao gồm ID và tên, đồng thời cung cấp các phương thức quản lý người dùng. Đây là lớp giúp xác định và phân loại người dùng trong hệ thống.

**Lớp WeatherData:** chịu trách nhiệm lưu trữ các thông tin thời tiết, chẳng hạn như nhiệt độ, độ ẩm, tốc độ gió, và các dữ liệu liên quan khác. Lớp này cũng tích hợp các phương thức xử lý dữ liệu để chuẩn bị cho quá trình sắp xếp.

**Lớp SortAlgorithm:** đảm nhận nhiệm vụ thực hiện các thuật toán sắp xếp dữ liệu thời tiết. Các thuật toán như Heap Sort hoặc Insertion Sort được triển khai trong lớp này để tổ chức dữ liệu theo cách hiệu quả nhất.

**Lớp Storage:** được sử dụng để quản lý lịch sử dữ liệu đã xử lý. Lớp này cung cấp các phương thức lưu trữ kết quả và hiển thị lại thông tin khi cần thiết, đảm bảo tính liên tục và khả năng truy xuất dữ liệu.

**Lớp Display:** đóng vai trò hiển thị thông tin và kết quả xử lý cho người dùng. Đây là nơi người dùng có thể tương tác với hệ thống, xem dữ liệu sắp xếp hoặc kết quả cuối cùng.

Các lớp này được liên kết với nhau thông qua quan hệ kết hợp (Association), cho phép chúng phối hợp chặt chẽ trong việc thực hiện các chức năng chung. Điều này đảm bảo hệ thống hoạt động trơn tru và hiệu quả, đáp ứng các yêu cầu của người dùng.

Chú thích: vì hình họa có thể nhỏ và khó nhìn, cô có thể xem trực tiếp trên ứng dụng powerdesinger bằng file: UMl class đính kèm

**3.3.3. Sơ đồ Sequence Diagram**

**Ảnh có chứa văn bản, biểu đồ, ảnh chụp màn hình, Song song

Mô tả được tạo tự động**

Hình 3. 6. Sơ đồ Sequence Diagram

Mô tả: Đây là sơ đồ trình tự (Sequence Diagram), mô tả luồng thông tin giữa các đối tượng trong hệ thống theo trình tự thời gian. Sơ đồ tập trung vào cách các đối tượng tương tác với nhau để hoàn thành các yêu cầu của người dùng.

Các đối tượng chính:

* User: Đại diện cho người dùng của hệ thống.
* API: Thành phần chịu trách nhiệm lấy dữ liệu thời tiết từ nguồn bên ngoài.
* WeatherData: Quản lý và xử lý dữ liệu thời tiết nhận được.
* SortAlgorithm: Thực hiện các thuật toán sắp xếp dữ liệu.
* Storage: Lưu trữ dữ liệu đã xử lý và sắp xếp.
* Display: Hiển thị kết quả cho người dùng.

Trình tự hoạt động:

1. **User**: Người dùng bắt đầu bằng cách nhập dữ liệu hoặc yêu cầu thông qua giao diện hệ thống (gửi yêu cầu đến Display).
2. **API**: Hệ thống gọi API để lấy dữ liệu thời tiết từ nguồn bên ngoài, ví dụ như OpenWeatherMap.
3. **WeatherData**: Dữ liệu nhận được từ API được chuyển đến lớp WeatherData để xử lý và chuẩn hóa.
4. **SortAlgorithm**: Sau khi xử lý, dữ liệu được gửi đến lớp SortAlgorithm, nơi áp dụng các thuật toán sắp xếp như Heap Sort hoặc Insertion Sort.
5. **Storage**: Dữ liệu đã sắp xếp được lưu trữ vào lớp Storage để phục vụ các yêu cầu truy xuất sau này.
6. **Display:** Cuối cùng, kết quả được chuyển đến lớp Display để hiển thị lên giao diện cho người dùng xem.

Quá trình này đảm bảo hệ thống thực hiện đầy đủ các chức năng từ nhận yêu cầu, xử lý dữ liệu, sắp xếp, lưu trữ đến hiển thị kết quả.

Chú thích: vì hình họa có thể nhỏ và khó nhìn, cô có thể xem trực tiếp trên ứng dụng powerdesinger bằng file: Sequence diagram đính kèm

**3.4 SƠ ĐỒ THỰC THỂ KẾT HỢP (ERD)**

**3.4.1. Sơ đồ ERD mức quan niệm**

**Ảnh có chứa văn bản, Phông chữ

Mô tả được tạo tự động**

Hình 3. 7.Sơ đồ ERD mức quan niệm

Chú thích: vì hình họa có thể nhỏ và khó nhìn, cô có thể xem trực tiếp trên ứng dụng powerdesinger bằng file: Conceptual Data\_1.cdm đính kèm

**3.4.2. Sơ đồ ERD mức vật lý**

**Ảnh có chứa văn bản, ảnh chụp màn hình, hàng, biểu đồ

Mô tả được tạo tự động**

Hình 3. 8.Sơ đồ ERD mức vật lý

Sơ Đồ Quan Hệ Thực Thể (Entity-Relationship Diagram - ERD) ở mức quan niệm. Nó bao gồm ba thực thể chính: WeatherData, SortAlgorithm, và UserRequest, cùng với các thuộc tính và mối quan hệ của chúng. Dưới đây là phần giải thích chi tiết:

1. Thực Thể WeatherData

* Thuộc tính (Attributes):
  + WeatherDataID: Mã định danh của dữ liệu thời tiết (kiểu số nguyên)
  + Temperature: Nhiệt độ (kiểu số nguyên)
  + Humidity: Độ ẩm (kiểu số thập phân)
  + Wind\_Speed: Tốc độ gió (kiểu số thập phân)

2. Thực Thể UserRequest

* Thuộc tính (Attributes):
  + RequestID: Mã định danh của yêu cầu (kiểu số nguyên)
  + TimeStamp: Dấu thời gian (ngày và giờ)

3. Thực Thể SortAlgorithm

* Thuộc tính (Attributes):
  + AlgorithmID: Mã định danh của thuật toán (kiểu số nguyên)
  + AlgorithmName: Tên của thuật toán (kiểu ký tự biến đổi tối đa 100 ký tự)
  + ExecutionTime: Thời gian thực hiện (kiểu số thập phân)

Các Mối Quan Hệ

* Handle: Thực thể này kết nối giữa WeatherData và SortAlgorithm.
* Request: Mối quan hệ này kết nối thực thể UserRequest với WeatherData.

Sơ đồ này mô tả cấu trúc quan niệm của hệ thống cơ sở dữ liệu, cho thấy cách các thực thể khác nhau liên quan và các thuộc tính chúng chứa. Điều này giúp hiểu rõ cách dữ liệu được quản lý và tương tác trong hệ thống.

Chú thích: vì hình họa có thể nhỏ và khó nhìn, cô có thể xem trực tiếp trên ứng dụng powerdesinger bằng file: Physic1.pdm đính kèm

**3.4.3. Sơ đồ ERD mức logic**

**Ảnh có chứa văn bản, Phông chữ, Giấy nhớ, ảnh chụp màn hình

Mô tả được tạo tự động**

Hình 3. 9.Sơ đồ ERD mức logic

Sơ Đồ Quan Hệ Thực Thể (Entity-Relationship Diagram - ERD) ở mức vật lý cho hệ thống xử lý dữ liệu thời tiết. Dưới đây là phần giải thích chi tiết về các thành phần chính trong sơ đồ:

1. Thực Thể WeatherData

* Thuộc tính (Attributes):
  + WeatherDataID: Mã định danh của dữ liệu thời tiết (kiểu số nguyên)
  + RequestID: Mã định danh của yêu cầu (kiểu số nguyên)
  + Temperature: Nhiệt độ (kiểu số nguyên)
  + Humidity: Độ ẩm (kiểu số thập phân)
  + Wind\_Speed: Tốc độ gió (kiểu số thập phân)

2. Thực Thể UserRequest

* Thuộc tính (Attributes):
  + RequestID: Mã định danh của yêu cầu (kiểu số nguyên)
  + TimeStamp: Dấu thời gian (ngày và giờ)

3. Thực Thể Handle

* Thuộc tính (Attributes):
  + AlgorithmID: Mã định danh của thuật toán (kiểu số nguyên)
  + WeatherDataID: Mã định danh của dữ liệu thời tiết (kiểu số nguyên)

4. Thực Thể SortAlgorithm

* Thuộc tính (Attributes):
  + AlgorithmID: Mã định danh của thuật toán (kiểu số nguyên)
  + AlgorithmName: Tên của thuật toán (kiểu ký tự biến đổi tối đa 100 ký tự)
  + ExecutionTime: Thời gian thực hiện (kiểu số thập phân)

Các Mối Quan Hệ

* Handle: Thực thể này kết nối giữa WeatherData và SortAlgorithm.
* Request: Mối quan hệ này kết nối thực thể UserRequest với Handle.

Sơ đồ này mô tả cấu trúc vật lý của một hệ thống cơ sở dữ liệu, cho thấy cách các thực thể khác nhau liên quan và các thuộc tính chúng chứa. Điều này giúp hiểu rõ cách dữ liệu được quản lý và tương tác trong hệ thống.

Chú thích: vì hình họa có thể nhỏ và khó nhìn, cô có thể xem trực tiếp trên ứng dụng powerdesinger bằng file: Logic1.Ild đính kèm.

**CHƯƠNG 4. THIẾT KÊ GIAO DIỆN VÀ CÀI ĐẶT**

# 4.1. THIẾT KẾ GIAO DIỆN

**4.1.1. Giao diện chính của của chương trình**

# Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số Mô tả được tạo tự động

**Hình 4.1 Giao diện chính của chương trình**

# Tại đây người dùng có thể lấy dữ liệu trực tiếp từ API trên OpenWeatherMap, lấy dữ liệu từ file có sẵn (file json). Hoặc nhập tay tùy vào kiểu dữ liệu người dùng muốn nhập dữ liệu bằng 3 cách trên. Sau khi đã chọn được kiểu dữ liệu để nhập vào rồi thì người dùng bắt đầu nhập dữ liệu theo ý muốn.

Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, số

Mô tả được tạo tự động

# Hình 4.2. Giao diện khi người dùng muốn nhập dữ liệu bằng cách lấy trực tiếp từ API

Khi người dùng muốn nhập dữ liệu bằng cách lấy trực tiếp từ API thì chỉ cần nhập tên thành phố cần lấy sau đó thì bên cột nhiệt độ, độ ẩm, tốc độ gió độ ẩm sẽ hiện thị và khi người dùng nhấn vào hiển thị kết quả lên bảng thì dữ liệu sẽ được trực tiếp gửi lên bảng.

# Ảnh có chứa văn bản, ảnh chụp màn hình, số, Phông chữ Mô tả được tạo tự động

# Hình 4.3. Giao diện người dùng muốn nhập dữ liệu bằng cách từ file

Khi người dùng nhấn vào chọn lấy dữ liệu từ file có sẵn xong rồi nhấn vào hiển thị lên kết quả như trên bảng của trên giao diện ( Lưu ý chỉ có thể đọc file json).

# Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số Mô tả được tạo tự động

# Hình 4.4 Giao diện người dùng muốn nhập dữ liệu bằng tay

Tại đây người dùng có thể nhập trực tiếp bằng tay vào các cột của thành phố đó tại đây có ràng buộc về dữ liệu đối với nhiệt độ thì từ -50 đến 50 độ C, tốc độ gió từ 0 đến 100m/s và độ ẩm thì từ 0 đến 100%. Đảm bảo cho việc đáp ứng thực tế cho các nguồn dữ liệu này.

# Ảnh có chứa văn bản, ảnh chụp màn hình, số, Phông chữ Mô tả được tạo tự động

# Hình 4.5. Giao diện người dùng muốn chọn sắp xếp theo nhiệt độ

Tại đây người dùng có thể chọn cách sắp xếp dữ liệu thời tiết theo nhiệt độ hay là tốc độ gió độ ẩm tùy thuộc vào như cầu của người dùng và sắp xếp cách sắp xếp.

# Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số Mô tả được tạo tự động

# Hình 4.6. Giao diện người dùng muốn chọn thuật toán sắp xếp

Ở đây người dùng có thể chọn 2 thuật đoán đó là Insertion Sort và Heap Sort cho việc sắp xếp thuật toán của mình.

# Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số Mô tả được tạo tự động

# Hình 4.7. Giao diện người dùng khi nhấn vào sắp xếp theo nhiệt độ Insertion Sort

Tại đây ta có thể thấy rằng sau khi người dùng chọn xong cách sắp xếp và thuật toán và nhấn sắp xếp thì bảng kết quả dữ liệu đã được sắp xếp theo yêu cầu của người dùng và hiển thị thời gian của thuật toán sau khi sắp xếp xong.

# Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Biểu tượng máy tính Mô tả được tạo tự động

# Hình 4.9 Giao diện người dùng muốn chọn sắp xếp theo tốc độ gió Insertion Sort

Ảnh có chứa văn bản, Phông chữ, số, phần mềm

Mô tả được tạo tự động

# Hình 4.10 Giao diện người dùng muốn chọn sắp xếp theo độ ẩm Insertion Sort

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

# Hình 4.11. Giao diện người dùng muốn chọn sắp xếp theo nhiệt độ Heap Sort

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

# Hình 4.12. Giao diện người dùng muốn chọn sắp xếp theo tốc độ gió Heap Sort

# Ảnh có chứa văn bản, phần mềm, Phông chữ, Biểu tượng máy tính Mô tả được tạo tự động Hình 4.13. Giao diện người dùng muốn chọn sắp xếp theo tốc độ ẩm Heap Sort

# Ảnh có chứa văn bản, ảnh chụp màn hình, số, Phông chữ Mô tả được tạo tự động

# Hình 4.13 Giao diện khi người dùng xóa 1 hàng đang chọn

Khi người dùng muốn xóa 1 hàng đang chọn nào đó thì người dùng có thể click vào hàng mình đang chọn sau đó ấn vào button xóa hàng đang chọn thì ngay lập từ hàng được chọn đó sẽ bị loại bỏ ngay tức khắc.

# Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Phông chữ Mô tả được tạo tự động

# Hình 4.9 Giao diện khi người dùng xóa tất cả các hàng

Sau khi nhấn vào thì tất cả dữ liệu được hiển thị lên bảng đều đã được xóa ngay tức thì.

# Ảnh có chứa văn bản, ảnh chụp màn hình, số, Phông chữ Mô tả được tạo tự động

# Hình 4.10. Giao diện người dùng khi chọn mô phỏng thuật toán trực quan

Sau khi ấn vào mô phỏng thì sẽ xuất hiện thêm 1 giao diện mới cho việc mô phỏng trực quan thuật toán sắp xếp dữ liệu thời tiết

# Ảnh có chứa ảnh chụp màn hình, văn bản, biểu đồ, Sơ đồ Mô tả được tạo tự động

# Hình 4.11. Giao diện khi người dùng mô phỏng trực quan thuật toán sắp xếp chèn

Tại đây người dùng có thể thấy được các cột là các dữ liệu thời tiết là biểu đồ cột bên tay phải màn hình người dùng có cách hoạt động về thuật toán và nút bắt đầu sắp xếp có thanh điều chỉnh tốc độ có thời gian khi mà thuậttoán sắp xếp xong, có nút chọn tăng dần và giảm dần cho phép người dùng lựa chọn phù hợp với yêu cầu người dùng.

# Ảnh có chứa ảnh chụp màn hình, văn bản, biểu đồ, Sơ đồ Mô tả được tạo tự động

# Hình 4.12. Giao diện khi sắp xếp thuật toán theo chèn thứ tự tăng dần

Ảnh có chứa ảnh chụp màn hình, văn bản, biểu đồ, Sơ đồ

Mô tả được tạo tự động

# Hình 4.13. Giao diện khi sắp xếp thuật toán chèn theo thứ tự giảm dần

Tương tự với sắp xếp chèn thì sắp xếp vun đống cũng như vậy nhưng trực quan về sắp xếp vun đống là sắp xếp bằng cây nhị phân dựa vào Max heap và Min heap.

# Ảnh có chứa ảnh chụp màn hình, biểu đồ, hàng, văn bản Mô tả được tạo tự động

# Hình 4.14. Giao diện khi người dùng mô phỏng trực quan thuật toán sắp xếp vun đống

Tại đây cũng giống như giao diện mô phỏng trực quan về thuật toán Insertion đều có tại đây người dùng có thể thấy được các cột là các dữ liệu thời tiết là dạng cây nhị phân khi chưa sắp xếp bên tay phải màn hình người dùng có cách hoạt động về thuật toán và nút bắt đầu sắp xếp có thanh điều chỉnh tốc độ có thời gian khi mà thuật toán sắp xếp xong, có nút chọn tăng dần và giảm dần cho phép người dùng lựa chọn phù hợp với yêu cầu người dùng.

# Ảnh có chứa ảnh chụp màn hình, biểu đồ, hàng Mô tả được tạo tự động

# Hình 4.15. Giao diện khi sắp xếp thuật toán vun đống thứ tự tăng dần

Sau khi sắp xếp xong thì có thời gian hiển thị bên dưới tương tự như vậy đối với sắp xếp giảm dần.

# Ảnh có chứa ảnh chụp màn hình, biểu đồ, hàng Mô tả được tạo tự động

# Hình 4.15 Giao diện khi sắp xếp thuật toán vun đống thứ tự giảm dần

Sau khi sắp xếp xong thì người dùng có thể thấy tất cả dữ liệu được lấy hiển thị lên ứng dụng đều được sắp xếp 1 cách trực quan và chi tiết.

# 4.2. CÀI ĐẶT CHƯƠNG TRÌNH

**4.2.1 Main.py**

Chương trình sử dụng PyQt6 để xây dựng một ứng dụng quản lý dữ liệu thời tiết, cho phép người dùng nhập dữ liệu thành phố và thông số thời tiết (nhiệt độ, tốc độ gió, độ ẩm) từ các nguồn khác nhau như nhập tay, API, hoặc file JSON. Giao diện người dùng bao gồm các nút chức năng để thêm, xóa, và sắp xếp dữ liệu, cùng với một bảng hiển thị các thông số thời tiết của các thành phố. Người dùng có thể chọn phương thức sắp xếp (Insertion Sort hoặc Heap Sort) và theo tiêu chí (nhiệt độ, tốc độ gió, độ ẩm), sau đó xem quá trình mô phỏng sắp xếp dữ liệu. Chương trình cũng hỗ trợ hiển thị và quản lý dữ liệu thời tiết từ API OpenWeatherMap và xử lý các lỗi như không tìm thấy thành phố hoặc sai định dạng dữ liệu, mang đến một công cụ trực quan và dễ sử dụng cho người dùng.

# 

**Hình 4.16. Khai báo thư viện và cửa sổ ứng dụng các nút**

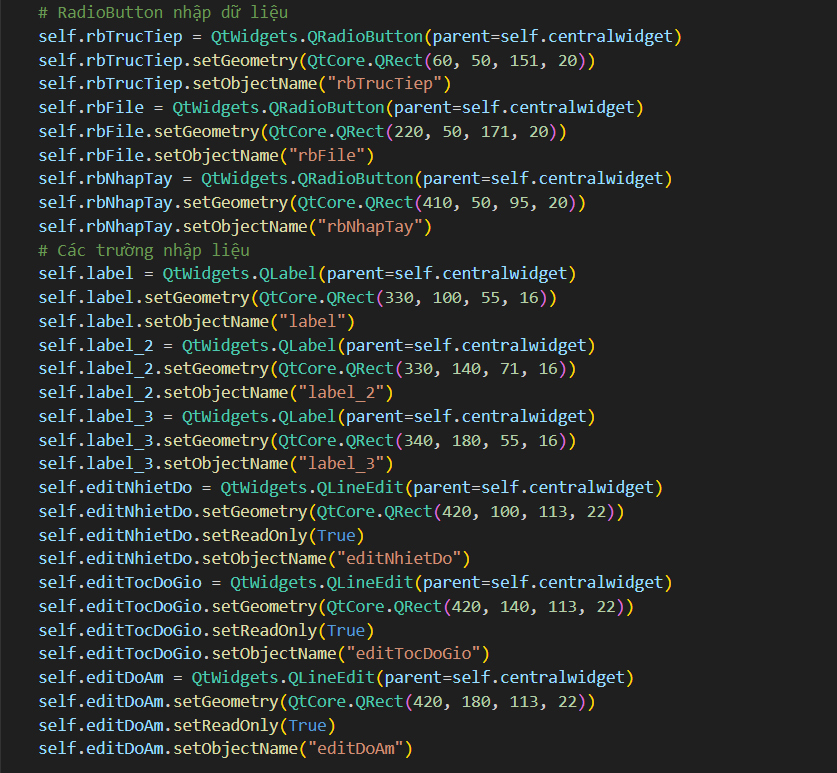
# **MainWindow:** Đây là cửa sổ chính của ứng dụng, có thể xem là khung chính chứa tất cả các widget (thành phần giao diện) của ứng dụng.

* **Centralwidget**: Là widget trung tâm của cửa sổ MainWindow, nơi tất cả các thành phần giao diện khác như nút bấm và hộp lựa chọn sẽ được đặt vào.
* **buttonDeleteRow:** Đây là nút "Xóa dòng", được tạo ra để người dùng có thể xóa một dòng trong bảng hoặc danh sách.
* **buttonDeleteAll**: Đây là nút "Xóa tất cả", cho phép người dùng xóa toàn bộ dữ liệu hoặc tất cả các dòng trong bảng/danh sách.
* **buttonVisualizationSort**: Nút này có chức năng "Hiển thị quá trình sắp xếp", giúp người dùng xem trực quan quá trình sắp xếp dữ liệu.

# **buttonSort:** Đây là nút "Sắp xếp", cho phép người dùng thực hiện việc sắp xếp dữ liệu theo một tiêu chí nhất định

* **cbTieuChi**: Đây là một hộp lựa chọn (ComboBox), cho phép người dùng chọn tiêu chí để sắp xếp dữ liệu. Người dùng có thể chọn các mục từ danh sách, mặc dù các mục hiện tại vẫn chưa được định nghĩa cụ thể.

# **cbLoaiSapXep**: Đây là một hộp lựa chọn (ComboBox) khác, cho phép người dùng chọn loại sắp xếp (tăng dần hay giảm dần, hoặc các lựa chọn khác).



# Hình 4.17 Tạo các nút nhập và các trường dữ liệu

* **rbTrucTiep**: Đây là một RadioButton cho phép người dùng lựa chọn phương thức nhập liệu "Trực tiếp".
* **rbFile:** Đây là một RadioButton khác cho phép người dùng lựa chọn phương thức nhập liệu từ "File".
* **rbNhapTay**: Đây là RadioButton cho phép người dùng lựa chọn nhập liệu "Nhập tay".
* **label**: Đây là nhãn (label) cho trường nhập liệu "Nhiệt độ".
* **label\_2**: Đây là nhãn cho trường nhập liệu "Tốc độ gió".
* **label\_3**: Đây là nhãn cho trường nhập liệu "Độ ẩm".
* **editNhietDo**: Đây là trường nhập liệu dạng LineEdit cho phép người dùng nhập giá trị "Nhiệt độ". Có nghĩa là người dùng không thể chỉnh sửa trực tiếp mà chỉ có thể xem giá trị.
* **editTocDoGio**: Đây là trường nhập liệu dạng LineEdit cho phép người dùng nhập giá trị "Tốc độ gió".
* **editDoAm**: Đây là trường nhập liệu dạng LineEdit cho phép người dùng nhập giá trị "Độ ẩm".

**Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động**

# **Hình 4.18** Tạo các nút cho giao diện

* **\_translate = QtCore.QCoreApplication.translate:** Đây là một cách thức để sử dụng cơ chế dịch ngôn ngữ của Qt, giúp chuyển đổi các chuỗi văn bản thành ngôn ngữ đã được định nghĩa trong ứng dụng. Hàm translate nhận vào hai đối số: tên của ứng dụng (ở đây là "MainWindow") và chuỗi văn bản cần dịch.
* **MainWindow.setWindowTitle**(\_translate("MainWindow", "MainWindow")): Đặt tiêu đề cửa sổ chính của ứng dụng là "MainWindow".
* **self.buttonDeleteRow.setText**(\_translate("MainWindow", "Xóa hàng đang chọn")): Đặt văn bản cho nút "Xóa hàng đang chọn".
* **self.buttonDeleteAll.setText**(\_translate("MainWindow", "Xóa tất cả hàng")): Đặt văn bản cho nút "Xóa tất cả hàng".
* **self.buttonVisualizationSort.setText**(\_translate("MainWindow", "Mô Phỏng")): Đặt văn bản cho nút "Mô Phỏng".
* **self.buttonSort.setText(\_translate("MainWindow", "Sắp xếp")):** Đặt văn bản cho nút "Sắp xếp".
* **self.cbTieuChi.setItemText**(0, \_translate("MainWindow", "Nhiệt độ")): Đặt tên cho mục đầu tiên trong ComboBox "Tiêu chí" là "Nhiệt độ".
* **self.cbTieuChi.setItemText**(1, \_translate("MainWindow", "Tốc độ gió")): Đặt tên cho mục thứ hai trong ComboBox "Tiêu chí" là "Tốc độ gió".
* **self.cbTieuChi.setItemText**(2, \_translate("MainWindow", "Độ ẩm")): Đặt tên cho mục thứ ba trong ComboBox "Tiêu chí" là "Độ ẩm".
* **self.cbLoaiSapXep.setItemText**(0, \_translate("MainWindow", "Insertion Sort")): Đặt tên cho mục đầu tiên trong ComboBox "Loại sắp xếp" là "Insertion Sort".
* **self.cbLoaiSapXep.setItemText**(1, \_translate("MainWindow", "Heap Sort")): Đặt tên cho mục thứ hai trong ComboBox "Loại sắp xếp" là "Heap Sort".
* **self.rbTrucTiep.setText**(\_translate("MainWindow", "Lấy dữ liệu trực tiếp")): Đặt văn bản cho RadioButton "Lấy dữ liệu trực tiếp".
* **self.rbFile.setText**(\_translate("MainWindow", "Lấy dữ liệu từ file có sẵn")): Đặt văn bản cho RadioButton "Lấy dữ liệu từ file có sẵn".
* **self.rbNhapTay.setText**(\_translate("MainWindow", "Nhập tay")): Đặt văn bản cho RadioButton "Nhập tay".
* **self.label.setText**(\_translate("MainWindow", "Nhiệt độ")): Đặt văn bản cho nhãn "Nhiệt độ".
* **self.label\_2.setText**(\_translate("MainWindow", "Tốc độ gió")): Đặt văn bản cho nhãn "Tốc độ gió".
* **self.label\_3.setText**(\_translate("MainWindow", "Độ ẩm")): Đặt văn bản cho nhãn "Độ ẩm".
* **self.pushButton.setText**(\_translate("MainWindow", "Nhập Thành Phố")): Đặt văn bản cho nút "Nhập Thành Phố".
* **self.pushButton\_2.setText**(\_translate("MainWindow", "Hiện Kết Quả Lên Bảng")): Đặt văn bản cho nút "Hiện Kết Quả Lên Bảng".

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động

# Hình 4.19. Ràng buộc dữ liệu cho người dùng khi nhập

Đoạn mã này định nghĩa một phương thức có tên **enable\_manual\_input** để quản lý trạng thái nhập liệu thủ công cho các trường dữ liệu trong giao diện người dùng. Nếu người dùng chọn **RadioButton** "Nhập tay" (rbNhapTay), phương thức sẽ kích hoạt chế độ nhập liệu cho ba trường thông tin (nhiệt độ, tốc độ gió, độ ẩm) bằng cách cho phép người dùng nhập dữ liệu vào các trường này và áp dụng các ràng buộc giá trị (ví dụ, nhiệt độ phải từ -50 đến 50°C, tốc độ gió từ 0 đến 100 m/s, độ ẩm từ 0 đến 100%). Ngược lại, nếu người dùng không chọn "Nhập tay", các trường nhập liệu sẽ bị vô hiệu hóa (không thể nhập dữ liệu) và các ràng buộc cũng được loại bỏ. Phương thức này giúp linh hoạt điều khiển tính năng nhập liệu trong giao diện dựa trên sự lựa chọn của người dùng.

# Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ Mô tả được tạo tự động

# Hình 4.20 Đọc dữ liệu từ API trên web OpenWeatherMap

Phần mã nàyđịnh nghĩa phương thức **fetch\_weather\_data** để lấy dữ liệu thời tiết từ API OpenWeatherMap và hiển thị thông tin thời tiết trong các trường nhập liệu trên giao diện người dùng. Phương thức đầu tiên kiểm tra xem người dùng có chọn RadioButton "Trực tiếp" hay không, sau đó lấy tên thành phố từ trường nhập liệu. Nếu thành phố chưa được nhập, một thông báo cảnh báo sẽ xuất hiện yêu cầu người dùng nhập tên thành phố. Tiếp theo, phương thức xây dựng URL để gửi yêu cầu tới API, và xử lý phản hồi từ API. Nếu thành phố không tìm thấy, một thông báo lỗi sẽ được hiển thị. Nếu thành công, các thông tin như nhiệt độ, tốc độ gió và độ ẩm sẽ được trích xuất và hiển thị trên giao diện người dùng sau khi chuyển đổi nhiệt độ sang độ C.

# Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm Mô tả được tạo tự động

**Hình 4.21 Đọc dữ liệu từ file JSON**

# Định nghĩa phương thức show\_data\_on\_table để hiển thị dữ liệu thời tiết lên bảng trong giao diện người dùng. Nếu người dùng chọn phương thức "Trực tiếp", dữ liệu thời tiết sẽ được lấy từ các trường nhập liệu và thêm vào bảng, nếu các trường dữ liệu không trống. Nếu chọn "File", phương thức sẽ đọc dữ liệu từ file JSON "thanhpho.json", trích xuất thông tin về nhiệt độ, tốc độ gió và độ ẩm, sau đó hiển thị dữ liệu này lên bảng. Nếu dữ liệu không đầy đủ hoặc có lỗi trong việc đọc file, các thông báo lỗi sẽ được hiển thị.

Ảnh có chứa văn bản, ảnh chụp màn hình

Mô tả được tạo tự động

# Hình 4.22. Tạo các nút chọn 2 thuật toán

Đoạn mã trong ảnh định nghĩa phương thức visualization\_sort\_window để sắp xếp dữ liệu trên bảng dựa trên tiêu chí và phương pháp sắp xếp người dùng chọn. Phương thức lấy thông tin về tiêu chí sắp xếp (nhiệt độ, tốc độ gió, độ ẩm) và phương pháp sắp xếp (Insertion Sort hoặc Heap Sort) từ các ComboBox. Sau đó, dữ liệu từ bảng được chuyển thành danh sách để dễ dàng xử lý, với mỗi dòng được trích xuất và chuyển đổi sang kiểu dữ liệu phù hợp. Dựa vào tiêu chí, phương thức chọn cột tương ứng để làm cơ sở sắp xếp dữ liệu. Nếu tiêu chí không hợp lệ, một thông báo cảnh báo sẽ hiển thị. Phương thức chuẩn bị dữ liệu để sắp xếp nhưng không thực hiện việc sắp xếp trực tiếp trong đoạn mã này.

# 

**Hình 4.23 Xóa dữ liệu trong bảng và khởi tạo giao diện**  
Đoạn mã này định nghĩa hai phương thức để xóa dữ liệu trong bảng. Phương thức delete\_selected\_row cho phép xóa hàng được chọn trong bảng, và nếu không có hàng nào được chọn, một thông báo cảnh báo sẽ xuất hiện. Phương thức delete\_all\_rows xóa tất cả các hàng trong bảng, và nếu bảng không có dữ liệu, một cảnh báo sẽ hiển thị. Cuối cùng, phần mã if \_\_name\_\_ == "\_\_main\_\_": khởi tạo ứng dụng Qt, tạo cửa sổ chính, hiển thị giao diện người dùng và tiếp tục chạy chương trình cho đến khi người dùng đóng cửa sổ.

# 4.2.2 VisualizationSort.py

Trong file code trên mô tả một widget PyQt6 được thiết kế để mô phỏng hai thuật toán sắp xếp: **Heap Sort** và **Insertion Sort**. Widget này sử dụng QGraphicsView và QGraphicsScene để vẽ các cấu trúc dữ liệu tương ứng với mỗi thuật toán, trong đó Heap Sort được thể hiện bằng cây nhị phân, còn Insertion Sort được thể hiện qua biểu đồ cột. Các nút trong cây nhị phân hoặc cột trong biểu đồ thay đổi màu sắc để chỉ ra phần tử đang được thao tác. Widget còn cung cấp các công cụ điều khiển như QSlider cho phép người dùng điều chỉnh tốc độ hoạt động của thuật toán và QComboBox để chọn thứ tự sắp xếp (tăng dần hoặc giảm dần). Mỗi bước của thuật toán, chẳng hạn như hoán đổi trong Heap Sort hoặc di chuyển phần tử trong Insertion Sort, được mô phỏng và hiển thị với thời gian dừng giữa các bước, giúp người dùng dễ dàng quan sát quá trình sắp xếp. Thông tin về các bước hoạt động của thuật toán và thời gian thực hiện được cập nhật liên tục trên giao diện, tạo ra một trải nghiệm tương tác và dễ hiểu cho người sử dụng khi theo dõi hoạt động của các thuật toán sắp xếp.

Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm

Mô tả được tạo tự động**Hình 4.24 Khai báo thư viện và tọa các nút**

# Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ Mô tả được tạo tự động

**Hình 4.25 Vẽ các cây nhị phân mô phỏng thuật toán Heap Sort**

Đoạn mã trên thực hiện một số chức năng liên quan đến việc cập nhật tốc độ sắp xếp, thay đổi thứ tự sắp xếp, và vẽ cây nhị phân biểu diễn thuật toán Heap. Dưới đây là phần giải thích bằng cách thay thế các từ trong đoạn mã:

Phương thức update\_speed được sử dụng để **cập nhật tốc độ sắp xếp** dựa trên giá trị hiện tại của thanh trượt (slider). Khi người dùng thay đổi giá trị của thanh trượt, phương thức này sẽ tính toán lại **tốc độ hoạt hình** bằng cách chia một giá trị cố định (1.6) cho giá trị của thanh trượt, từ đó điều chỉnh tốc độ hiển thị quá trình sắp xếp.

Phương thức toggle\_order thực hiện việc **thay đổi thứ tự sắp xếp**, có thể là **tăng dần** hoặc **giảm dần**, dựa vào chỉ mục hiện tại của lựa chọn trong **ComboBox** (order\_combobox). Nếu lựa chọn là **tăng dần**, giá trị của biến ascending sẽ được đặt là True, ngược lại nếu là **giảm dần**, giá trị của nó sẽ là False.

Phương thức draw\_heap\_tree dùng để **vẽ cây nhị phân** đại diện cho **Heap**. Cây này được vẽ trên **Cảnh đồ họa** (QGraphicsScene). Đầu tiên, phương thức này **xóa** mọi đối tượng cũ trong cảnh, sau đó tính toán số tầng của cây (dựa trên độ dài của dữ liệu) và xác định vị trí của các nút cây. Các nút cây được vẽ dưới dạng **hình elip**, và nếu nút cần được làm nổi bật (như khi hoán đổi trong thuật toán), màu sắc của nó sẽ được thay đổi thành **đỏ**, ngược lại sẽ là **xanh dương**. Các đường nối giữa các nút cây được vẽ bằng cách sử dụng **đường thẳng** (QGraphicsLineItem), từ đó biểu diễn mối quan hệ cha-con trong cấu trúc cây nhị phân

Ảnh có chứa văn bản, ảnh chụp màn hình

Mô tả được tạo tự động

# Hình 4.26 Vẽ biểu đồ cho thuật toán Insertion Sort

Đoạn mã trên mô phỏng thuật toán sắp xếp **Insertion Sort** và **Heap Sort** với giao diện đồ họa, cho phép người dùng xem trực quan quá trình sắp xếp của các thuật toán này.

Trong phương thức insertion\_sort\_visualization, thuật toán **Insertion Sort** được thực hiện trên dữ liệu đầu vào data. Đầu tiên, hàm tính toán độ dài của dữ liệu và ghi lại thời gian bắt đầu. Sau đó, thuật toán thực hiện qua từng phần tử trong dữ liệu (từ phần tử thứ 2 trở đi) và so sánh nó với các phần tử trước đó để tìm vị trí thích hợp. Khi một phần tử cần di chuyển, thuật toán **di chuyển các phần tử lớn hơn** một vị trí sang phải cho đến khi tìm thấy vị trí thích hợp. Sau mỗi lần di chuyển, hàm sẽ **vẽ lại biểu đồ cột** thể hiện dữ liệu sau khi thay đổi, đồng thời cập nhật **thông báo hoạt động** hiển thị trên giao diện, mô tả phần tử nào đang được di chuyển. Khi phần tử được chèn vào đúng vị trí, nó sẽ được hiển thị lại trên biểu đồ và thông báo sẽ được thay đổi để mô tả việc phần tử đã được chèn vào đúng vị trí.

Khi tất cả các phần tử được xử lý, thời gian sắp xếp sẽ được tính toán và hiển thị, đồng thời **hoạt động sắp xếp** sẽ được thông báo là hoàn tất.

Trong phương thức start\_sorting, khi người dùng bắt đầu quá trình sắp xếp, nó kiểm tra loại thuật toán cần sử dụng (sort\_type). Nếu là **Heap Sort**, phương thức sẽ gọi hàm heap\_sort\_visualization để thực hiện sắp xếp với thuật toán Heap Sort. Nếu là **Insertion Sort**, phương thức sẽ gọi insertion\_sort\_visualization để bắt đầu sắp xếp với thuật toán Insertion Sort. Trước khi bắt đầu, **thông báo hoạt động** sẽ được cập nhật để hiển thị thuật toán đang được thực hiện.

# 4.2.3. SortingAlgorithm.py

Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Hệ điều hành

Mô tả được tạo tự độngẢnh có chứa văn bản, ảnh chụp màn hình

Mô tả được tạo tự động

# Hình 4.27 Thuật toán Heap Sort

# Việc thực hiện thuật toán Heap Sort để sắp xếp dữ liệu trong một mảng. Thuật toán này bắt đầu bằng việc xây dựng một heap hợp lệ từ mảng, sử dụng hàm heapify để duy trì tính chất của heap. Sau khi mảng được chuyển đổi thành heap, thuật toán hoán đổi phần tử đầu tiên (phần tử lớn nhất hoặc nhỏ nhất) với phần tử cuối cùng chưa sắp xếp và tiếp tục gọi lại heapify cho phần mảng còn lại để duy trì tính chất heap. Quá trình này tiếp tục cho đến khi mảng được sắp xếp hoàn toàn. Thuật toán này có thể được áp dụng cho mảng dữ liệu phức tạp bằng cách sử dụng một chỉ mục (key index) để xác định phần tử so sánh. Thêm vào đó, hàm draw\_data\_callback cho phép vẽ lại mảng sau mỗi bước hoán đổi, giúp mô phỏng trực quan quá trình sắp xếp.

# Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm Mô tả được tạo tự động

**Hình 4.28 Thuật toán Insertion**

# Thuật toán Insertion Sort, một thuật toán sắp xếp đơn giản nhưng hiệu quả trong việc xử lý các mảng nhỏ hoặc đã gần được sắp xếp. Thuật toán này làm việc bằng cách duyệt qua các phần tử trong mảng từ trái qua phải, mỗi lần chèn một phần tử vào vị trí thích hợp trong phần mảng đã được sắp xếp. Hàm insertion\_sort nhận vào các tham số như data (dữ liệu cần sắp xếp), key\_index (chỉ số khóa dùng để so sánh phần tử), n (số phần tử trong mảng cần sắp xếp), và các tham số khác cho việc điều chỉnh thứ tự sắp xếp (tăng dần hoặc giảm dần). Sau mỗi lần hoán đổi, hàm sẽ gọi draw\_data\_callback để vẽ lại mảng với màu sắc khác nhau, giúp người dùng dễ dàng theo dõi quá trình sắp xếp. Cụ thể, phần tử đang di chuyển được tô màu vàng, phần tử hiện tại được so sánh tô màu xanh, và các phần tử còn lại được tô màu đỏ. Cuối cùng, khi chèn phần tử vào đúng vị trí, phần tử đó sẽ được tô màu xanh lá cây để thể hiện rằng quá trình chèn đã hoàn thành.

**CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHẤT TRIỂN**

# 5.1 KẾT LUẬN

Ứng dụng giải thuật sắp xếp để sắp xếp dữ liệu thời tiết, đặc biệt là qua việc áp dụng Heap Sort và Insertion Sort, đã đạt được kết quả khả quan trong việc thực hiện chức năng cơ bản của mình. Cả hai thuật toán đều cho thấy những điểm mạnh riêng biệt trong việc xử lý và sắp xếp dữ liệu. Heap Sort, với độ phức tạp thời gian O(n log n), là một lựa chọn hiệu quả khi xử lý các bộ dữ liệu lớn, đảm bảo tính ổn định và tối ưu về mặt hiệu suất. Ngược lại, Insertion Sort mặc dù có độ phức tạp cao hơn trong trường hợp dữ liệu lớn (O(n²)), nhưng lại tỏ ra rất hữu ích khi làm việc với các bộ dữ liệu nhỏ hoặc các dãy dữ liệu đã gần như sắp xếp sẵn. Qua quá trình thực hiện, chúng ta cũng nhận thấy rằng ứng dụng hiện tại đã đáp ứng được nhu cầu sắp xếp dữ liệu cơ bản nhưng vẫn còn nhiều không gian để cải tiến về hiệu suất, giao diện người dùng, và tính năng hỗ trợ. Tổng thể, ứng dụng đã đạt được mục tiêu ban đầu và tạo nền tảng vững chắc cho việc phát triển thêm các tính năng trong tương lai.

5.2. HƯỚNG PHÁT TRIỂN

# Một trong những hướng phát triển quan trọng là tối ưu hóa giao diện người dùng (UI). Hiện tại, giao diện của ứng dụng có phần đơn giản, không hoàn toàn tối ưu cho người dùng khi cần thao tác với các bộ dữ liệu lớn. Để nâng cao trải nghiệm người dùng, giao diện có thể được thiết kế lại với các tính năng như tìm kiếm nhanh, lọc dữ liệu theo tiêu chí (ví dụ: tìm kiếm theo thành phố, nhiệt độ, độ ẩm), hoặc thao tác kéo và thả để người dùng có thể dễ dàng thay đổi thứ tự dữ liệu. Thêm vào đó, ứng dụng có thể phát triển thêm các tính năng như chế độ xem biểu đồ (biểu đồ nhiệt độ, độ ẩm theo thời gian), giúp người dùng có cái nhìn trực quan hơn về sự thay đổi thời tiết trong suốt thời gian.

# Một hướng phát triển khác là mở rộng các thuật toán sắp xếp, đặc biệt là bổ sung các thuật toán tối ưu hơn như Quick Sort, Merge Sort, hay thậm chí các thuật toán phân tán để xử lý dữ liệu trong môi trường đám mây. Điều này không chỉ giúp tăng cường khả năng xử lý dữ liệu mà còn giúp người dùng có thể lựa chọn thuật toán phù hợp nhất tùy theo tình huống. Bên cạnh đó, một cải tiến rất cần thiết là tích hợp các nguồn dữ liệu thời tiết đa dạng từ các dịch vụ thời tiết khác nhau như OpenWeatherMap, AccuWeather, hay Weatherstack, giúp nâng cao tính chính xác và phong phú của dữ liệu đầu vào. Việc tích hợp các API dự báo thời tiết theo thời gian thực sẽ mang lại trải nghiệm hữu ích hơn cho người dùng, giúp họ theo dõi và nhận thông tin mới nhất về thời tiết. Cuối cùng, để mở rộng khả năng ứng dụng, có thể phát triển thêm các tính năng như thông báo khi có sự thay đổi đột ngột về thời tiết, hoặc dự báo xu hướng thời tiết cho những ngày tới, giúp người dùng có thể chuẩn bị kế hoạch phù hợp.

5.3 HẠN CHẾ

# Dù đã hoàn thiện chức năng cơ bản, ứng dụng vẫn tồn tại một số hạn chế quan trọng cần được khắc phục. Đầu tiên, vấn đề đầu vào dữ liệu không đồng nhất là một trong những khó khăn lớn. Dữ liệu đầu vào có thể đến từ nhiều nguồn khác nhau, và nếu không được chuẩn hóa đúng cách, ứng dụng sẽ gặp phải các lỗi trong quá trình xử lý và sắp xếp, ví dụ như dữ liệu thiếu hoặc sai định dạng. Hơn nữa, ứng dụng vẫn chưa có tính năng tự động cập nhật dữ liệu thời tiết từ các nguồn API, điều này khiến người dùng phải cập nhật thủ công, gây bất tiện và không đảm bảo tính thời gian thực của dữ liệu. Giao diện chưa được bắt mắt và bố trí hợp lí

# Thứ hai, mặc dù Heap Sort là một giải thuật rất hiệu quả với độ phức tạp O(n log n), nhưng khi làm việc với dữ liệu cực lớn (hàng triệu hoặc hàng tỷ bản ghi), ứng dụng vẫn có thể gặp phải tình trạng tắc nghẽn tài nguyên hệ thống. Các thuật toán cần được tối ưu hóa để có thể xử lý các bộ dữ liệu khổng lồ mà không làm giảm hiệu suất hoặc gây ảnh hưởng tới các thao tác khác trong hệ thống. Điều này đặc biệt quan trọng khi ứng dụng được triển khai trên các nền tảng có giới hạn tài nguyên như thiết bị di động hay các máy chủ với cấu hình không mạnh.

# Ngoài ra, khả năng mở rộng của ứng dụng vẫn còn hạn chế, đặc biệt khi phải xử lý một lượng lớn dữ liệu thời tiết từ nhiều nguồn khác nhau. Điều này có thể gây ảnh hưởng đến tốc độ và tính chính xác trong việc sắp xếp và phân tích dữ liệu. Hơn nữa, một số tính năng cần thiết để nâng cao hiệu suất của ứng dụng như caching, load balancing, hay tối ưu hóa bộ nhớ vẫn chưa được triển khai. Do đó, để đáp ứng nhu cầu sử dụng trong môi trường thực tế, ứng dụng cần có thêm các cải tiến về mặt công nghệ và kiến trúc hệ thống.

# Cuối cùng, vấn đề bảo mật dữ liệu là một yếu tố cần chú ý. Việc xử lý và lưu trữ dữ liệu thời tiết có thể gặp phải các nguy cơ về bảo mật nếu không có các biện pháp bảo vệ như mã hóa và xác thực người dùng.

**TÀI LIỆU THAM KHẢO** (GeeksforGeeks, 2024)

TS.Nguyễn Đức Thuần, Cấu trúc dữ liệu và giải thuật

TS. Nguyễn Đức Thuần, Kỹ thuật lập trình.

OpenWeatherMap, "Weather API Documentation:<https://home.openweathermap.org/>

Nguyễn Thị Hòa (2020), *Lập trình Python cơ bản và nâng cao*, Nhà xuất bản Bách Khoa Hà Nội.

GeeksforGeeks (2024), Heap Sort Algorithm: <https://www.geeksforgeeks.org/heap-sort/> GeeksforGeeks (2024), Insertion Sort Algorith:<https://www.geeksforgeeks.org/insertion-sort-algorithm/>

**TÀI LIỆU TIẾNG ANH**

Goodrich, M. T., and Tamassia, R. (2014), *Algorithm Design and Applications.*