**ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

**--------------**



**BÁO CÁO BÀI TẬP LỚN**

**LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG**

Chủ đề: *“*Biểu diễn thuật toán sắp xếp trên mảng*”*

Nhóm 17

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Họ và Tên | MSSV | Email |
| 1 | Nguyễn Nhật Minh | 20225886 | Minh.nn225886@sis.hust.edu.vn |
| 2 | Trần Đại Quang Minh | 20225889 | Minh.tdq225889@sis.hust.edu.vn |
| 3 | Phạm Tuấn Minh | 20225744 | Minh.pt225744@sis.hust.edu.vn |
| 4 | Nguyễn Tuấn Minh | 20225742 | Minh.nt225742@sis.hust.edu.vn |
| 5 | Nguyễn Đức Minh | 20225885 | Minh.nd225885@sis.hust.edu.vn |

**Mục Lục**

[**LỜI NÓI ĐẦU 3**](#_Toc186058187)

[**PHẦN 1. PHÂN CÔNG CÔNG VIỆC 4**](#_Toc186058188)

[**PHẦN 2. BIỂU ĐỒ UML 5**](#_Toc186058189)

[**I. UseCase Diagram 5**](#_Toc186058190)

[**II. Class Diagram 5**](#_Toc186058191)

[**PHẦN 3. KỸ THUẬT VÀ CÔNG NGHỆ ĐÃ ÁP DỤNG 6**](#_Toc186058192)

[**1. Nguyên tắc Lập trình Hướng Đối Tượng (OOP) 6**](#_Toc186058193)

[**a) Đóng gói (Encapsulation): 6**](#_Toc186058194)

[**b) Kế thừa (Inheritance): 6**](#_Toc186058195)

[**c) Đa hình (Polymorphism): 6**](#_Toc186058196)

[**d) Trừu tượng (Abstraction): 6**](#_Toc186058197)

[**2. Thuật toán và các kiến thức sử dụng 6**](#_Toc186058198)

[**a. Các Thuật toán Sắp xếp: 6**](#_Toc186058199)

[**b. Kiến thức Cơ bản và Cài đặt: 7**](#_Toc186058200)

[**Các Thư viện và Package: 7**](#_Toc186058201)

[**HƯỚNG DẪN SỬ DỤNG 8**](#_Toc186058202)

# LỜI NÓI ĐẦU

Ứng dụng này tập trung vào việc giải thích ba thuật toán sắp xếp quan trọng, bao gồm:

1. Sắp xếp nổi bọt (Bubble Sort):

○ Là thuật toán đơn giản, hoạt động dựa trên việc so sánh từng cặp phần tử liền kề trong danh sách và hoán đổi vị trí của chúng nếu sai thứ tự. Quá trình này lặp lại cho đến khi toàn bộ danh sách được sắp xếp.

○ Mặc dù dễ hiểu và dễ triển khai, sắp xếp nổi bọt có hiệu suất thấp, đặc biệt là với tập dữ liệu lớn, do phải thực hiện nhiều lần duyệt qua danh sách.

○ Ưu điểm: Đơn giản, dễ cài đặt.

○ Nhược điểm: Thời gian thực thi chậm (độ phức tạp O(n²)).

2. Sắp xếp nhanh (Quick Sort):

○ Thuộc nhóm thuật toán chia để trị, quick sort chọn một phần tử làm chốt (pivot) và phân chia mảng thành hai phần: các phần tử nhỏ hơn pivot và các phần tử lớn hơn pivot. Sau đó, thuật toán tiếp tục áp dụng sắp xếp nhanh cho từng phần.

○ Đây là một trong những thuật toán sắp xếp nhanh nhất trong thực tế, thường được sử dụng cho các tập dữ liệu lớn.

○ Ưu điểm: Hiệu suất cao, trung bình O(n log n).

○ Nhược điểm: Trường hợp xấu nhất có thể đạt O(n²) nếu chọn pivot không phù hợp.

3. Sắp xếp chèn (Insertion Sort):

○ Hoạt động bằng cách chia mảng thành hai phần: phần đã sắp xếp và phần chưa sắp xếp. Mỗi lần, thuật toán lấy một phần tử từ phần chưa sắp xếp và chèn vào vị trí đúng trong phần đã sắp xếp.

○ Phù hợp cho các danh sách nhỏ hoặc các danh sách đã gần như sắp xếp sẵn.

○ Ưu điểm: Dễ triển khai, hiệu quả với dữ liệu nhỏ hoặc dữ liệu đã được sắp xếp một phần.

○ Nhược điểm: Hiệu suất thấp với dữ liệu lớn, độ phức tạp O(n²).

Ứng dụng không chỉ cung cấp giải thích lý thuyết mà còn **minh họa chi tiết** từng thuật toán, cho phép **chạy thực tế** theo yêu cầu của người dùng. Quá trình thực thi sẽ **hiển thị từng bước thực hiện** của thuật toán, giúp người dùng dễ dàng theo dõi và hiểu rõ cách thức hoạt động.

Thông qua ứng dụng, người dùng sẽ nắm được cách thức hoạt động, ưu và nhược điểm của từng thuật toán, từ đó áp dụng hiệu quả trong thực tiễn.

# PHẦN 1. PHÂN CÔNG CÔNG VIỆC

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Họ và Tên | MSSV | Đóng góp | Đóng góp chung |
| 1 | Nguyễn Nhật Minh | 20225886 | * Làm sơ đồ use case (50%) * Cài đặt packet SortAlgorithm (10%) * Cài đặt packet Menu (50%) * Làm báo cáo (50%) | 18% |
| 2 | Trần Đại Quang Minh | 20225889 | * Làm sơ đồ use case (50%) * Cài đặt packet SortAlgorithm (10%) * Cài đặt packet Menu (50%) * Làm báo cáo (50%) | 17% |
| 3 | Phạm Tuấn Minh | 20225744 | * Làm class diagram (60%) * Cài đặt packet SortInputGUI * Làm slide (20%) * Kiểm tra, fix bug | 25% |
| 4 | Nguyễn Tuấn Minh | 20225742 | * Làm class diagram (20%) * Cài đặt packet SortAlgorithm (40%) * Làm slide (40%) | 20% |
| 5 | Nguyễn Đức Minh | 20225885 | * Làm class diagram (20%) * Cài đặt packet SortAlgorithm (40%) * Làm slide (40%) | 20% |

# PHẦN 2. BIỂU ĐỒ UML

## I. UseCase Diagram

A diagram of a person

Description automatically generated

## II. Class Diagram

A diagram of a software application

Description automatically generated with medium confidence

# PHẦN 3. KỸ THUẬT VÀ CÔNG NGHỆ ĐÃ ÁP DỤNG

1. **Nguyên tắc Lập trình Hướng Đối Tượng (OOP)**
2. **Đóng gói (Encapsulation):**
   * **Bảo vệ dữ liệu:**Mảng array và vùng hiển thị arrayDisplay trong lớp SortInputGUI được đóng gói, tránh việc truy cập trực tiếp từ bên ngoài. Các phương thức như generateRandomArray() và inputArrayManually() quản lý và thay đổi mảng, bảo vệ dữ liệu.
   * **Đóng gói hành vi:**Hành vi như tạo mảng ngẫu nhiên hoặc nhập mảng thủ công được đóng gói trong các phương thức, giúp người dùng chỉ cần sử dụng chức năng mà không cần biết chi tiết cách thức.
3. **Kế thừa (Inheritance):**

* **Tái sử dụng mã:**Các lớp VisualBubbleSortFX, VisualInsertionSortFX, và VisualQuickSortFX có thể kế thừa từ lớp SortAlgorithm, tái sử dụng các phương thức chung như start() mà không phải viết lại mã cho từng thuật toán.
* **Mở rộng dễ dàng:**Việc kế thừa giúp mở rộng mã mà không thay đổi lớp SortInputGUI. Thêm thuật toán mới chỉ cần tạo lớp mới kế thừa từ SortAlgorithm, giúp giao diện người dùng không cần thay đổi.

1. **Đa hình (Polymorphism):**
   * **Thay đổi hành vi đối tượng:** Đa hình được thể hiện khi người dùng chọn thuật toán sắp xếp. Các đối tượng thuật toán sẽ sử dụng phương thức start() để thực hiện thuật toán, nhưng hành vi sẽ khác nhau tùy thuộc vào thuật toán chọn.
   * **Đa hình trong GUI**:Các đối tượng thuật toán có hành vi khác nhau khi gọi phương thức start(), giúp tăng tính linh hoạt và không cần thay đổi mã trong giao diện người dùng.
2. **Trừu tượng (Abstraction):**
   * **Che giấu chi tiết thực hiện:**Các chi tiết thuật toán như Bubble Sort, Insertion Sort, và Quick Sort được ẩn trong các lớp như VisualBubbleSortFX, người dùng chỉ cần chọn thuật toán.
   * **Giao diện rõ ràng:**Lớp SortInputGUI cung cấp giao diện rõ ràng để người dùng nhập mảng và chọn thuật toán mà không cần biết các chi tiết nội bộ của các thuật toán.
3. **Thuật toán và các kiến thức sử dụng**
   1. **Các Thuật toán Sắp xếp:**
      * **Bubble Sort:**Tìm cặp phần tử liên tiếp không đúng thứ tự và đổi chỗ chúng. Quá trình này được lặp lại cho đến khi mảng được sắp xếp hoàn toàn.
      * **Insertion Sort:**Chia mảng thành hai phần: một phần đã được sắp xếp và phần còn lại chưa. Sau đó, chèn từng phần tử vào vị trí thích hợp trong phần đã sắp xếp.
      * **Quick Sort:**Chọn một phần tử làm pivot, sau đó phân chia mảng thành hai phần: các phần tử nhỏ hơn pivot và các phần tử lớn hơn pivot. Thuật toán tiếp tục áp dụng đệ quy cho hai phần này cho đến khi mảng được sắp xếp.
   2. **Kiến thức Cơ bản và Cài đặt:**

* **Lý thuyết sắp xếp:**Các thuật toán như Bubble Sort, Insertion Sort và Quick Sort đều là các thuật toán sắp xếp cơ bản với cách tiếp cận khác nhau để sắp xếp mảng.
* **Cấu trúc dữ liệu Java:**Trong đoạn mã bạn cung cấp, arrayDisplay sử dụng TextArea để hiển thị mảng, còn array là một mảng số nguyên, được sử dụng để lưu trữ các phần tử mà người dùng nhập vào hoặc tạo ra ngẫu nhiên. Các cấu trúc như ChoiceBox được dùng để chọn thuật toán sắp xếp, và Button cho phép thực hiện hành động tương tác với người dùng.
* **Kiến trúc Phần mềm (MVC):**Mô hình MVC được áp dụng khi phân chia mã thành các phần như:
  + **Model:** Các lớp như VisualBubbleSortFX, VisualInsertionSortFX, và VisualQuickSortFX thực hiện công việc xử lý thuật toán và quản lý dữ liệu.
  + **View:** Lớp SortInputGUI đại diện cho giao diện người dùng, hiển thị các yếu tố giao diện như nút bấm và hộp chọn.
  + **Controller:** Lớp SortInputGUI cũng đóng vai trò như controller, xử lý các sự kiện và liên kết giữa các lớp mô hình và giao diện người dùng

### Các Thư viện và Package:

* **application.UI:**Lớp SortInputGUI quản lý giao diện người dùng, cho phép nhập mảng và chọn thuật toán sắp xếp, cũng như thao tác với cửa sổ hiển thị.
* **application.model:**Dù không có lớp Node và Edge trong mã, có thể giả định chúng quản lý cấu trúc dữ liệu đồ thị, bao gồm các đỉnh và cạnh.
* **application.algorithm:**Các lớp như VisualBubbleSortFX, VisualInsertionSortFX, và VisualQuickSortFX thực hiện các thuật toán sắp xếp và hiển thị quá trình sắp xếp cho người dùng.

# HƯỚNG DẪN SỬ DỤNG

**1. Khởi động ứng dụng**

* Khi bạn khởi động ứng dụng, một cửa sổ chính với tiêu đề **"Main Menu"** sẽ xuất hiện. Trên giao diện chính, bạn sẽ thấy ba nút: **Input**, **Help**, và **Quit**.

**2. Các chức năng chính**

* **Input**:
  + Khi nhấn vào nút **Input**, một cửa sổ mới sẽ mở ra, yêu cầu người dùng nhập một mảng số nguyên hoặc chọn mảng ngẫu nhiên. Đây là bước chuẩn bị dữ liệu đầu vào cho thuật toán sắp xếp.
  + Sau khi chọn hoặc nhập mảng, bạn có thể tiếp tục để thực hiện việc sắp xếp dữ liệu.
* **Help**:
  + Nút **Help** sẽ hiển thị thông tin hỗ trợ cho người dùng về cách sử dụng ứng dụng. Một cửa sổ cảnh báo sẽ xuất hiện, cung cấp hướng dẫn về cách thức hoạt động của chương trình, bao gồm cách nhập dữ liệu, chọn thuật toán sắp xếp, và theo dõi quá trình sắp xếp.
  + Các thuật toán sắp xếp có sẵn trong ứng dụng bao gồm **Bubble Sort**, **Insertion Sort**, và **Quick Sort**. Bạn có thể lựa chọn một trong các thuật toán này để xem cách chúng sắp xếp mảng dữ liệu đã nhập.
* **Quit**:
  + Nút **Quit** sẽ yêu cầu người dùng xác nhận trước khi thoát ứng dụng. Một hộp cảnh báo sẽ xuất hiện yêu cầu bạn xác nhận việc đóng ứng dụng. Bạn có thể chọn **OK** để thoát hoặc **Cancel** để quay lại.

**3. Chọn thuật toán sắp xếp**

* Sau khi nhập mảng dữ liệu, ứng dụng sẽ hiển thị dữ liệu dưới dạng các khối hình chữ nhật với kích thước tương ứng với giá trị của mỗi phần tử trong mảng.
* Quá trình sắp xếp sẽ được thực hiện trực quan với ba thuật toán:
  + **Bubble Sort**: Các phần tử sẽ được so sánh và hoán đổi vị trí nếu chúng không đúng thứ tự, dần dần "nổi" các phần tử lớn nhất lên cuối mảng.
  + **Insertion Sort**: Mỗi phần tử sẽ được so sánh và chèn vào vị trí đúng trong phần mảng đã được sắp xếp.
  + **Quick Sort**: Mảng được phân chia thành các phần nhỏ hơn, sau đó các phần tử được sắp xếp xung quanh một giá trị trụ (pivot) rồi tiếp tục phân chia và sắp xếp.

**4. Quan sát quá trình sắp xếp**

* Trong suốt quá trình sắp xếp, bạn sẽ thấy các phần tử được tô màu để minh họa các bước so sánh và hoán đổi. Các khối màu sẽ thay đổi theo thời gian để chỉ ra những bước trong thuật toán.
  + **Màu vàng**: Các phần tử đang được so sánh.
  + **Màu đỏ**: Các phần tử đang được hoán đổi.
  + **Màu xanh lá cây**: Các phần tử đã được sắp xếp xong.

**5. Kết thúc và thoát**

* Khi thuật toán hoàn thành việc sắp xếp mảng, các phần tử sẽ được tô màu xanh lá cây để báo hiệu quá trình sắp xếp đã thành công.
* Bạn có thể nhấn **Quit** để thoát khỏi ứng dụng sau khi hoàn thành.