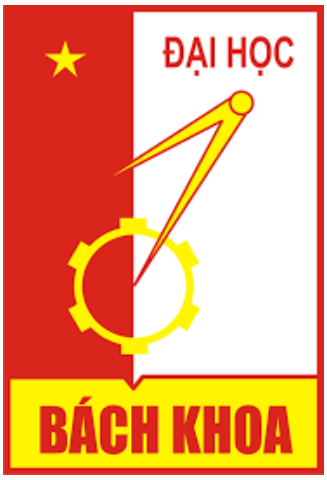
Trường Công nghệ thông tin và Truyền thông Đại Học Bách Khoa Hà Nội



Đề tài: Mô tả thuật toán sắp xếp với mảng

Môn: Lập trình hướng đối tượng

GV hướng dẫn: TS. Nguyễn Thị Thu Trang

Nhóm 3

Nguyễn Bá Anh	20215260
Nguyễn Duy Anh	20225596
Nguyễn Hải Anh	20225597
Nguyễn Kỳ Anh	20225783
Nguyễn Phúc Anh	20225784

Hà Nội, ngày 25 tháng 12 năm 2024

Table of contents

IT3103-151965

Nhóm tác giả và phân chia công việc

Họ và tên	Công việc thực hiện	Đánh giá
1. Nguyễn Bá Anh MSSV: 20215260	• Thiết kế UseCase Diagram • Code logic class SelectionSort.java • Viết báo cáo	Hoàn thành đúng tiến độ
2. Nguyễn Duy Anh MSSV: 20225596	• Thiết kế Class Diagram • Code logic class SortingAlgorithm và MergeSort • Code UI class ShellSort và MergeSort	Hoàn thành đúng tiến độ
3. Nguyễn Hải Anh MSSV: 20225597	• Thiết kế Class Diagram • Làm slide • Code class SortingPanel	Hoàn thành đúng tiến độ
4. Nguyễn Kỳ Anh MSSV: 20225783	• Quay video demo • Code logic class ShellSort • Viết README.md	Hoàn thành đúng tiến độ
5. Nguyễn Phúc Anh MSSV: 20225784	• Thiết kế Class Diagram • Code class MainScreenPanel và InputHandling • Code UI SelectionSort	Hoàn thành đúng tiến độ

- Link project: https://github.com/PhucAnh0502/LTHDT.20241-03
- Trong quá trình làm project, chúng em đã lấy cảm hứng từ

VisualGo : https://visualgo.net/en

GeeksforGeeks :

https://www.youtube.com/watch?v=xWBP4lzkoyM

https://www.youtube.com/watch?v=SHcPqUe2GZM

https://www.youtube.com/watch?v=SHcPqUe2GZM

Nhóm 03

1 GIỚI THIỆU BÀI TOÁN

1.1 Giới thiệu đề tài

SortingApplication là một ứng dụng sắp xếp mảng trực quan, được phát triển với mục đích giúp người dùng dễ dàng hiểu và theo dõi các thuật toán sắp xếp phổ biến như **Selection Sort**, **Merge Sort**, và **Shell Sort**. Dự án này cung cấp giao diện trực quan, cho phép người dùng nhập mảng và quan sát các bước sắp xếp qua từng thuật toán, từ việc so sánh các phần tử cho đến các bước hoán đổi.

SortingApplication đặc biệt hữu ích cho sinh viên và những người mới học về cấu trúc dữ liệu và thuật toán, giúp họ hình dung rõ ràng hơn về các thuật toán sắp xếp và cách thức hoạt động của chúng trong môi trường lập trình.

1.2. Các tính năng

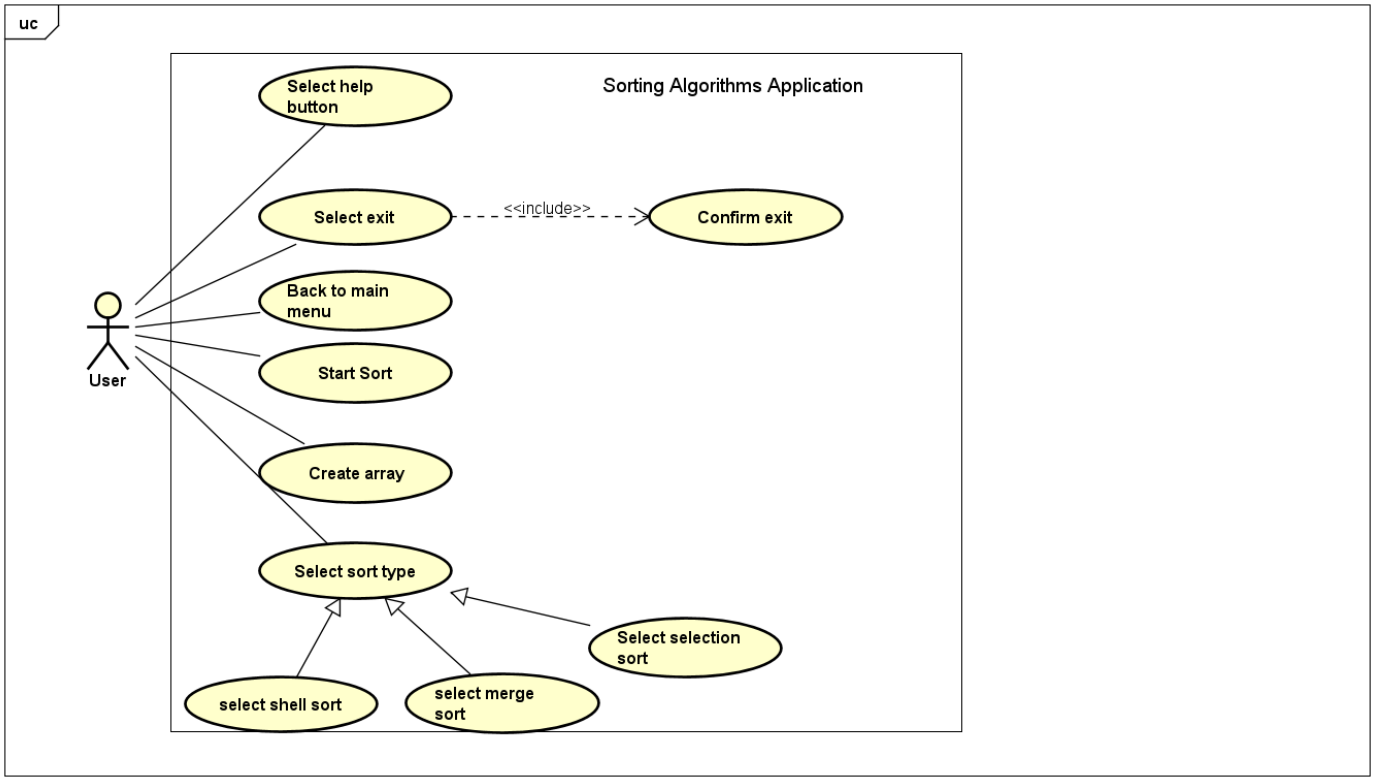
- Tính năng chính của chương trình là thực hiện việc sắp xếp mảng theo các thuật toán sắp xếp selection sort, shell sort, merge sort
- Người dùng sẽ lựa chọn thuật toán sắp xếp, sau đó có thể tạo mảng random hoặc nhập mảng mong muốn
- Chương trình sẽ thể hiện các bước sắp xếp và hiển thị quá trình sắp xếp

1.3. Công cụ sử dụng

- Ngôn ngữ lập trình: Java
- Version Control: Git (Github)
- GUI: Java Swing

2. Phân Tích Biểu Đồ Use Case

2.1. Biểu đồ usecase



2.2. Đặc tả một số Use case

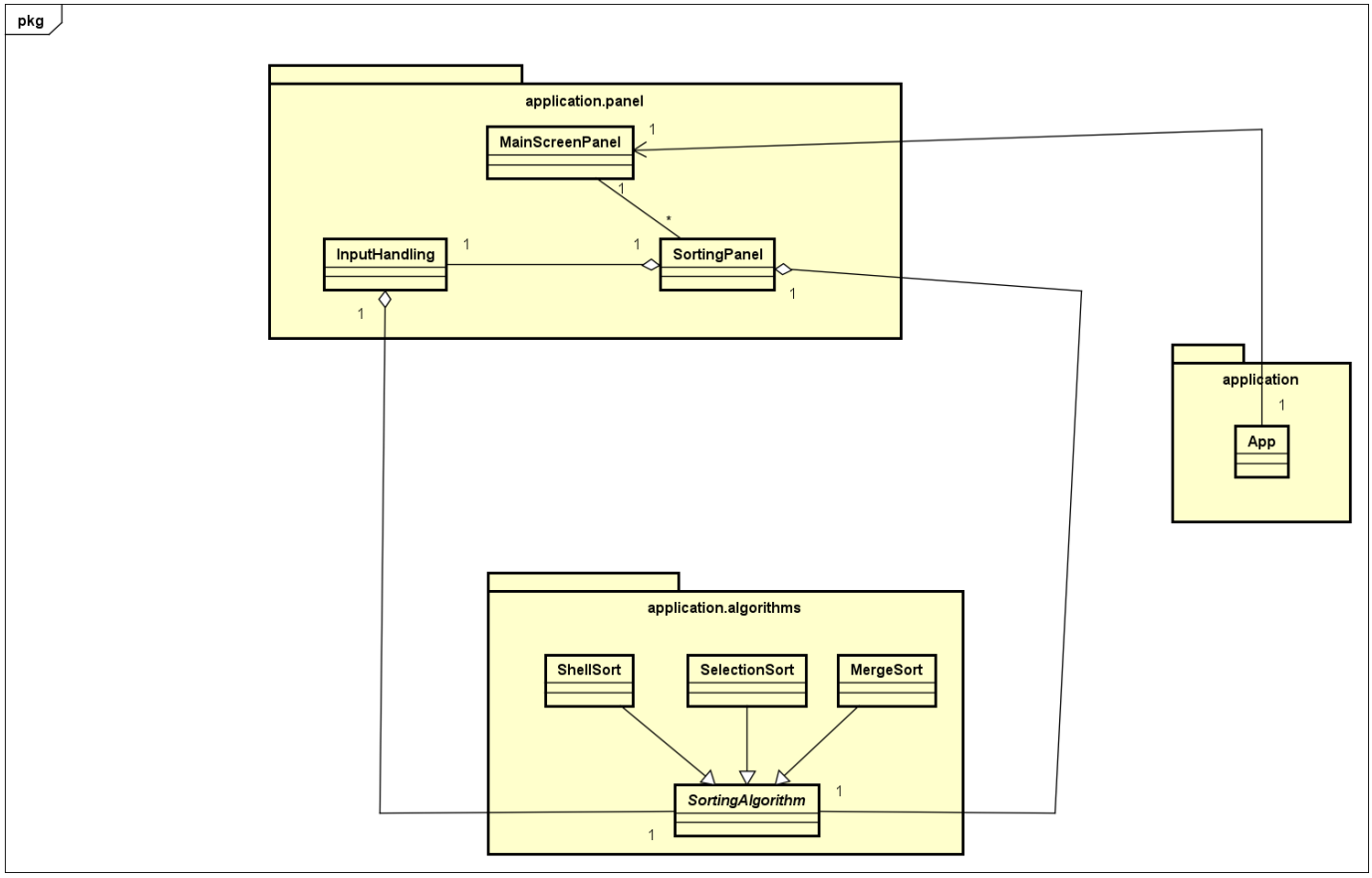
Khi sử dụng ứng dụng này, người dùng có thể thực hiện nhiều thao tác trên đó:

- **Chọn thuật toán (Select sort type):** Khi mở ứng dụng, người dùng được yêu cầu chọn 1 trong 3 thuật toán sắp xếp: Merge sort, Selection Sort và Shell Sort.
- **Tạo mảng (Create array):** Sau khi chọn thuật toán, người dùng sẽ được đưa đến màn tạo dữ liệu. Ở đây người có thể lựa chọn tự nhập dữ liệu hoặc tạo dữ liệu ngẫu nhiên.
- **Bắt đầu thuật toán sắp xếp (Start sort):** Sau khi tạo dữ liệu, người dùng được đưa đến màn biểu diễn thuật toán sắp xếp. Ở đây người dùng nhấn vào nút bắt đầu thuật toán để chương trình chạy ứng với thuật toán và dữ liệu của người dùng
- **Quay lại menu chính (Back to main menu):** Sau khi chương trình đã sắp xếp và cho xem kết quả, người dùng có thể nhấn nút quay lại menu chính để bắt đầu chương trình lại lần nữa.

- **Mở menu trợ giúp (Select help button):** Người dùng có thể mở menu trợ giúp ở trên thanh menu để đọc về thông tin ứng dụng
- **Thoát chương trình (Select exit):** Người dùng có thể nhấn vào nút exit ở trên thanh menu và xác nhận thoát ứng dụng

3. Biểu đồ Class Diagram

3.1. Biểu đồ Chung Class Diagram



Phân tích Biểu đồ Chung Class Diagram:

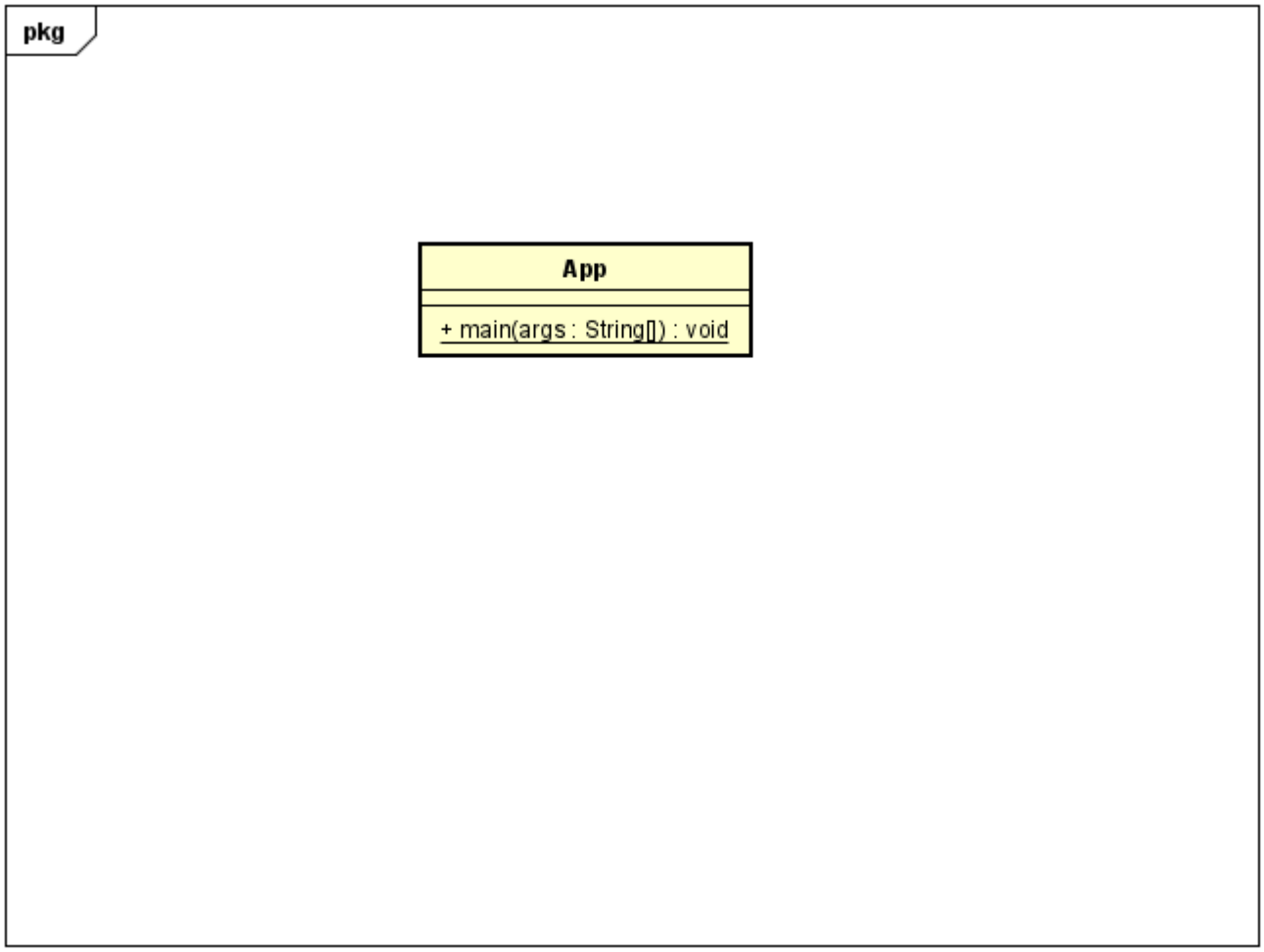
- **Package application:** Package này chứa một class duy nhất là class App. Class App được dùng để khởi tạo chương trình lên.
- **Package application.panel:**
  - Package này chứa 3 class được sử dụng làm làm các màn chính của ứng dụng.
  - Class MainScreenPanel được sử dụng để tạo giao diện và logic cho Main menu

- Class InputHandling được sử dụng để xử lý các dữ liệu đầu vào của người dùng
- Class SortingPanel được sử dụng để tạo ra màn hình hiển thị quá trình sắp xếp

- **Package application.algorithms**: Package này chứa 4 class trong đó có 1 abstract class đó là class SortingAlgorithm. 3 class còn lại MergeSort, SelectionSort, ShellSort kế thừa class SortingAlgorithm.

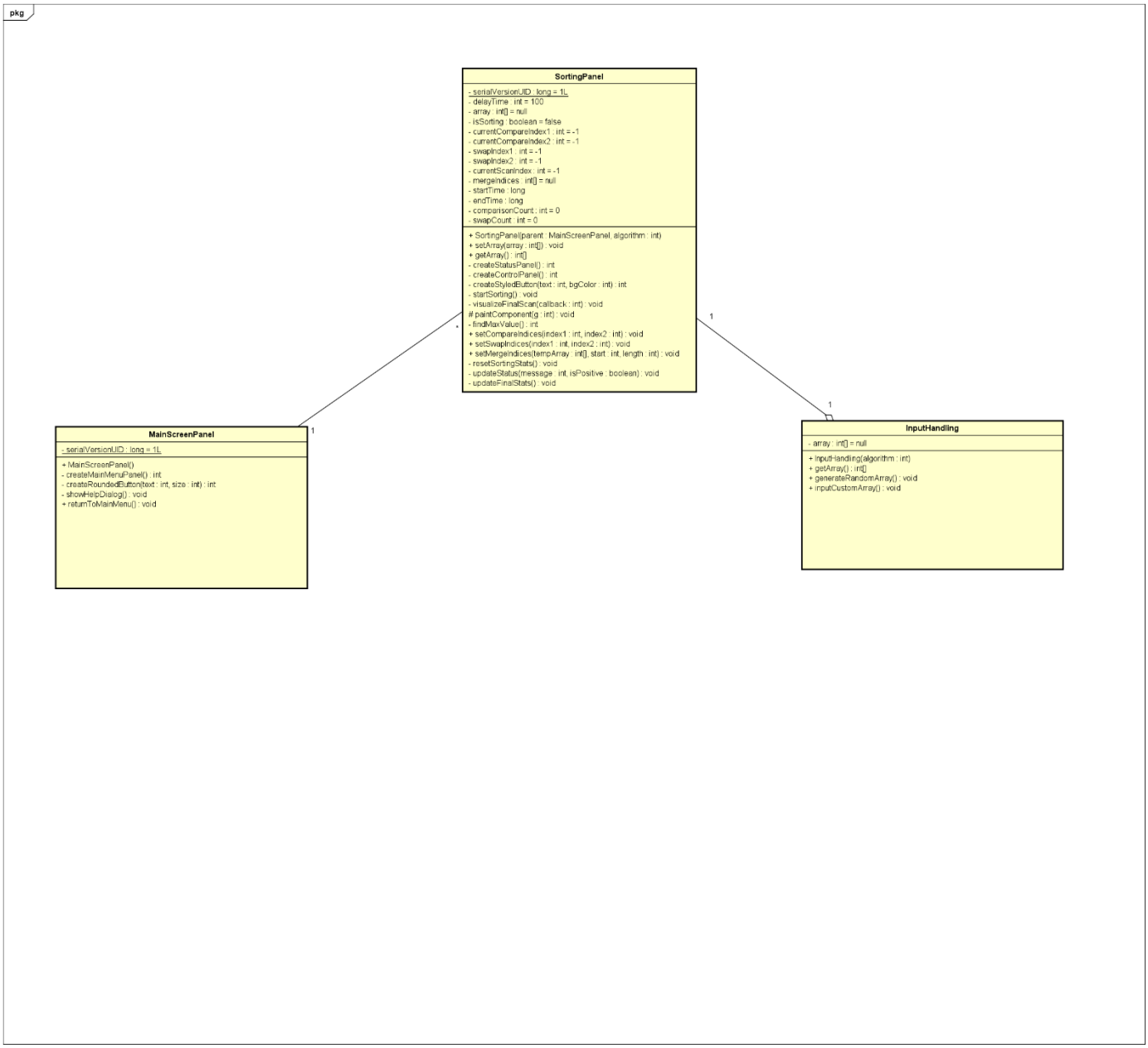
3.2. Biểu đồ Chi tiết Class Diagram

3.2.1. Package application



Package application có một class duy nhất là class **App**. Nhiệm vụ của class **App** là bắt đầu chạy ứng dụng.

3.2.2. Package application.panel



Package application.panel có tất cả 3 class chính: **MainScreenPanel**, **SortingPanel**, **InputHandling**.

MainScreenPanel: Class MainScreenPanel là một JFrame, nghĩa là nó đại diện cho cửa sổ chính của ứng dụng GUI. Nó chịu trách nhiệm quản lý các màn hình khác nhau (panel) để hiển thị giao diện trực quan hóa thuật toán sắp xếp. Lớp này sử dụng **CardLayout** để chuyển đổi giữa các panel.

- **MainScreenPanel() – *Constructor***

  - Khởi tạo cửa sổ chính, cài đặt CardLayout và thêm các panel cho từng thuật toán (Selection Sort, Merge Sort, Shell Sort).
- **createMainMenuPanel() – *Private***
  - Tạo panel menu chính với tiêu đề, các nút chọn thuật toán (Selection, Merge, Shell Sort), và nút Help/Quit.
  - Xử lý sự kiện chuyển đổi panel bằng CardLayout.
- **createRoundedButton(String text, Dimension size) – *Private***
  - Tạo nút JButton bo tròn với văn bản và kích thước tùy chỉnh.
  - Dùng Graphics2D để vẽ và tùy chỉnh giao diện nút.
- **showHelpDialog() – *Private***
  - Hiển thị hộp thoại trợ giúp (JOptionPane) với hướng dẫn sử dụng ứng dụng.
- **returnToMainMenu() – *Public***
  - Trở về panel menu chính (MAIN_MENU) từ các màn hình khác bằng cardLayout.show().

SortingPanel: Class SortingPanel là một thành phần giao diện đồ họa (GUI) sử dụng Jpanel để trực quan hóa các thuật toán sắp xếp. Nó cung cấp chức năng hiển thị, điều khiển và trực quan hóa quá trình sắp xếp mảng theo thời gian thực.

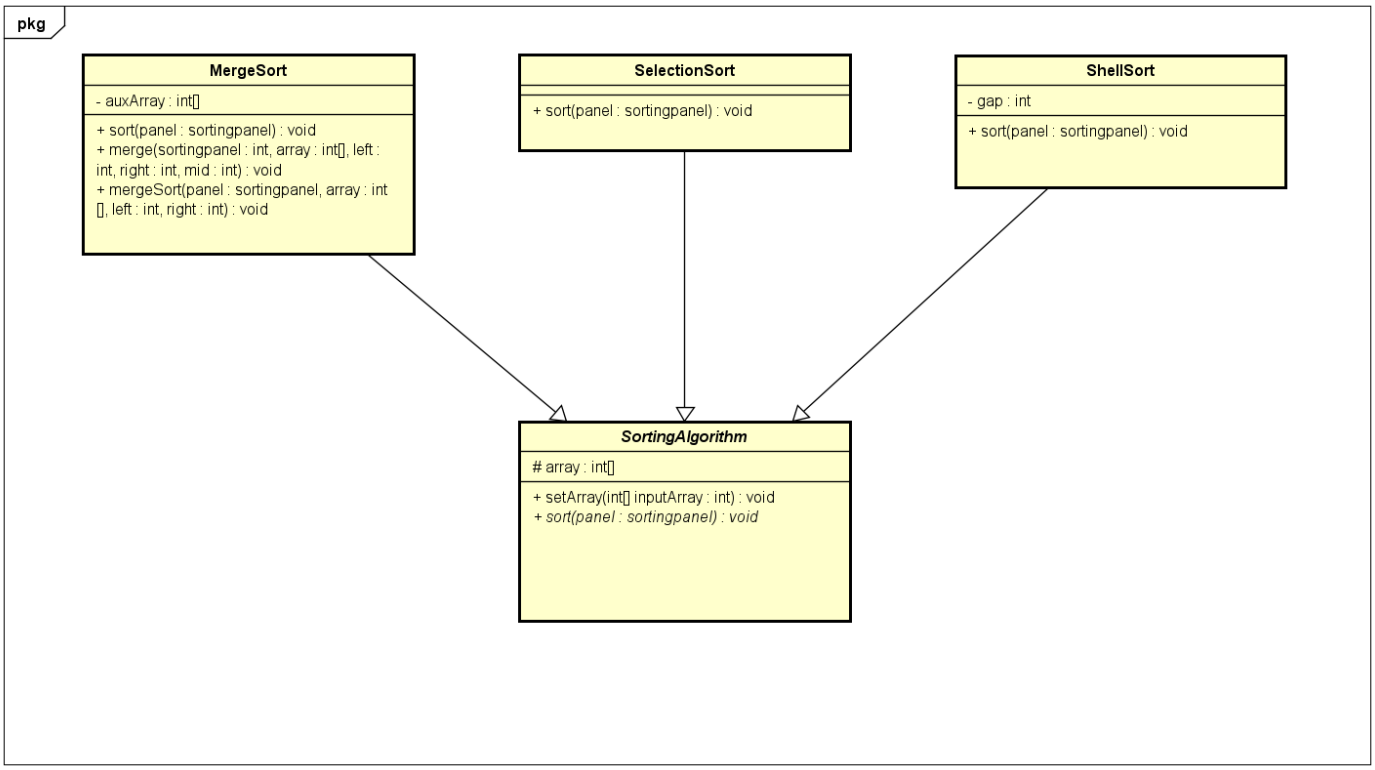
- **setArray(int[] array)**
  - Cập nhật mảng dữ liệu và vẽ lại giao diện.
- **createStatusPanel()**
  - Tạo panel chứa trạng thái và thống kê.
- **createControlPanel()**
  - Tạo các nút điều khiển (Tạo mảng ngẫu nhiên, Nhập mảng, Bắt đầu sắp xếp, Quay lại).
- **startSorting()**
  - Thực hiện sắp xếp trong luồng riêng (Thread).
  - Tính toán thời gian thực thi và cập nhật trạng thái sau khi sắp xếp xong.
- **paintComponent(Graphics g)**
  - Vẽ trực quan các cột biểu thị giá trị của mảng và tô màu theo trạng thái so sánh hoặc hoán đổi.
- **setCompareIndices(int index1, int index2)**
  - Cập nhật chỉ số đang so sánh, vẽ lại và thêm độ trễ.
- **setSwapIndices(int index1, int index2)**
  - Cập nhật chỉ số đang hoán đổi, thực hiện hoán đổi thực tế và trực quan hóa.
- **setMergeIndices(int[] tempArray, int start, int length)**
  - Cập nhật trạng thái của mảng trong thuật toán Merge Sort.
- **resetSortingStats()**: Đặt lại bộ đếm và thời gian.
- **updateStatus(String message, boolean isPositive)**: Cập nhật thông báo trạng thái.
- **updateFinalStats()**: Tính toán thời gian thực hiện và cập nhật số lần so sánh/hoán

đổi.

InputHandling: Mã này xử lý việc nhập mảng để sắp xếp trong ứng dụng Java. Đây là phần hỗ trợ cho phần mềm trực quan hóa thuật toán sắp xếp.

- **getArray()** – Trả về mảng hiện tại.
- **generateRandomArray()**
  - Nhập kích thước mảng từ hộp thoại.
  - Tạo mảng ngẫu nhiên (1-100) và gán cho thuật toán sắp xếp.
- **inputCustomArray()**
  - Nhập mảng từ hộp thoại (chuỗi số, cách nhau bởi dấu phẩy).
  - Kiểm tra và chuyển đổi thành mảng số nguyên.
  - Không cho phép số âm hoặc đầu vào không hợp lệ.

3.2.3. Package application.algorithms



Package application.algorithms bao gồm 3 class **MergeSort**, **SelectionSort** và **ShellSort** và 1 **abstract class SortingAlgorithm**.

SortingAlgorithm:

- **Attribute array**
  - Mảng array lưu trữ các phần tử cần sắp xếp.
- **Method setArray(int[] inputArray)**
  - Tạo bản sao của mảng đầu vào để tránh sửa đổi mảng gốc.
- **Method sort(SortingPanel panel)** ***(abstract)***

- Định nghĩa trừu tượng cho thuật toán sắp xếp, sẽ được hiện thực trong các class con.

SelectionSort:

- **sort(SortingPanel panel)**

  - Phương thức chính thực hiện thuật toán **Selection Sort**.
  - Nhận vào một đối tượng SortingPanel để truy xuất mảng và cập nhật trực quan hóa quá trình sắp xếp.
  - Các bước thực hiện trong phương thức này:
    - Lấy mảng từ SortingPanel.
    - Kiểm tra mảng có hợp lệ không.
    - Duyệt qua mảng để tìm phần tử nhỏ nhất trong phần chưa sắp xếp.
    - Hoán đổi phần tử nhỏ nhất với phần tử đầu tiên của phần chưa sắp xếp.
    - Cập nhật lại mảng trong **SortingPanel**.

MergeSort:

- **sort(SortingPanel panel)**
  - Mô tả: Đây là phương thức chính thực hiện thuật toán Merge Sort và quản lý quá trình trực quan hóa.
  - Chức năng:
    - Lấy mảng từ SortingPanel.
    - Kiểm tra mảng có hợp lệ không (không phải null và có ít nhất 2 phần tử).
    - Khởi tạo mảng phụ auxArray.
    - Gọi phương thức đệ quy mergeSort để chia mảng thành các phần nhỏ hơn và sau đó gộp lại.
    - Cập nhật mảng đã sắp xếp vào SortingPanel sau khi quá trình hoàn tất.
- **mergeSort(SortingPanel panel, int[] array, int left, int right)**
  - Mô tả: Phương thức đệ quy thực hiện việc chia mảng thành các phần nhỏ hơn cho đến khi chỉ còn một phần tử.
  - Chức năng:
    - Phân chia mảng thành các phần con nhỏ hơn.
    - Gọi đệ quy để xử lý các phần mảng con.
    - Sau khi phân chia xong, gọi phương thức merge để gộp lại các phần mảng con đã được sắp xếp.
- **merge(SortingPanel panel, int[] array, int left, int mid, int right)**
  - Mô tả: Phương thức gộp lại hai mảng con đã được sắp xếp (mảng con trái và mảng con phải) thành một mảng hoàn chỉnh.
  - Chức năng:
    - Sao chép mảng gốc vào mảng phụ auxArray để giúp quá trình gộp dễ dàng hơn.
    - So sánh các phần tử từ hai mảng con và gộp chúng vào mảng gốc.
    - Cập nhật trực quan hóa quá trình gộp qua các phương thức như **panel.setCompareIndices()** và **panel.setMergeIndices().**

ShellSort:

- **sort(SortingPanel panel) thực hiện các bước chính sau:**

- Lấy mảng từ **SortingPanel.**
- Kiểm tra mảng có hợp lệ không (không phải null và có ít nhất 2 phần tử).
- Thực hiện thuật toán Shell Sort bằng cách chia mảng thành các phần nhỏ hơn, so sánh và hoán đổi các phần tử trong các vòng lặp con.
- Trực quan hóa quá trình so sánh và hoán đổi qua các phương thức như **panel.setCompareIndices()** và **panel.setSwapIndices().**
- Cập nhật mảng vào SortingPanel sau mỗi vòng lặp và sau khi hoàn tất quá trình sắp xếp.

4 KẾT LUẬN

- Ưu điểm:

- Đáp ứng được các chức năng cơ bản của chương trình sắp xếp mảng theo thuật toán sắp xếp

- Người dùng có thể lựa chọn thuật toán sắp xếp, nhập mảng và nhận được kết quả sắp xếp cùng qúa trình sắp xếp.

- Nhược điểm:

- Thiết kế UI còn có thể cải thiện hơn

- Chưa hiển thị được giao diện với các trường hợp input là số âm

- Mới chỉ có 1 kiểu dữ liệu là integer