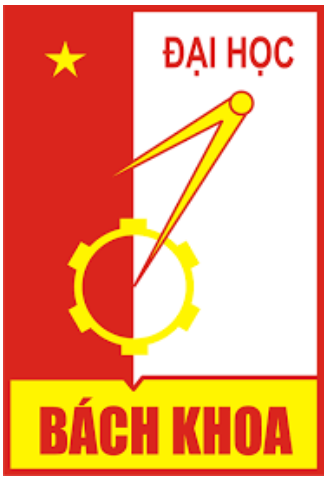
Trường Công nghệ thông tin và Truyền thông
Đại Học Bách Khoa Hà Nội



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 26

Nguyễn Đức Phong	20194642
Nguyễn Hải Phong	20215114
Đào Sỹ Phúc	20215117
Đỗ Quang Phúc	20194646

Hà Nội, ngày 05 tháng 01 năm 2023

Mục lục

Nhóm tác giả và phân chia công việc

Họ và tên	Công việc thực hiện	Đánh giá
1. Nguyễn Đức Phong MSSV: 20194642	• Xử lý UI phần thao tác với dữ liệu (màn hình nhập dữ liệu, các bước sắp xếp và xuất dữ liệu) • Viết báo cáo	Hoàn thành đúng tiến độ
2. Nguyễn Hải Phong MSSV: 20215114	• Xử lý UI phần menu (màn hình main menu, help menu) • Làm slide thuyết trình	Hoàn thành đúng tiến độ
3. Đào Sỹ Phúc MSSV: 20215117	• Xử lý back-end các phần:	Hoàn thành đúng tiến độ
4. Đỗ Quang Phúc MSSV: 20194646	• Xử lý back-end các phần:	Hoàn thành đúng tiến độ
5. Công việc cùng thực hiện	• Lên ý tưởng cho project	Hoàn thành đúng tiến độ

- Link project: https://github.com/ghostoninternet/OOLT.VN.20231-26

Trong quá trình làm project, chúng em đã tham khảo giao diện UI từ project này trên github: https://github.com/thanhlam104/Sorting-Algorithm-Visualization?fbclid=IwAR34jqRnPEWNH-2u_7_gkq1ZHdzr7yrvkCAGfss13LJGVnH42zqCMIKVr68

1 GIỚI THIỆU BÀI TOÁN

1.1 Giới thiệu đề tài

Các thuật toán sắp xếp là nền tảng của cấu trúc dữ liệu và giải thuật. Nắm chắc được cách vận hành và áp dụng các thuật toán sắp xếp sẽ giúp người học lập trình có được kỹ năng và nền tảng vững chắc trong tương lai.

Nhận thấy rằng các kiến thức từ môn học Lập trình hướng đối tượng có thể được áp dụng để giải quyết bài toán sắp xếp mảng bằng thuật toán sắp xếp. Vì vậy, Nhóm 26 chúng em mạnh dạn thực hiện đề tài xây dựng chương trình "Mô tả thuật toán sắp xếp với mảng" nhằm học hỏi, củng cố và phát triển các kĩ năng về thực hành lập trình hướng đối tượng.

1.2. Các tính năng

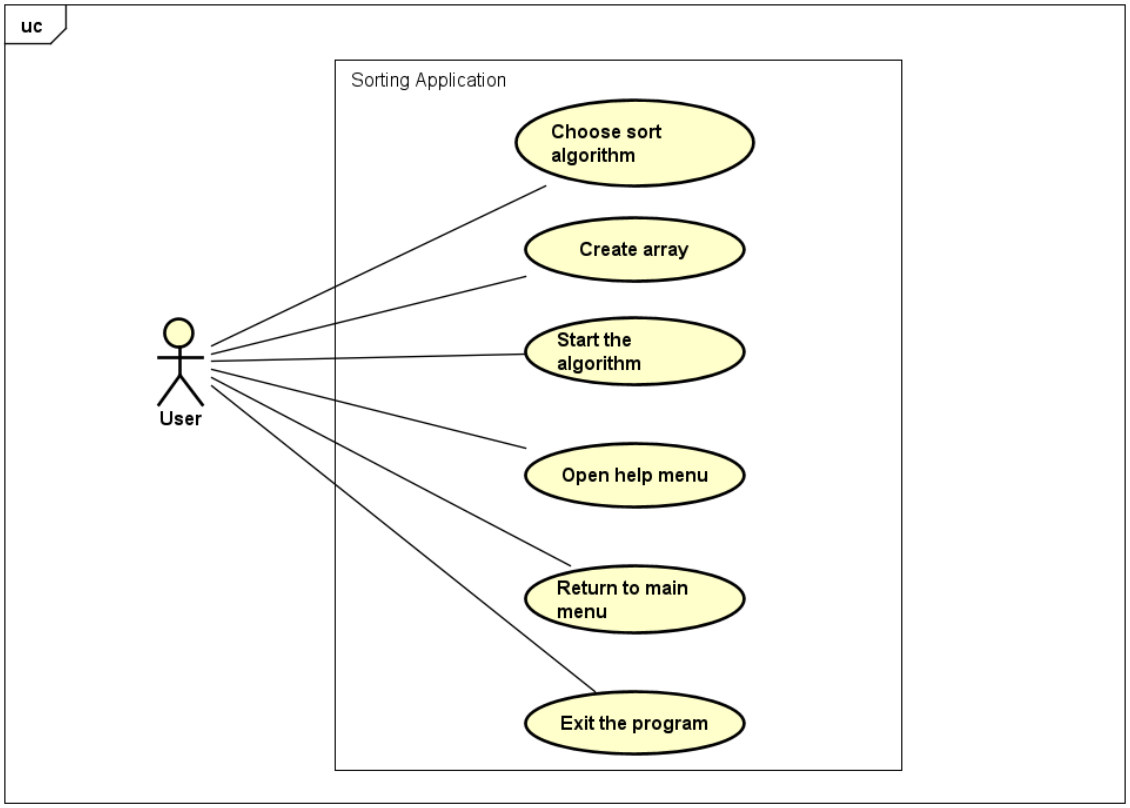
- Tính năng chính của chương trình là thực hiện việc sắp xếp mảng theo các thuật toán sắp xếp có sẵn.
- Người dùng sẽ lựa chọn thuật toán sắp xếp, nhập số lượng phần tử của mảng, nhập các phần tử của mảng.
- Chương trình sẽ thể hiện các bước sắp xếp và xuất ra kết quả sắp xếp

1.3. Công cụ sử dụng

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

2. Phân Tích Biểu Đồ Use Case

2.1. Biểu đồ usecase



2.2. Đặc tả một số Use case

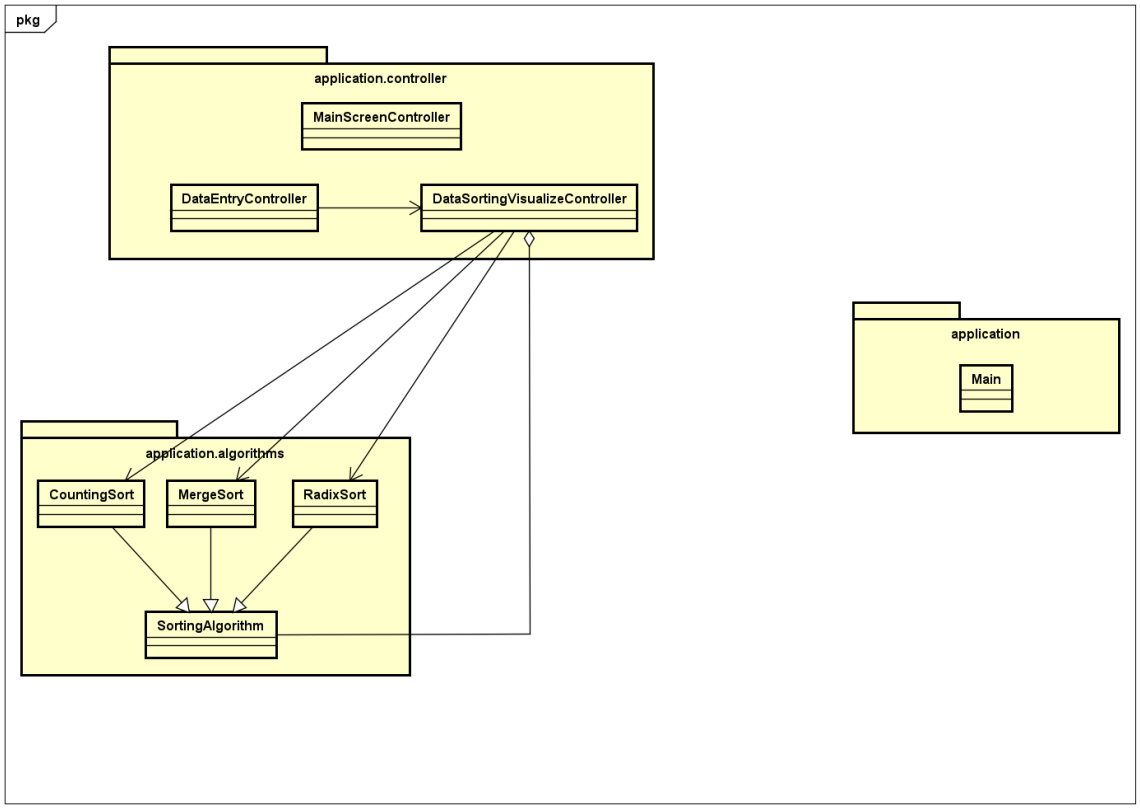
Khi sử dụng ứng dụng này, người dùng có thể thực hiện nhiều thao tác trên đó:

- **Chọn thuật toán (Choose sort algorithm):** Khi mở ứng dụng, người dùng được yêu cầu chọn 1 trong 3 thuật toán sắp xếp: Merge sort, Counting sort và Radix sort.
- **Tạo mảng (Create array):** Sau khi chọn thuật toán, người dùng sẽ được đưa đến màn tạo dữ liệu. Ở đây người dùng nhập số lượng phần tử, có thể lựa chọn tự nhập dữ liệu hoặc tạo dữ liệu ngẫu nhiên.
- **Bắt đầu thuật toán sắp xếp (Start the algorithm):** 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 (Return 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 (Open help menu):** Người dùng có thể mở menu trợ giúp ở trên thanh menu để đọc về thông tin ứng dụng
- **Thoát chương trình (Exit the program):** Người dùng có thể nhấn vào nút exit ở trên thanh menu để thoát ứng dụng

3. Biểu đồ Class Diagram

3.1. Biểu đồ Chung Class Diagram



Phân tích Biểu đồ Chung Class Diagram:

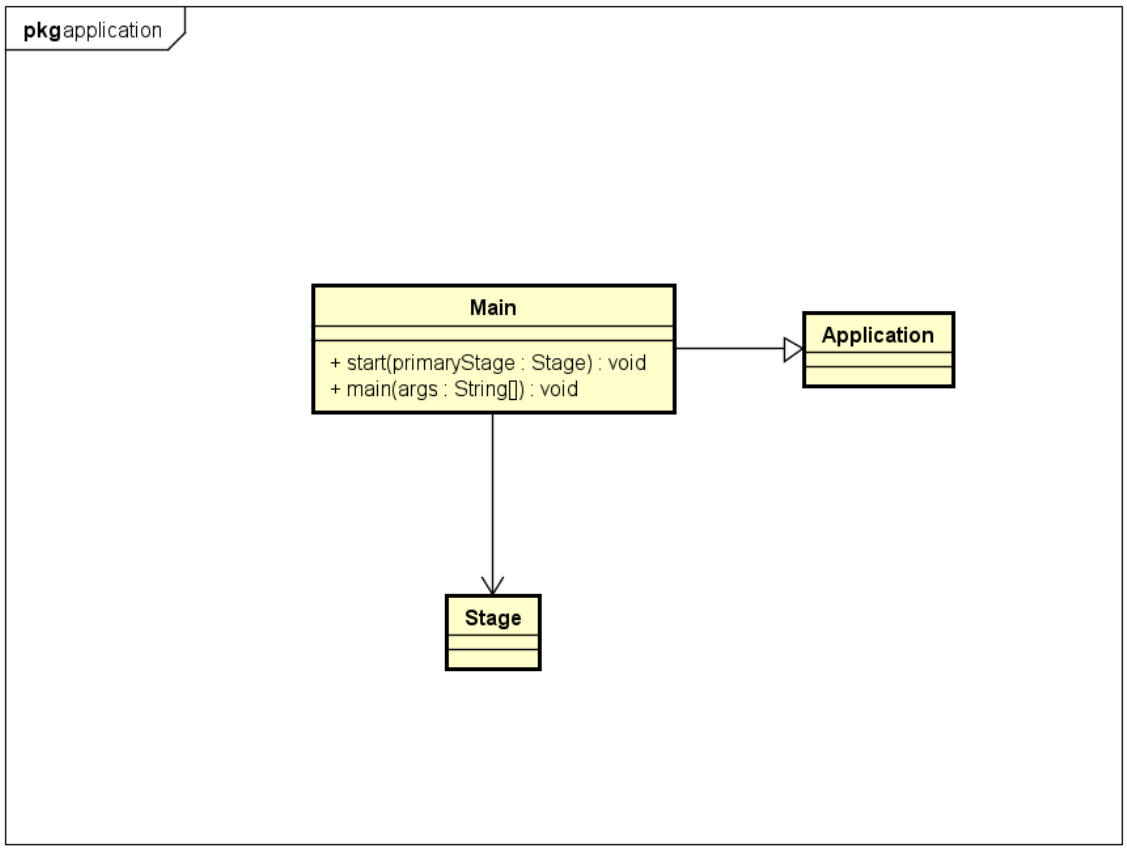
- **Package application:** Package này chứa một class duy nhất là class Main. Class Main được dùng để khởi tạo chương trình lên.
- **Package application.controller:**
  - Package này chứa 3 class được sử dụng làm controller cho các UI.
  - Class MainScreenController được sử dụng để làm controller cho màn menu chính

- Class DataEntryController được sử dụng để làm controller cho màn tạo dữ liệu. Class DataEntryController có sử dụng class DataSortingVisualizationController trong các phương thức.
- Class DataSortingVisualizationController được sử dụng để làm controller cho màn biểu diễn các bước chạy thuật toán sắp xếp. Class này có sử dụng 3 thuật toán tương ứng với 3 class là MergeSort, RadixSort và CountingSort. Ngoài ra class DataSortingVisualizationController còn sử dụng class SortingAlgorithm để tạo thuộc tính kiểu SortingAlgorithm

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

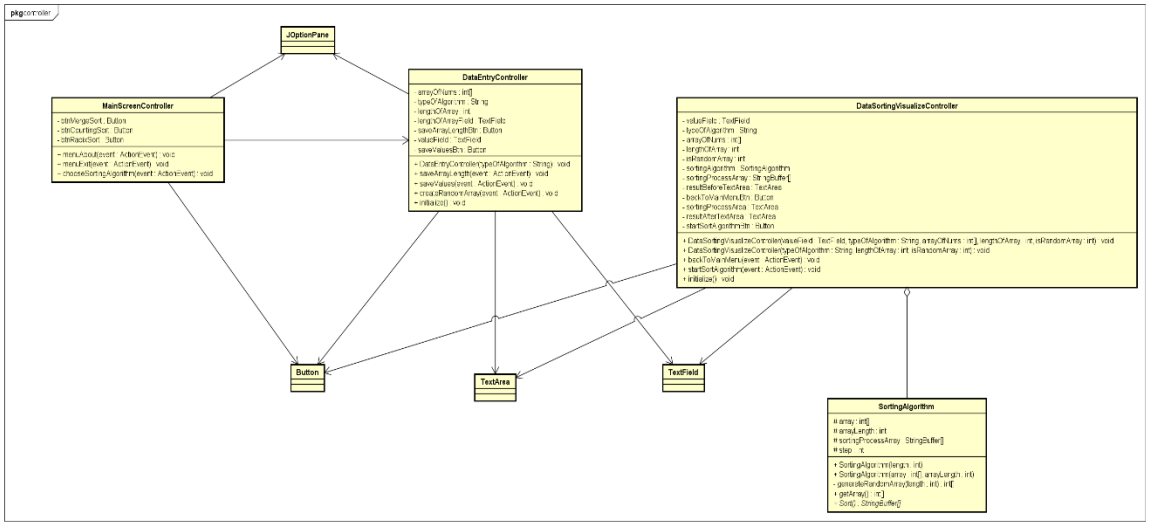
3.2. Biểu đồ Chi tiết Class Diagram

3.2.1. Package application



Package application có một class duy nhất là class **Main**. Class main kế thừa từ class **Application** của JavaFx và có sử dụng class **Stage**. Trong class Main có **override** phương thức **start** của class Application. Nhiệm vụ của class Main là bắt đầu chạy ứng dụng.

3.2.2. Package application.controller



Package application.controller có tất cả 3 class chính: **MainScreenController**, **DataEntryController**, **DataSortingVisualizeController**. Cả 3 lớp này đều sử dụng các class khác từ JavaFx, JoptionPane và class SortingAlgorithm.

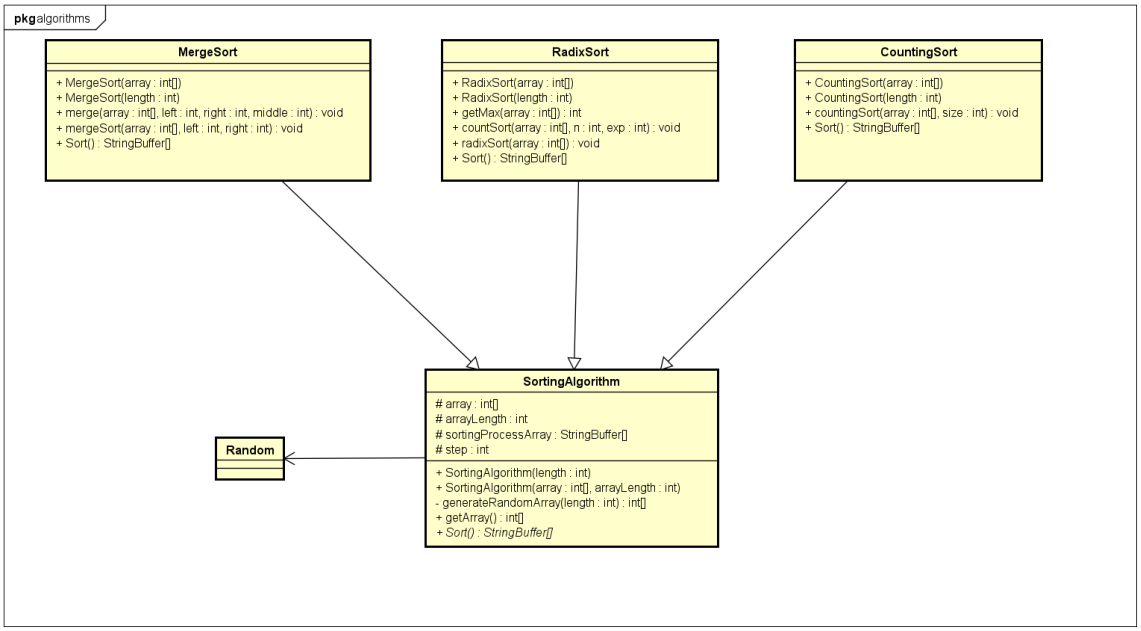
- **MainScreenController:** Class này được dùng để làm controller cho màn menu chính. Class có 3 attribute đều là các component của JavaFx. 3 method có các nhiệm vụ chính sau: chooseSortingAlgorithm
  - **menuAbout(ActionEvent event):** method này dùng để xử lý sự kiện người dùng nhấn vào nút mở menu trợ giúp.
  - **menuExit(ActionEvent event):** method này dùng để xử lý sự kiện người dùng nhấn vào nút thoát.
  - **chooseSortingAlgorithm(ActionEvent event):** method này dùng để xử lý sự kiện người dùng nhấn vào 1 trong 3 nút chọn thuật toán sắp xếp. Khi nhấn vào bất kỳ lựa chọn nào, chương trình sẽ lấy ra lựa chọn thuật toán của người dùng, lưu vào biến và truyền cho controller của màn tạo dữ liệu. Đồng thời sau khi nhấn vào, chương trình sẽ đóng màn menu chính, mở ra màn tạo dữ liệu.
- **DataEntryController:** class này được dùng để làm controller cho màn tạo dữ liệu. Class có 7 attribute, trong đó 3 attribute là do người dùng tạo và 4 attribute là component của JavaFx. Class có 1 constructor, 4 method tất cả.
  - **Attribute arrayOfNums:** dùng để lưu trữ mảng số nguyên do người dùng nhập vào
  - **Attribute typeOfAlgorithm:** dùng để lưu trữ thuật toán sắp xếp mà người dùng đã chọn

- **Attribute lengthOfArray:** dùng để lưu trữ số lượng phần tử mà người dùng nhập vào
- **Constructor DataEntryController(String typeOfAlgorithm):** constructor của class **DataEntryController** với tham số là thuật toán sắp xếp mà người dùng đã chọn
- **Method saveArrayLength(ActionEvent event):** method xử lý sự kiện người dùng nhấn nút saveArrayLengthBtn. Method này xử lý dữ liệu là số lượng phần tử người dùng đã nhập vào, bao gồm bước validate dữ liệu có bị để trống, có bị nhập dữ liệu nào khác số nguyên hay nhập số nguyên âm, nhập số lượng vượt quá giới hạn cho phép.
- **Method saveValues(ActionEvent event):** method xử lý sự kiện người dùng nhấn nút saveValuesBtn. Method này xử lý dữ liệu là các số nguyên người dùng nhập vào, bao gồm bước validate dữ liệu có bị để trống, có nhập thừa số lượng, có nhập dữ liệu nào khác ngoài dữ liệu số nguyên hay không.
- **Method createRandomArray(ActionEvent event):** method này xử lý sự kiện người dùng nhấn nút createRandomArrayBtn. Method này mở luôn màn biểu diễn các bước sắp xếp của thuật toán, truyền vào đó số lượng phần tử người dùng đã nhập để hỗ trợ việc sinh mảng số nguyên ngẫu nhiên.
- **Method initialize():** method này được gọi sau constructor, dùng để đặt các trạng thái ban đầu cho màn tạo dữ liệu: vô hiệu các nút saveValuesBtn và createRandomArrayBtn.

- **DataSortingVisualizeController:** class này được dùng làm controller cho màn biểu diễn các bước sắp xếp thuật toán. Class này có 12 attribute, trong đó 7 attribute do người dùng định nghĩa và 5 attribute là các component của JavaFx. Class còn có 2 constructor và 3 method.
  - **Attribute valueField:** dùng để lưu trữ giá trị mà người dùng đã nhập vào TextField ở màn tạo dữ liệu.
  - **Attribute typeOfAlgorithm:** dùng để lưu trữ thuật toán sắp xếp mà người dùng đã chọn
  - **Attribute lengthOfArray:** dùng để lưu trữ số lượng phần tử mà người dùng nhập vào
  - **Attribute arrayOfNums:** dùng để lưu trữ mảng số nguyên do người dùng nhập vào
  - **Attribute isRandomArray:** được dùng để lưu trữ trạng thái người dùng có đang muốn tạo mảng ngẫu nhiên hay không
  - **Attribute sortingAlgorithm:** được dùng để lưu trữ object được tạo ra từ các class MergeSort, RadixSort và CountingSort.

- **Attribute sortingProcessArray:** dùng để lưu trữ mảng các bước sắp xếp thuật toán
- **Constructor DataSortingVisualizeController (TextField valueField, String typeOfAlgorithm, int arrayOfNums[], int lengthOfArray, int isRandomArray):** constructor được class DataEntryController sử dụng khi người dùng tự tạo mảng dữ liệu
- **Constructor DataSortingVisualizeController (String typeOfAlgorithm, int lengthOfArray, int isRandomArray):** constructor được class DataEntryController sử dụng khi người dùng muốn sinh mảng dữ liệu ngẫu nhiên
- **Method backToMainMenu(ActionEvent event):** method xử lý sự kiện người dùng nhấn vào nút backToMainMenuBtn, đưa người dùng quay lại menu chính
- **Method startSortAlgorithm(ActionEvent event):** method xử lý sự kiện người dùng nhấn vào nút startSortAlgorithmBtn
- **Method initialize():** method được gọi ngay sau constructor, dùng để thiết lập các trạng thái, dữ liệu cần thiết cho màn biểu diễn các bước sắp xếp. Method này khi được gọi sẽ tạo ra các object từ các class MergeSort, RadixSort và CountingSort, lưu vào biến sortingAlgorithm, ghi dữ liệu vào TextField resultBeforeTextArea.

3.2.3. Package application.algorithms



Package application.algorithms bao gồm 3 class **MergeSort**, **RadixSort** và **CountingSort** và 1 **abstract class SortingAlgorithm**.

- **MergeSort**: Class kế thừa class SortingAlgorithm, dùng để cài đặt thuật toán Merge sort. Class bao gồm 2 constructor, 2 method và 1 method được override từ class SortingAlgorithm.
  - **Constructor MergeSort(int[] array, int arrayLength):** constructor được sử dụng để tạo ra object của class MergeSort trong trường hợp người dùng tự tạo mảng dữ liệu
  - **Constructor MergeSort(int arrayLength):** constructor được sử dụng để tạo object của class MergeSort trong trường hợp người dùng muốn sinh mảng ngẫu nhiên
  - **Method merge (int [] array, int left, int right, int middle):** method trộn hai mảng trái và phải sau khi đã chia của thuật toán trộn
  - **Method mergeSort (int [] array, int left, int right):** method bắt đầu thuật toán Merge sort
  - **Method Sort():** method override từ class SortingAlgorithm, dùng cho class DataSortingVisualizeController gọi để bắt đầu thuật toán Merge sort.
- **RadixSort:** Class kế thừa class SortingAlgorithm, dùng để cài đặt thuật toán Radix sort. Class bao gồm 2 constructor, 3 method và 1 method được override từ class SortingAlgorithm.
  - **Constructor RadixSort (int [] array, int arrayLength):** constructor được sử dụng để tạo ra object của class RadixSort trong trường hợp người dùng tự tạo mảng dữ liệu
  - **Constructor RadixSort (int arrayLength):** constructor được sử dụng để tạo object của class RadixSort trong trường hợp người dùng muốn sinh mảng ngẫu nhiên
  - **Method getMax(int [] array):** method được sử dụng để tìm phần tử lớn nhất trong mảng, hỗ trợ cho quá trình sắp xếp
  - **Method countSort (int [] array, int n, int exp, StringBuffer sortingProcess):** method hỗ trợ thuật toán Radix sort
  - **Method radixSort(int [] array):** method thực hiện thuật toán Radix sort
  - **Method Sort():** method override từ class SortingAlgorithm, dùng cho class DataSortingVisualizeController gọi để bắt đầu thuật toán Radix sort.
- **CountingSort:** Class kế thừa class SortingAlgorithm, dùng để cài đặt thuật toán Counting sort. Class bao gồm 2 constructor, 1 method và 1 method được override từ class SortingAlgorithm.
  - **Constructor CountingSort(int [] array, int arrayLength):** constructor được sử dụng để tạo ra object của class CountingSort trong trường hợp người dùng tự tạo mảng dữ liệu

- **Constructor CountingSort(int length):** constructor được sử dụng để tạo object của class CountingSort trong trường hợp người dùng muốn sinh mảng ngẫu nhiên
- **Method countingSort (int [] array, int size):** method dùng để chạy thuật toán Counting sort
- **Method Sort():** method override từ class SortingAlgorithm, dùng cho class DataSortingVisualizeController gọi để bắt đầu thuật toán Counting sort.

- **SortingAlgorithm:** abstract class dùng để định nghĩa ra các attribute và method dùng chung cho các thuật toán sắp xếp khác. Class có 4 attribute ở mức protected, 5 method trong đó có 1 method ở mức private và 1 abstract method. Class SortingAlgorithm có sử dụng class Random từ package java.util.Random để sinh ngẫu nhiên 1 mảng với số lượng phần tử do người dùng cung cấp.
  - **Attribute array:** dùng để lưu trữ mảng dữ liệu do người dùng nhập vào hoặc do tự sinh ngẫu nhiên
  - **Attribute arrayLength:** dùng để lưu trữ số lượng phần tử người dùng nhập vào
  - **Attribute sortingProcessArray:** dùng để lưu trữ mảng các bước của thuật toán sắp xếp
  - **Attribute step:** dùng để lưu trữ bước thứ mấy của thuật toán
  - **Constructor SortingAlgorithm(int[] array, int arrayLength):** constructor được sử dụng trong trường hợp người dùng tự tạo mảng dữ liệu
  - **Constructor SortingAlgorithm(int length):** constructor được dùng trong trường hợp người dùng muốn sinh ngẫu nhiên một mảng
  - **Method generateRandomArray(int length):** method dùng để sinh ngẫu nhiên một mảng số nguyên với độ dài cho trước
  - **Method getArray():** getter của class SortingAlgorithm, dùng để lấy mảng dữ liệu **array**
  - **Method Sort():** abstract method để các class kế thừa class SortingAlgorithm cài đặt thuật toán 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 mô tả các bước sắp xếp từ chương trình.

- Nhược điểm:

- Phần mô tả các bước sắp xếp chưa được hoàn thiện, chương trình mới chỉ cung cấp cho người dùng các bước sắp xếp mà không có hình ảnh minh họa

- Thiết kế UI còn có thể cải thiện hơn