





Selection Sort

PROBLEM

Explaining and visualizing different sorting algorithms



Merge Sort

Design

Design an UI for the problem with different features



Shell Sort

TARGET

Help users have a better insight about how algorithms work

0

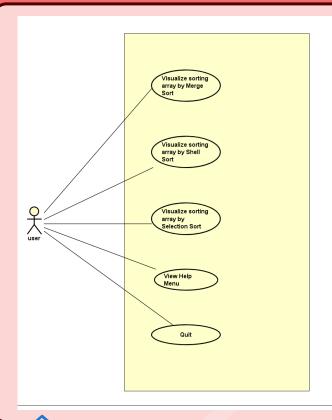
USE CASE DIAGRAM

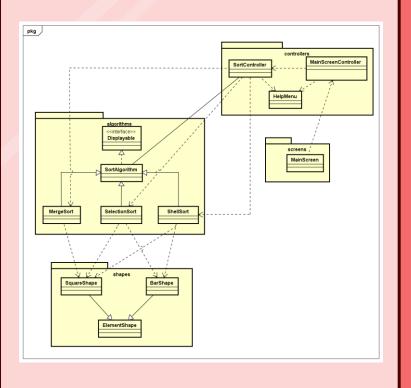
Users can start visualizing by selecting a sort type

- Take all user command from the GUI
- Notice the user if there is anything wrong with his input
- o Run the designated algorithm
- Run/Reset the visualization based on user command.

Users can view the help menu

Users can quit the program





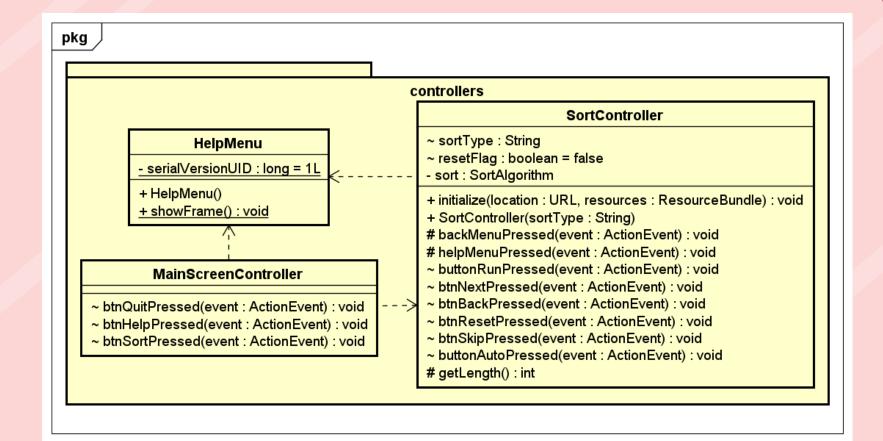
CLASS DIAGRAM

Package "algorithms": store 3 sorting algorithms for visualizing, and an abstract class SortAlgorithm to store common operations, attributes

Package "controllers": store all the screen controllers

Package "screens": store the main screen class of the application

Package "shapes": store the abstract class "ElementShape" and 2 derived shapes, namely SquareShape and BarShape.



algorithms

SortAlgorithm # n:int # arr : int[] # displayType : String # curSteps : int # numSteps : int # progressField : TextArea ---- + displayStartScreen(): void # pane : Pane + getN(): int + getCurSteps(): int + getNumSteps(): int + SortAlgorithm(inputArr : int[], inputPane : Pane, inputProgressField : TextArea, inputDisplayType : String) # swap(posi : int, posj : int) : void

SelectionSort

<<interface>>

Displayable

+ displayFinishScreen(): void

+ nextStep(): void

+ previousStep(): void

startX : double

startY : double

- ratio : double - max : double

steps : int[][]

arrState : int[][] - flags : int[]

addState(i:int, j:int, flag:int):void

+ ShellSort(inputArr: int[], inputPane: Pane, inputProgressField: TextArea, displayType: String)

ShellSort

+ displayStartScreen(): void

+ nextStep() : void

+ previousStep() : void

+ displayFinishScreen(): void

+ drawBottom(arr : int[]) : void

+ drawArray(arr : int[], pos : int[], c : Color[]) : void

+ drawArrayAnimation(arr : int[], pos : int[], X : double, Y : double, type : int) : void

X : double

- Y : double flags : String∏

loops : int[][] = new int[100][2]

- arrState : int∏∏

+ SelectionSort(inputArr : int[], inputPane : Pane, inputProgressField : TextArea, inputDisplayType : String)

+ nextStep() : void

+ displayStartScreen(): void + previousStep() : void

+ displayFinishScreen() : void

+ drawElement(element : int, X : double, Y : double, c : Color, c2 : Color) : StackPane

+ drawElement(height : int, c : Color, X : double, Y : double) : StackPane

+ drawArray(arr : int[], seperate : int, minIndex : int, index : int, X : double, Y : double) : StackPane[]

+ drawArray(arr : int[], seperate : int, index : int, X : double, Y : double, minIndex : int) : StackPane[]

MergeSort

- cloneArr : int[]

startX : double

- startY : double

- steps : double[[[]

instructions : String[]

+ MergeSort(inputArr:int[], inputPane: Pane, inputProgressField: TextArea, inputDisplayType: String)

- merge(start : int, end : int, mid : int, midX : double, startY : double, startLeftX : double, startRightX : double) : void

- merge_sort(start : int, end : int, startX : double, endX : double, startY : double) : void

- assignSteps(args1 : double, args2 : double, args3 : double, args4 : double, color : int, instruction : int) : void

+ displayStartScreen(): void

+ nextStep() : void

+ previousStep(): void

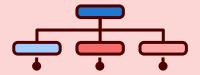
+ displayFinishScreen(): void

- drawArray(arr : int[], c : Color[], startX : double, startY : double) : void

- displayStep(stepNum : int) : void

pkg shapes **BarShape** + BarShape(X : double, Y : double, height : double, c : Color) + BarShape(X : double, Y : double, height : double, c1 : Color, c2 : Color, element : int) ElementShape - rectangle : Rectangle + movingXY(X : double, Y : double) : TranslateTransition + movingX(X : double) : TranslateTransition + movingY(Y : double) : TranslateTransition **SquareShape** + SquareShape(X : double, Y : double, c1 : Color, c2 : Color, element : int)

OOP TECHNIQUES





Inheritance

- In package algorithms: three sort types inherit from SortAlgorithm
- In package shapes: two shape type inherit from ElementShape

Polymorphism

- Method polymorphism: different drawElement and drawArray in SelectionSort method to visualize different shapes
- Object polymorphism: All specific sort algorithms are upcasted to the abstract class SortAlgorithm, which implements the interface Displayable

