Mariam Tsirekidze

Id: 823460489

Report

Sort-O-Matic

Problem:

Create a windowed program that will give the user a choice of generating an array of integers (or doubles) that is:

- random
- data is in ascending order
- data is in descending order

The contents of the array should then be displayed on the form. The size of the array is up to you. You can allow the user to choose the size, but make sure it's not so big such that the contents can't be displayed on the form.

Allow the user to choose from a variety of sorting routines (implement at least 3 of these for your program):

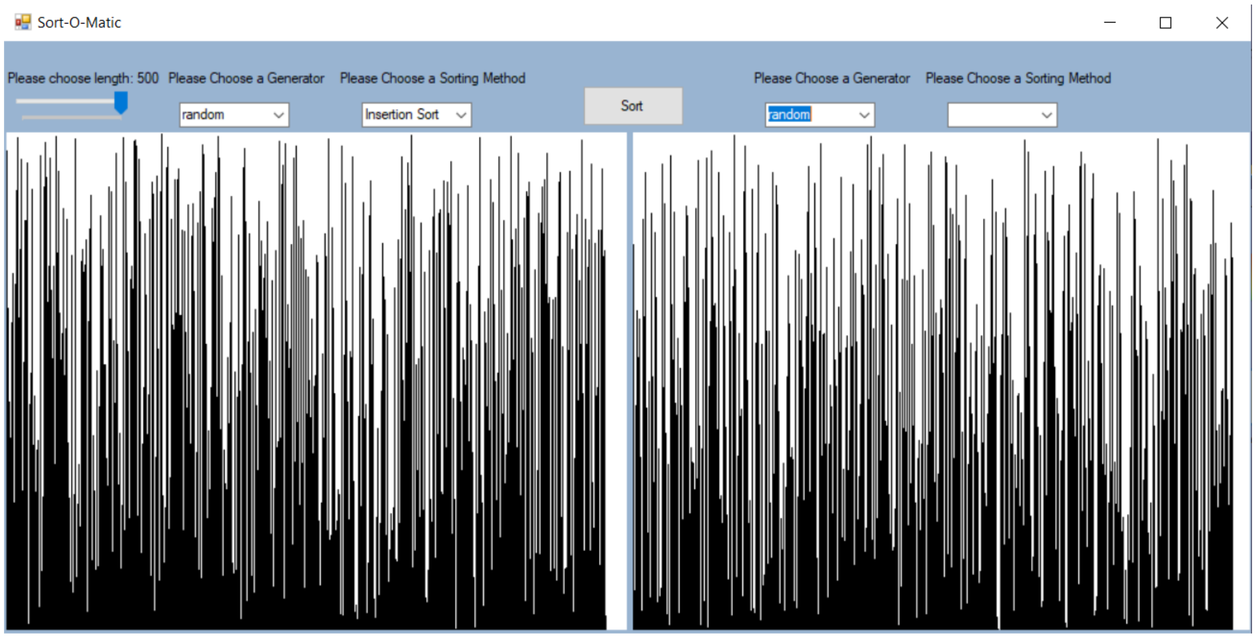
- insertion sort
- quick sort
- selection sort
- Bubble sort (My choice)

Your program should show at the very least how the data in the array is moved, step by step, based on the chosen sorting algorithm. This is where you should incorporate "animation." For each step you can require user input before continuing to the next step or you can use a timer to provide a sufficient delay between the steps.

Solution:

I created form which has size picker, thus the user is able to choose the size of the array, you can choose the size between 200-500. Then the user should choose structure of data for both graph: random, in ascending order or in descending order. Then user should choose sorting method for both graph: Insertion Sort, Quick sort, Bubble Sort (Added according to my choice), Selection Sort. Final step is to press Sort button.

Both Graphs are starting sorting at the same time, in that case I needed asynchronous code that's why I used `System.Threading.Tasks;` and used `Task.Run(() =>{});` for asynchronous workflow.


Sort-O-Matic
Please choose length: 500
Please Choose a Generator
random
Please Choose a Sorting Method
Insertion Sort
Sort
Please Choose a Generator
random
Please Choose a Sorting Method

I have interface interface ISortEngine which has one function void DoWork(int[] MyArray, Graphics graph, int UpperBound);

All my 4 classes implements ISortEngine interface:

- class SelectionSort : ISortEngine
  1. For each item in the array
  2. assume the first item is the smallest value
  3. Cycle through the rest of the array
  4. If any of the remaining values are smaller, find the smallest of these
  5. Swap the found-smallest value with the current value
- class QuickSort:ISortEngine
  1. pivot is last element
  2. For each element from index of smaller element to the last element
  3. If current element is smaller than the pivot
  4. swap values
- class InsertionSort : ISortEngine
  1. while is not sorted
  2. for every element i that is from 1 to array length
  3. while the next number is greater then previous, swap values.
- class BubbleSort : ISortEngine
  1. while the array is not sorted
  2. for every number that is from 0 to array length
  3. if current element is greater then next then swap.

In my form class I have private properties:

int[] MyArray; ------ for first array

int[] MyArray2;-----for second array

Graphics graph;----for first graph

Graphics graph2;--for second graph

int length;----------- for length of both array

string method1;----save which sort method is selected for first graph

string method2;--- save which sort method is selected for second graph

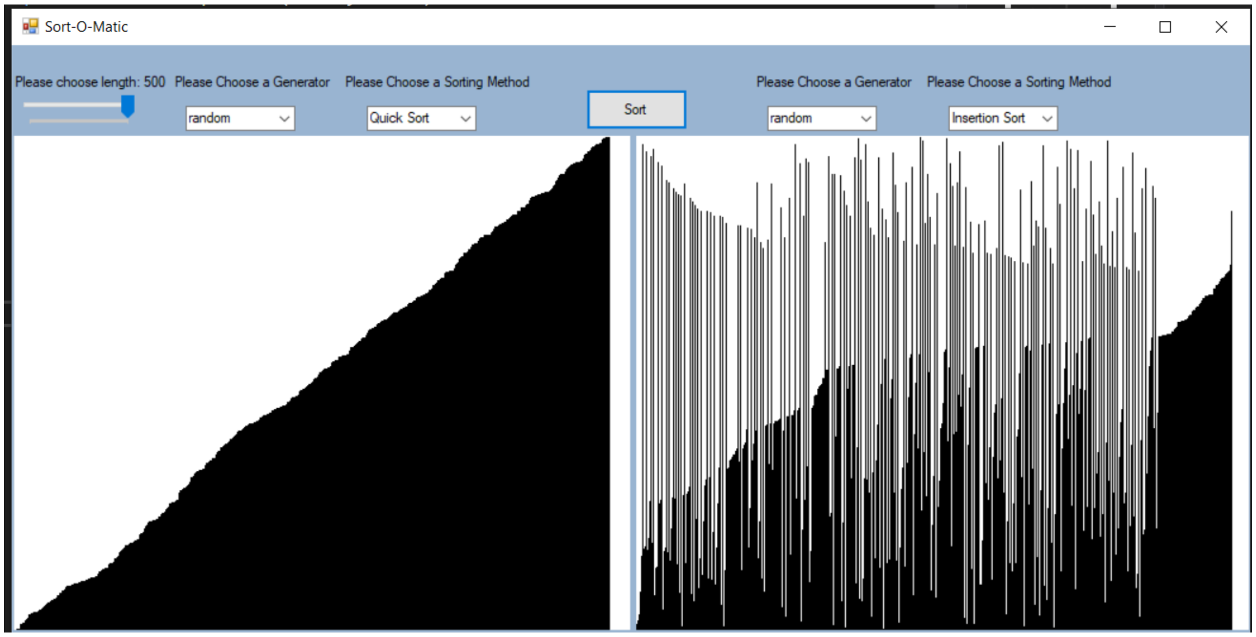
in the method private void trackBar1_Scroll(object sender, EventArgs e) I am assign this.length with the value picked by the user

in the method private void comboBox2_SelectedIndexChanged(object sender, EventArgs e) I am defining this.method by the sorting method with the user picked

in the method private void Generator_SelectedIndexChanged(object sender, EventArgs e) I am defining the sorting routine and output it on the panel

in my method private async void button1_Click(object sender, EventArgs e) if the sort button is cklicked then start specific sorting method for both arrays and output it on the panels.

Because I have asynchronous code and two panels we can compare two sorting methods with each other like this, quick sort is way better than insertion in case of unordered data



Please choose length: 500

Please Choose a Generator: random

Please Choose a Sorting Method: Quick Sort

Please Choose a Generator: random

Please Choose a Sorting Method: Insertion Sort