GUIDE
QUICKSORT PROGRAM

BY RAFAŁ MEISEL
Date: 06.06.2017

1. Program description:

Application was created to sort numeric data randomly generated or from file.
It was used QuickSort algorithm with Algorithmic complexity n(log(n)).

In the code are 2 classes: Main (where is main control of program) and Sorting (where actualy sorting is happen) and 1 layout: GUI.fxml.

Main class is :
- connecting logic with layout
- controlling buttons and text fields
- show information about finished tasks
- reading file with data and generating random data.

Sorting class is:
- sorting with algorithm (it is on the end of this guide)
- exchange (swap) the data in array list
- saving results to file

GUI layout consists of:
- HBOX and VBOX
- Button control
- Text Fields
- Labels

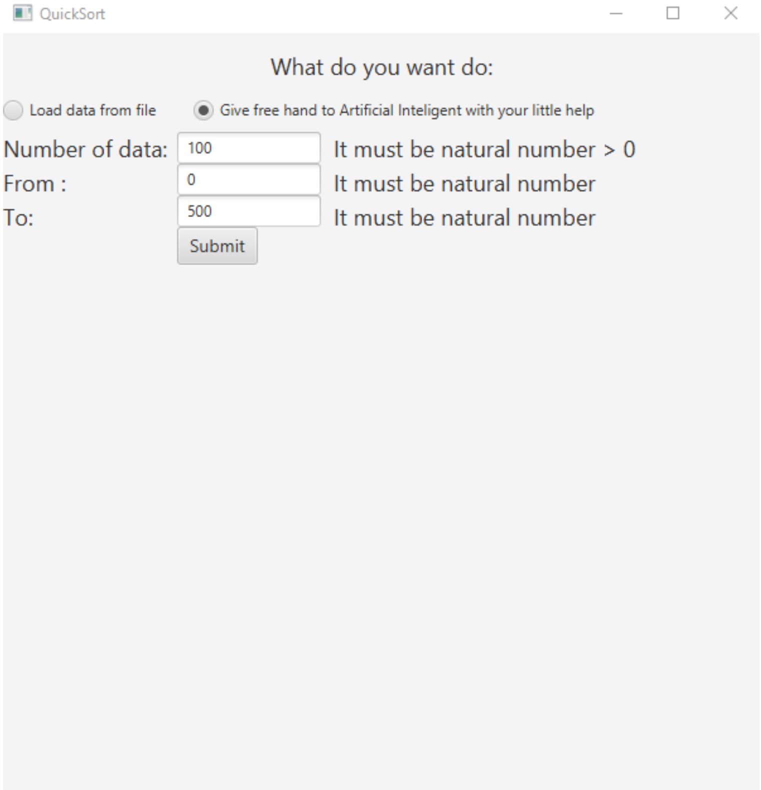
2.Lauch program

Aplication consists of 1 files:
-QuickSort.jar (executable file)

Before lauch application you must be sure that you have these files.

To run application, you need to double-click on QuickSort.jar.
(Information: if input.txt don't exists - it's create automatically).

After that, you will see main windows of application.

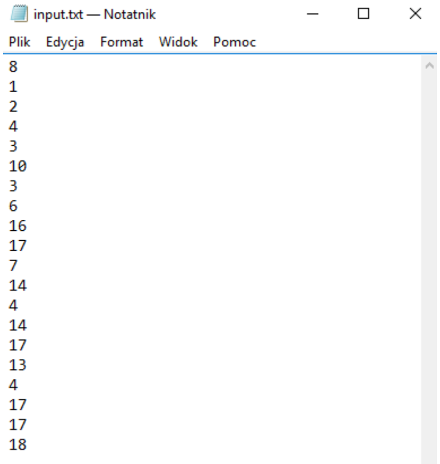


What do you want do:

○ Load data from file ● Give free hand to Artificial Inteligent with your little help

Number of data: 100 It must be natural number > 0
From : 0 It must be natural number
To: 500 It must be natural number

Example data from input.txt

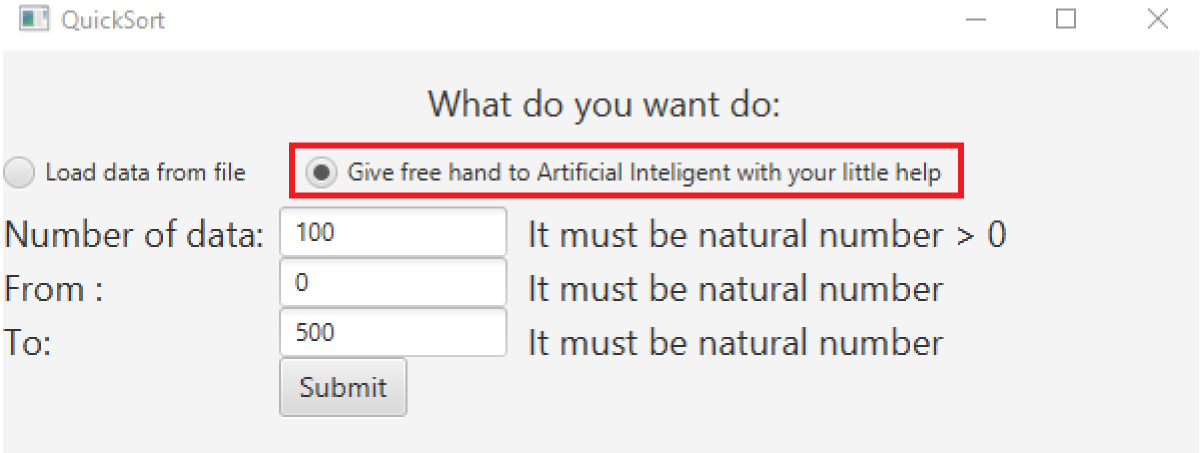


```
8
1
2
4
3
10
3
6
16
17
7
14
4
14
17
13
4
17
17
18
```

3.Select method to input data:

In program are 2 methods to input data:
- you can modify file "input.txt" by yourself
- you can input parameters and application will generate input data automatically

a) Random generate data
Below is option to generate input data. As parameter you have to give:
Number of data - how many data (numbers) it have to generate
From - lower index of range
To - higher index of range
*Remember that all values must be natural number!

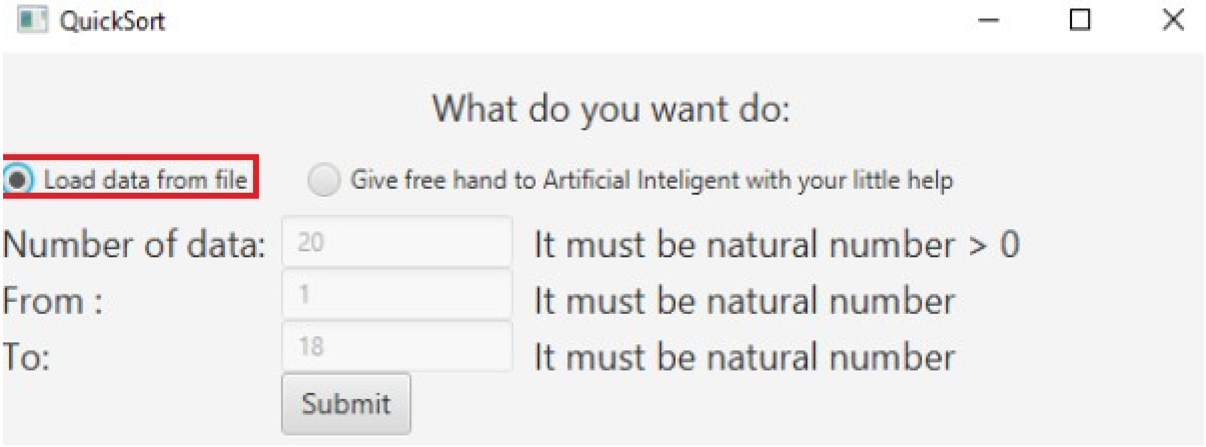


What do you want do:

○ Load data from file   ● Give free hand to Artificial Inteligent with your little help

Number of data: 100 It must be natural number > 0
From : 0 It must be natural number
To: 500 It must be natural number

b) Data loaded from file "input.txt"
In second option, application read input file that we prepared earlier. When we choose this option we can see that in all boxes, already are values.



What do you want do:

● Load data from file   ○ Give free hand to Artificial Inteligent with your little help

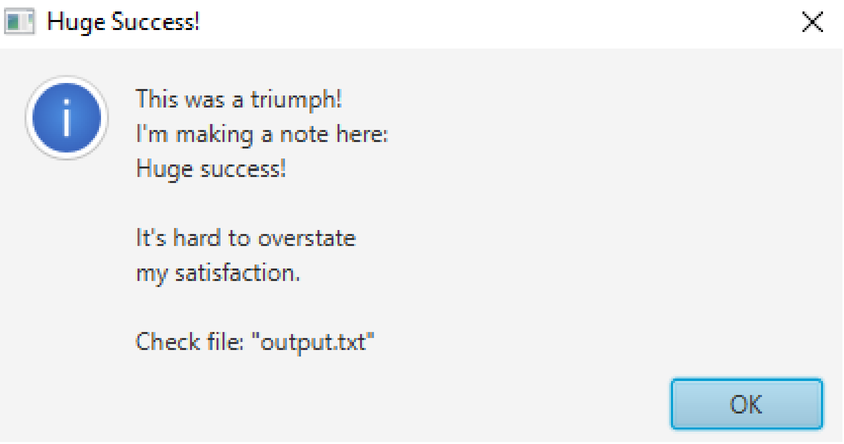
Number of data: 20 It must be natural number > 0
From : 1 It must be natural number
To: 18 It must be natural number

*Remember that all data in "input.txt" file have to be one under one (like on the picture in chapter 1 - "Lauch program").

Finally, you should click "Submit" button to sort data with QuickSort algorithm.

4.Results:

When program finish sorting our data, you get information:



This was a triumph!
I'm making a note here:
Huge success!

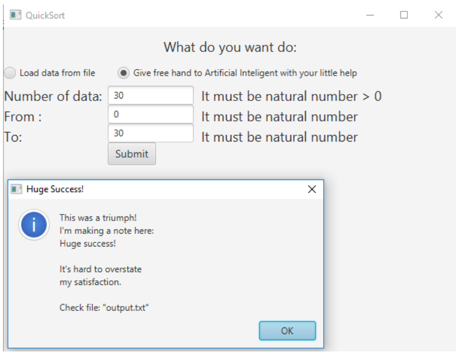
It's hard to overstate
my satisfaction.

Check file: "output.txt"

In case that nothing happen - check if you write correct data into text fields or in input file.

If you compare resultats, you can see that program worked properly.

For custom data:



What do you want do:

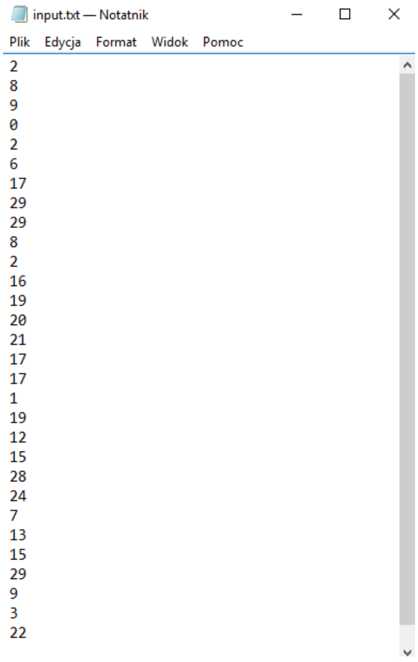
○ Load data from file ● Give free hand to Artificial Inteligent with your little help

Number of data: 30 It must be natural number > 0
From : 0 It must be natural number
To: 30 It must be natural number

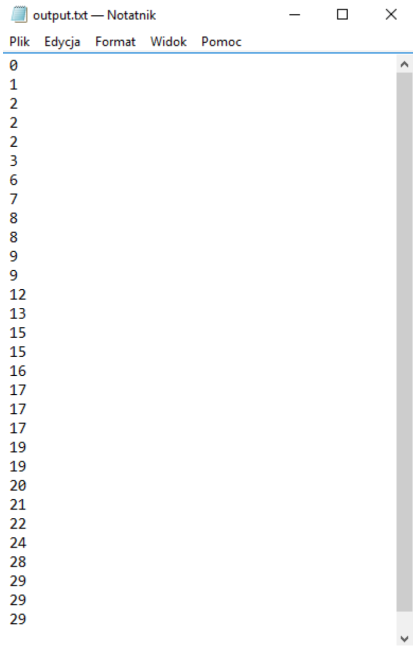
This was a triumph!
I'm making a note here:
Huge success!

It's hard to overstate
my satisfaction.

Check file: "output.txt"

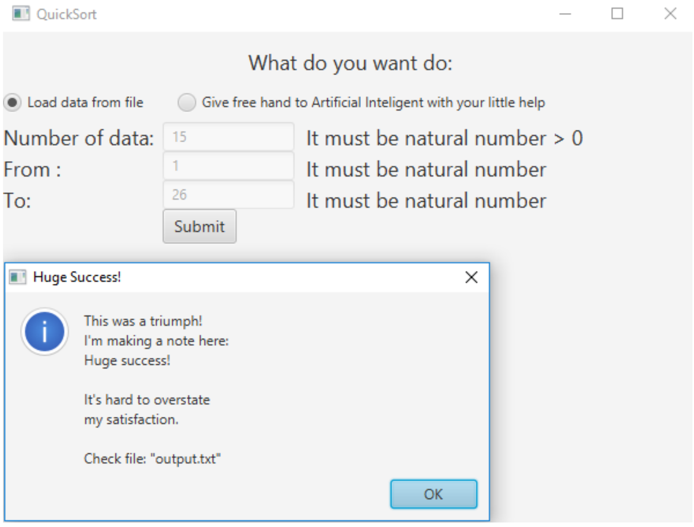


```
2
8
9
0
2
6
17
29
29
8
2
16
19
20
21
17
17
1
19
12
15
28
24
7
13
15
29
9
3
22
```



```
0
1
2
2
2
3
6
7
8
8
9
9
12
13
15
15
16
17
17
17
19
19
20
21
22
24
28
29
29
29
```

For loaded data:



What do you want do:

● Load data from file ○ Give free hand to Artificial Inteligent with your little help

Number of data: 15 It must be natural number > 0

From : 1 It must be natural number

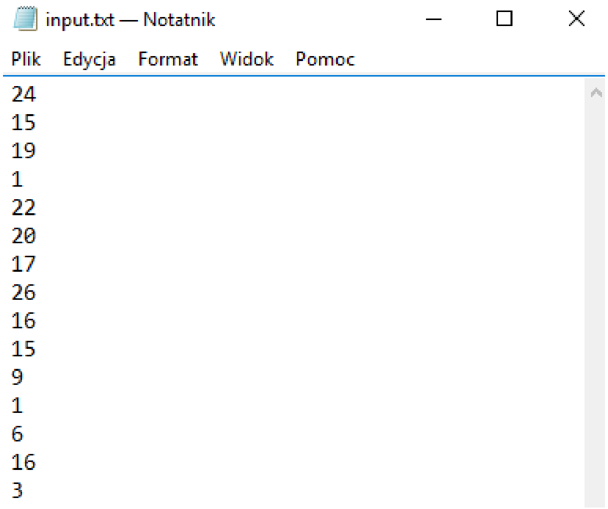
To: 26 It must be natural number

Huge Success!

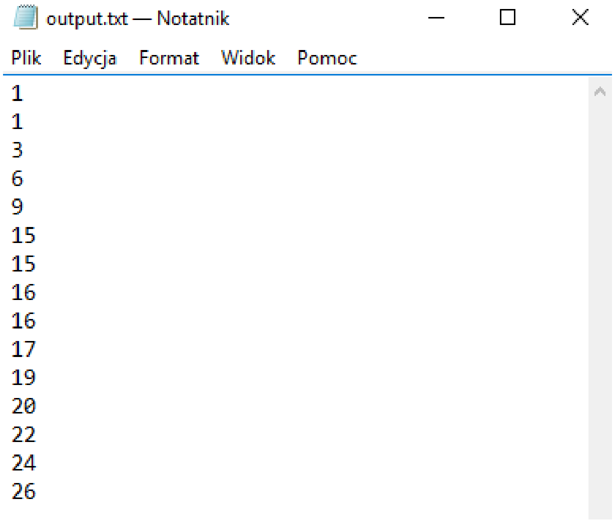
This was a triumph!
I'm making a note here:
Huge success!

It's hard to overstate
my satisfaction.

Check file: "output.txt"



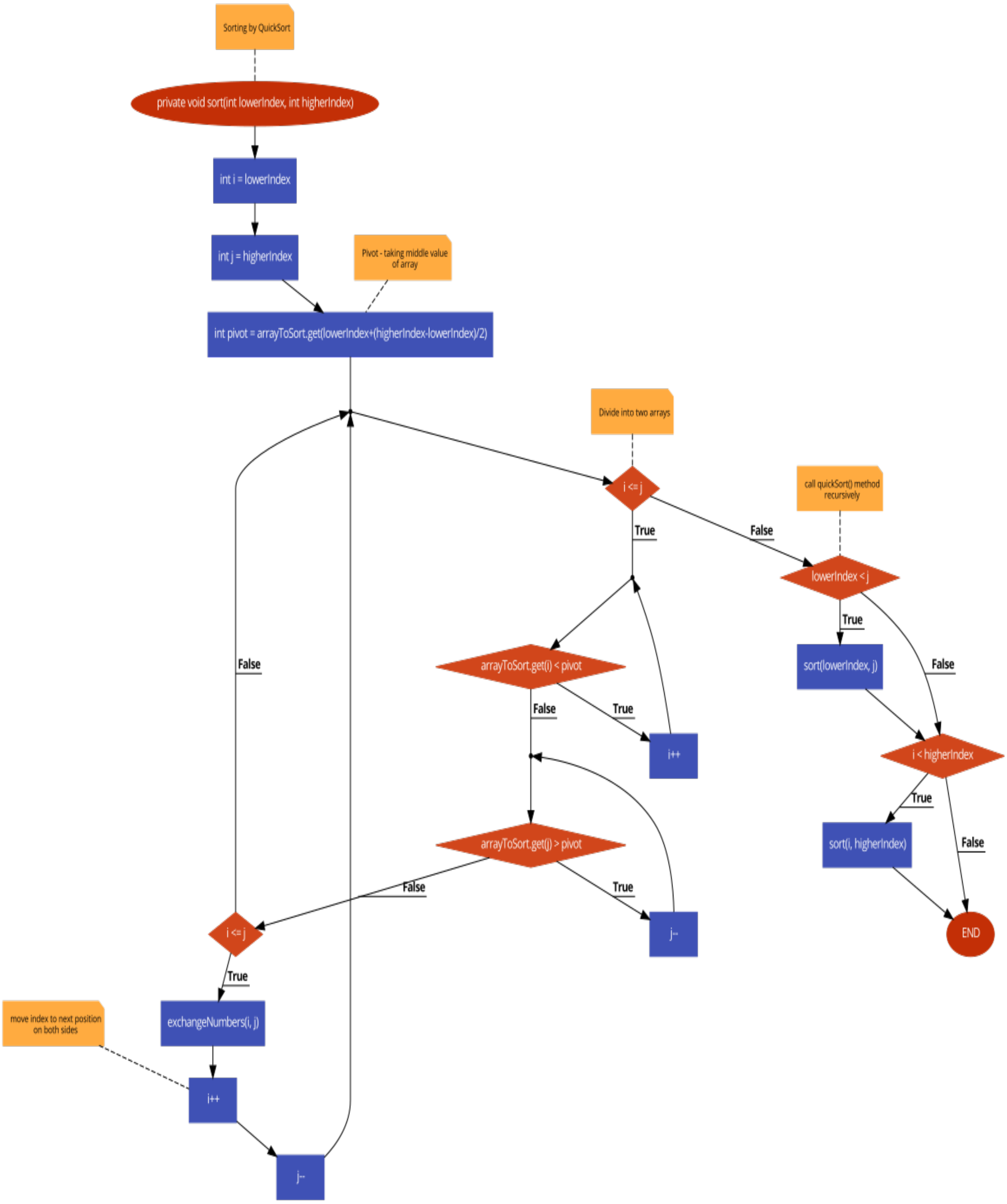
```
24
15
19
1
22
20
17
26
16
15
9
1
6
16
3
```



```
1
1
3
6
9
15
15
16
16
17
19
20
22
24
26
```

5.Theory:

Application is using QuickSort algorithm. It was implemented in this way:


Sorting by QuickSort
private void sort(int lowerIndex, int higherIndex)
int i = lowerIndex
int j = higherIndex
Pivot - taking middle value of array
int pivot = arrayToSort.get(lowerIndex+(higherIndex-lowerIndex)/2)
Divide into two arrays
i <= j
True
False
call quickSort() method recursively
lowerIndex < j
True
sort(lowerIndex, j)
False
i < higherIndex
True
sort(i, higherIndex)
False
END
False
arrayToSort.get(i) < pivot
False
True
i++
arrayToSort.get(j) > pivot
False
True
j--
i <= j
True
exchangeNumbers(i, j)
move index to next position on both sides
i++
j--