İZMİR KÂTİP ÇELEBİ UNIVERSITY

2010

INTRODUCTION TO ALGORITHM

EEE321

Python Project

Beyza Sayracı

170403034

Volkan Kılıç

PYTHON FINAL PROJECT

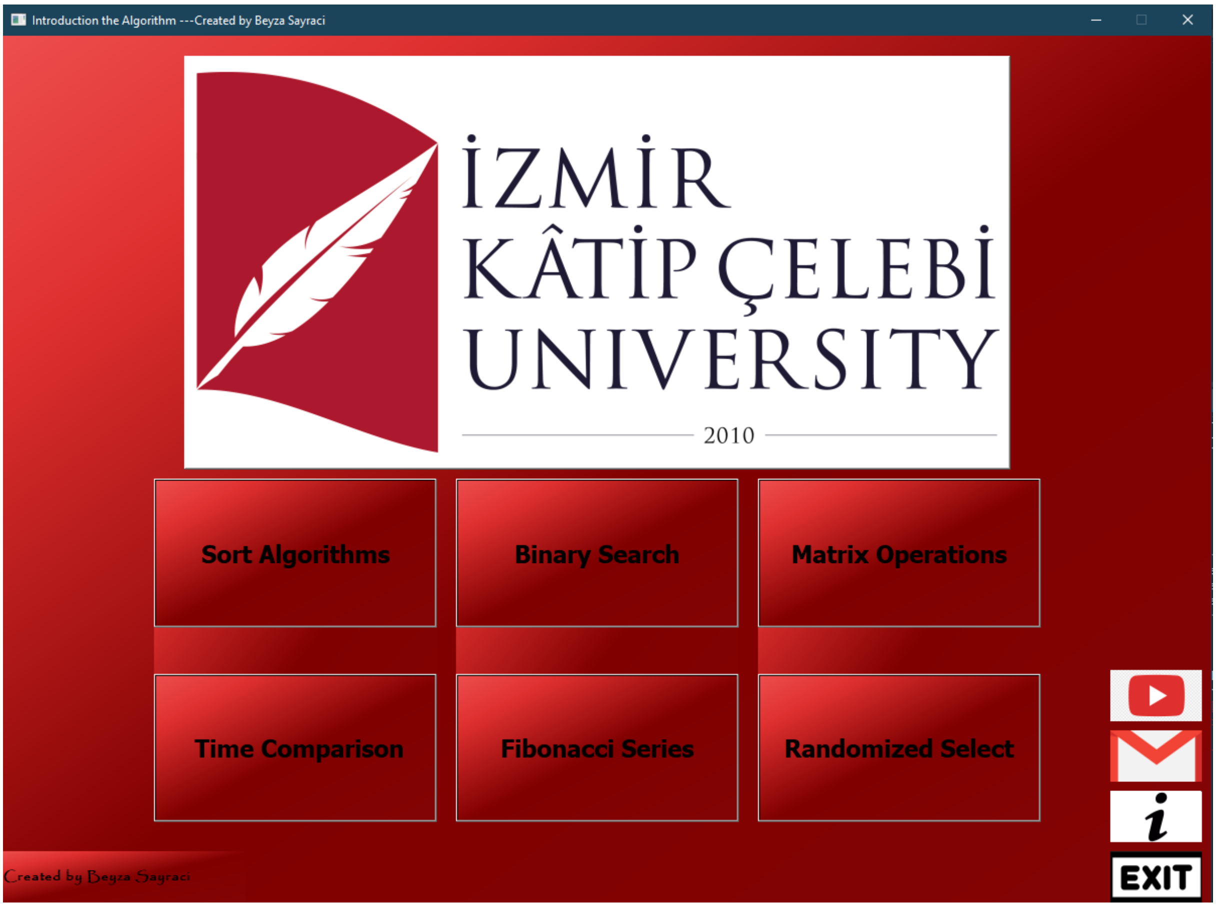
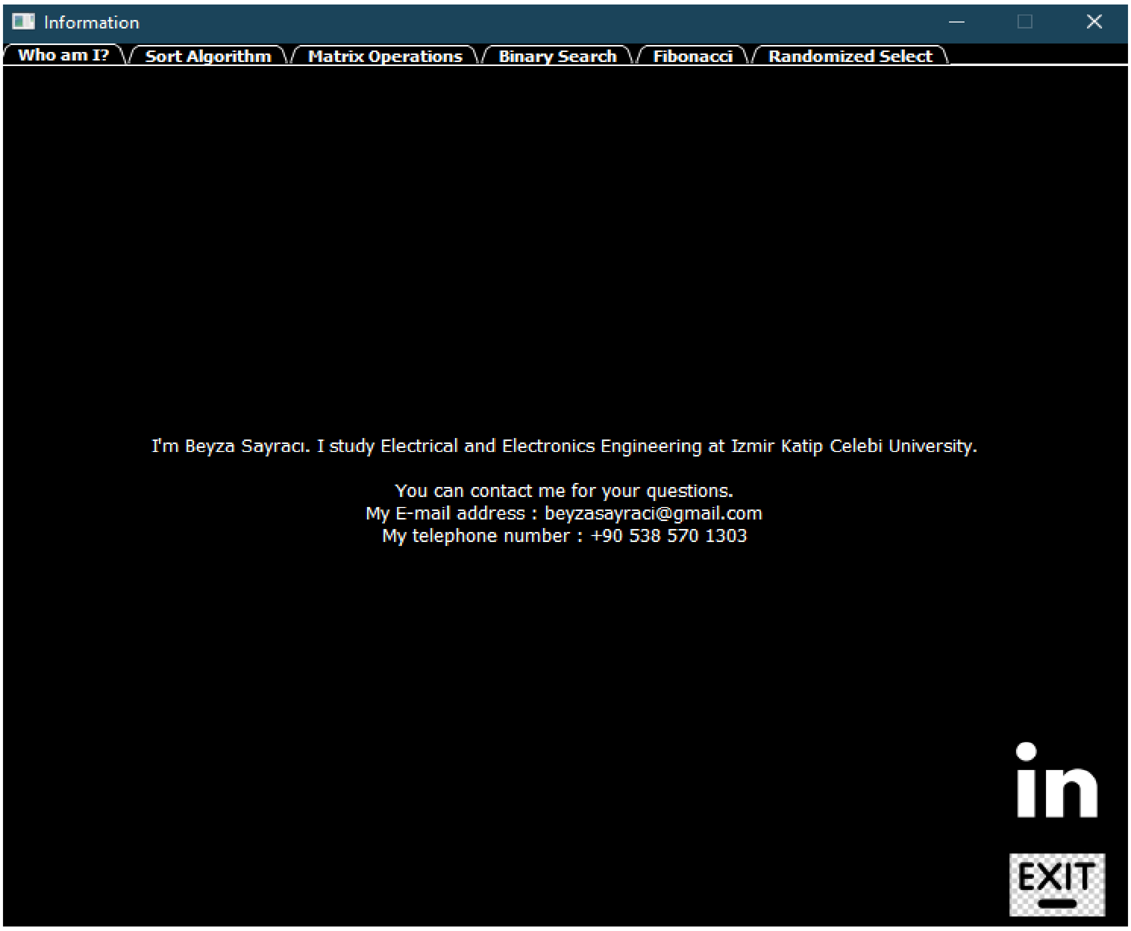


Figure 1

We will examine the python part of my algorithm lesson project homework. A login screen [Figure 1] welcomes us. There are buttons for the screens we want to see here. We can view the screens by clicking them. The information section is available here [Fig. 2]. You are directed to the pages I use on the information page. There is also information on how to use those pages here. You can send your feedback to me via the Gmail section [Fig. 3]. You can access my linkedin page with the Linkedin button, our youtube page with the link with the youtube button and our school page with the ikçü logo. There is also a section on the info page where I briefly mentioned myself.



I'm Beyza Sayracı. I study Electrical and Electronics Engineering at Izmir Katip Celebi University.

You can contact me for your questions.
My E-mail address : beyzasayraci@gmail.com
My telephone number : +90 538 570 1303

Figure 2

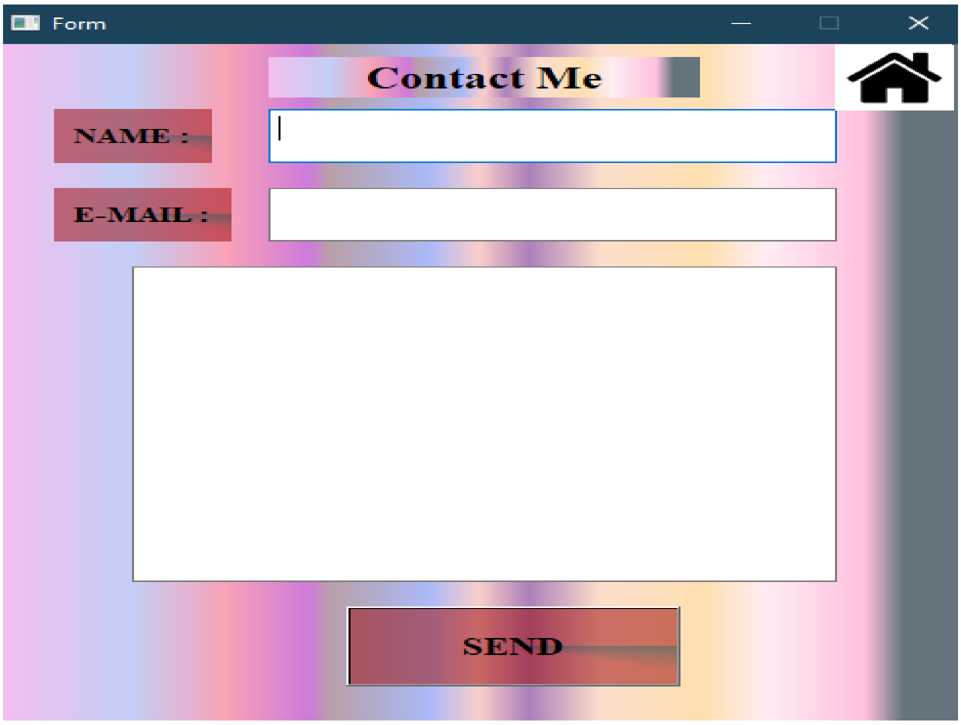
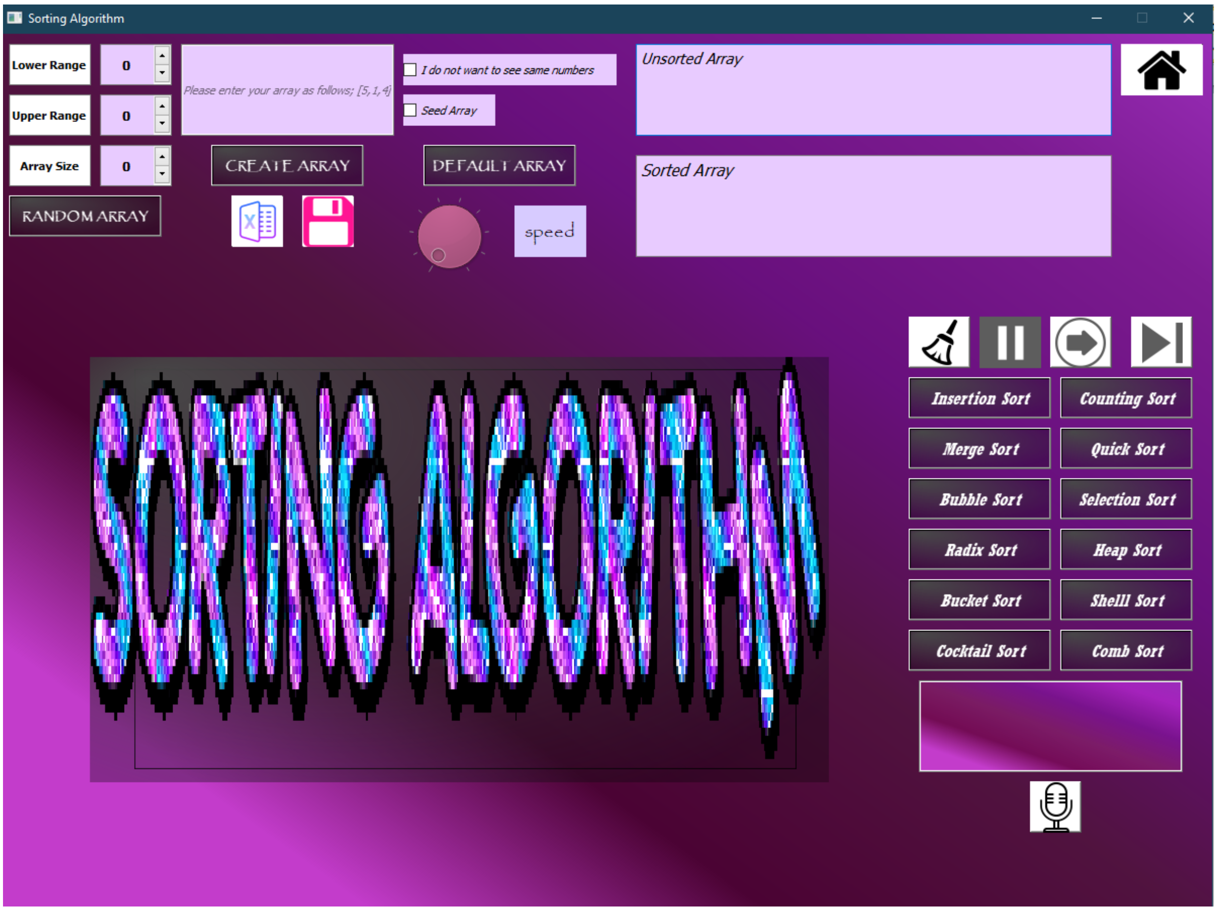


Figure 3

When we come to our first page, sorting algorithms meet us [Figure 4].


Sorting Algorithm

Lower Range: 0

Upper Range: 0

Array Size: 0

☐ I do not want to see same numbers

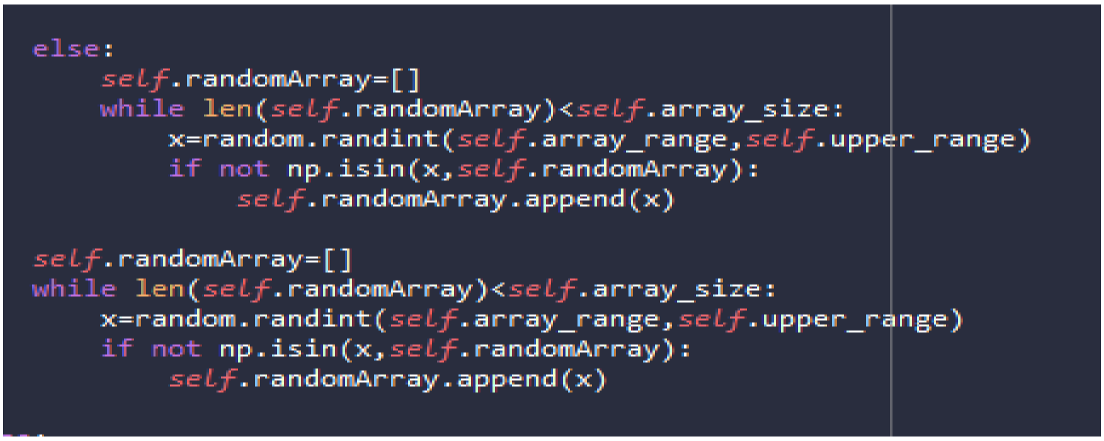
☐ Seed Array

Figure 4

There are various sorting algorithms here. I added 4 more. These are selection sort, comb sort, cocktail sort and shell sort. We can create an unsorted array in 5 different ways : by entering manually, by default, randomly, by enter voice and by txt, csv and excel file.

First, let's examine our random array button. There are three options we have to set : array size, upper bound and lower bound. Opening error message if they are equal to zero. We have the following option for the random array button. The option of having the same numbers or not the same numbers. If we don't want the same numbers, we click button. We let's choose see same numbers . If upper bound not greater than lower bound, open the error message box pops up. When array size, lower bound, and upper bound are appropriate, an array is formed. If we select the seed option this saves our array. When you enter the same array size, upper bound, lower bound values on another computer and select the see same numbers option, you will see the same array. For the seed option, I used the random.seed(0) function in the code. To obtain a unique array, the difference between upper bound and lower bound must be greater than array size. If you don't follow this rule, an error message appears.

If I show you how I make **unique array** in our code:



```
else:
    self.randomArray=[]
    while len(self.randomArray)<self.array_size:
        x=random.randint(self.array_range,self.upper_range)
        if not np.isin(x,self.randomArray):
            self.randomArray.append(x)

self.randomArray=[]
while len(self.randomArray)<self.array_size:
    x=random.randint(self.array_range,self.upper_range)
    if not np.isin(x,self.randomArray):
        self.randomArray.append(x)
```

Now let's examine the default array button. With this button, an array is created without us doing anything. The seed and unique array rules in the random array are also valid in the default array. It is integrated into the code in the same way.

In the default array, as in the random array, when we click the seed button, you save the array, and when you enter the same array size, upper bound and lower bound values on another computer, you view the same array.

Let's come to the Create array button. When we click the button after entering our numbers here, our array is formed. If we enter a string, we will see a warning message. So it only prints integer values.

In our voice array input button, we say the numbers we want to enter after clicking the button. In this way, our numbers appear in the bar and text field in the form of a list.

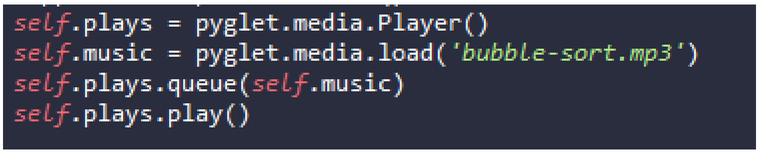
Our other feature is to get array in xlsx, txt and csv file or save the array we created.

Now let's talk about what our skip, stop and continue buttons do by ordering any of our series.

When we click on the stop button, our animation in motion stops. In addition, when we run any sort algorithm, all buttons are disabled except skip, stop-continue and clear buttons. It becomes enabled again after the sequence is finished. Our other button is the continue button, which resumes our animation from where it left off.

With the skip button banana we can sort our array without waiting for the animation. We do this by clearing the graph with the cla function and reprinting the ordered state of the array on the graph.

We also hear a music during animation. We do this with this code:



```
self.plays = pyglet.media.Player()
self.music = pyglet.media.load('bubble-sort.mp3')
self.plays.queue(self.music)
self.plays.play()
```

Other features for my bar animation are;

1-Speed

Thanks to Speed, we can adjust the speed of our animation during animation.

2-Graph Name

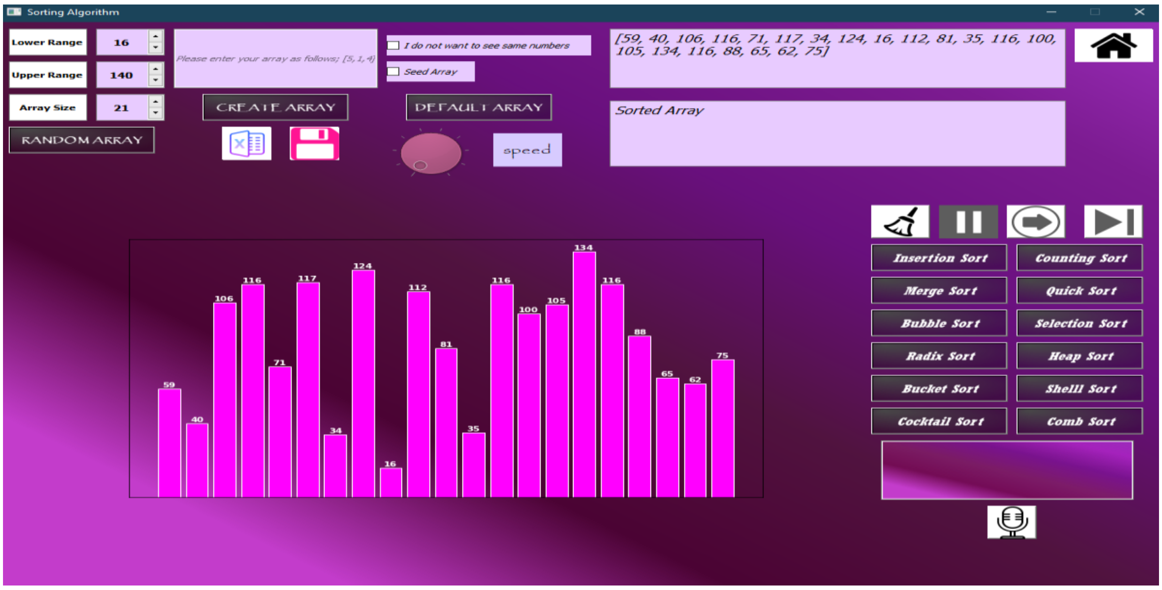
After the animation is finished, the name of the animation is written on the graphic, whichever animation we list.

4- Clear Button

Thanks to the Clear button, we can clean our interface.

5- Gif Effect

There is a gif in our graphics area



Sorting Algorithm

Lower Range 16

Upper Range 140

Array Size 21

Please enter your array as follows; [5, 1, 4]

□ I do not want to see same numbers

□ Seed Array

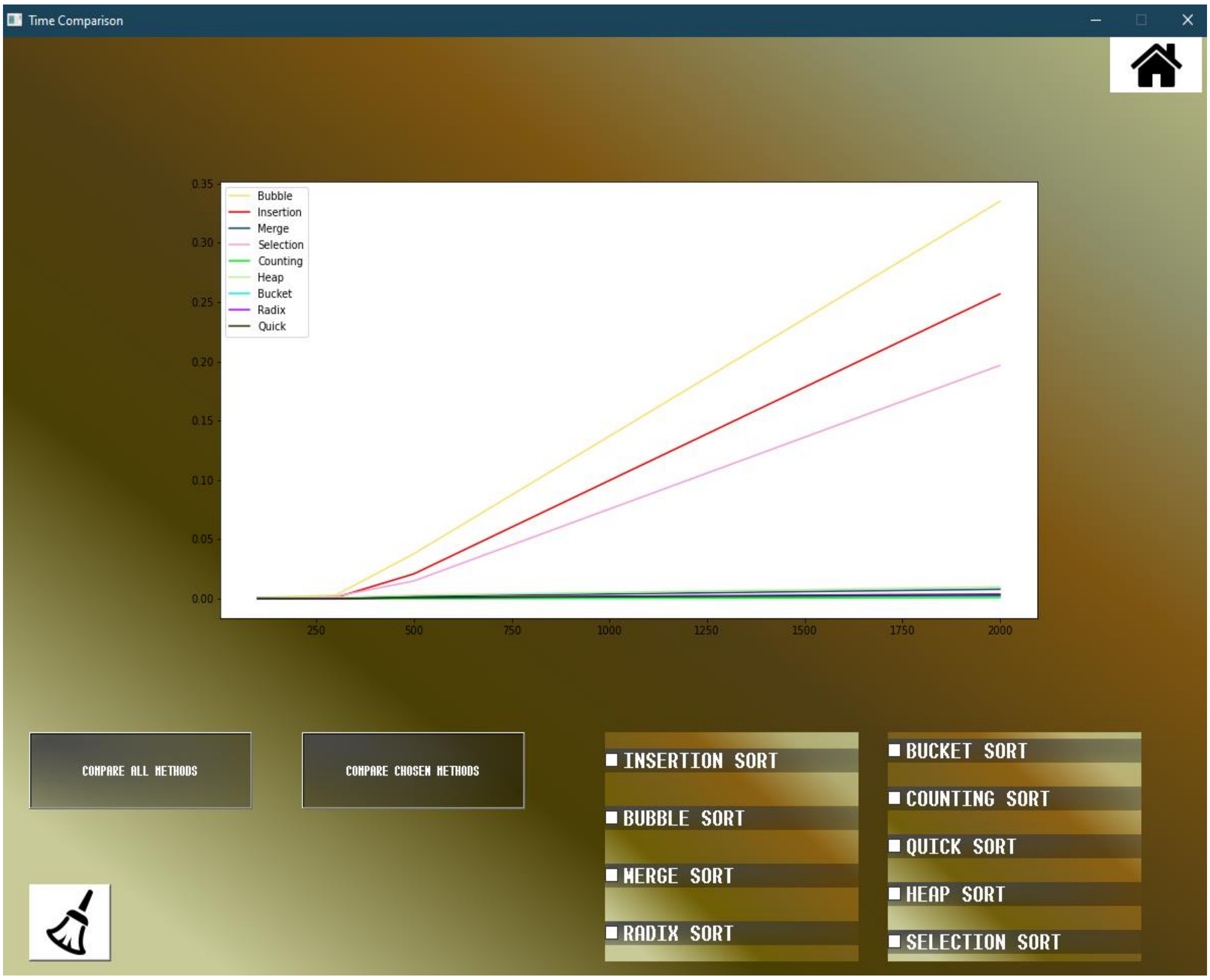
[59, 40, 106, 116, 71, 117, 34, 124, 16, 112, 81, 35, 116, 100, 105, 134, 116, 88, 65, 62, 75]

Sorted Array

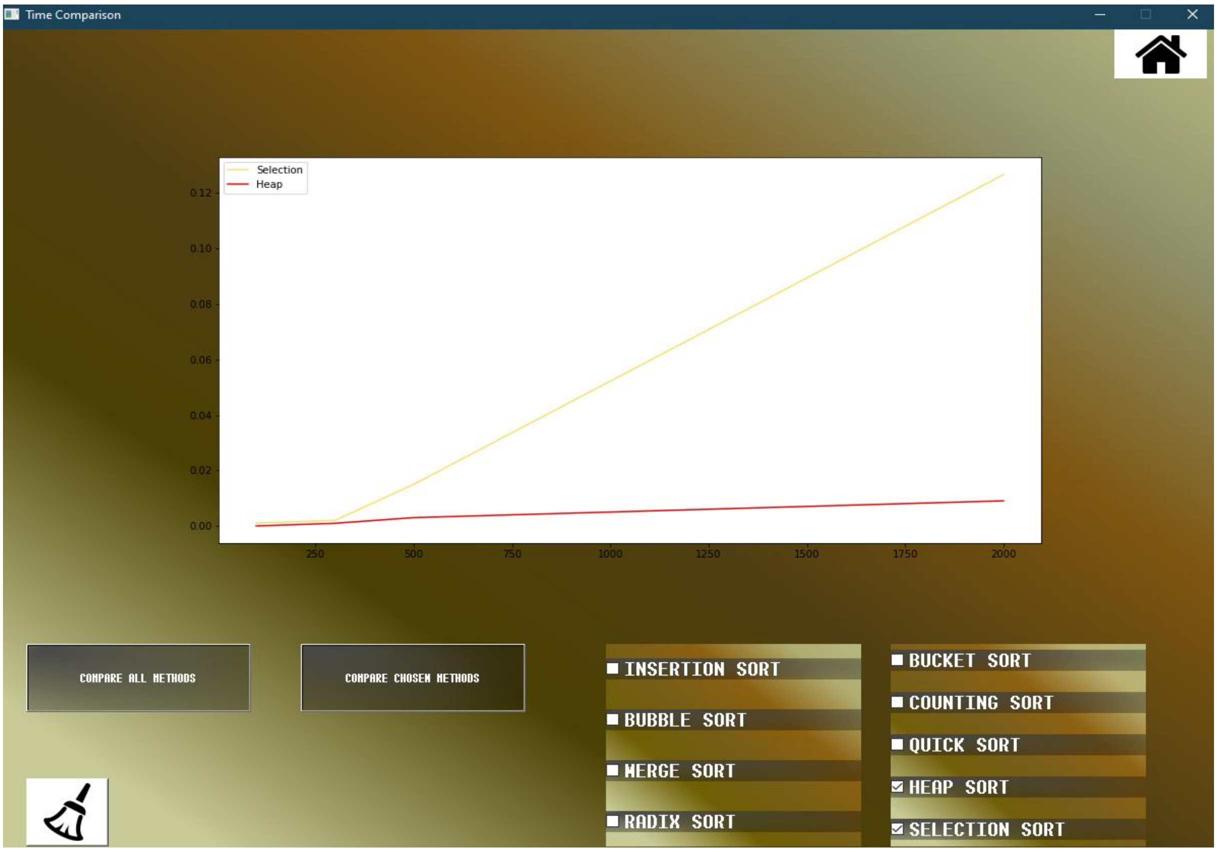
When we come to our second page, time comparison meet us

Here the times of the sorting algorithms are compared. After entering the range, size and iteration values, it compares all sort algorithms if we say compare all methods, compares the sort algorithms we choose if we say compare chosen methods.

I clicked compare all methods;



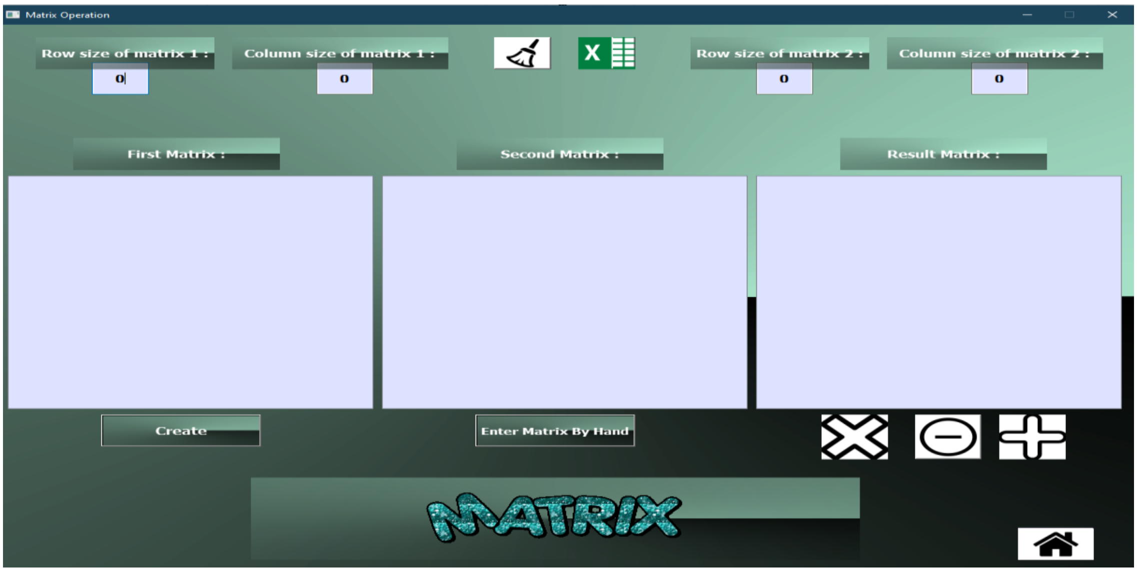
I clicked compare chosen methods;



Here, too, I used the random.seed(0) function all computers, the same time comparison can be seen.

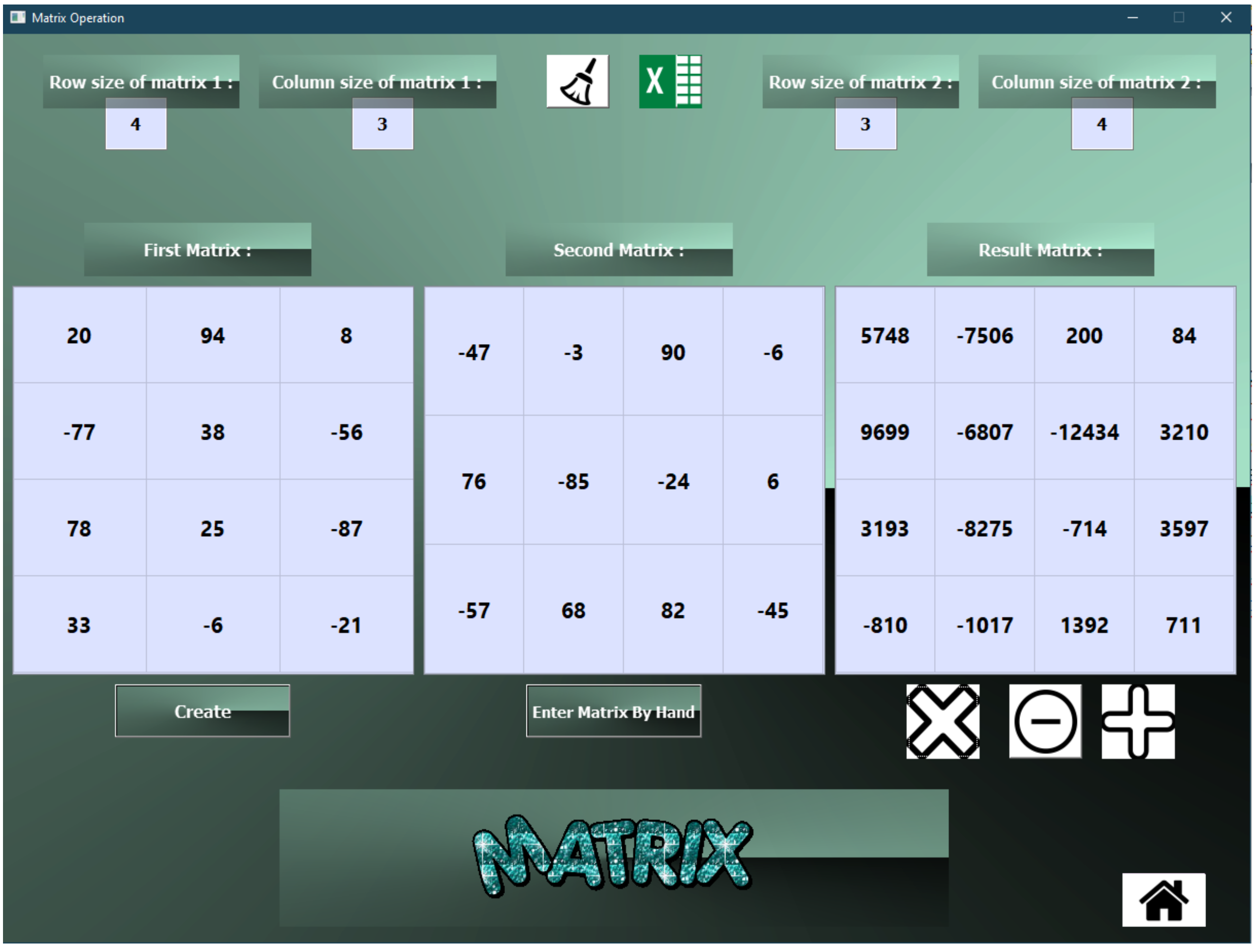
Clear button is again useful for cleaning the entire interface. When I click on the Home button, we go back to the main page.

When we come to our third page, matrix operations meet us :



There are matrix multiplication, sum and subtraction operations here. When the length of the row and column obeys the rules, operations occur. If he doesn't follow the rules, error messages will appear. Also, another feature I added is the exact fit of the matrices to the dimensions we create and the average of the numbers. We can also create matrix in two different ways. These can be randomly and manually entering the matrix. We can save our result matrix in excel file. There is a gif on our homepage. We can clear the interface by clicking the Clear button. By clicking the Home button, we return to the home page.

Below is an example of matrix multiplication:



Row size of matrix 1 : 4

Column size of matrix 1 : 3

Row size of matrix 2 : 3

Column size of matrix 2 : 4

First Matrix :

20	94	8
-77	38	-56
78	25	-87
33	-6	-21

Second Matrix :

-47	-3	90	-6
76	-85	-24	6
-57	68	82	-45

Result Matrix :

5748	-7506	200	84
9699	-6807	-12434	3210
3193	-8275	-714	3597
-810	-1017	1392	711

When we come to our fourth page, fibonacci meet us [Figure 5];

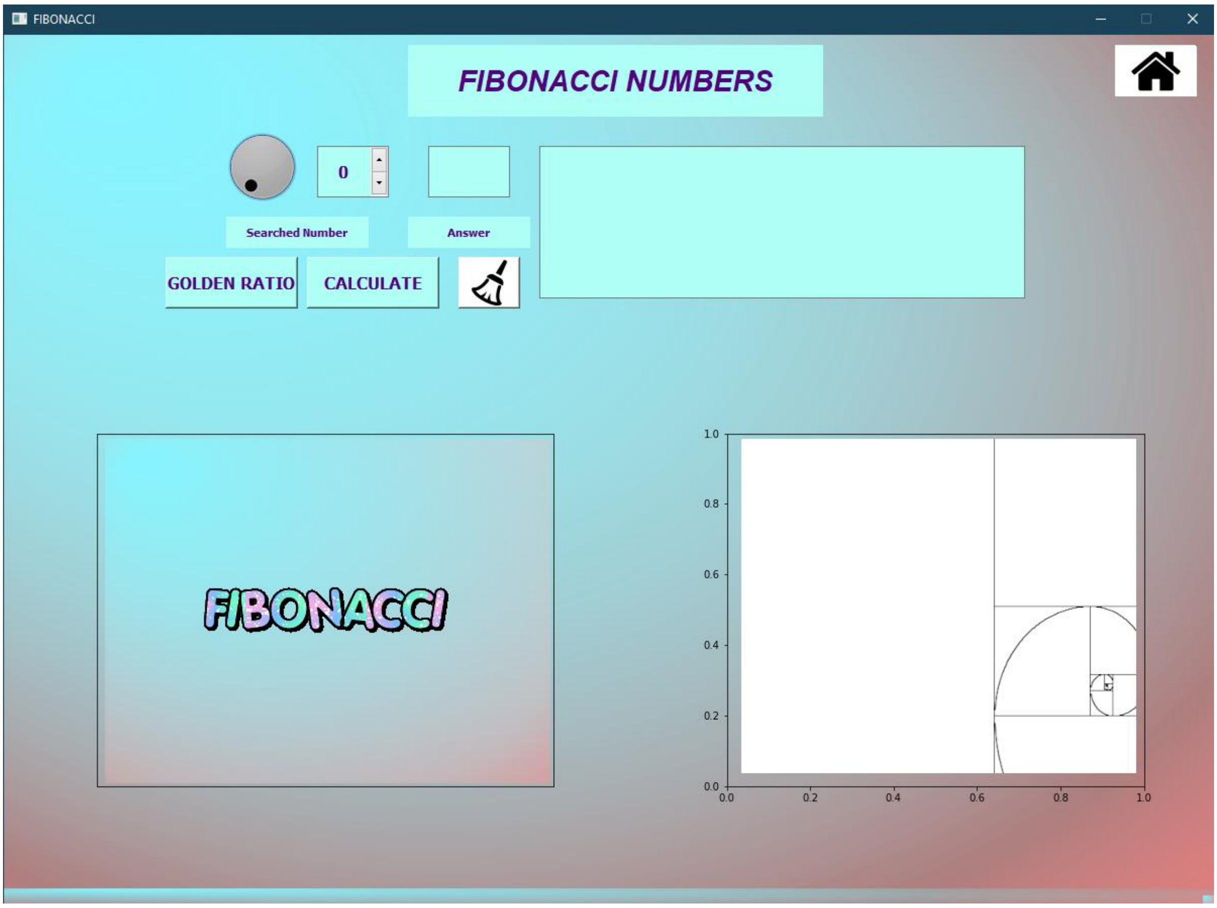
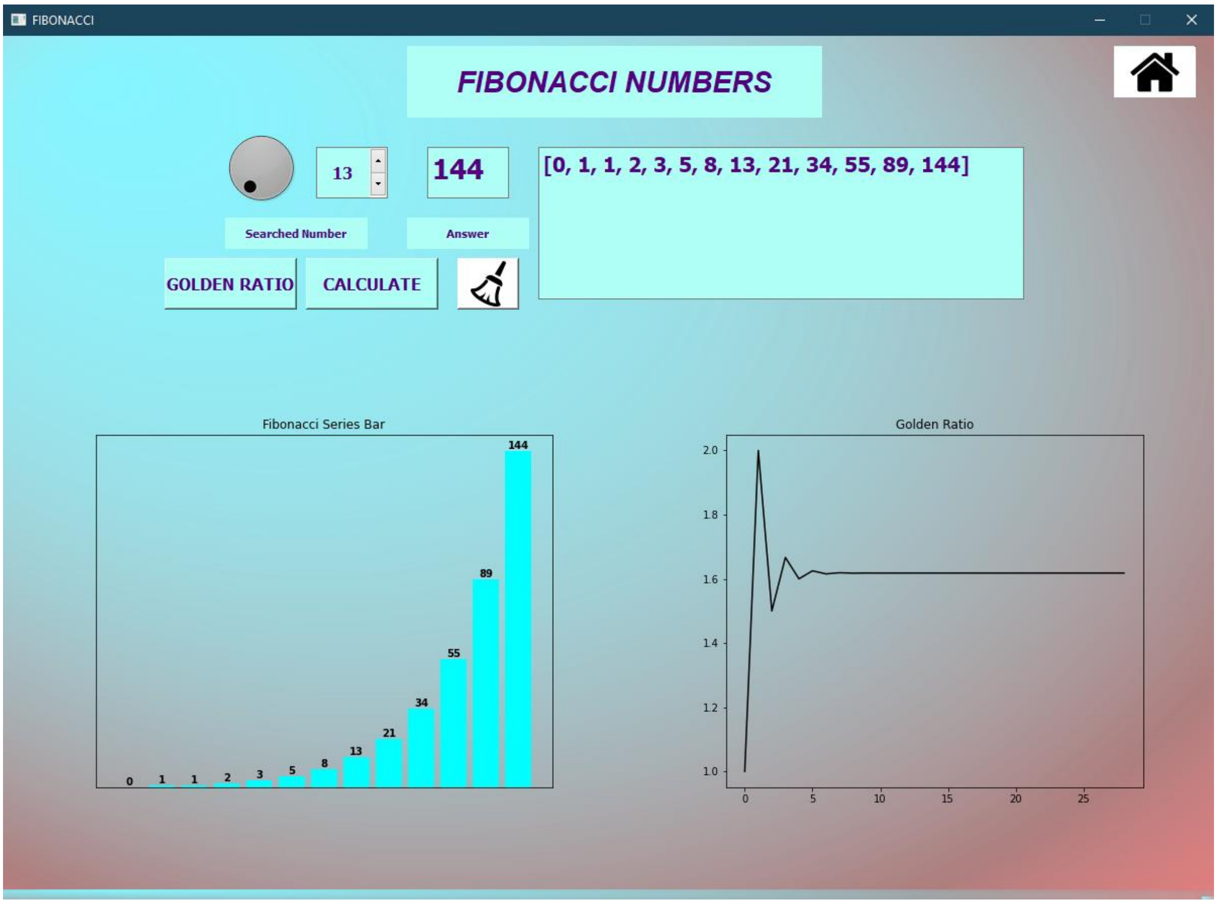


Figure 5

When we enter the index we want to see in the Fibonacci series, we can see both the number and the list. We also observe the list in the form of bars. It's available on the golden ratio chart on the same page. The Clear button clears the whole screen, the information box gives information about the fibonacci series. There are also gifs on our page. We go back to the home page with the Home button.



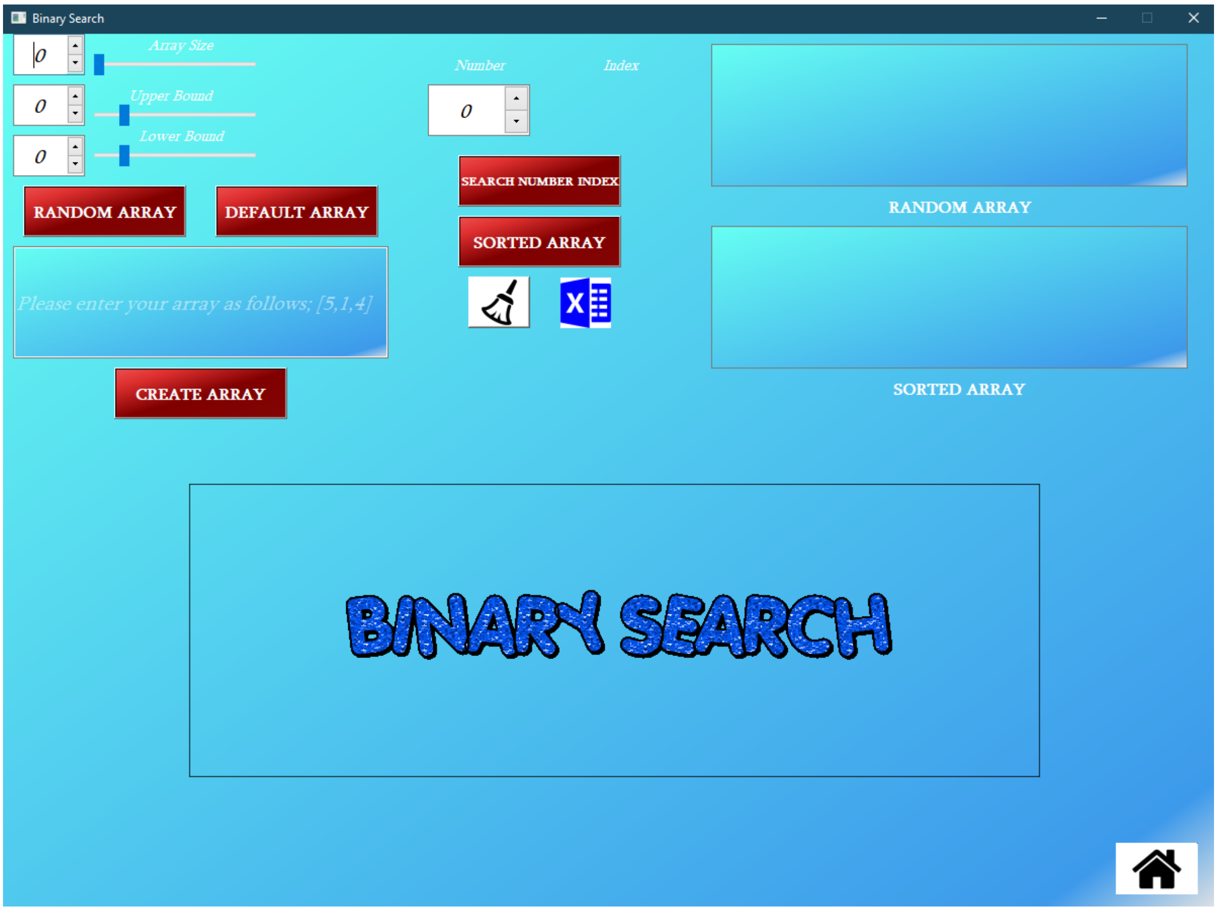
FIBONACCI NUMBERS

Searched Number: 13

Answer: 144

[0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144]

When we come to our fifth page, Binary search meet us;



Array Size: 0

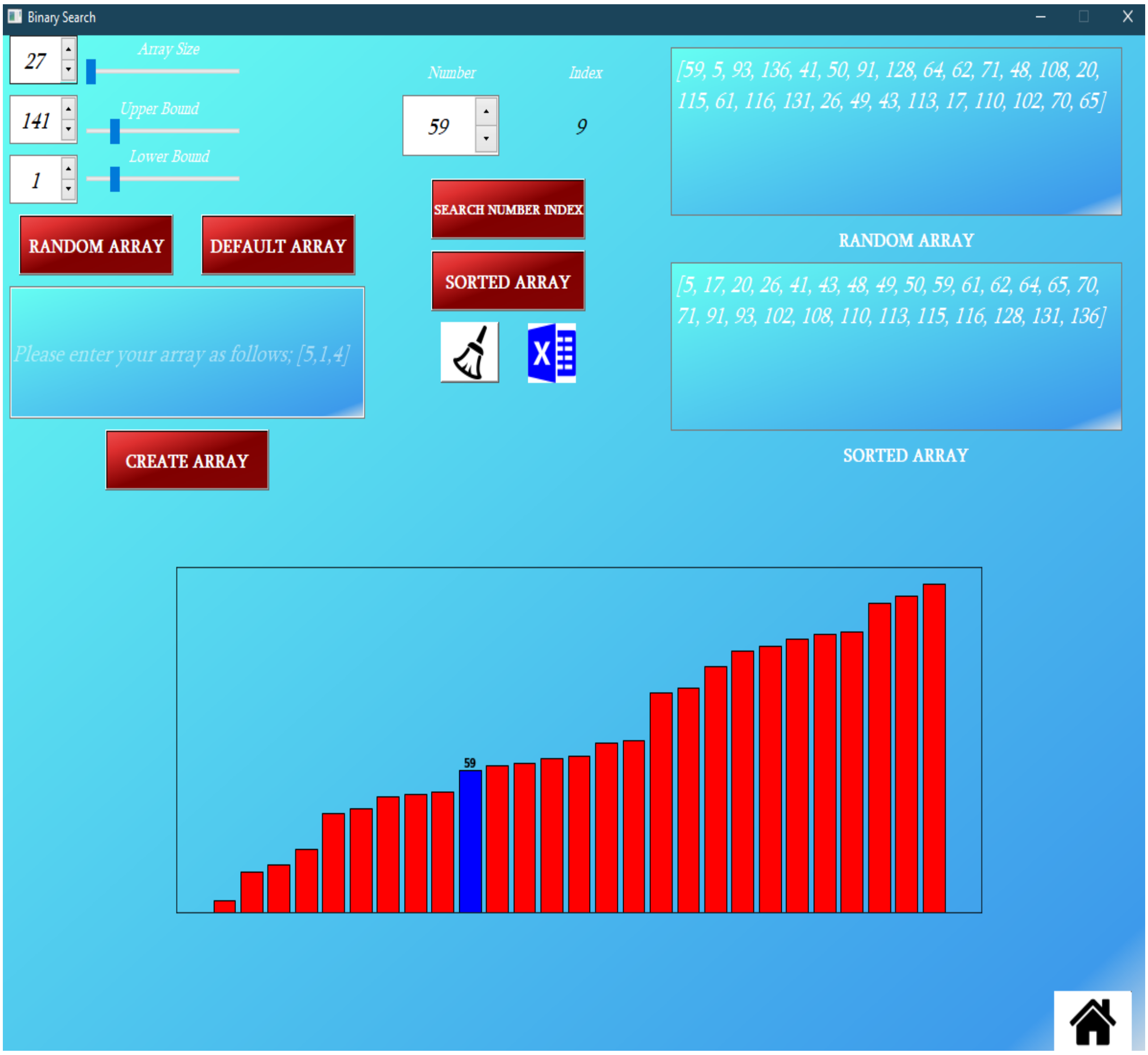
Upper Bound: 0

Lower Bound: 0

Number: 0

Please enter your array as follows; [5,1,4]

In our binary search screen, we can create three types of arrays. Random array, create array and default array. Also, if we want, we can add an array array from excel. Error messages in the sorting algorithm are also present here. Also, in the arrays we have created, the same numbers never appear. There are completely different numbers coming up. I did this with the functions I use in the sorting algorithm. We observe our unsorted and sorted arrays both as a list and a bar. Also, when we enter the number we are looking for, the sorted array bar also paints the number. If the number we are looking for is not available in the list, the color disappears. Our clear button clears the whole screen. We go back to the home page with the Home button.



Binary Search

Array Size 27

Upper Bound 141

Lower Bound 1

Number 59

Index 9

Please enter your array as follows; [5,1,4]

[59, 5, 93, 136, 41, 50, 91, 128, 64, 62, 71, 48, 108, 20, 115, 61, 116, 131, 26, 49, 43, 113, 17, 110, 102, 70, 65]

RANDOM ARRAY

[5, 17, 20, 26, 41, 43, 48, 49, 50, 59, 61, 62, 64, 65, 70, 71, 91, 93, 102, 108, 110, 113, 115, 116, 128, 131, 136]

SORTED ARRAY

When we come to our sixth page, Randomized Select meet us [Figure 6]

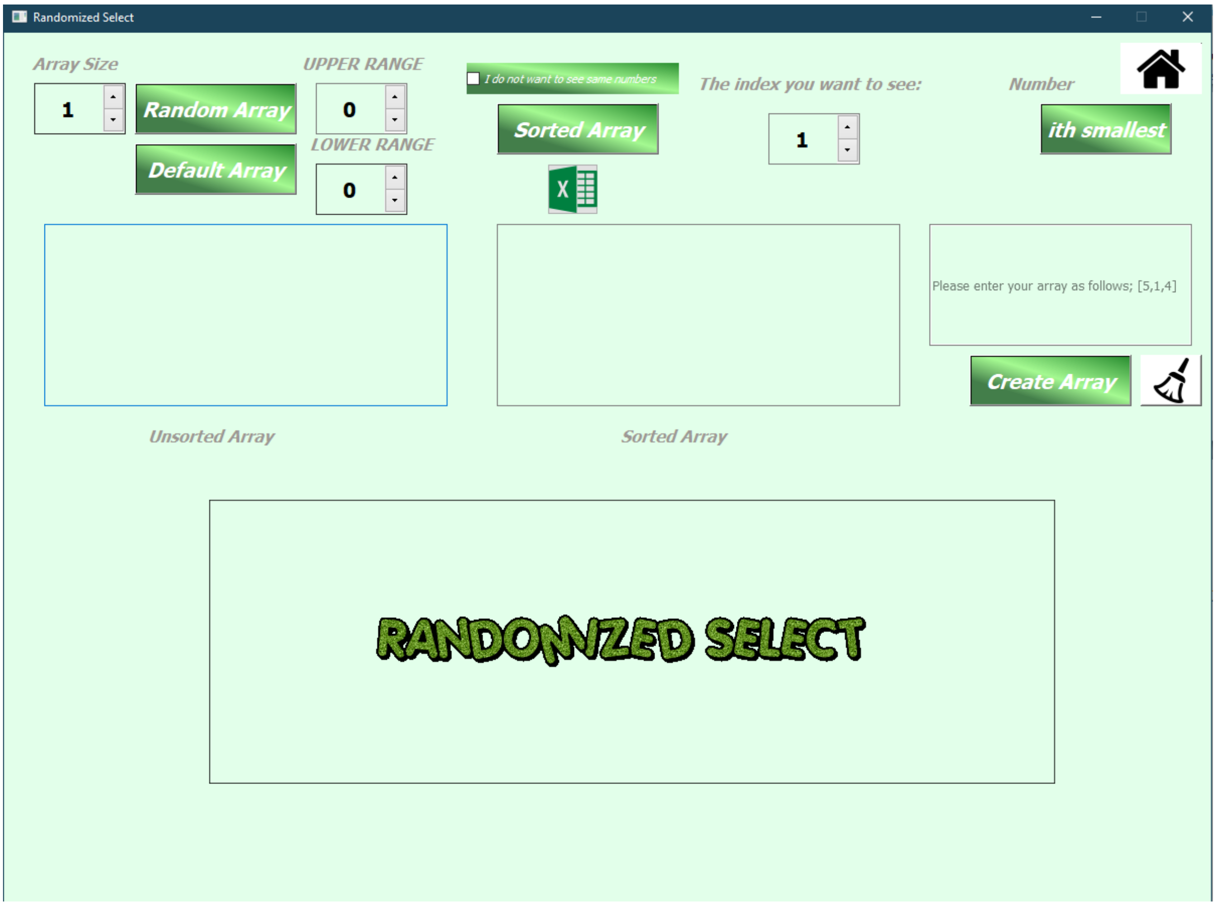
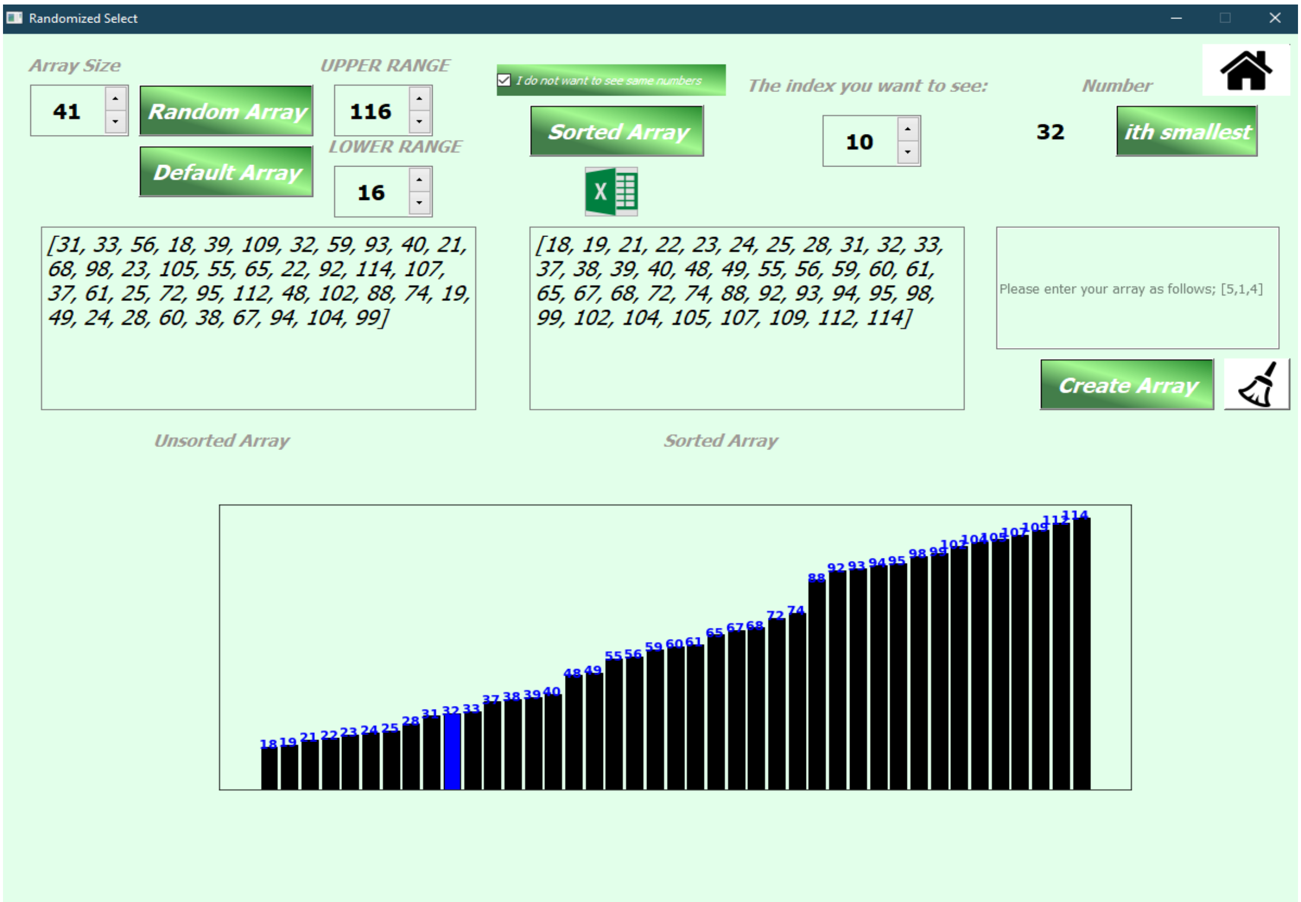


Figure 6

In our randomized select screen, we can create three types of arrays. Random array, create array and default array. Also, if we want, we can add an array array from excel. Error messages in the sorting algorithm are also present here. . If we click on the I do not see same numbers button, there are no similar numbers in the array we created. I did this with the functions I use in the sorting algorithm. We observe our unsorted and sorted arrays both as a list and a bar. When we enter the index we are looking for, the number in that index is painted in the sorted array bar. If the index we are looking for is not available, an error message is given. Our clear button clears the whole screen. We go back to the home page with the Home button.



Array Size: 41

UPPER RANGE: 116

LOWER RANGE: 16

☑ I do not want to see same numbers

The index you want to see: 10

Number: 32

[31, 33, 56, 18, 39, 109, 32, 59, 93, 40, 21, 68, 98, 23, 105, 55, 65, 22, 92, 114, 107, 37, 61, 25, 72, 95, 112, 48, 102, 88, 74, 19, 49, 24, 28, 60, 38, 67, 94, 104, 99]

[18, 19, 21, 22, 23, 24, 25, 28, 31, 32, 33, 37, 38, 39, 40, 48, 49, 55, 56, 59, 60, 61, 65, 67, 68, 72, 74, 88, 92, 93, 94, 95, 98, 99, 102, 104, 105, 107, 109, 112, 114]

Please enter your array as follows; [5,1,4]

Additional features:

1- I made gradient color backgrounds.

2- Resize screen

3- I destroyed the graphic lines and graphic display

4- I added gifs

5- My interface cannot be trolled.

6- Average property of numbers above bars