SELECTION SORT PROJECT

Selection Sort is a sorting algorithm that finds the minimum value in the array for each iteration of the loop. Then this minimum value is swapped with the current array element. This procedure is followed until the array is sorted.

**Time Complexity:**

The time complexity of Selection Sort is O(N2) as there are two nested loops:

\* One loop to select an element of Array one by one = O(N)

\* Another loop to compare that element with every other Array element = O(N)

Therefore, overall complexity = O(N) \* O(N) = O(N\*N) = O(N2)

A Linear search algorithim so worst case is o(n^2)

Ordering from smallest to bigger

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 18 | 7 | 21 | -2 | 19 |

Example data set:

To analyse data set first of all we will look at Array [0]:18

So, we will reach more small number from 18 as -2

And then we will swap these two numbers.

And new data set looks like anymore:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| -2 | 7 | 21 | 18 | 19 |



Therefore, we have defined the smallest value [0] index in the array.

then we have to analyse index [1] in array namely value 7 at index [1].

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| -2 | 7 | 21 | 18 | 19 |

So, if there is ever small number one (value) than 7 in other next indices, we will swap the places of these values but in this array- list as you can see second smallest value is 7 so new list looks like this. And then, we will switch to next index to analyse.

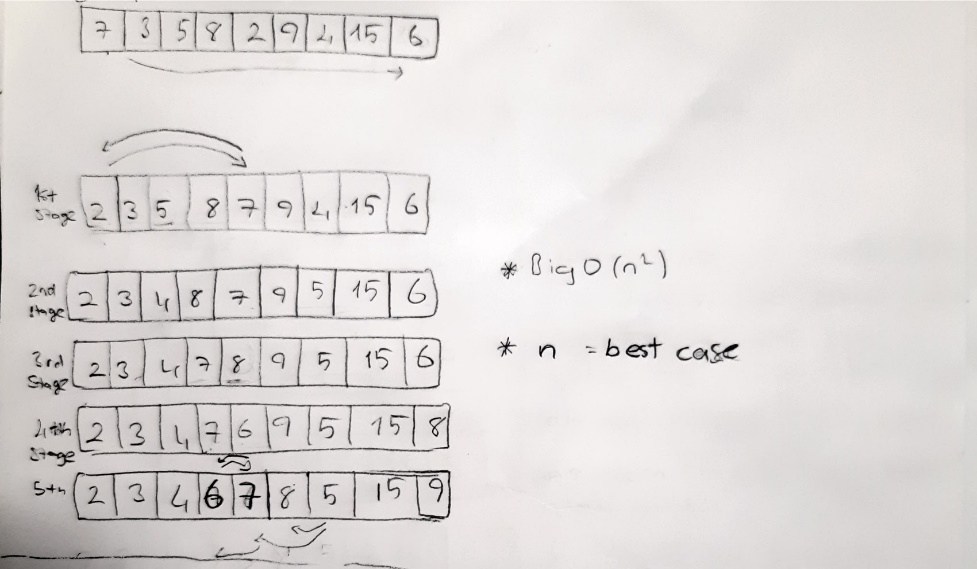
End of the all processes we sorted the array-list by using selection sorting method.



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| -2 | 7 | 18 | 19 | 21 |

PATİKA DEV DATA SRTUCTURES AND ALGORİTHİ

Selection sort



First stage

Replace the places of 2 nad 7 value

Second stage

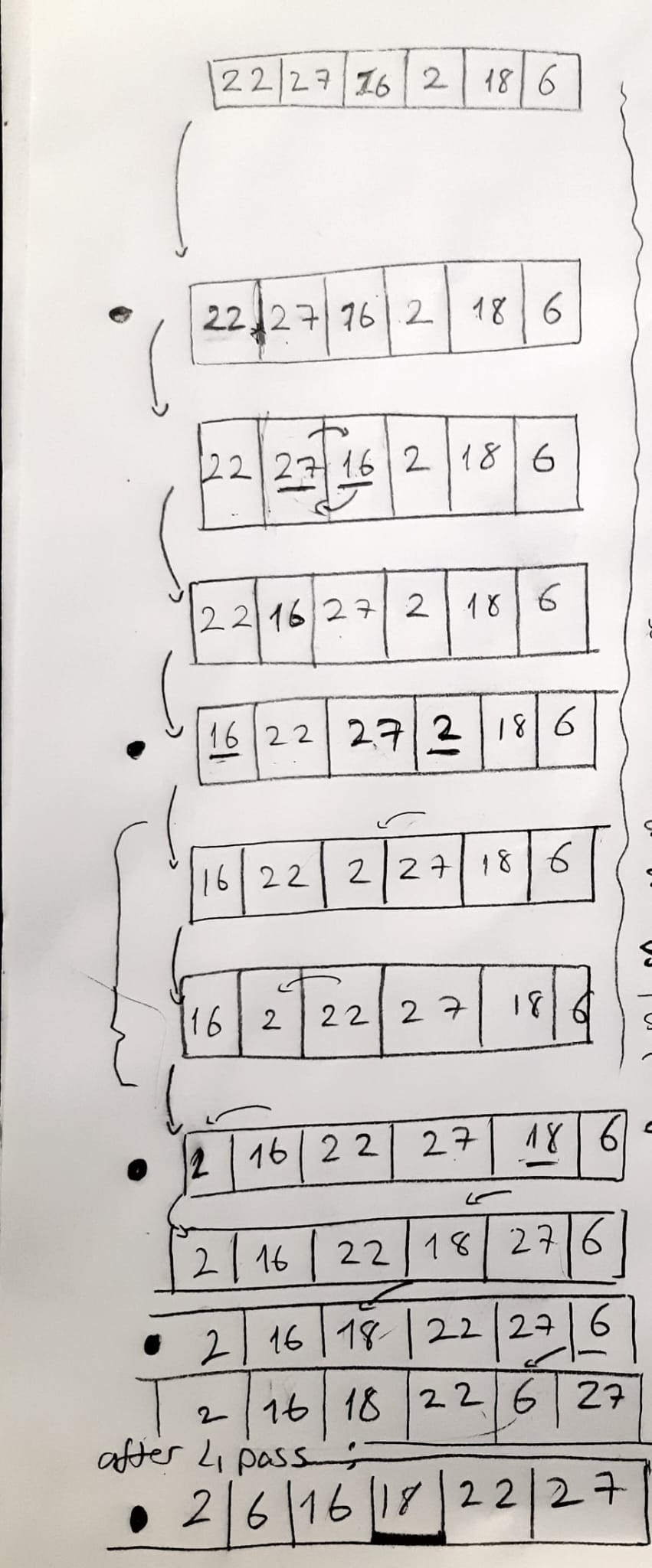
1st index 3 stay in same index because of there is no other value smaller than 3. And 2nd index 5 is changing place with 4

Step by step this processes is implemented.

End of the all implementation :

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2 | 3 | 4 | 7 | 6 | 9 | 5 | 15 | 8 |

Project 1 -ınsertion sort

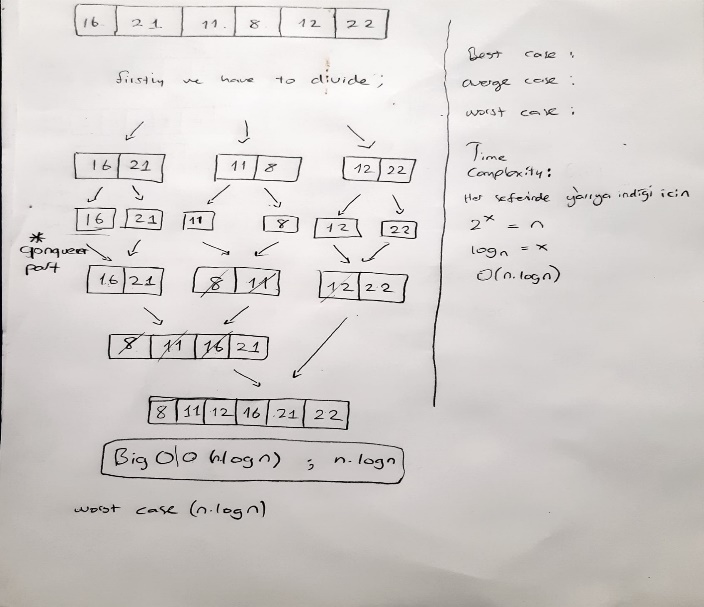


Big o: O (n^2)

As you can see the sorted list we can say

Our Time Complexity is in average case

Project 2



PROJECT 3

Binary Tree Searching Algorithim

Firstly, we’ll choose one node as a root node :4

And then we’ll define the right and left child nodes of each node accordingly the value size comparison.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 5 | 1 | 8 | 3 | 0 | 9 | 4 | 2 |

4

first parent node-root node :4

5

3

2

7

8

1

0

9