

***CA-3 Project report***

***on***

***Heap sort***

***Submitted by***

*Aniket Kumar*

*12001472*

*KM092*

***Under guidance of***

*Dr. Navneet Kaur*

*School of Computer Science & Engineering,*

*Lovely Professional University, Punjab*

***Introduction***

*The project is based on* ***heap sort.*** *What is Heap? Heap is a specialized tree-based data structure which is essential an almost complete tree that satisfies the heap property.*

*There are two types of heaps:*

1. *Min heap*
2. *Max heap.*

*Priority queue cam be efficiently implemented on binary heap.*

*One of the applications of the heap is* ***heap sort.*** *It uses binary heap to sort an array in O(N logN) time complexity. If we sort min heap, we will get sorted elements in ascending order and if we sort max heap, we will get sorted elements in descending order.*

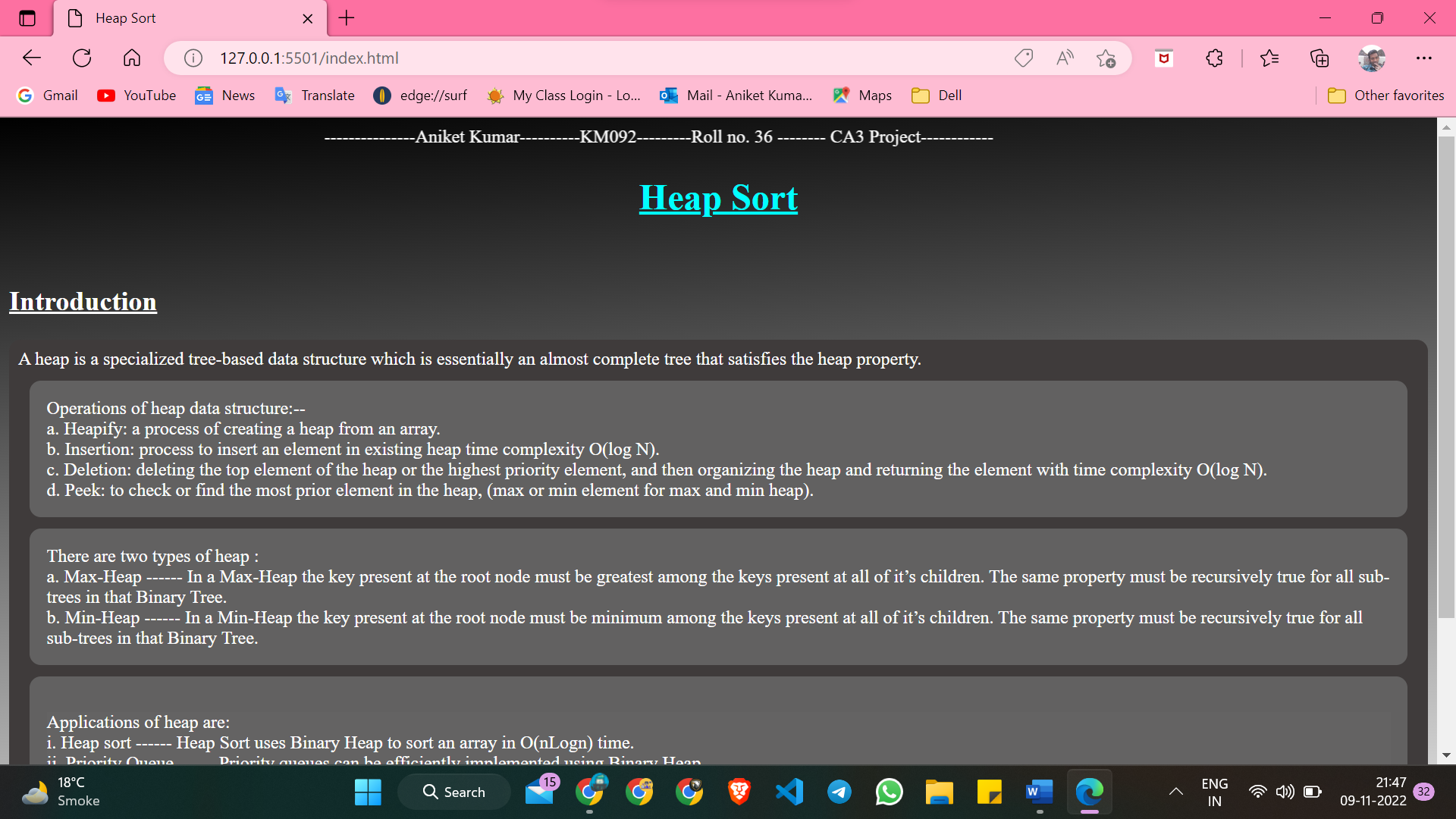
***Technologies used***

*In this project I have used HTML (HyperText Markup Language) , CSS (Cascading Style Sheets) and JS (Java Script). HTML is used to for making the page structure, CSS is used to give the design to the page and JS is used to perform the operations.*

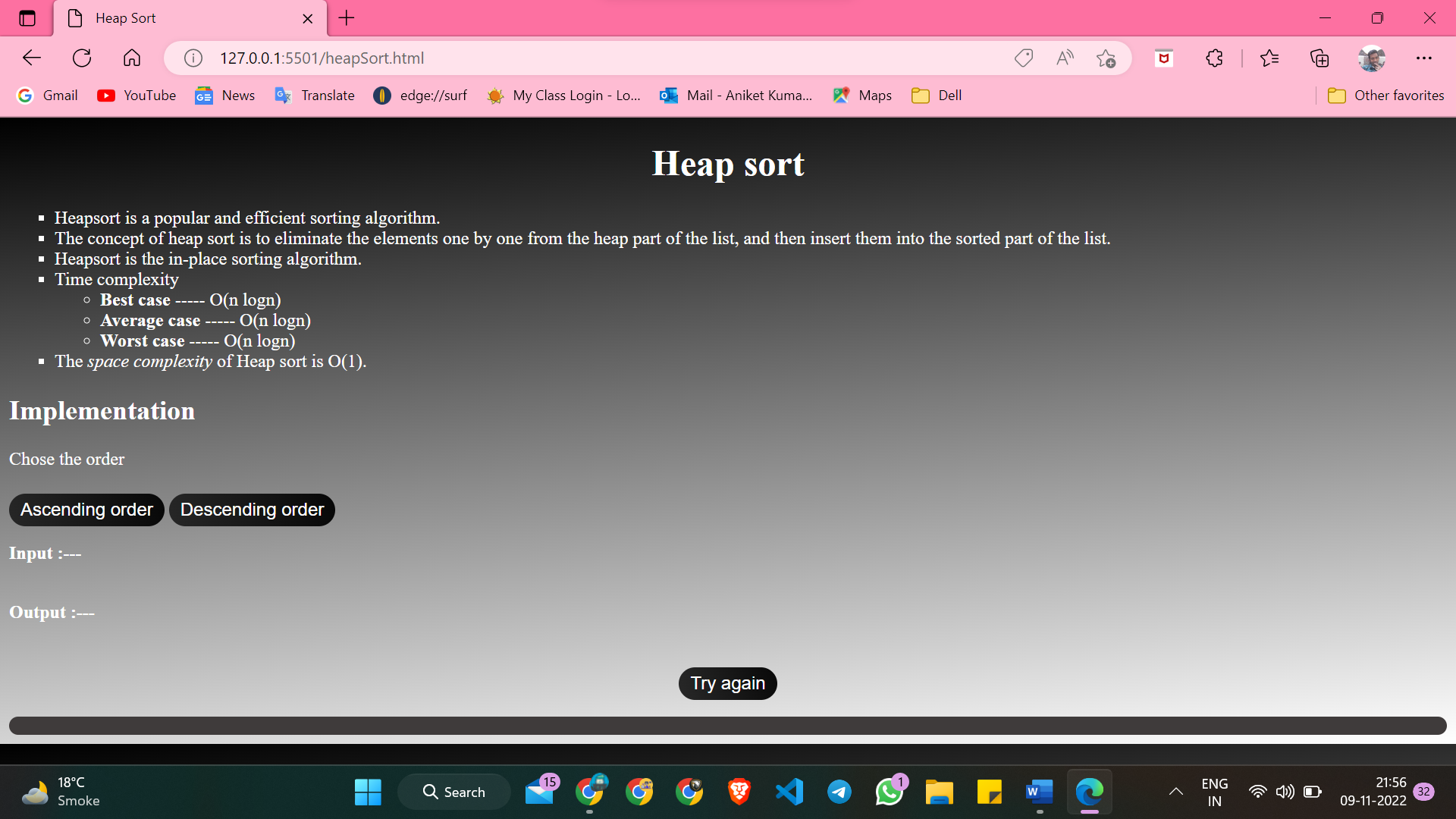
***Project details:***

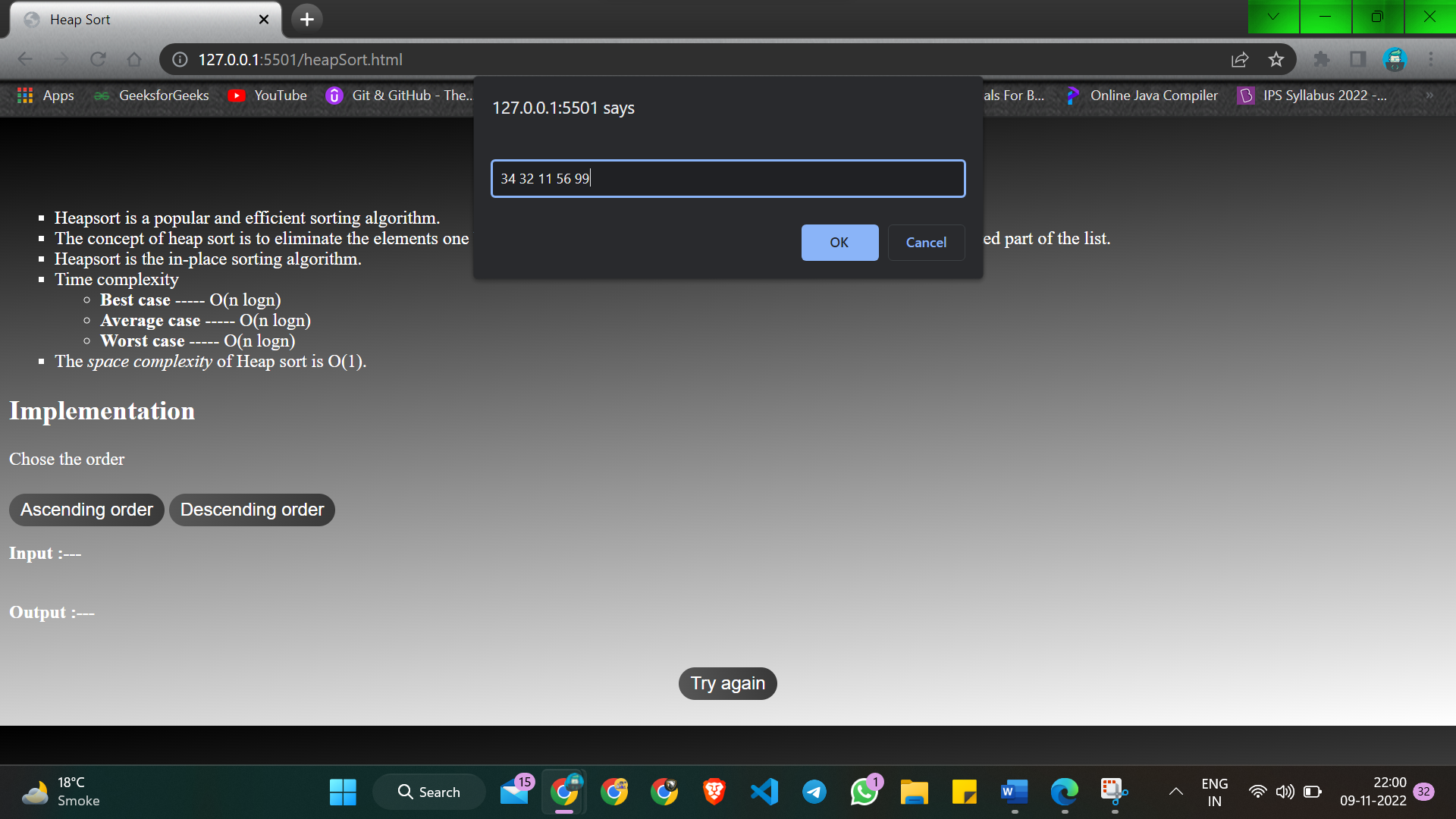
*First page i.e., index.html is the initial page which consists of the explanation about heap. In second page i.e., heapsort.html it is the page that consists of the information and the sorting operation in both ascending and descending order.*

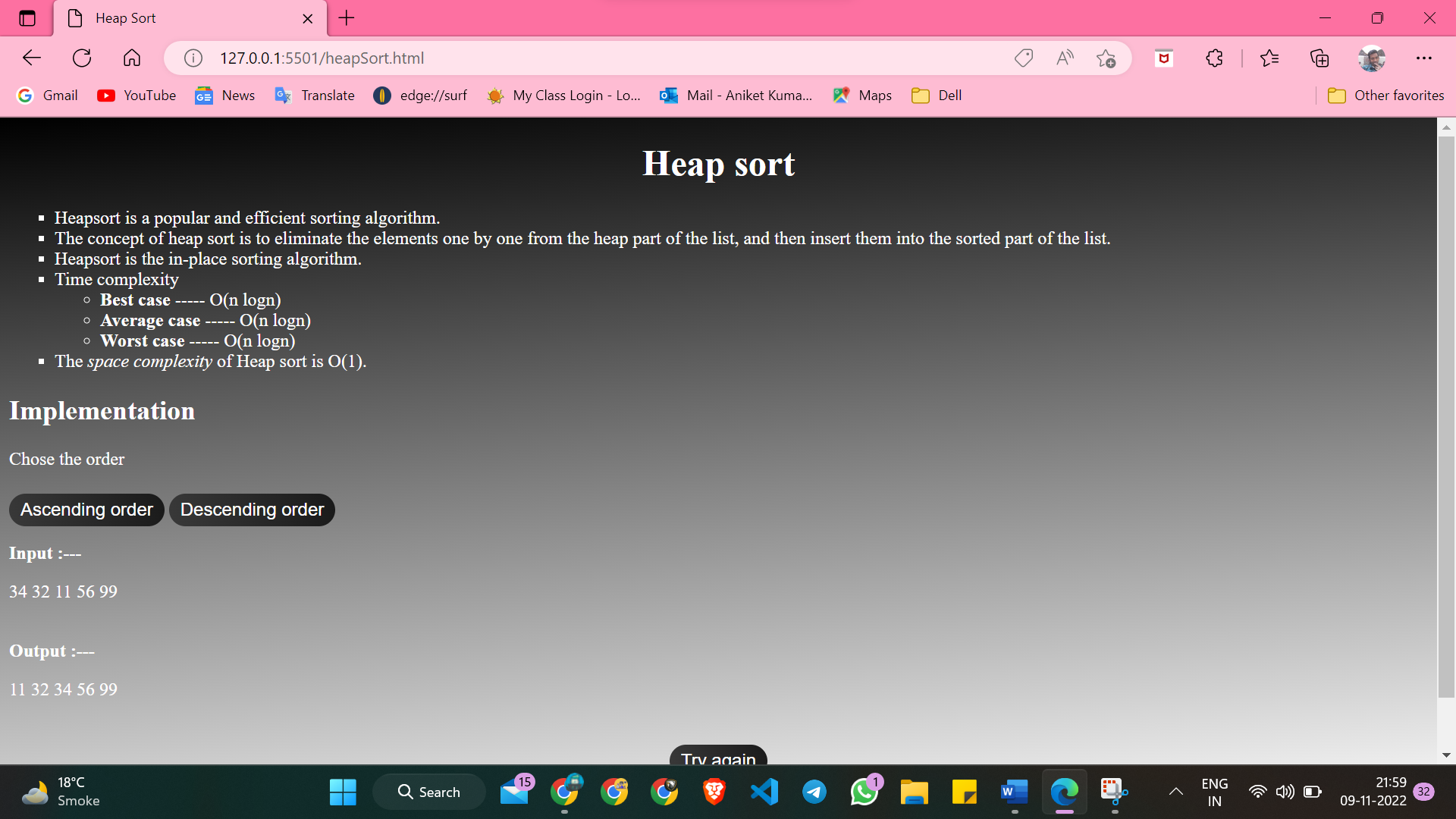
*Below are the screenshots of the pages*

*1st page: *

*2nd page:*

**

**

**

***Code:***

1. *index.html*
2. <!DOCTYPE html>
3. <html lang="en">
4. <head>
5. <meta charset="UTF-8">
6. <meta http-equiv="X-UA-Compatible" content="IE=edge">
7. <meta name="viewport" content="width=device-width, initial-scale=1.0">
8. <title>Heap Sort</title>
9. <link rel="stylesheet" href="style.css">
10. <script src="script.js"></script>
11. </head>
12. <body>
13. <!-- Student details-->
14. <marquee scrollamount="10">---------------Aniket Kumar----------KM092---------Roll no. 36 -------- CA3 Project------------</marquee>
15. <!-- Main heading-->
16. <h1 id="hed1">Heap Sort</h1>
17. <br>
18. <!--Introduction-->
19. <h2 id="intr1">Introduction</h2>
20. <div>
21. A heap is a specialized tree-based data structure which is essentially an almost complete tree that satisfies the heap property.
22. <br>
23. <div class="intr2">
24. Operations of heap data structure:-- <br> a. Heapify: a process of creating a heap from an array.<br> b. Insertion: process to insert an element in existing heap time complexity O(log N). <br> c. Deletion: deleting the top element of the heap
25. or the highest priority element, and then organizing the heap and returning the element with time complexity O(log N).<br> d. Peek: to check or find the most prior element in the heap, (max or min element for max and min heap).
26. </div>
27. <div class="intr2">
28. There are two types of heap : <br> a. <a href="max\_heap1.png">Max-Heap</a> ------ In a Max-Heap the key present at the root node must be greatest among the keys present at all of it’s children. The same property must be recursively true for
29. all sub-trees in that Binary Tree.
30. <br> b. <a href="min\_heap1.png">Min-Heap</a> ------ In a Min-Heap the key present at the root node must be minimum among the keys present at all of it’s children. The same property must be recursively true for all sub-trees in that Binary
31. Tree.
32. </div>
33. <div class="intr2">
34. <p id="pr1">Applications of heap are: <br> i. Heap sort ------ Heap Sort uses Binary Heap to sort an array in O(nLogn) time.<br> ii. Priority Queue ------ Priority queues can be efficiently implemented using Binary Heap.<br> iii. It is also used in graph
35. algorithms like Dijkstra’s Shortest Path and Prim’s Minimum Spanning Tree. </p>
36. </div>
37. </div><br>
38. <button class="btn" onclick="funpvpg()">Previous</button>
39. <button class="btn" onclick="funxtpg()">Next</button>
40. </body>
41. </html>

*2. heapSort.html*

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Heap Sort</title>

    <link rel="stylesheet" href="style.css">

    <script src="script.js"></script>

</head>

<body>

    <h1 id="pg2id1">Heap sort</h1>

    <ul type="square">

        <li>Heapsort is a popular and efficient sorting algorithm.</li>

        <li>The concept of heap sort is to eliminate the elements one by one from the heap part of the list, and then insert them into the sorted part of the list.</li>

        <li>Heapsort is the in-place sorting algorithm.</li>

        <li>Time complexity

            <ul>

                <li><b>Best case</b> ----- O(n logn)</li>

                <li><b>Average case</b> ----- O(n logn)</li>

                <li><b>Worst case</b> ----- O(n logn)</li>

            </ul>

        </li>

        <li>The <i>space complexity</i> of Heap sort is O(1).</li>

    </ul>

    <h2>Implementation</h2>

    <p>Chose the order </p>

    <button class="btn2" onclick="ascending()">Ascending order</button>

    <button class="btn2" onclick="descending()">Descending order</button>

    <br><b>Input :---</b>

    <p id="inpt"></p>

    <br><b>Output :--- </b>

    <p id="otpt"></p>

    <br>

    <center><button class="btn2" onclick="reloadpg()">Try again</button></center>

</body>

</html>

*3. style.css*

\* {

*color*: white;

*font-family*: 'Times New Roman', Times, serif;

}

body {

*background*: linear-gradient(175deg, black, grey, white);

}

#hed1 {

*color*: aqua;

*text-align*: center;

*text-decoration*: underline;

}

#intr1 {

*text-decoration*: underline;

}

div {

*background-color*: rgb(63, 59, 59);

*border-radius*: 10px;

*padding*: 8px;

}

.intr2 {

*background-color*: rgba(101, 99, 99, 0.945);

*border-radius*: 10px;

*padding*: 15px;

*margin*: 10px;

}

.intr2 a {

*text-decoration*: none;

}

#pr1 {

*background-color*: rgba(101, 99, 99, 0.945);

}

.btn {

*appearance*: none;

*border*: 0;

*padding*: 10px 35px;

*display*: inline-block;

*color*: white;

*text-transform*: uppercase;

*border-radius*: 500px;

*font-size*: 14px;

*font-weight*: 400;

*font-family*: Arial, Helvetica, sans-serif;

*background*: linear-gradient(250deg, black, grey);

*background-size*: 400% 400%;

*animation*: AnimationName 2s ease infinite;

}

@keyframes *AnimationName* {

    0% {

*background-position*: 0% center;

    }

    100% {

*background-position*: 100% center;

    }

}

#pg2id1 {

*text-align*: center;

}

.btn2 {

*appearance*: none;

*border*: 0;

*padding*: 5px 10px;

*display*: inline-block;

*color*: white;

*text-transform*: none;

*border-radius*: 20px;

*font-size*: 16px;

*font-weight*: 400;

*font-family*: Arial, Helvetica, sans-serif;

*background*: linear-gradient(250deg, black, grey);

*background-size*: 400% 400%;

*animation*: AnimationName 10s ease infinite;

*margin-top*: 5px;

*margin-bottom*: 15px;

}

*4. script.js*

*function* funpvpg() {

    alert("You are still on initial page")

}

*function* funxtpg() {

    window.open("heapSort.html", "\_self");

}

//var n;

// n = Number(document.getElementById("input1").value);

// function input1() {

//     window.alert("Use space to seperate the elements")

//     let array = prompt().split(' ');

//     let arrayLength = array.length;

//     document.getElementById('elementinpt').innerHTML += array.join(' ');

// }

//let newArray = [];

*function* ascending() {

    window.alert("Ascending order");

*let* array = prompt().split(' ');

*let* arrayLength = array.length;

*let* newArray = [];

    document.getElementById('inpt').innerHTML += array.join(' ');

*function* setDoughter(*iter*) {

*var* iLeft = *iter* \* 2 + 1;

*var* iRight = *iter* \* 2 + 2;

*var* res = 0;

        if (iLeft < arrayLength - 1) {

            setDoughter(iLeft);

        }

        if (array[iLeft] < array[iRight] && array[iLeft] < array[*iter*]) {

            res = array[*iter*];

            array[*iter*] = array[iLeft];

            array[iLeft] = res

        }

        if (iRight < arrayLength - 1) {

            setDoughter(iRight);

        }

        if (array[iLeft] > array[iRight] && array[iRight] < array[*iter*]) {

            res = array[*iter*];

            array[*iter*] = array[iRight];

            array[iRight] = res;

        }

        if (array[iLeft] > array[iRight]) {

            res = array[iLeft];

            array[iLeft] = array[iRight];

            array[iRight] = res;

        }

    }

*function* start() {

*var* i = 0;

        while (array[0] != null) {

            setDoughter(0);

            newArray.push(array[0]);

            array.splice(0, 1);

        }

    }

    start();

    document.getElementById('otpt').innerHTML += newArray.join(' ');

}

*function* descending() {

    window.alert("Descending order");

*let* array = prompt().split(' ');

*let* arrayLength = array.length;

*let* newArray = [];

    document.getElementById('inpt').innerHTML += array.join(' ');

*function* setDoughter(*iter*) {

*var* iLeft = *iter* \* 2 + 1;

*var* iRight = *iter* \* 2 + 2;

*var* res = 0;

        if (iLeft < arrayLength - 1) {

            setDoughter(iLeft);

        }

        if (array[iLeft] > array[iRight] && array[iLeft] > array[*iter*]) {

            res = array[*iter*];

            array[*iter*] = array[iLeft];

            array[iLeft] = res

        }

        if (iRight < arrayLength - 1) {

            setDoughter(iRight);

        }

        if (array[iLeft] < array[iRight] && array[iRight] > array[*iter*]) {

            res = array[*iter*];

            array[*iter*] = array[iRight];

            array[iRight] = res;

        }

        if (array[iLeft] < array[iRight]) {

            res = array[iLeft];

            array[iLeft] = array[iRight];

            array[iRight] = res;

        }

    }

*function* start() {

*var* i = 0;

        while (array[0] != null) {

            setDoughter(0);

            newArray.push(array[0]);

            array.splice(0, 1);

        }

    }

    start();

    document.getElementById('otpt').innerHTML += newArray.join(' ');

}

*function* reloadpg() {

    location.reload();

}

*GitHub link :* [*https://github.com/Anibaba4296/INT219\_Ca3\_Project*](https://github.com/Anibaba4296/INT219_Ca3_Project)