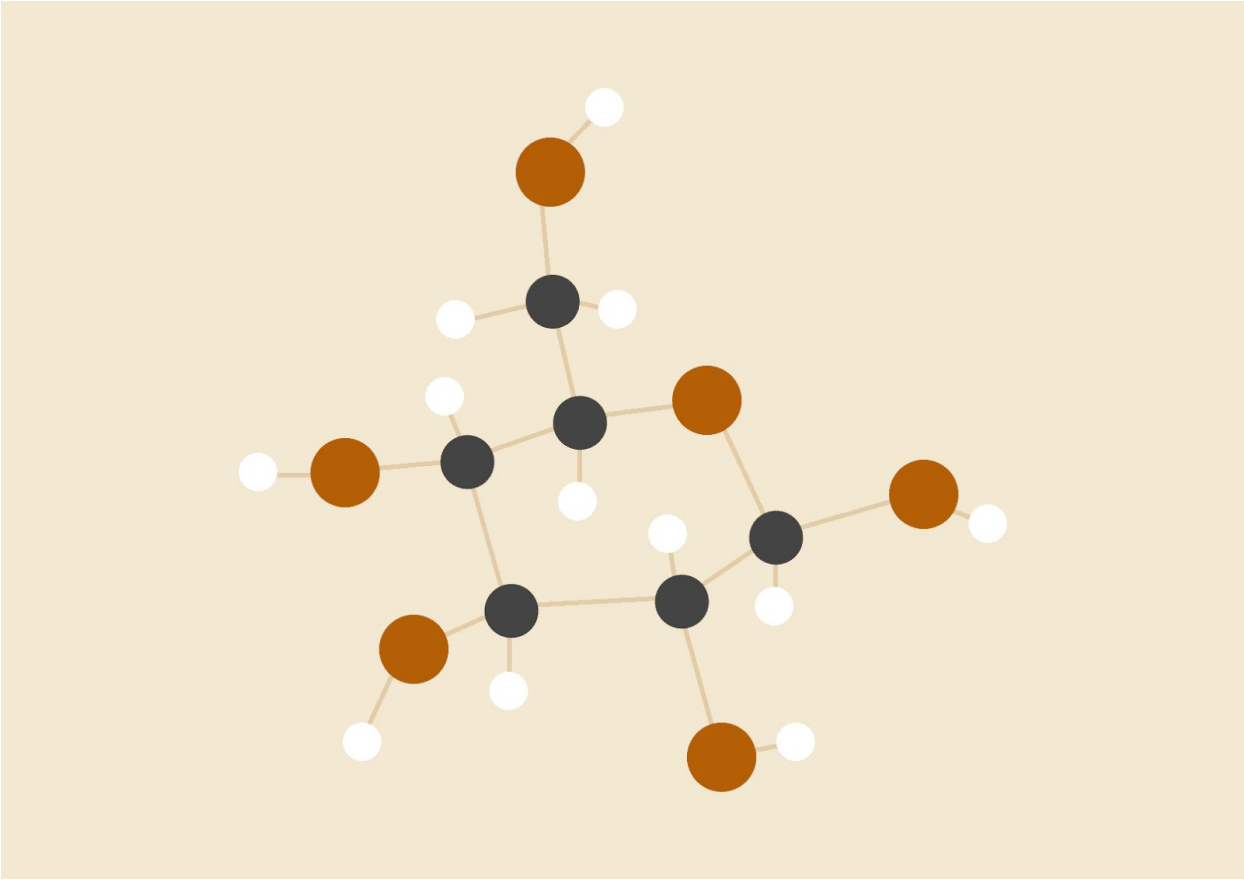
ASSIGNMENT 7

Data Structures Laboratory



Rishi Chordia

18118052

BTech CSE

Problem Statement 1:

Given: n 2D points and two orthogonal polygons.
Problem: Find the set of points lie inside the overlapping region (rectangular) of the two given orthogonal polygons. Write a program in Java to solve the above problem applying k-d tree data structure.

Algorithms and Implementation:

- KD Tree is used as a data structure to store the points
- The root splits it based on the x-axis and the child splits based on y-axis and so on.
- Range search is employed to search for points lying in a polygon
- If a region completely lies within the rectangle, the entire subtree is printed
- The leaf nodes are checked to see if it lies inside.

Snapshots and Computation Time

```
arki1418@rishi-G5:~/Documents/L7$ javac q1.java
arki1418@rishi-G5:~/Documents/L7$ java q1
Enter number of points: 10
Enter the points:
4.3 4.1
5 5.8
5.2 3
4.3 8
6 7.7
7.7 2.2
6.8 4.4
8.1 3.6
7.3 8
7.5 6.6
(4.3,8.0)
(4.3,4.1)
(5.0,5.8)
(5.2,3.0)
(6.0,7.7)
(6.8,4.4)
(7.3,8.0)
(7.5,6.6)
(7.7,2.2)
(8.1,3.6)
Enter number of sides of polygon 1: 2
3.5 5.1
6.5 8.4
Enter number of sides of polygon 2: 6
4.1 2.2
6.7 2.2
6.7 4.3
5.4 4.3
5.4 8.7
4.1 8.7
The Points lying in the overlapping Region:
[(4.3,8.0), (5.0,5.8)]
```

Problem Statement 2:

Given n values in an array and two index values, find the result of the following queries 1. minimum value 2. maximum value 3. sum 4. update by adding 4 with each element, within the given index range using Segment tree. Also implement the brute-force method and compare the execution time of both the methods. A segment tree is a data structure used for storing information about intervals, range or segments. It facilitates efficient range querying in O(log n), where n is the size of the given problem.

Algorithms and Implementation:

- **A segment tree is used as a data structure to store the array of elements in a segmented fashion**
- **A segment tree is much faster than a brute force approach**
- **Execution time in segment tree faster**
- **Execution time for in brute force approach is Slower...Pls check snapshots.**

Snapshots and Computation Time

Brute Force

```
arki1418@rishi-G5:~/Documents/L7$ java bruteproblem2
6
2 5 1 4 9 3
1 - Find minimum value
2 - Find maximum value
3 - Find sum
4 - Update by adding 4
5 - Exit
1
Enter range
Index 1 : 3
Index 2 : 5
Minimum is : 3
Execution time :23827090 ns
1 - Find minimum value
2 - Find maximum value
3 - Find sum
4 - Update by adding 4
5 - Exit
2
Enter range
Index 1 : 2
Index 2 : 4
Maximum is : 9
Execution time :876561 ns
1 - Find minimum value
2 - Find maximum value
3 - Find sum
4 - Update by adding 4
5 - Exit
3
Enter range
Index 1 : 2
Index 2 : 5
Sum is : 17
Execution time :980604 ns
1 - Find minimum value
2 - Find maximum value
3 - Find sum
```

```
Execution time :876561 ns
1 - Find minimum value
2 - Find maximum value
3 - Find sum
4 - Update by adding 4
5 - Exit
3
Enter range
Index 1 : 2
Index 2 : 5
Sum is : 17
Execution time :980604 ns
1 - Find minimum value
2 - Find maximum value
3 - Find sum
4 - Update by adding 4
5 - Exit
4
Enter range
Index 1 : 1
Index 2 : 4
Execution time :2168 ns
1 - Find minimum value
2 - Find maximum value
3 - Find sum
4 - Update by adding 4
5 - Exit
1
Enter range
Index 1 : 1
Index 2 : 4
Minimum is : 5
Execution time :156667 ns
1 - Find minimum value
2 - Find maximum value
3 - Find sum
4 - Update by adding 4
5 - Exit
5
```

Segment Trees Implementation

```
arki1418@rishi-G5:~/Documents/L7$ javac problem2.java
arki1418@rishi-G5:~/Documents/L7$ java problem2
6
2 5 1 4 9 3
1 - Find minimum value
2 - Find maximum value
3 - Find sum
4 - Update by adding 4
5 - Exit
1
Enter range
Index 1 : 3
Index 2 : 5
Minimum is : 3
Execution time :237367 ns
1 - Find minimum value
2 - Find maximum value
3 - Find sum
4 - Update by adding 4
5 - Exit
2
Enter range
Index 1 : 2
Index 2 : 4
Maximum is : 9
Execution time :190634 ns
1 - Find minimum value
2 - Find maximum value
3 - Find sum
4 - Update by adding 4
5 - Exit
3
Enter range
Index 1 : 2
Index 2 : 5
Sum is : 17
Execution time :182056 ns
1 - Find minimum value
2 - Find maximum value
```

```
2 - Find maximum value
3 - Find sum
4 - Update by adding 4
5 - Exit
3
Enter range
Index 1 : 2
Index 2 : 5
Sum is : 17
Execution time :182056 ns
1 - Find minimum value
2 - Find maximum value
3 - Find sum
4 - Update by adding 4
5 - Exit
4
Enter range
Index 1 : 1
Index 2 : 4
Execution time :13048 ns
1 - Find minimum value
2 - Find maximum value
3 - Find sum
4 - Update by adding 4
5 - Exit
1
Enter range
Index 1 : 1
Index 2 : 4
Minimum is : 5
Execution time :205440 ns
1 - Find minimum value
2 - Find maximum value
3 - Find sum
4 - Update by adding 4
5 - Exit
5
arki1418@rishi-G5:~/Documents/L7$
```