Indian Institute of Technology Roorkee

Department of Computer Science and Engineering

CSN-261: Data Structures Laboratory (Autumn 2019-2020)

Lab Assignment-7 (L7) **Date: September 25, 2019** **Duration: 3 Weeks**

General Instructions:

1. Every Lab Assignment will be performed by the students individually. No group formation is required and the evaluations will be done every week for the students individually.

Submission and Evaluation Instructions:

1. **Submit your** zipped folder (**<filename>.zip** or **<filename>.tar.gz**) through your account in Moodle through the submission link for this Lab Assignment in Moodle course site: https://moodle.iitr.ac.in/course/view.php?id=46.
2. **Hard deadline for Final submission in Moodle:** **October 16, 2019 (1:00 pm Indian Time).** For any submission after Final Deadline, 20% marks will be deducted (irrespective of it is delayed by a few seconds or a few days). The key to success is starting early. You can always take a break, if you finish early.
3. The submitted zipped folder (**<filename>.zip** or **<filename>.tar.gz**) must contain the following:
   (a) The source code files in a folder
   (b) A report file (**<filename>.DOC** or **<filename>.PDF**) should contain the details like:
      i. Title page with details of the student
      ii. Problem statements
      iii. Algorithms and data structures used in the implementation
      iv. Snapshots of running the codes for each problem
4. The submission by each student will be checked with others' submission to identify any copy case (using such detection software). If we detect that the code submitted by a student is a copy (partially or fully) of other's code, then the total marks obtained by one student will be divided by the total number of students sharing the same code.

Instructions for L7:

1. Objective of this Lab Assignment is to make the students familiar with different data structures while coding the programs in the Java language to solve some real-life problems.
2. The students are expected to have a basic knowledge of data structures and the Java programming language.
3. The student will have to demonstrate and explain the coding done for this Lab Assignment in the next laboratory class to be held on **October 16, 2019** for evaluation.

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.

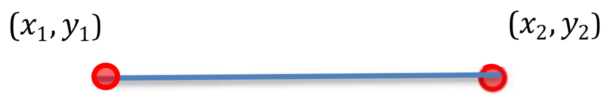
Write a program in Java to solve the above problem applying k-d tree data structure.

Graphviz/applet

Point representation:

$p_i(x_i, y_i)$

Line representation:

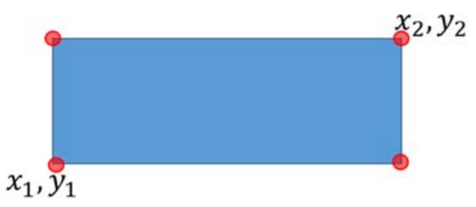


Polygon representation:

Sides = 4

<bottom-left, top-right>

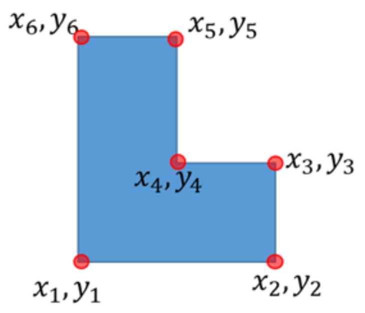
$\{ (x_1, y_1) , (x_2, y_2) \}$



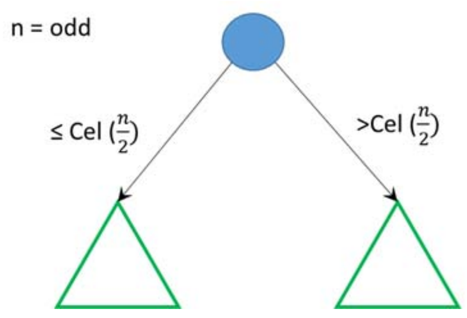
Sides > 4

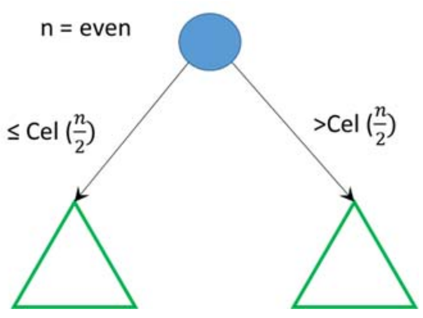
anti-clock wise order, starting from bottom-left

$\{ (x_1, y_1), (x_2, y_2), \ldots , (x_n, y_n) \}$



Constraints while building the k-d tree:



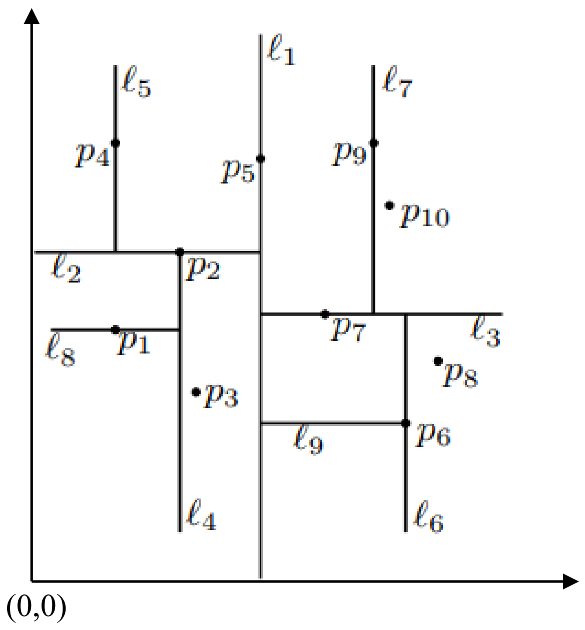


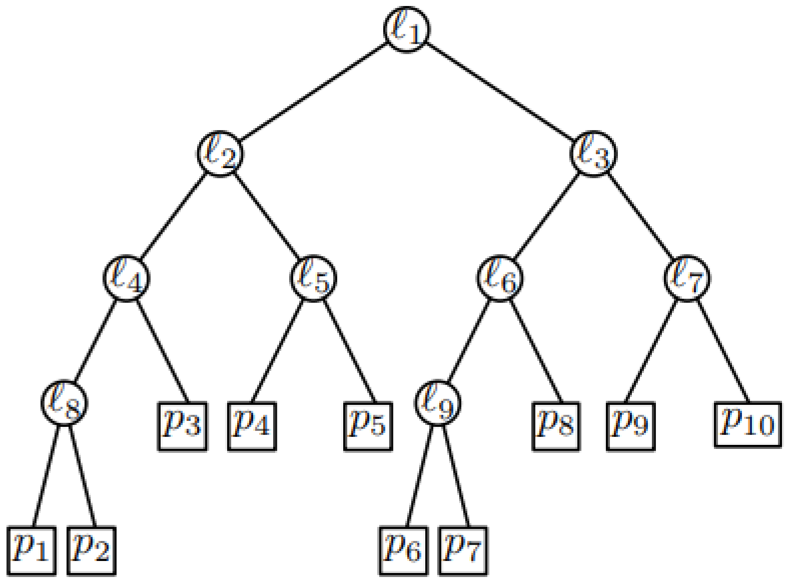
Example:

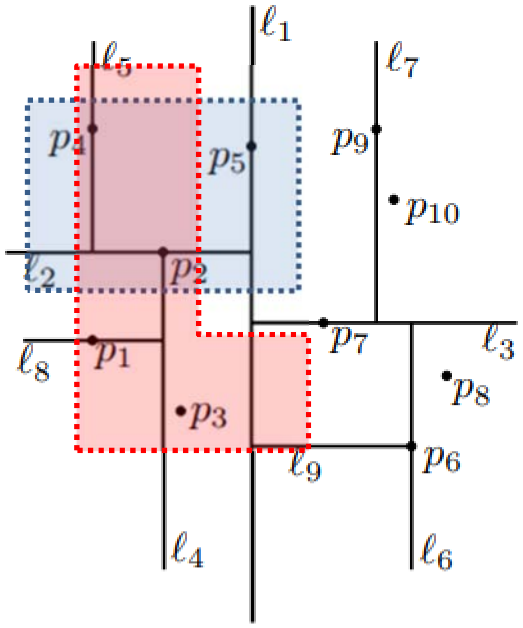
$p_1 = (4.3,4.1)$, $p_2 = (5,5.8)$, $p_3 = (5.2,3)$, $p_4 = (4.3,8)$, $p_5 = (6,7.7)$, $p_6 = (7.7,2.2)$, $p_7 = (6.8,4.4)$, $p_8 = (8.1,3.6)$, $p_9 = (7.3,8)$, $p_{10} = (7.5,6.6)$

Poly1 = {(3.5,5.1) , (6.5,8.4) }

Poly2 = {(4.1,2.2), (6.7,2.2), (6.7,4.3), (5.4,4.3), (5.4,8.7), (4.1,8.7)}







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.

Example:

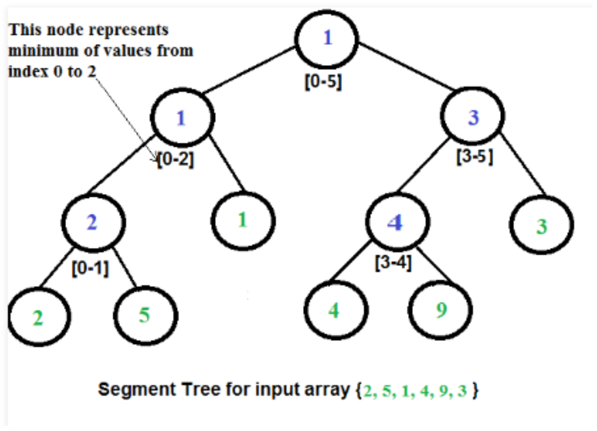
Input:

[2, 5, 1, 4, 9, 3]

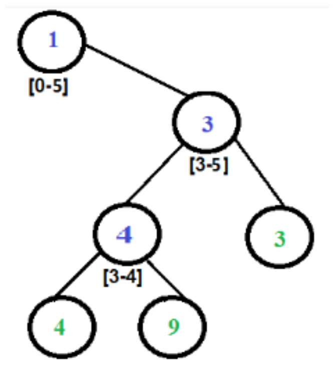
Index$_1$ = 3

Index$_2$ = 5

Output: 3


This node represents minimum of values from index 0 to 2
1
[0-5]
1
[0-2]
3
[3-5]
2
[0-1]
1
4
[3-4]
3
2
5
4
9
Segment Tree for input array {2, 5, 1, 4, 9, 3 }

Segment tree



Searched segment tree