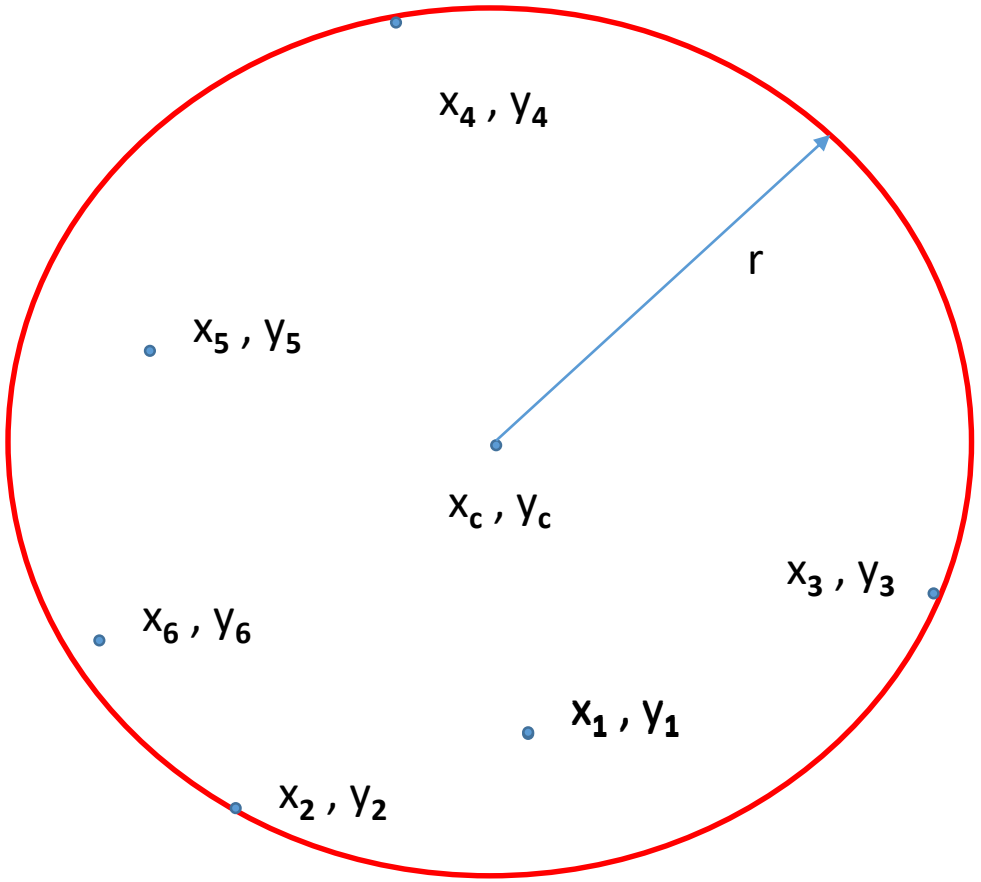
# Assignment 1

Submission Instruction:

E-mail: pds2016autumn@gmail.com
Subject Line: A1_ROLLNO
Attachments: A1_ROLLNO_P1.c/.cpp
A1_ROLLNO_P2.c/.cpp
Deadline: 11-Jan-2024 11:59 P.M.




Problem 1: Find the minimum area circle that encloses n points two dimensional Euclidean plane.

Input: n

{(x1, y1), (x2, y2), …, (xn, yn)}

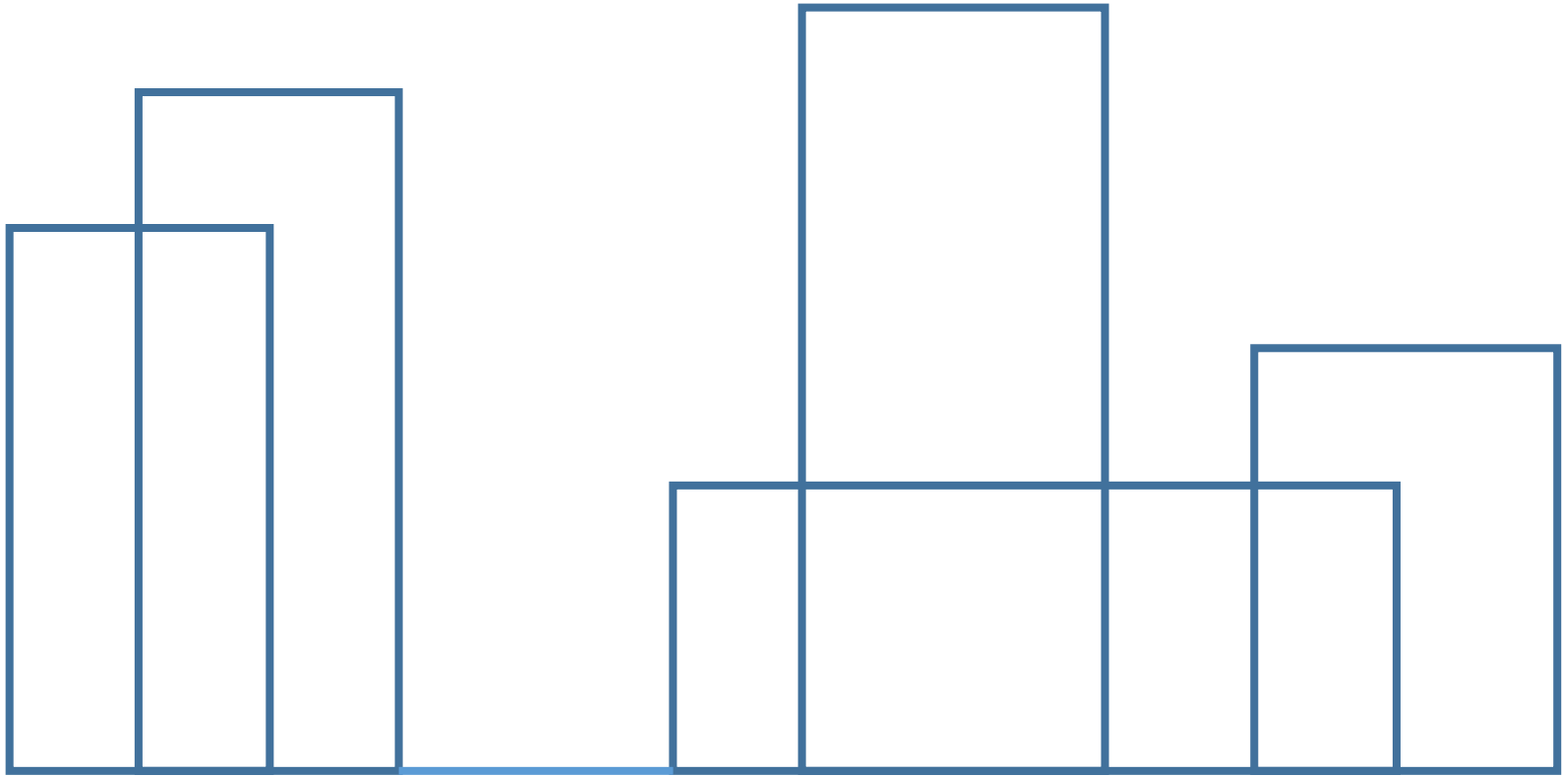
Output: $x_c$ , $y_c$ , r (radius)

Outline of an Algorithm: $O(n^4)$

1. Given any three points, a unique circle can be drawn through them (Circumcircle)
2. For each set of three points ($\binom{n}{3}$ sets), find the circle, and check whether it encloses all n points. If no, ignore it. If yes, calculate its area. Find the min area circle min1.
3. Given any two points as the endpoints of a diameter, a unique circle can be drawn through them.
4. For each set of two points ($\binom{n}{2}$ sets), find the circle, and check whether it encloses all n points. If no, ignore it. If yes, calculate its area. Find the min area circle min2.
5. Ans = min(min1, min2)

Time Complexity: $O(n^4)$

- Problem 2: Given a set of n rectangles on x-axis, remove the hidden lines to mark the boundary only.

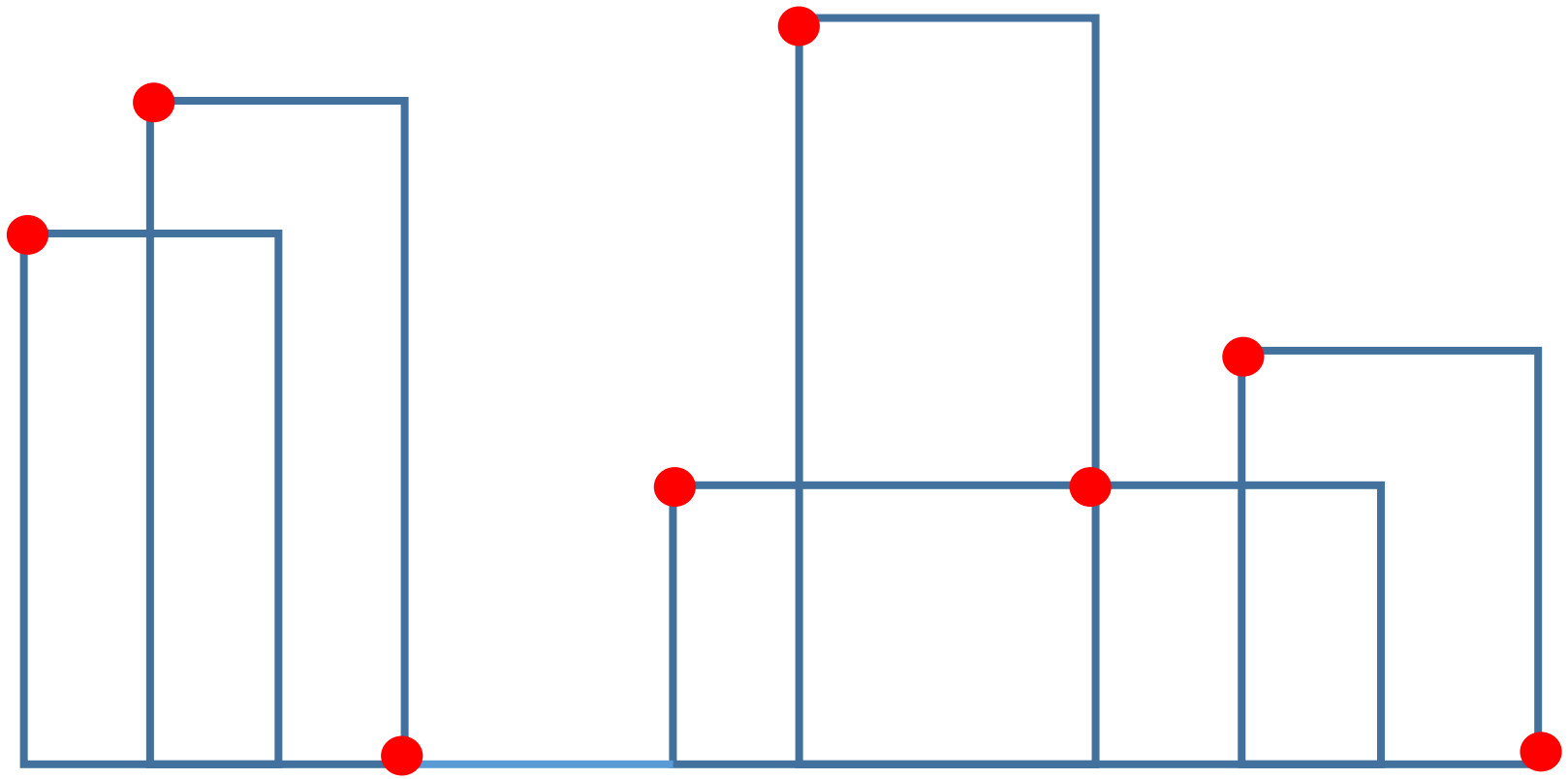


Input:

<3, 7, 10>
<5, 9, 12>
<12, 20, 5>
<13, 15, 15>
<19, 22, 8>

<startX, endX, height>
Input is sorted in non-descending order of startX

# Hidden Line Removal: Output

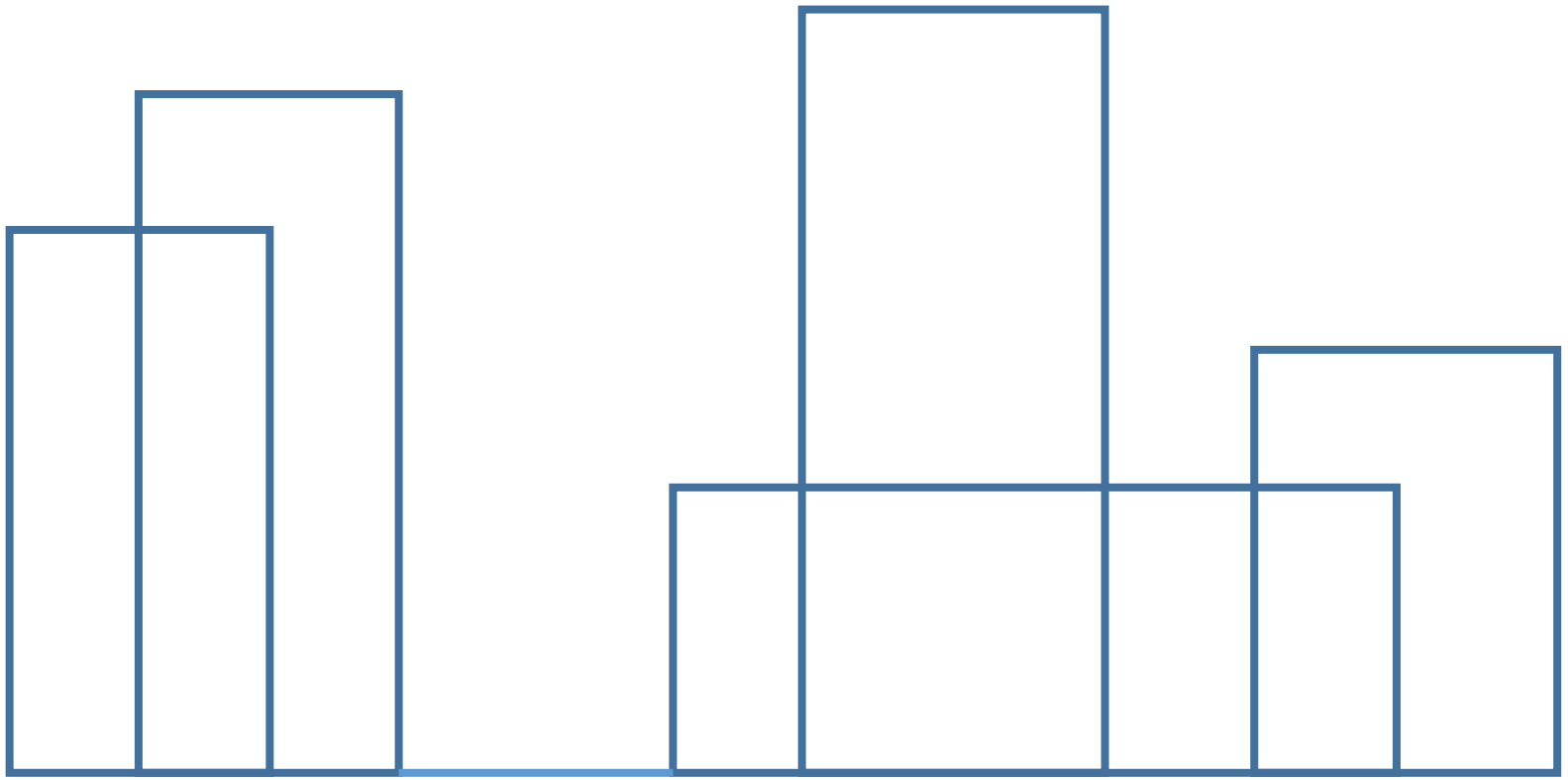


<3, 10>
<5, 12>
<9, 0>
<12, 5>
<13, 15>
<15, 5>
<19, 8>
<22, 0>

<X, H>

Output is sorted in ascending order of X

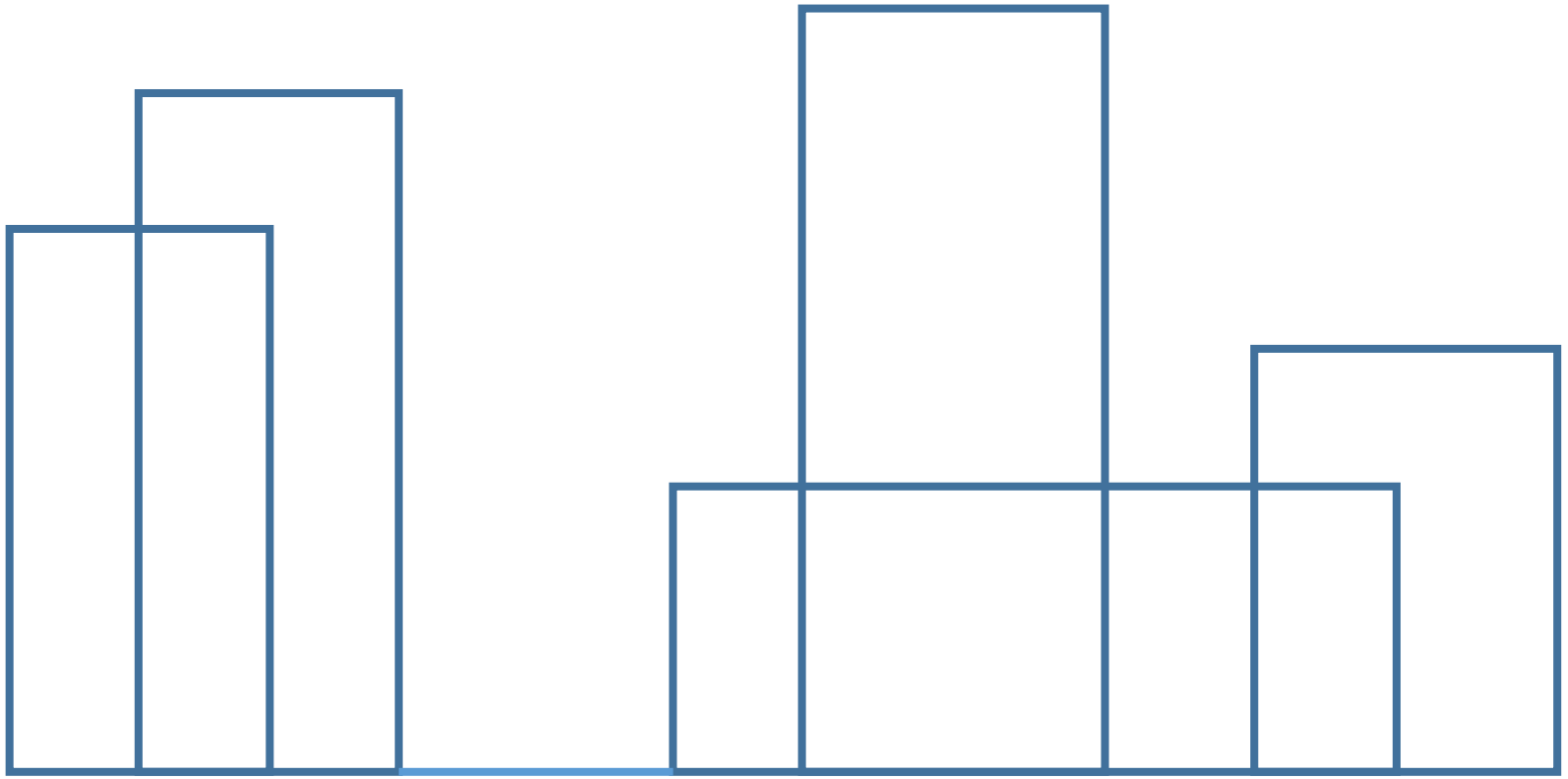
# Hidden Line Removal: Sweep Line Algorithm



<3, 10, s>
<7, 10, e>
<5, 12, s>
<9, 12, e>
<12, 5, s>
<20, 5, e>
<13, 15, s>
<15, 15, e>
<19, 8, s>
<22, 8, e>

<startX, endX, height>
Input is sorted in non-decreasing order of startX

# Hidden Line Removal: Algorithm

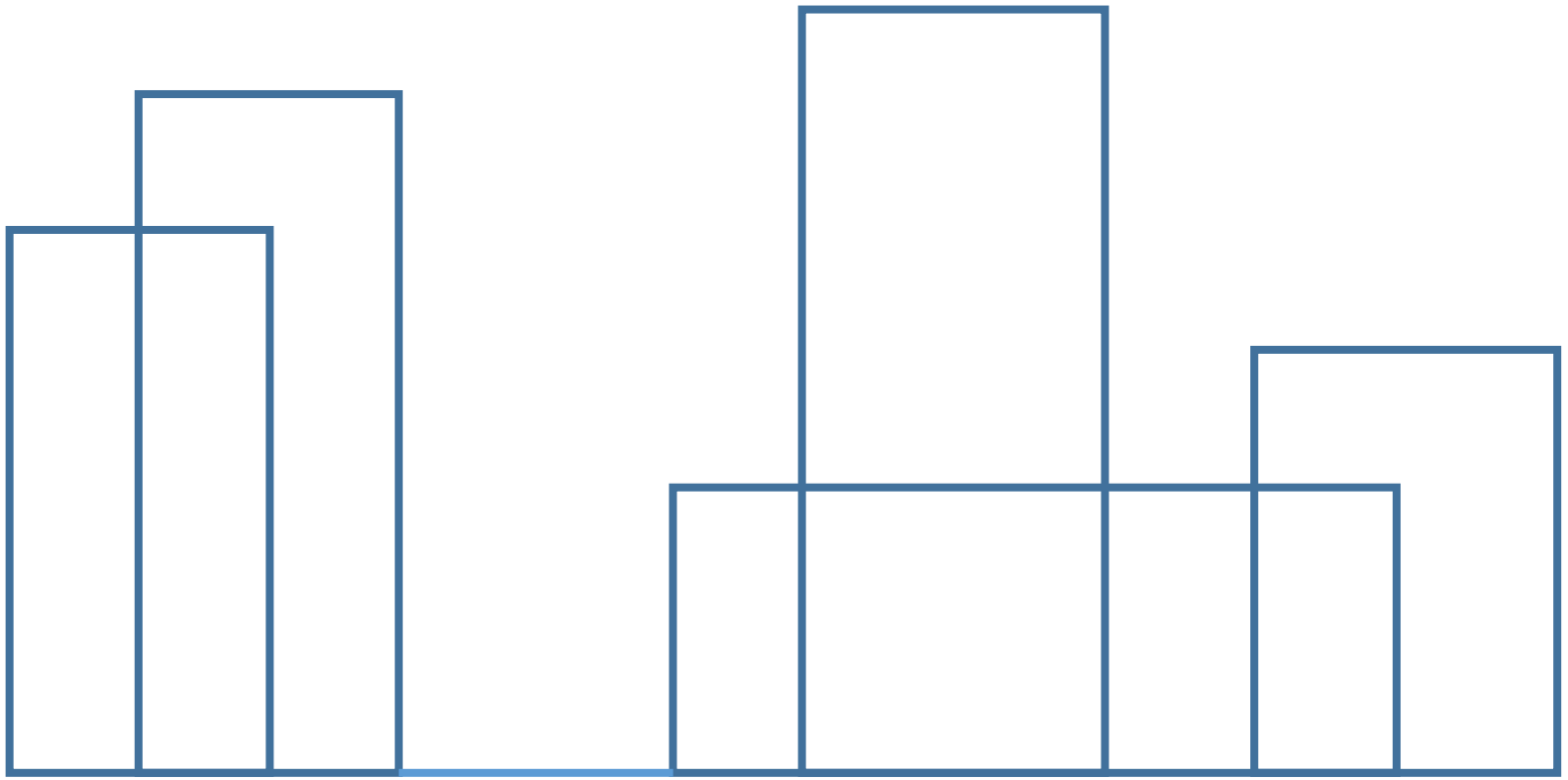


<startX, endX, height>
Input is sorted in non-decreasing order of startX

Sort X-coordinates

<3, 10, s>
<5, 12, s>
<7, 10, e>
<9, 12, e>
<12, 5, s>
<13, 15, s>
<15, 15, e>
<19, 8, s>
<20, 5, e>
<22, 8, e>

# Hidden Line Removal: Algorithm

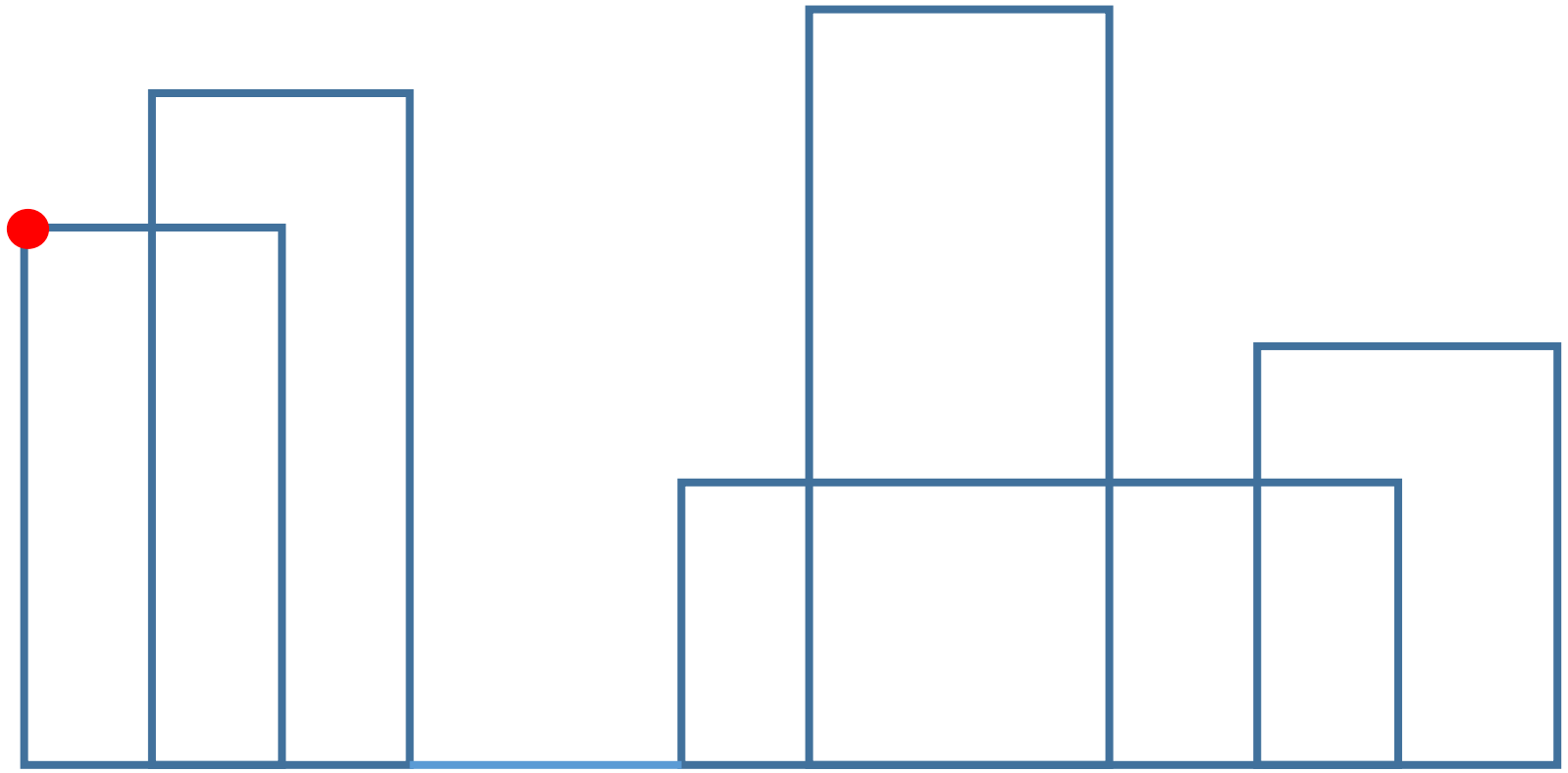


Maximum Height = 0  Heap = {0}

Scan the points from left to right, and insert/delete heights into/from Max-Heap as follows:

<3, 10, s>
<5, 12, s>
<7, 10, e>
<9, 12, e>
<12, 5, s>
<13, 15, s>
<15, 15, e>
<19, 8, s>
<20, 5, e>
<22, 8, e>

# Hidden Line Removal: Algorithm

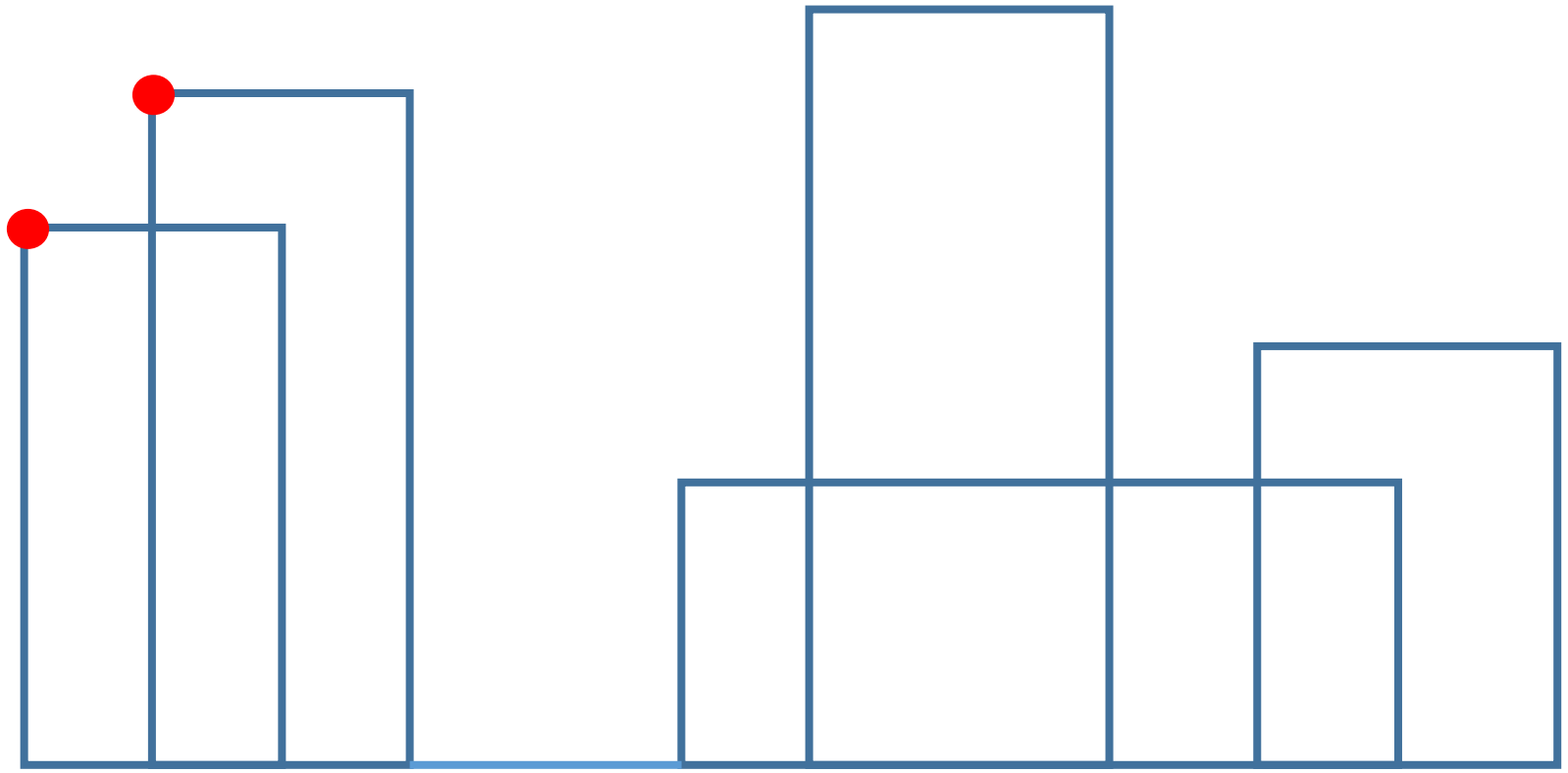


Maximum Height = 10 Heap = {10, 0} Point = (3, 10)

Start points are inserted into the heap. If the maximum height changes, then the point belongs to the boundary

<3, 10, s>
<5, 12, s>
<7, 10, e>
<9, 12, e>
<12, 5, s>
<13, 15, s>
<15, 15, e>
<19, 8, s>
<20, 5, e>
<22, 8, e>

# Hidden Line Removal: Algorithm

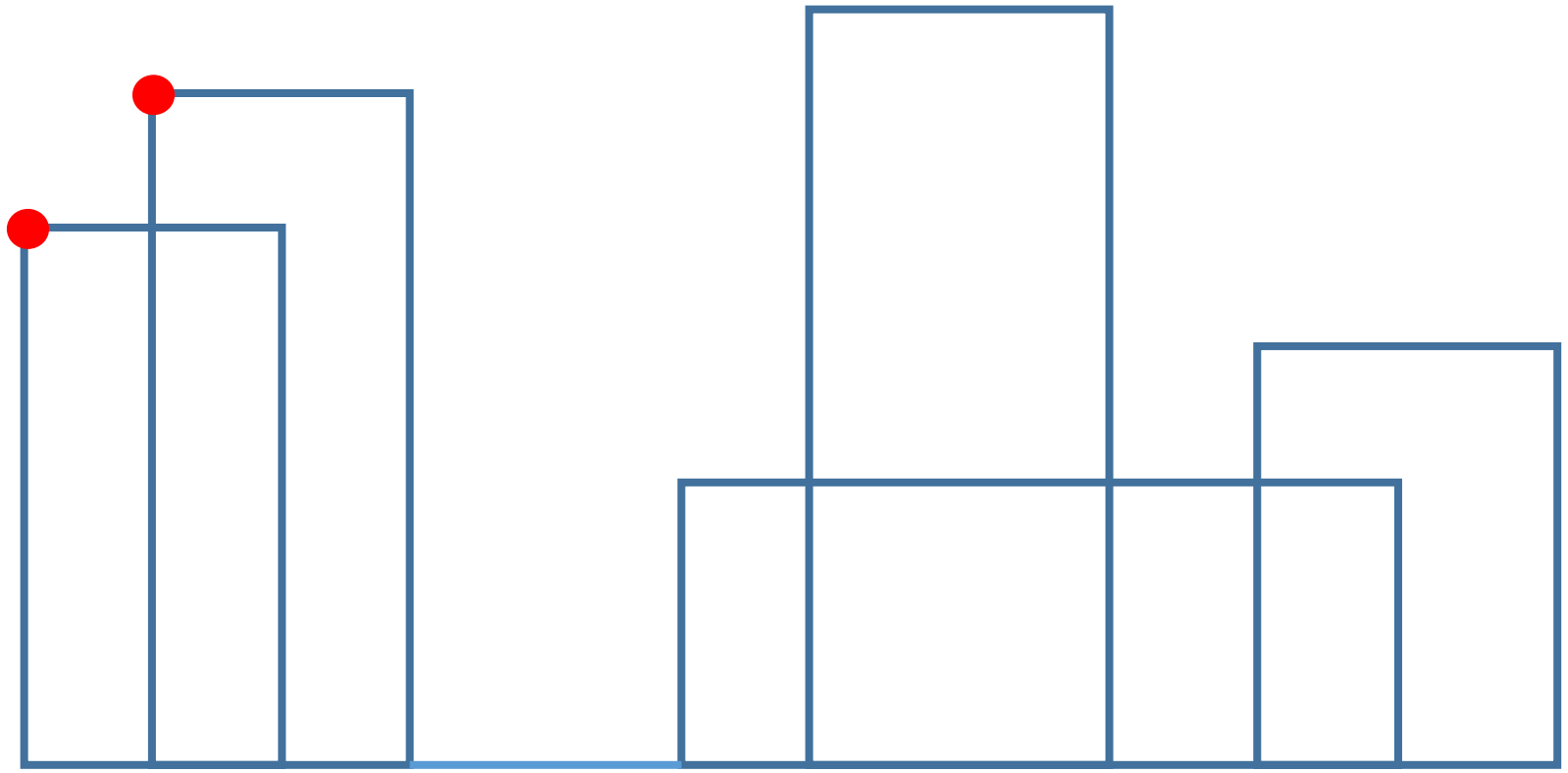


Maximum Height = 12 Heap = {12, 10, 0} Point = (5, 12)

Start points are inserted into the heap. If the maximum height changes, then the point belongs to the boundary

<3, 10, s>
<5, 12, s>
<7, 10, e>
<9, 12, e>
<12, 5, s>
<13, 15, s>
<15, 15, e>
<19, 8, s>
<20, 5, e>
<22, 8, e>

# Hidden Line Removal: Algorithm

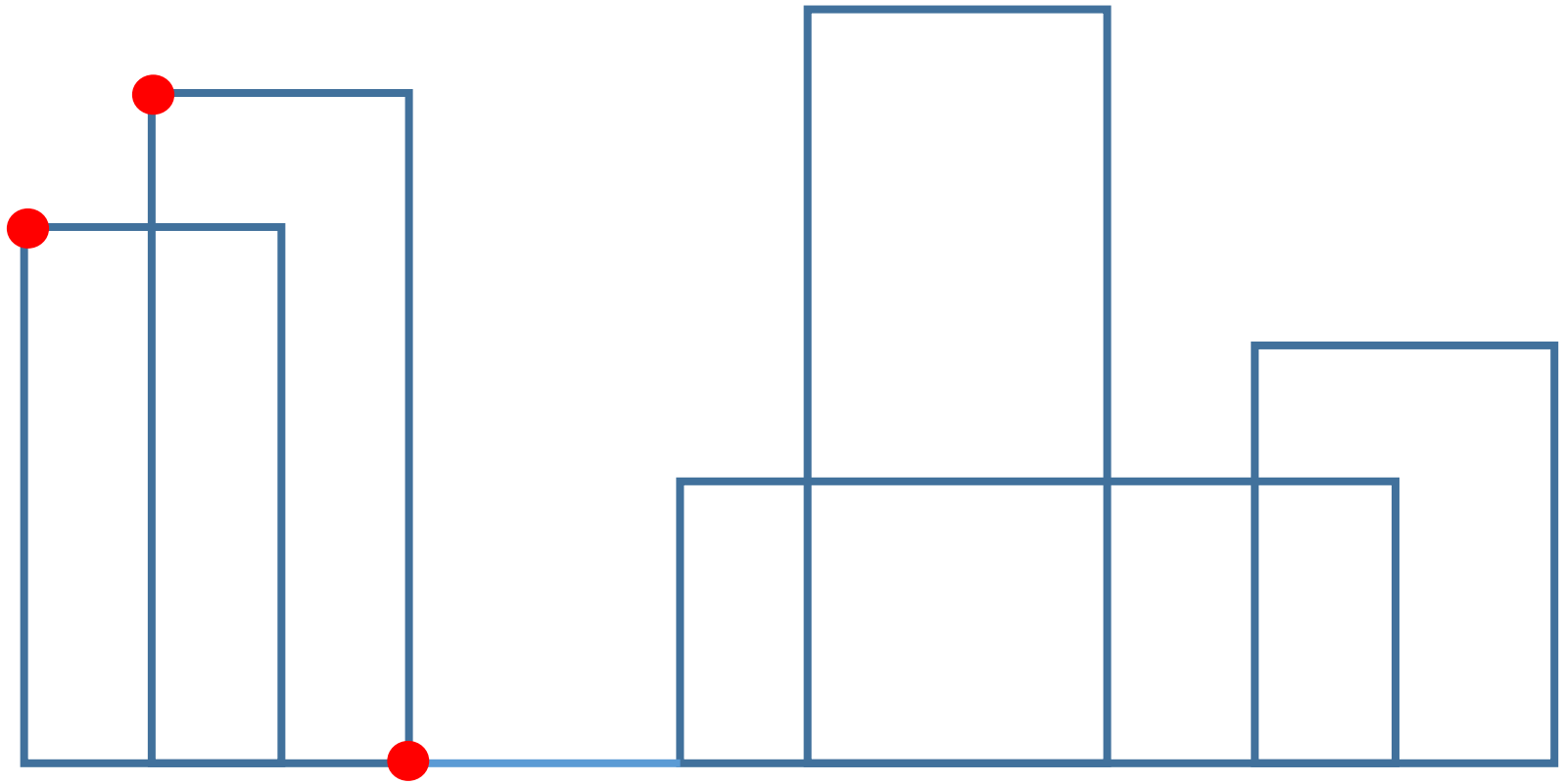


Maximum Height = 12 Heap = {12, 0}

End points are deleted from the heap. If the maximum height changes, then the point belongs to the boundary

<3, 10, s>
<5, 12, s>
<7, 10, e>
<9, 12, e>
<12, 5, s>
<13, 15, s>
<15, 15, e>
<19, 8, s>
<20, 5, e>
<22, 8, e>

# Hidden Line Removal: Algorithm

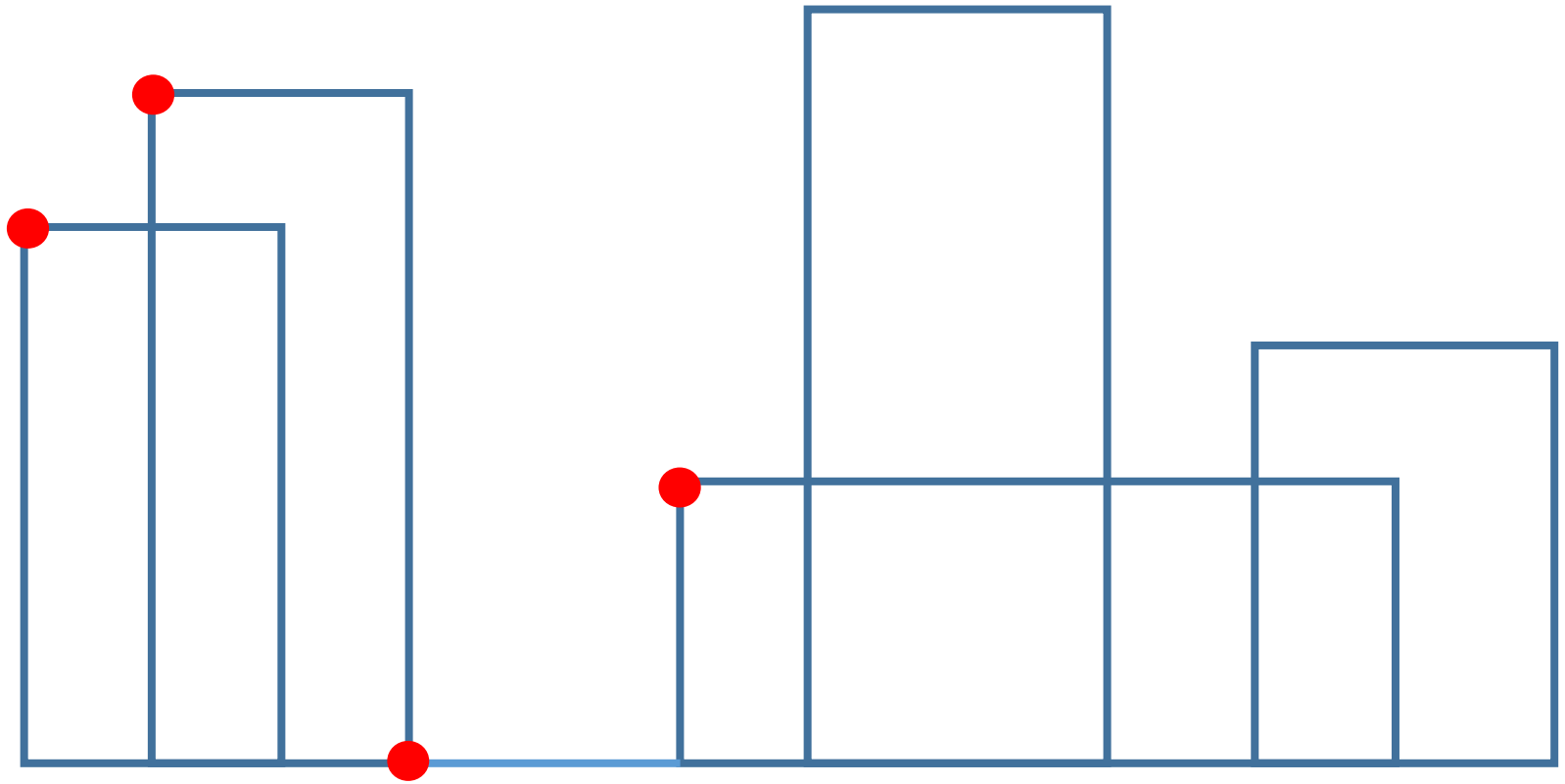


Maximum Height = 0 Heap = {0}  Point = (9, 0)

End points are deleted from the heap. If the maximum height changes, then the point belongs to the boundary

<3, 10, s>
<5, 12, s>
<7, 10, e>
<9, 12, e>
<12, 5, s>
<13, 15, s>
<15, 15, e>
<19, 8, s>
<20, 5, e>
<22, 8, e>

# Hidden Line Removal: Algorithm

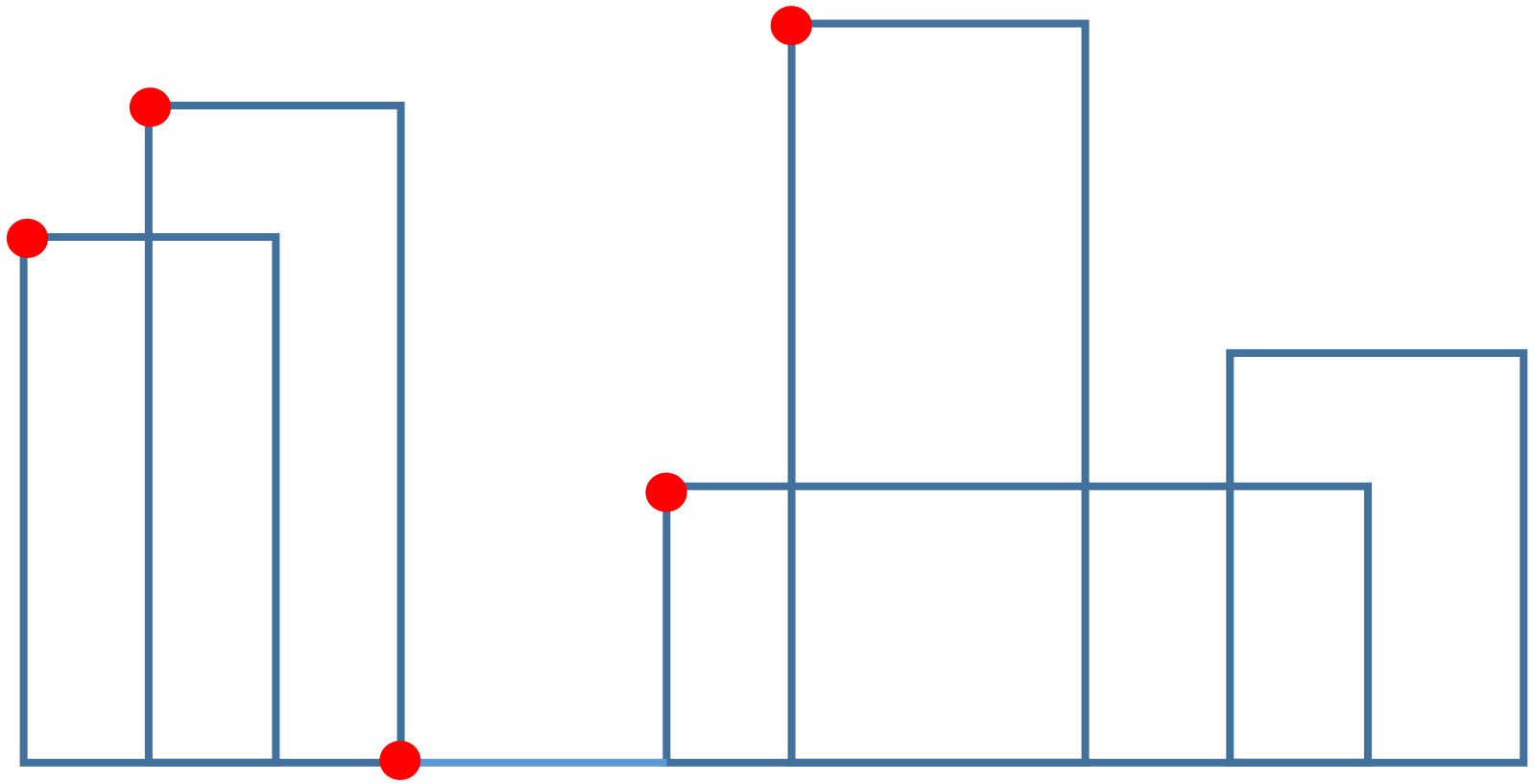


Maximum Height = 5 Heap = {5, 0}  Point = (12, 5)

Start points are inserted into the heap. If the maximum height changes, then the point belongs to the boundary

<3, 10, s>
<5, 12, s>
<7, 10, e>
<9, 12, e>
<12, 5, s>
<13, 15, s>
<15, 15, e>
<19, 8, s>
<20, 5, e>
<22, 8, e>

# Hidden Line Removal: Algorithm

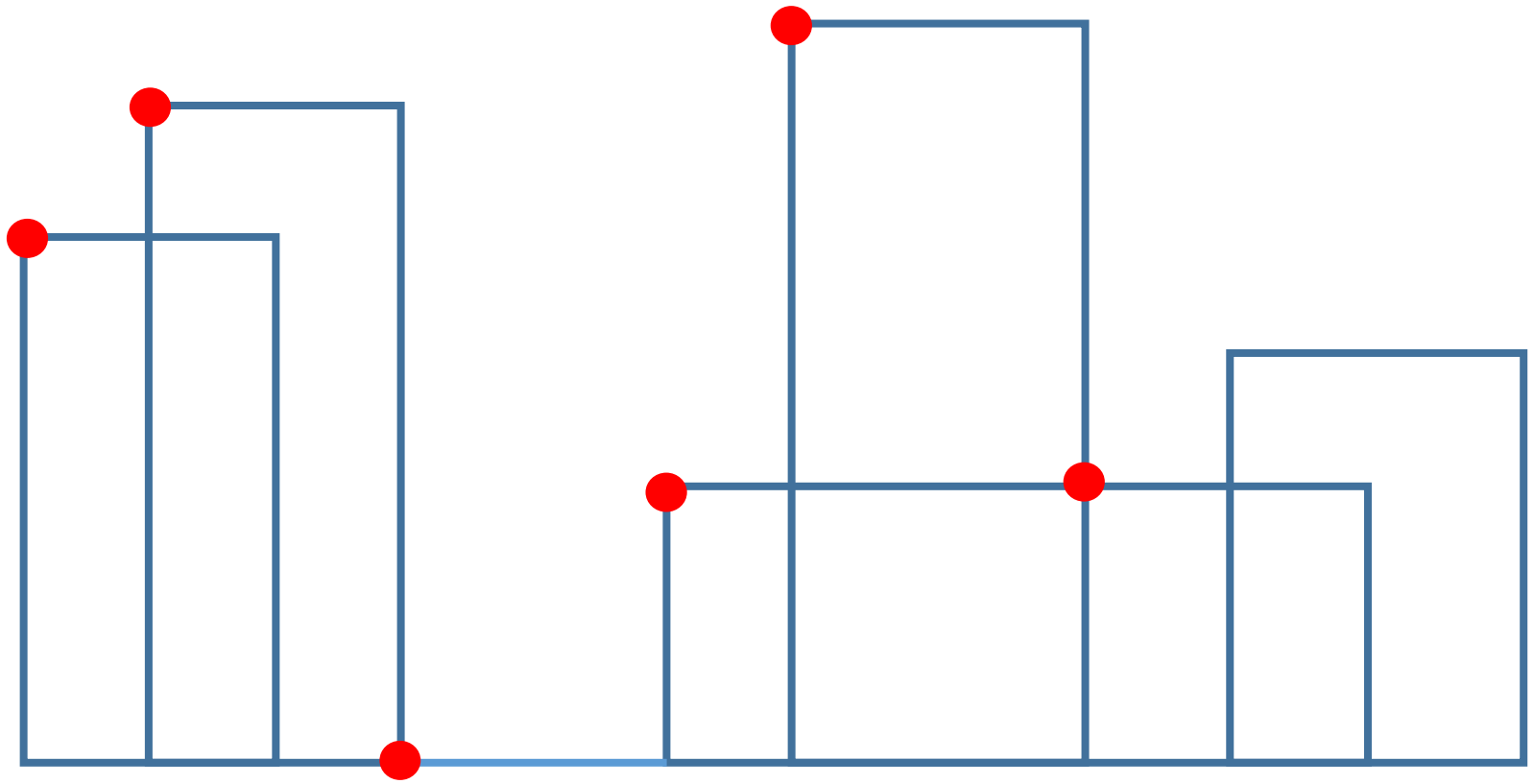


Maximum Height = 15 Heap = {15, 5, 0}  Point = (13, 15)

Start points are inserted into the heap. If the maximum height changes, then the point belongs to the boundary

<3, 10, s>
<5, 12, s>
<7, 10, e>
<9, 12, e>
<12, 5, s>
<13, 15, s>
<15, 15, e>
<19, 8, s>
<20, 5, e>
<22, 8, e>

# Hidden Line Removal: Algorithm

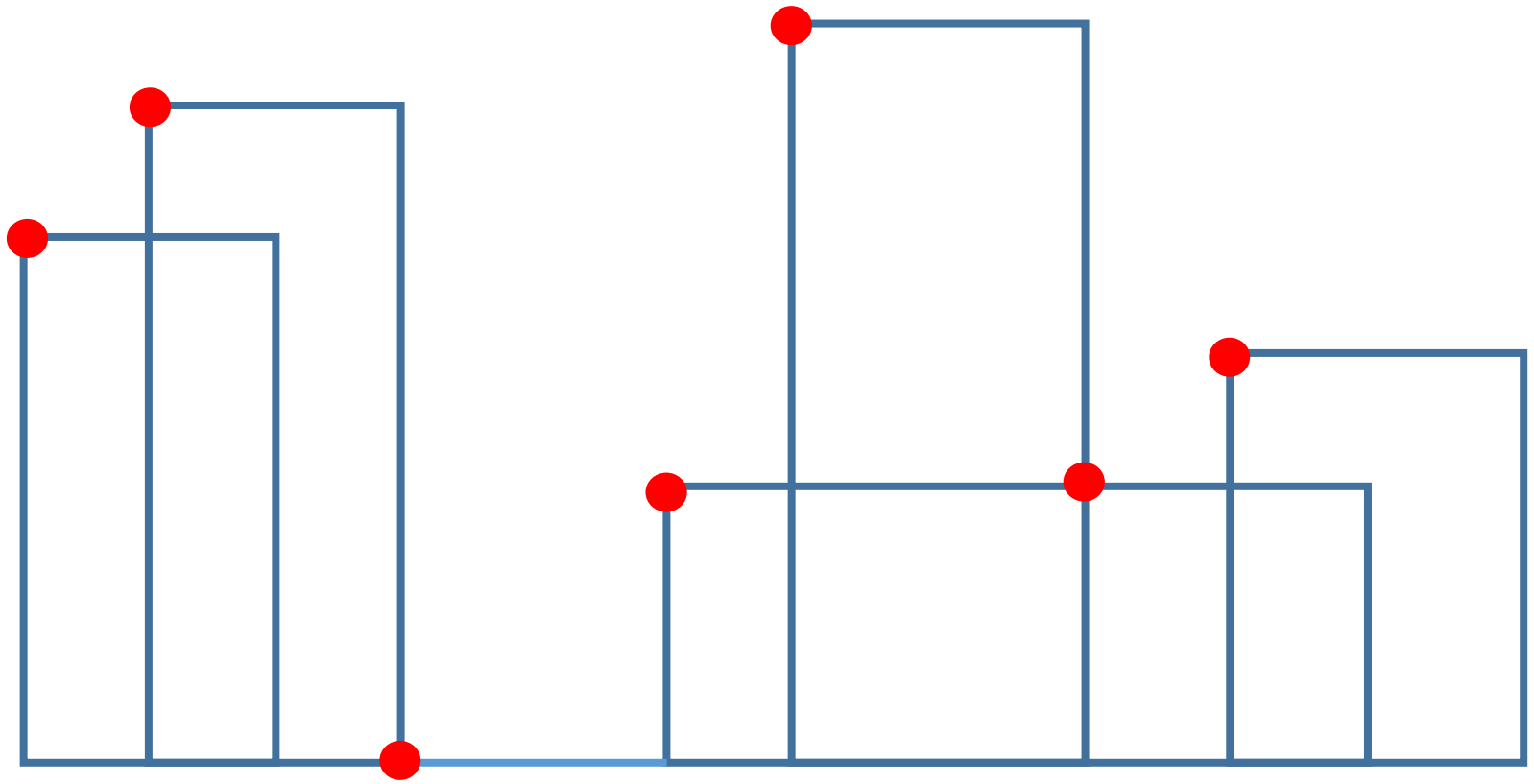


Maximum Height = 5 Heap = {5, 0}  Point = (15, 5)

End points are deleted from the heap. If the maximum height changes, then the point belongs to the boundary

<3, 10, s>
<5, 12, s>
<7, 10, e>
<9, 12, e>
<12, 5, s>
<13, 15, s>
<15, 15, e>
<19, 8, s>
<20, 5, e>
<22, 8, e>

# Hidden Line Removal: Algorithm

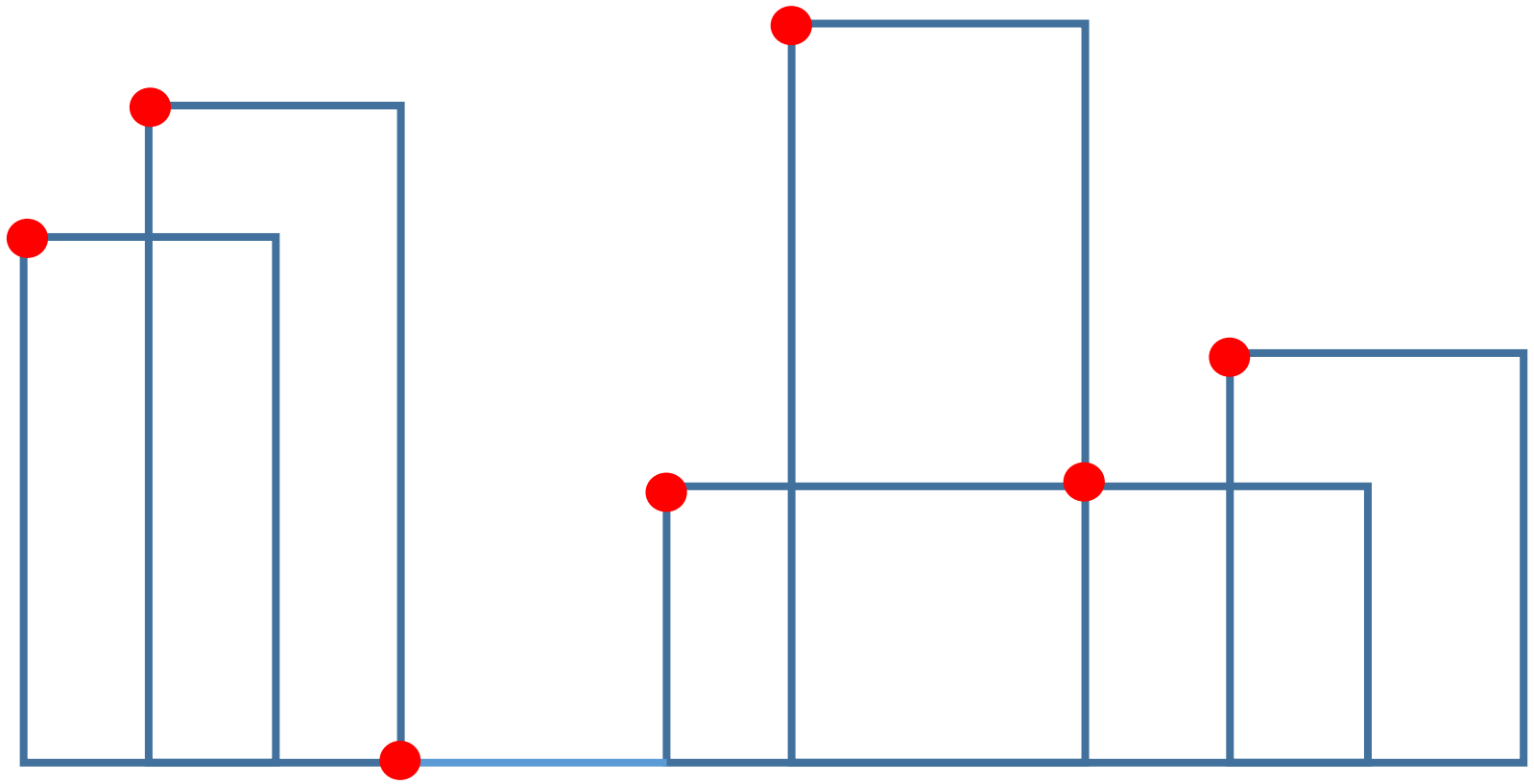


Maximum Height = 8 Heap = {8, 5, 0}  Point = (19, 8)

Start points are inserted into the heap. If the maximum height changes, then the point belongs to the boundary

<3, 10, s>
<5, 12, s>
<7, 10, e>
<9, 12, e>
<12, 5, s>
<13, 15, s>
<15, 15, e>
<19, 8, s>
<20, 5, e>
<22, 8, e>

# Hidden Line Removal: Algorithm

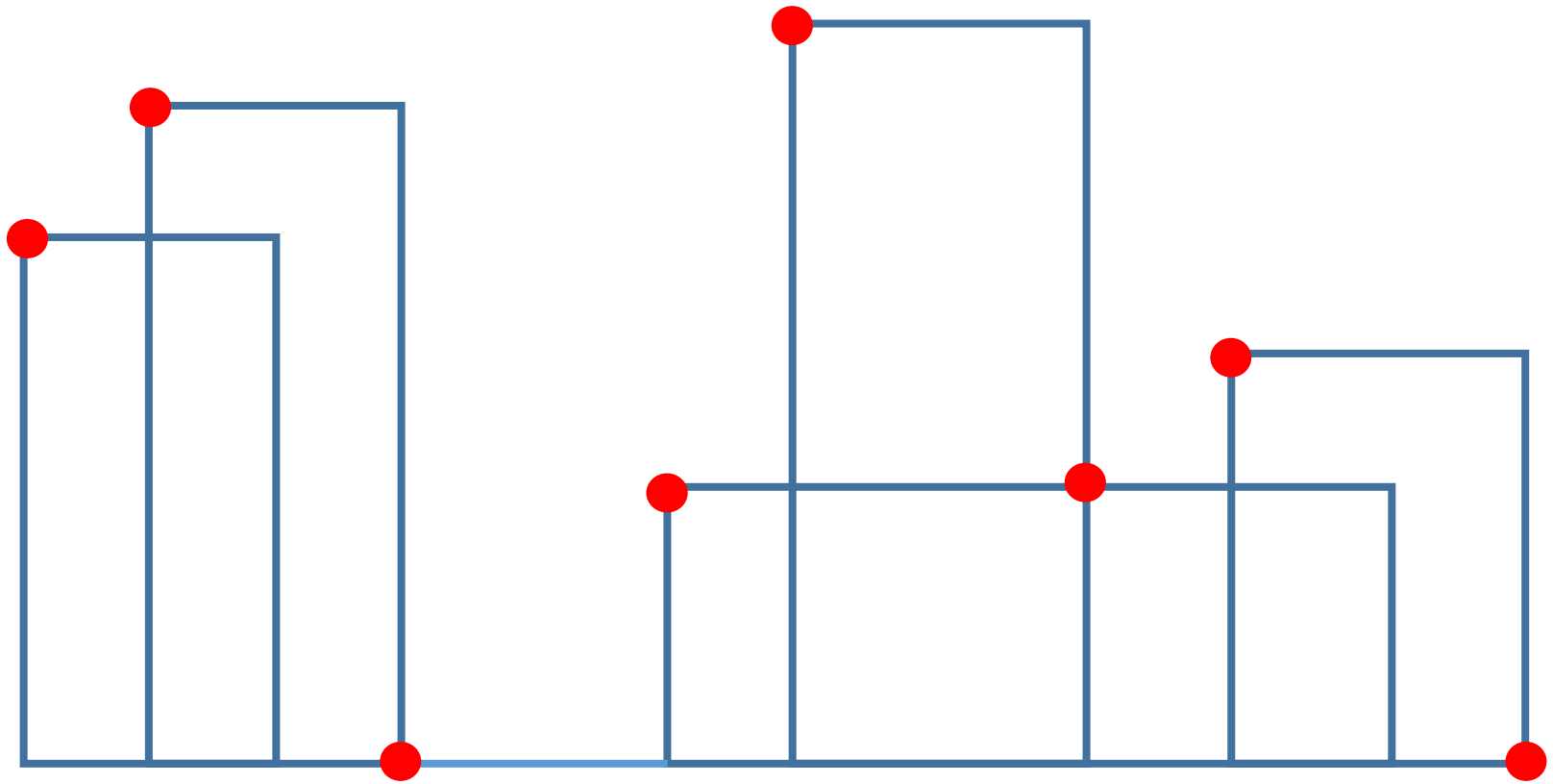


Maximum Height = 8 Heap = {8, 0}

End points are deleted from the heap. If the maximum height changes, then the point belongs to the boundary

<3, 10, s>
<5, 12, s>
<7, 10, e>
<9, 12, e>
<12, 5, s>
<13, 15, s>
<15, 15, e>
<19, 8, s>
<20, 5, e>
<22, 8, e>

# Hidden Line Removal: Algorithm



Maximum Height = 0 Heap = {0} Point = (22, 0)

End points are deleted from the heap. If the maximum height changes, then the point belongs to the boundary

<3, 10, s>
<5, 12, s>
<7, 10, e>
<9, 12, e>
<12, 5, s>
<13, 15, s>
<15, 15, e>
<19, 8, s>
<20, 5, e>
<22, 8, e>

# Special Cases