Lab Manual 07 Stack Problems

1. Largest Rectangle in Histogram

Problem Statement:
You are given an array heights[] of size n, where each element represents the height of a bar in a histogram. Each bar has a width of 1. Find the largest rectangular area that can be formed in the histogram.

Example:

Input:
heights = [2, 1, 5, 6, 2, 3]

Output:
10

Explanation:
The largest rectangle is formed by heights [5, 6] (from index 2 to 3) with a width of 2 and a height of 5, so the area is 5 * 2 = 10.

Hints: Use a monotonic increasing stack to keep track of indices and calculate the max area.

2. Trapping Rainwater Problem

Problem Statement:
Given an array heights[] of size n, where heights[i] represents the height of the building at index i, determine the amount of rainwater trapped between the buildings after rainfall.

Example:

Input:
heights = [0, 1, 0, 2, 1, 0, 1, 3, 2, 1, 2, 1]

Output:
6

Explanation:
Water is trapped at indices 2, 4, 5, 6, 9, 10 with units [1, 1, 2, 1, 1, 1], totaling 6 units.

Hints: Use a monotonic decreasing stack or track left max and right max heights.

3. Find Celebrity in a Party (Stack Approach)

Problem Statement:
You are given n people at a party, labeled as 0 to n-1. A celebrity is a person who:
1. Knows nobody at the party.
2. Is known by everyone at the party.

You are given a function knows(a, b) which returns True if a knows b, and False otherwise. Find the celebrity in O(n) time complexity using a stack.

Example:

Input (Matrix Representation of knows(a, b)):
M = [[0, 1, 1], [0, 0, 1], [0, 0, 0]]

Output:
2

Explanation:
Person 2 is known by everyone (0 and 1) and does not know anyone, so 2 is the celebrity.

Hints: Use a stack to eliminate non-celebrities and verify the last candidate.

4. Design a Special Stack with Two Stacks

Problem Statement:
Design a stack that supports the following operations in O(1) time:
1. push(x): Push an element onto the stack.
2. pop(): Remove the top element.
3. get_min(): Get the minimum element in the stack.
4. get_max(): Get the maximum element in the stack.

Example:

Input:
s = SpecialStack()
s.push(5)
s.push(1)
s.push(3)

Output:
print(s.get_min()) # 1
print(s.get_max()) # 5

Hints: Use two stacks: main_stack for storing elements and min_stack/max_stack for tracking min/max values.