Lab Practice Sheet 3: Stacks

CS G526: Advanced Algorithms & Complexity

Instructor: Tulasimohan Molli **Date:** 2026-09-26

! Evaluation Instructions for Students

- **Practice all 8 problems below.**
- In the next lab session, **one question will be chosen at random** for on-the-spot coding and evaluation.
- Your code will be evaluated by running it against automated hidden test cases.
- You may write your solutions in **Python**, **C++**, or **Java**.
- **Library Usage:** You are allowed and encouraged to use built-in standard libraries for data structures (e.g., `stack` or `vector` in C++, `collections.deque` or `list` in Python, `Stack` or `ArrayDeque` in Java).
- Standard I/O (reading from standard input `stdin` and printing to standard output `stdout`) is required.

Q1: Valid Parentheses & Bracket Matching

Description: Given a string S containing only the characters `'('`, `')'`, `'{'`, `'}'`, `'['`, and `']'`, determine whether the bracket sequence is valid.

A bracket sequence is valid if: 1. Every open bracket is closed by a matching bracket of the same type. 2. Open brackets are closed in the correct Last-In, First-Out (LIFO) order. 3. Every closing bracket corresponds to a preceding open bracket of the same type.

- **Input Format:**
  - A single line containing the string S.
- **Output Format:**
  - Print `True` if the string is valid, or `False` otherwise.

i Sample Input 1

```
()[]{}
```

i Sample Output 1

```
True
```

i Sample Input 2

```
([)]
```

i Sample Output 2

```
False
```

Q2: Next Greater Element (Monotonic Decreasing Stack)

Description: Given an array of N integers, find the **Next Greater Element (NGE)** for each element. The Next Greater Element of an element x is the first element strictly to its right in the array that is strictly greater than x. If no greater element exists to its right, output `-1` for that element.

Solve this in $O(N)$ time using a monotonic stack.

- **Input Format:**
  - Line 1: An integer N, the size of the array.
  - Line 2: N space-separated integers representing the array.
- **Output Format:**
  - A single line containing N space-separated integers, where the i-th integer is the next greater element for the i-th element of the input.

Sample Input 1

```
4
4 5 2 25
```

Sample Output 1

```
5 25 25 -1
```

Sample Input 2

```
5
13 7 6 12 10
```

Sample Output 2

```
-1 12 12 -1 -1
```

Q3: Next Smaller Element

Description: Given an array of N integers, find the **Next Smaller Element (NSE)** for each element. The Next Smaller Element of an element x is the first element strictly to its right in the array that is strictly smaller than x. If no smaller element exists to its right, output `-1` for that position.

Solve this in $O(N)$ time using a monotonic stack.

- **Input Format:**
  - Line 1: An integer N, the size of the array.
  - Line 2: N space-separated integers representing the array.
- **Output Format:**
  - A single line containing N space-separated integers representing the next smaller element for each index.

Sample Input 1

```
5
4 2 1 5 3
```

Sample Output 1

```
2 1 -1 3 -1
```

Sample Input 2

```
4
5 6 7 8
```

Sample Output 2

```
-1 -1 -1 -1
```

Q4: Daily Temperatures (Span to Next Warmer Day)

Description: Given an array of N integers representing daily temperatures, return an array A such that $A[i]$ is the number of days you have to wait after the i-th day to get a warmer temperature. If there is no future day for which the temperature is warmer, set $A[i] = 0$.

Solve this in $O(N)$ time using a monotonic stack storing indices.

- **Input Format:**
  - Line 1: An integer N, the number of days.
  - Line 2: N space-separated integers representing the daily temperatures.
- **Output Format:**
  - A single line containing N space-separated integers representing the wait times.

Sample Input 1

```
8
73 74 75 71 69 72 76 73
```

Sample Output 1

```
1 1 4 2 1 1 0 0
```

Sample Input 2

```
4
30 40 50 60
```

Sample Output 2

```
1 1 1 0
```

Q5: Evaluate Reverse Polish Notation (Postfix Expression)

Description: Evaluate the value of an arithmetic expression in **Reverse Polish Notation (RPN)**. Valid operators are `+`, `-`, `*`, and `/`. Each operand may be an integer or another expression. Division between two integers should truncate toward zero.

Solve this in $O(N)$ time using an evaluation stack.

- **Input Format:**
  - Line 1: An integer N, the number of tokens in the RPN expression.
  - Line 2: N space-separated tokens (integers or operators `+`, `-`, `*`, `/`).
- **Output Format:**
  - A single integer representing the final evaluated value.

Sample Input 1

```
5
2 1 + 3 *
```

Sample Output 1

```
9
```

Explanation: $((2 + 1) \times 3) = 9$.

Sample Input 2

```
5
4 13 5 / +
```

Sample Output 2

```
6
```

Explanation: $(4 + (13/5)) = 4 + 2 = 6$.

Q6: Min Stack with $O(1)$ Operations

Description: Design a stack data structure that supports `push`, `pop`, `top`, and retrieving the minimum element in $O(1)$ time.

You are given a sequence of Q operations. The operations are: - `push X`: Push integer X onto the stack. - `pop`: Remove the element on top of the stack. - `top`: Print the top element of the stack. - `getMin`: Print the minimum element currently present in the stack.

- **Input Format:**
  - Line 1: An integer Q, the number of operations.
  - Next Q lines: Each line contains one of the operations (`push X`, `pop`, `top`, or `getMin`).
- **Output Format:**
  - For each `top` and `getMin` operation, print the resulting integer on a new line.

Sample Input

```
7
push -2
push 0
push -3
getMin
pop
top
getMin
```

Sample Output

```
-3
0
-2
```

Q7: Asteroid Collision

Description: You are given an array of N integers representing asteroids in a row. For each asteroid, the absolute value represents its size, and the sign represents its direction (positive meaning moving right, negative meaning moving left). All asteroids move at the same speed.

Find out the state of the asteroids after all collisions: - If two asteroids meet, the smaller one explodes. - If both are the same size, both explode. - Two asteroids moving in the same direction or moving away from each other will never meet.

- **Input Format:**
  - Line 1: An integer N, the number of asteroids.
  - Line 2: N space-separated integers representing the asteroid array.
- **Output Format:**
  - A single line containing the remaining asteroids separated by spaces. If no asteroids survive, print `Empty`.

Sample Input 1

```
3
5 10 -5
```

Sample Output 1

```
5 10
```

Explanation: The 10 and -5 collide resulting in 10. The 5 and 10 never collide.

Sample Input 2

```
3
8 -8
```

Sample Output 2

```
Empty
```

Sample Input 3

```
3
10 2 -5
```

Sample Output 3

```
10
```

Q8: Largest Rectangle in Histogram

Description: Given an array of N non-negative integers representing the histogram's bar heights where the width of each bar is 1, find the area of the largest rectangle that can be formed in the histogram.

Solve this in $O(N)$ time using a monotonic stack.

- **Input Format:**
  - Line 1: An integer N, the number of bars.
  - Line 2: N space-separated integers representing the heights of the bars.
- **Output Format:**
  - A single integer representing the maximum rectangular area.

Sample Input 1

```
6
2 1 5 6 2 3
```

Sample Output 1

```
10
```

Explanation: The largest rectangle is formed by bars of height 5 and 6: area $= 5 \times 2 = 10$.

Sample Input 2

```
4
2 4 2 1
```

Sample Output 2

```
6
```