Lab Assignment: Sort a Stack Using Another Stack

Problem Statement

Given a stack of integers, sort it such that the smallest elements are on the top. Only standard stack operations are allowed: `push()`, `pop()`, `peek()`, and `isEmpty()`. The solution must use **only one additional stack** as temporary storage.

Stack Operations Required

- `push(Stack *s, int value)` – Pushes an integer onto the stack.
- `int pop(Stack *s)` – Pops and returns the top element of the stack.
- `int peek(Stack *s)` – Return the top element without removing it.
- `int isEmpty(Stack *s)` – Checks if the stack is empty.
- `void init(Stack *s)` – Initializes the stack.

Algorithm

1. Initialize a temporary empty stack.
2. While the original stack is not empty:
    - Pop an element from the original stack (let's call it `temp`).
    - While the temporary stack is not empty and its top element is greater than `temp`:
        - Pop from the temporary stack and push it back to the original stack.
    - Push `temp` into the temporary stack.
3. At the end, the temporary stack will have the sorted elements in ascending order with the smallest at the top.

Example

Input Stack (Top → Bottom): `34, 3, 31, 98, 92, 23`
Output Stack (Top → Bottom): `3, 23, 31, 34, 92, 98`