Lab Assignment: Application of Stack - Postfix Expression Evaluation

Problem Statement

Given a valid postfix expression consisting of single-digit operands and binary operators (+, −, *, /), evaluate the result using a stack. The algorithm must simulate the step-by-step process using standard stack operations..

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 isEmpty(Stack *s)` – Checks if the stack is empty.
- `void init(Stack *s)` – Initializes the stack.

Algorithm

1. Initialize an empty stack.
2. Traverse the postfix expression from left to right.
3. For each symbol:
    - If it is an operand, push it onto the stack.
    - If it is an operator:
        - Pop the top two elements.
        - Apply the operator: `result = operand2 operator operand1`
        - Push the result back onto the stack.
4. At the end, the result is the top of the stack.

Example

Postfix Expression: `5 1 2 + 4 * + 3 -`
Evaluation Steps:

```
→ 1 + 2 = 3
→ 3 * 4 = 12
→ 5 + 12 = 17
→ 17 - 3 = 14
```

Result: 14

Sample Input and Output

- **Input**: `5 1 2 + 4 * + 3 -`
- **Output**: `14`