Postfix Expression Evaluation - Detailed Breakdown

Introduction

This document explains how to evaluate a postfix expression using a stack in C++. A step-by-step breakdown of the code, along with a dry run and test cases, is provided for better understanding.

Code Snippet

```
#include<iostream>
#include<string>
#include<stack>
using namespace std;

int priority(char ch){
  if(ch=='+'||ch=='-') return 1;
  else if(ch=='*'||ch=='/')return 2;
}

int eval(int v1,int v2,char ch){
  if(ch=='+') return v1+v2;
  else if(ch == '-')return v1-v2;
  else if(ch == '*') return v1*v2;
  else return v1/v2;
}

int main(){
  string s = "126+4*8/+3-";
  stack<int> val;
  for(int i=0;i<s.length();i++){
    if(s[i]>=48 && s[i]<=57){
      val.push(s[i]-48);
    }
    else{
        int v2 = val.top();
        val.pop();
        char ch = s[i];
        int v1 = val.top();
        val.pop();
        int ans = eval(v1,v2,ch);
        val.push(ans);
        }
    }
    cout<<val.top();
    return 0;
}
```

Thought Process Behind the Code

1. If the character is a digit, convert it to an integer and push it onto the stack.

2. If an operator is found, pop the top two values from the stack and apply the operator.
3. Push the result back onto the stack.
4. At the end, the stack will contain the final result.

Dry Run (Step-by-Step Execution)

For the input: 126+4*8/+3-

1. Push 1, 2, and 6 onto the stack.
2. Apply '+': (2+6) = 8, push 8.
3. Push 4.
4. Apply '*': (8*4) = 32, push 32.
5. Push 8.
6. Apply '/': (32/8) = 4, push 4.
7. Apply '+': (8+4) = 12, push 12.
8. Push 3.
9. Apply '-': (12-3) = 9, final result.

Final Output

9

Short Notes

- Digits are pushed onto the stack.
- Operators apply operations to the top two stack values.
- The stack's final value is the result.
- Works efficiently using a Last-In-First-Out (LIFO) structure.