Problem Statement:

Ek arithmetic expression diya gaya hai jo infix notation me likha hai, jaise:

2 + 6 * 4 / 8 - 3

Hume is expression ko evaluate karna hai using **stacks**.

Approach (Thought Process):

1. **Two Stacks Use Karenge:** Ek **values ka stack** (val) aur ek **operators ka stack** (op).
2. **Expression Traverse Karna:** Ek-ek character par iterate karenge.
    - **Agar digit mile:** usko val stack me daal dena.
    - **Agar operator mile:** tab check karna ki op stack me precedence kaunsa zyada hai.
        - Agar op ka top precedence se kam hai, to naya operator push kar do.
        - Nahi to, pehle op stack ka evaluation karo phir naya operator push karo.
3. **End me Bache Hue Operators Ko Evaluate Karna.**
4. **Final Result val.top() me Milega.**

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;
  return 0;
}

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 = "2+6*4/8-3";
  stack<int> val;
  stack<char> op;
  for(int i=0;i<s.length();i++){
    if(s[i]>=48 && s[i]<=57){
      val.push(s[i]-48);
    }
    else{
      if(op.size()==0 || priority(op.top())<priority(s[i])){
        op.push(s[i]);
      }
      else{
        while(op.size()>0 && priority(op.top())>=priority(s[i])){
          int v2 = val.top(); val.pop();
          char ch = op.top(); op.pop();
          int v1 = val.top(); val.pop();
          int ans = eval(v1,v2,ch);
          val.push(ans);
        }
        op.push(s[i]);
      }
    }
  }
  while(op.size()>0){
    int v2 = val.top(); val.pop();
    char ch = op.top(); op.pop();
    int v1 = val.top(); val.pop();
```

```
   int ans = eval(v1,v2,ch);
   val.push(ans);
  }
  cout<<val.top();
  return 0;
}
```

Dry Run Table:

Step	**Character**	**val Stack**	**op Stack**	**Action**
1	'2'	[2]	[]	Push 2 into val
2	'+'	[2]	[+]	Push + into op
3	'6'	[2,6]	[+]	Push 6 into val
4	'*'	[2,6]	[+,*]	Push * into op (higher precedence)
5	'4'	[2,6,4]	[+,*]	Push 4 into val
6	'/'	[2,6,4]	[+,*]	Since * >= /, perform 6 * 4 = 24, push 24 into val
7	'8'	[2,24,8]	[+/]	Push 8 into val
8	'-'	[2,3]	[-]	Perform 24 / 8 = 3, push 3 into val
9	'3'	[2,3,3]	[-]	Push 3 into val
10	End	[2]	[]	Perform 3 - 3 = 0, Perform 2 + 0 = 2

Correct Answer: 2

Flaws in Code:

1. **Priority Function Return Issue:** Agar invalid operator aaye to function garbage value return karega. Isko fix karne ke liye return 0 add karna chahiye.
2. **Multi-Digit Numbers Handle Nahi Ho Rahe:** Agar input me koi multi-digit number aaye jaise 12+34, to ye code sirf 1 aur 2 alag maan lega.
3. **Division by Zero Handle Nahi Hai:** Agar kabhi v2 == 0 ho gaya, to program crash ho sakta hai.
4. **Operator Stack Overflow/Underflow Handle Nahi Kiya:** Agar operator precedence sahi se manage nahi hua to infinite loop lag sakta hai.

Final Output:

```
Result: 2
```