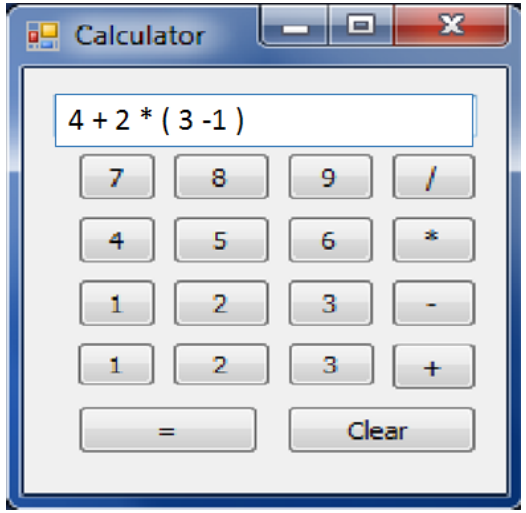
PART 1
Problem: **Why We Need Stack in Calculator Problem**

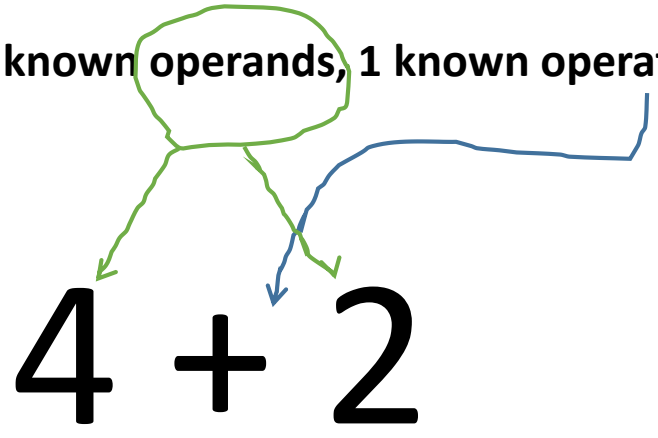


we will learn how *stack* is used in doing artihmetic calculation by a calculator.

Our calculator will receive the arithmetic expression at once, and then display the final result.

Arithmetic calculation:

2 known operands, 1 known operator.

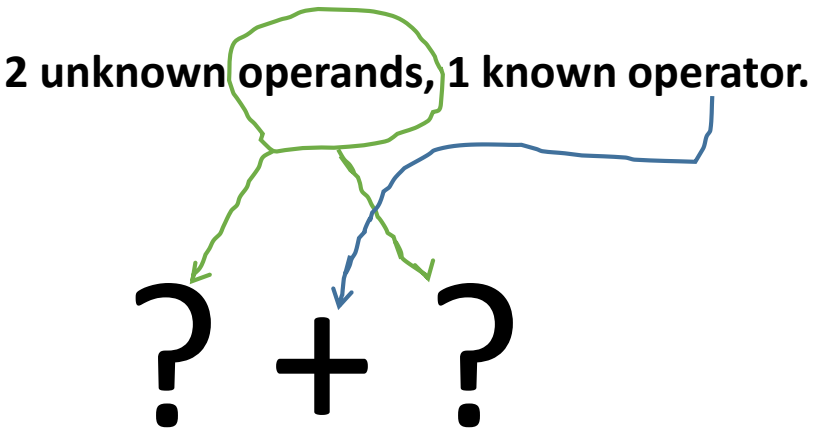


Possible Java program

```
int result = 0;

result = 4 + 2;
System.out.println("Result: " + result);
```

Arithmetic calculation:



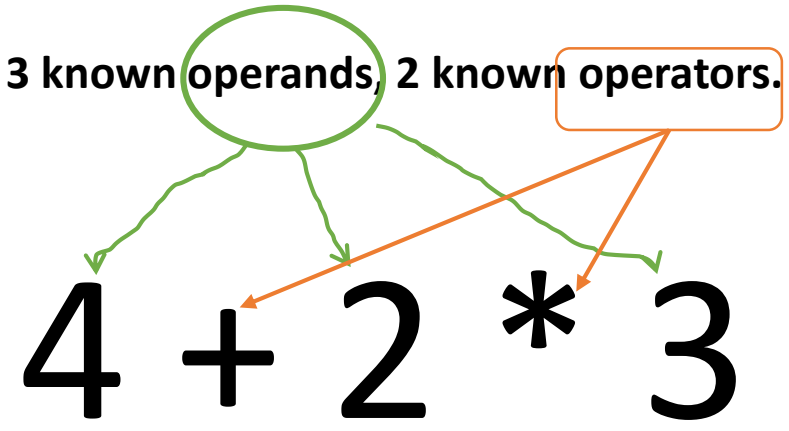
Algorithm
Get the value of 2 operands
Get the result from the calculation
Print the result

Possible Java program

```
int a, b, result = 0;

Scanner sc = new Scanner (System.in);
a= sc.nextInt();
b= sc.nextInt();
result = a + b;
System.out.println("Result: " + result);
```

Arithmetic calculation:



Algorithm
Get the result from the calculation
Print the result

Possible Java program

```
int result = 0;

result = 4 + 2 * 3 ;
System.out.println("Result: " + result);
```

Arithmetic calculation:

3 unknown operands, 2 known operators.

Algorithm
Get the value of the three operands
Get the result from the calculation
Print the result

Possible Java program

```
int a, b, c, result = 0;

Scanner sc = new Scanner (System.in);
a= sc.nextInt();
b= sc.nextInt();
c= sc.nextInt();
result = a + b * c ;
System.out.println("Result: " + result);
```

Arithmetic calculation:

3 unknown operands, 2 unknown operator.

? op ? op ?

Possible Java program

?

Arithmetic calculation:

3 unknown operands, 2 unknown operator.

? op ? op ?

We might think off this algorithm:

Get the value of the three operands a, b, c
Get the type of operators op1, op2
Get the result from for the 1st operation, value ← a op1 b
Get the result from the 2nd operation, result ← value op2 c
Print the result

Possible Java program

```
int a, b, c, value=0, result = 0;
char op1, op2;
Scanner sc = new Scanner (System.in);
a= sc.nextInt();
op1 = sc.next().charAt(0);
b= sc.nextInt();
op2= sc.next().charAt(0);
c= sc.nextInt();
switch (op1){
case '+': value = a + b; break;
case '-': value = a - b; break;
case '*': value = a * b; break;
case '/': value = a / b; break;
}
System.out.println("Value: " + value);
switch (op2) {
case '+': result = value + c; break;
case '-': result = value - c; break;
case '*': result = value * c; break;
case '/': result = value / c; break;
}
System.out.println("Result: " + result);
```

Problem!!

	Algorithm	Math
$4 + 2 * 3 = ?$	**18**	**10**
$4 * 2 + 3 = ?$	**11**	**11**

Using the algorithm, we will get wrong answer

Arithmetic calculation:

? op ? op ? op ? op ? ….

Use stack to handle the operator precedence

1. Get input: Infix expression.
2. Convert Infix → Postfix.
3. Calculate result: Postfix expression
4. Output result.

Example:

Input infix exp: 4 + 2 * 3

Convert infix-postfix: 4 + 2 * 3 → 4 2 3 * +

Calculate postfix exp: = 4 2 3 * +

= 4 6 +

= 10

Output: 10