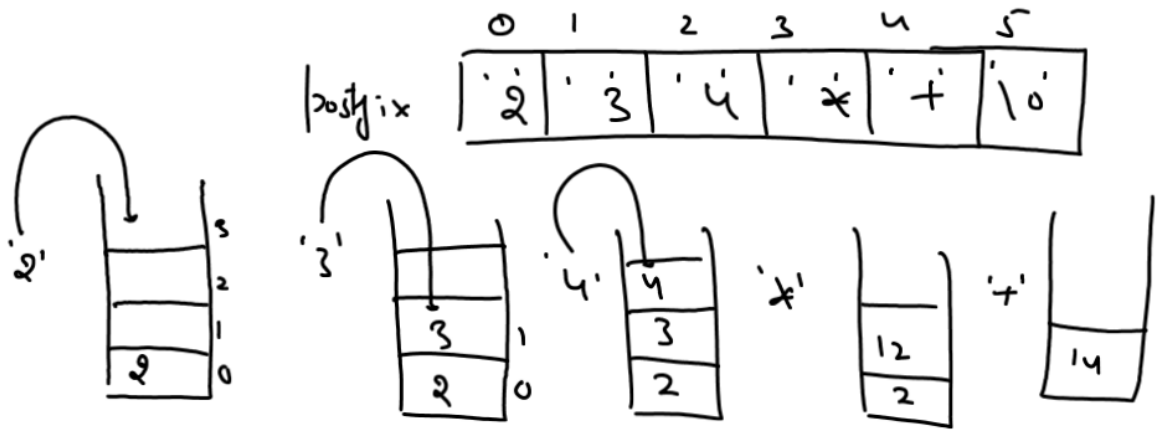
Algorrtihm For Evaluating a Postfix Expression



5 + 6 / 2 - 1 * 4 + 7

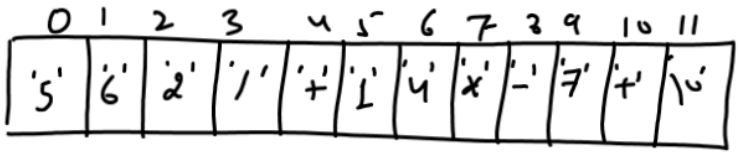
5 + 6 2 / - 1 * 4 + 7

5 + 6 2 / - 1 4 * + 7

5 6 2 / + - 1 4 * + 7

5 6 2 / + 1 4 * - + 7

5 6 2 / + 1 4 * - 7 +



Ele	Stack
'5'	5
'6'	5, 6
'2'	5, 6, 2
'/'	5, 3
'+'	8

Ele	Stack
'1'	8, 1
'4'	8, 1, 4
'*'	8, 4
'-'	4
'7'	4, 7
'+'	11
end	11 ← Ans

Algorrtihm For Evaluating a Postfix Expression

1. Scan the given postfix expression from left to right , one character at a time.
2. Check wherther it is an operator or operand.

3. If it is an operand, then PUSH it in the Stack and goto Step 5

4. If it is an operator then:
    a. POP the top 2 operands from the Stack
    b. Apply the operator on them
    c. PUSH the result back in the Stack

5. Repeat steps from 1 to 4 until POSTFIX expression ends.

6. Finally , POP the remaining last element from the Stack and return it as the answer of the expression

7.Finish

Implementing Evaluation Of Postfix Expression

```
struct Stack
{
    float arr[10];
    int tos;
};
void push(struct Stack *,float);
float pop(struct Stack *);
int isoperand(char);
float calculate(float,float,char);
float evaluate(char [ ]);
int main()
{
char postfix[20];
float ans;
printf("Enter valid postfix expression:");
scanf("%s",postfix);
ans=evaluate(postfix);
printf("Result is %f",ans);
return 0;
}
```

```
float evaluate(char postfix[20])
{
  struct Stack S;
  int i;
  char ch;
  float op1,op2,result;
  S.tos=-1;
 for(i=0;postfix[i]!='\0';i++)
{
     ch=postfix[i];
     // continue the code
```