PROLOG ACADEMY

- **DATA STRUCTURE**

By- MASEERA ALI

- Book followed - Data structures by Seymour Lipschutz (Schaum Series)

LET’S START!

INFIX TO POSTFIX

Suppose Q is an arithmetic expression written in infix notation.This algo finds the equivalent postfix expression P.

1.Push “(“ on to the stack and add “)” to the end of Q.

2.Scan Q from left to right and repeat Step 3 to 6 for each element of Q until the stack is empty.

3.if an operand is encountered, add it to P.

4.If a left parenthesis is encountered, push it onto the stack.

5.If an operator is encountered,then :

a. Repeatedly pop from STACK and add to P each operator(on the top of stack) which has the same precedence as or higher precedence than your operator

b.Add operator to stack.

[End of if structure]

6.If a right parenthesis is encountered then:

a.Reapedly pop from stack and Add to P each operator (on the top of stack) until a left parenthesis is encountered.

b.Remove the left parenthesis.[Do not add the left paenthesis to P]

[End of if Structure]

[End of step2 loop].

7.Exit.

CODE IN C

```
#include<stdio.h>
#include<string.h>

char stack [30];
int top = -1;

void push (char sym)
{
        if (top == 29)
        {
                printf ("\n stack is overflow");
                return;
        }
        top++;
        stack[top]=sym;
}

char pop ( )

{

char i;

if (top ==-1)

{

printf ("\n stack is empty");

return(0);

}
        i=stack [top];
        top--;
        return (i);
}

int prec (char ch)
{
        if (ch=='^')
                return (5);
        if (ch=='*' || ch=='/')
                return (4);
        if (ch=='+' || ch=='-')
                return (3);
        else
                return (2);
}
```

```
void infix_to_postfix (char infix [ ])
{
        int length;
       int index = 0, pos= 0;
        char symbol, temp;
        char postfix [50];
        length = strlen(infix);
        infix[length]=')';
        push('(');
        while (index <= length)
        {
                symbol = infix[index];
                switch (symbol)
                {
                        case '(' : push (symbol);
                                break;
                        case ')' : temp = pop ( );
                                   while (temp != '(')
                                   {
                                        postfix [pos]=temp;
                                        pos++;
                                        temp=pop ();
                                   }
                                break;
                        case '+' :
                        case '-' :
                        case '*' :
                        case '/' :
                        case '^' :
                                while (prec (stack[top]) >= prec (symbol))
                                {
                                        temp = pop();
                                        postfix [pos]= temp;
                                        pos++;
                                }
                                push (symbol);
                                break;
                        default:postfix [pos++] = symbol;
                                break;
                }
                index++;
        }
        postfix [pos++] = '\0';
        printf("\nEquivalent postfix expression is:\n");
        puts (postfix);
        return;
}
```

```
void main ( )
{
        char infix [30];
        printf ("\n Enter the infix expression:\n");
        gets (infix);
        infix_to_postfix (infix);
}
```

EVALUATION OF POSTFIX

This algo finds the Value of the arithematic expresssion P written in postfix notation.

1.Add right parenthesis “)” at the end of P

2.Scan P from the right and repeat step 3 and 4 for each element of P until “)” is encountered.

3.If an operand is encountered, Put iton stack.

4.If an operator is encountered, then:

a.Remove the top two elements of stack,where A is the top element and B is the next to top element.

b.Evaluate B op. A.

c.Place the result of (b) back on stack.

[End of If structure]

[End of step 2 loop]

5.Set Value equal to the top element on stack.

6.Exit

CODE IN C

```
#include<stdio.h>

#include<string.h>

#include<math.h>

int stack [30];

int top = -1;

void push (int num)

{

   if (top == 29)

{

  printf ("\n stack is overflow");

   return;

}

  top++;

  stack[top]=num;

}
```

```
   int pop ( ){

     int i;

     if (top ==-1){

    printf ("\n stack is empty");

    return(0);

   }     i=stack [top];
         top--;
         return (i);
}
void main( )
{
         char S[80];
         int i, n, x=0, y=0;
         printf ("\n Enter the valid postfix notation :");
         gets(S);
         n = strlen(S);
         printf ("The value of the postfix notation is :");
         for (i=0;i<n; i++)
         {
                  switch (S[i])
                  {
                           case '+' :y=pop();
                                    x=pop();
                                    x=x+y;
                                    push(x);
                                    break;
```

```
                case '-' :y=pop();
                          x=pop();
                          x=x-y;
                          push(x);
                          break;
                 case '*' :y=pop();
                          x=pop();
                          x=x*y;
                          push(x);
                          break;
                 case '/' :y=pop();
                          x=pop();
                          x=x/y;
                          push(x);
                          break;
                 case '^' :y=pop();
                          x=pop();
                          x=pow(x,y);
                          push(x);
                          break;


            default :if (S[i]>=48 && S[i]<=57)
                          push(S[i]-48);

        }
    }
    printf("%d\n", pop());
}
```