**Exercise 4: Implement a program to convert infix notation to postfix notation using stack.**

#include<stdio.h>

#include<stdlib.h>

#include<ctype.h>

#include<string.h>

#define SIZE 100

char stack[SIZE];

int top = -1;

void push(char item)

{

if(top >= SIZE-1)

{

printf("\nStack Overflow.");

}

else

{

top = top+1;

stack[top] = item;

}

}

char pop()

{

char item ;

if(top <0)

{

printf("stack under flow: invalid infix expression");

getchar();

exit(1);

}

else

{

item = stack[top];

top = top-1;

return(item);

}

}

int is\_operator(char symbol)

{

if(symbol == '^' || symbol == '\*' || symbol == '/' || symbol == '+' || symbol =='-')

{

return 1;

}

else

{

return 0;

}

}

int precedence(char symbol)

{

if(symbol == '^')

{

return(3);

}

else if(symbol == '\*' || symbol == '/')

{

return(2);

}

else if(symbol == '+' || symbol == '-')

{

return(1);

}

else

{

return(0);

}

}

void InfixToPostfix(char infix\_exp[], char postfix\_exp[])

{

int i, j;

char item;

char x;

push('(');

strcat(infix\_exp,")");

i=0;

j=0;

item=infix\_exp[i];

while(item != '\0')

{

if(item == '(')

{

push(item);

}

else if( isdigit(item) || isalpha(item))

{

postfix\_exp[j] = item;

j++;

}

else if(is\_operator(item) == 1)

{

x=pop();

while(is\_operator(x) == 1 && precedence(x)>= precedence(item))

{

postfix\_exp[j] = x;

j++;

x = pop();

}

push(x);

push(item);

}

else if(item == ')')

{

x = pop();

while(x != '(')

{

postfix\_exp[j] = x;

j++;

x = pop();

}

}

else

{

printf("\nInvalid infix Expression.\n");

getchar();

exit(1);

}

i++;

item = infix\_exp[i];

}

if(top>0)

{

printf("\nInvalid infix Expression.\n");

getchar();

exit(1);

}

if(top>0)

{

printf("\nInvalid infix Expression.\n");

getchar();

exit(1);

}

postfix\_exp[j] = '\0';

}

int main()

{

char infix[SIZE], postfix[SIZE];

printf("\nEnter Infix expression : ");

gets(infix);

InfixToPostfix(infix,postfix);

printf("Postfix Expression: ");

puts(postfix);

return 0;

}

Output:

**Infix and Postfix Conversion using array**

#include <stdio.h>

//#include <ctype.h>

#define SIZE 50

char stack[SIZE];

int top=-1;

push(char elem)

{

stack[++top]=elem;

}

char pop()

{

return(stack[top--]);

}

int pr(char symbol)

{

if(symbol == '^')

{

return(3);

}

else if(symbol == '\*' || symbol == '/')

{

return(2);

}

else if(symbol == '+' || symbol == '-')

{

return(1);

}

else

{

return(0);

}

}

void main()

{

char infix[50],postfix[50],ch,elem;

int i=0,k=0;

printf("Enter Infix Expression : ");

scanf("%s",infix);

push('#');

while( (ch=infix[i++]) != '\0')

{

if( ch == '(') push(ch);

else

if(isalnum(ch)) postfix[k++]=ch;

else

if( ch == ')')

{

while( stack[top] != '(')

postfix[k++]=pop();

elem=pop(); /\* Remove ( \*/

}

else

{ /\* Operator \*/

while( pr(stack[top]) >= pr(ch) )

postfix[k++]=pop();

push(ch);

}

}

while( stack[top] != '#')

postfix[k++]=pop();

postfix[k]='\0';

printf("\nPostfix Expression = %s\n",postfix);

}