POSTFIX TO PREFIX

Algorithm:- 1. Declare set of operators. 2. Initialize an empty stack. 3. To convert INFIX to POSTFIX follow the following steps 4. Scan the infix expression from left to right. 5. If the scanned character is an operand, output it. 6. Else, If the precedence of the scanned operator is greater than the precedence of the operator in the stack(or the stack is empty or the stack contains a ‘(‘ ), push it. 7. Else, Pop all the operators from the stack which are greater than or equal to in precedence than that of the scanned operator. After doing that Push the scanned operator to the stack. 8. If the scanned character is an ‘(‘, push it to the stack. 9. If the scanned character is an ‘)’, pop the stack and output it until a ‘(‘ is encountered, and discard both the parenthesis. 10. Pop and output from the stack until it is not empty. 11. To convert INFIX to PREFIX follow the following steps 12. First, reverse the infix expression given in the problem. 13. Scan the expression from left to right. 14. Whenever the operands arrive, print them. 15. If the operator arrives and the stack is found to be empty, then simply push the operator into the stack. 16. Repeat steps 6 to 9 until the stack is empty

CODE:-

#include<stdio.h>

#include<conio.h>

#include<string.h>

#include<stdlib.h>

#define MAX 20

char str[MAX], stack[MAX];

int top = -1;

void push(char c)

{

stack[++top] = c;

}

char pop()

{

return stack[top--];

}

// A utility function to check if the given character is operand

int checkIfOperand(char ch)

{

return (ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z');

}

//function to check if it is an operator

int isOperator(char x)

{

switch (x) {

case '+':

case '-':

case '/':

case '\*':

return 1;

}

return 0;

}

void postfixToprfix()

{

int n, i, j = 0;

char c[20];

char a, b, op;

printf("Enter the postfix expression:\n");

scanf("%s", str);

n = strlen(str);

for (i = 0; i < MAX; i++)

stack[i] = '\0';

printf("Prefix expression is:\t");

for (i = n - 1; i >= 0; i--)

{

if (isOperator(str[i]))

{

push(str[i]);

} else

{

c[j++] = str[i];

while ((top != -1) && (stack[top] == '#'))

{

a = pop();

c[j++] = pop();

}

push('#');

}

}

c[j] = '\0';

i = 0;

j = strlen(c) - 1;

char d[20];

while (c[i] != '\0') {

d[j--] = c[i++];

}

printf("%s\n", d);

}

int main()

{

postfixToprfix();

return 0;

}

OUTPUT:-

ENTER THE POSTFIX EXPRESSION:

AB^CR/+

PREFIX EXPRESSION IS:

B+^/CR