Rajalakshmi Engineering College

Name: RANNESH KHUMAR B R
Email: 240701422@rajalakshmi.edu.in
Roll no: 2116240701
Phone: 9042350670
Branch: REC
Department: I CSE FD
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 3_CY

Attempt : 1
Total Mark : 30
Marks Obtained : 30

Section 1 : Coding

1. Problem Statement

Suppose you are building a calculator application that allows users to enter mathematical expressions in infix notation. One of the key features of your calculator is the ability to convert the entered expression to postfix notation using a Stack data structure.

Write a function to convert infix notation to postfix notation using a Stack.

Input Format

The input consists of a string, an infix expression that includes only digits(0-9), and operators(+, -, *, /).

Output Format

The output displays the equivalent postfix expression of the given infix expression.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1+2*3/4-5

Output: 123*4/+5-

Answer

```
// You are using GCC
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<ctype.h>
#define max 100

typedef struct{
   int top;
   char items[max];
}Stack;

void ini(Stack*stack){
   stack->top=-1;
}

int isEmpty(Stack*stack){
   return stack->top==-1;
}
int isFull(Stack*stack){
   return stack->top==max-1;
}

void push(Stack*stack,char item){
   if(stack->top<max-1){
      stack->items[++stack->top]=item;
   }
}

char pop(Stack*stack){
   if(!isEmpty(stack)){
```

```
        return stack->items[stack->top--];
    }
    return '\0';
}
char peek(Stack*stack){
    if(!isEmpty(stack)){
        return stack->items[stack->top];
    }
    return '\0';
}
int prec(char op){
    switch(op){
        case'+':
        case'-':
            return 1;
        case'/':
        case'*':
            return 2;
        case'^':
            return 3;
        default:
            return 0;
    }
}

void infixtopost(char*infix,char*postfix){
    Stack stack;
    ini(&stack);
    int j=0;

    for(int i=0;infix[i];i++){
        char c=infix[i];
        if(isalnum(c)){
            postfix[j++]=c;
        }
        else if(c=='('){
            push(&stack,c);
        }
        else if(c==')'){
            while(!isEmpty(&stack)&&peek(&stack)!='('){
                postfix[j++]=pop(&stack);
            }
```

```
            pop(&stack);
        }
        else{
            while(!isEmpty(&stack)&&prec(peek(&stack))>= prec(c)){
                postfix[j++]=pop(&stack);
            }
            push(&stack,c);
        }
    }

    while(!isEmpty(&stack)){
        postfix[j++]=pop(&stack);
    }
    postfix[j]='\0';
}

int main(){
    char infix[max];
    char postfix[max];

    printf(" ");
    fgets(infix,sizeof(infix),stdin);
    infix[strcspn(infix,"\n")]=0;
    infixtopost(infix,postfix);
    printf("%s\n",postfix);
}
```

Status : Correct **Marks : 10/10**

2. Problem Statement

Latha is taking a computer science course and has recently learned about infix and postfix expressions. She is fascinated by the idea of converting infix expressions into postfix notation. To practice this concept, she wants to implement a program that can perform the conversion for her.

Help Latha by designing a program that takes an infix expression as input

and outputs its equivalent postfix notation.

Example

Input:

(3+4)5

Output:

34+5

Input Format

The input consists of a string, the infix expression to be converted to postfix notation.

Output Format

The output displays a string, the postfix expression equivalent of the input infix expression.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: A+B*C-D/E
Output: ABC*+DE/-

Answer

```
// You are using GCC
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<ctype.h>
#define max 100

typedef struct{
    int top;
    char items[max];
} Stack;
void ini(Stack *stack){
```

```
    stack->top=-1;
}
int isEmpty(Stack *stack){
    return stack->top ==-1;
}

void push(Stack*stack,char item){
    if(stack->top <max-1){
        stack->items[++stack->top]=item;
    }
}

char pop(Stack*stack){
    if(!isEmpty(stack)){
        return stack->items[stack->top--];
    }
    return '\0';
}

char peek(Stack *stack){
    if(!isEmpty(stack)){
        return stack->items[stack->top];
    }
    return'\0';
}

int pre(char op){
    switch(op){
        case'+':case'-': return 1;
        case'*':case'/':return 2;
        default: return 0;
    }
}

void ItoP(char*infix,char*postfix){
    Stack stack;
    ini(&stack);
    int j=0;

    for(int i=0;infix[i];i++){
        char c=infix[i];
        if(isalnum(c)){
```

```
            postfix[j++]=c;
        }
        else if(c=='('){
            push(&stack,c);
        }
        else if(c==')'){
            while(!isEmpty(&stack)&&peek(&stack)!='('){
                postfix[j++]=pop(&stack);
            }
            pop(&stack);
        }
        else{
            while(!isEmpty(&stack)&&pre(peek(&stack))>= pre(c)){
                postfix[j++]=pop(&stack);}
        push(&stack,c);
        }
    }
    while(!isEmpty(&stack)){
        postfix[j++]=pop(&stack);
    }
    postfix[j]='\0';
}

int main(){
char infix[max];
char postfix[max];
printf("");
fgets(infix,sizeof(infix),stdin);
infix[strcspn(infix,"\n")]=0;
ItoP(infix,postfix);
printf("%s\n",postfix);
}
```

Status : Correct **Marks : 10/10**

3. Problem Statement

In an educational setting, Professor Smith tasks Computer Science students with designing an algorithm to evaluate postfix expressions efficiently, fostering problem-solving skills and understanding of stack-based computations.

The program prompts users to input a postfix expression, evaluates it, and displays the result, aiding students in honing their coding abilities.

Input Format

The input consists of the postfix mathematical expression.

The expression will contain real numbers and mathematical operators (+, -, *, /), without any space.

Output Format

The output prints the result of evaluating the given postfix expression.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 82/

Output: 4

Answer

```
// You are using GCC
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<ctype.h>
#include<math.h>
#define max 100

int stack[max];
int top=-1;

void push(int val){
   if(top<=max-1){
      stack[++top]=val;
   }
}

int pop(){
```

```
    if(top!=-1){
        return stack[top--];
    }
}

int isop(char ch){
    return ch=='+'||ch=='-'||ch=='*'||ch=='/';
}

int eval(char*exp){
    for(int i=0;exp[i]!='\0';i++){
        char ch=exp[i];
        if(isdigit(ch)){
            push((int)(ch-'0'));
        }
        else if(isop(ch)){
            int val2=pop();
            int val1=pop();
            int res;

            switch(ch){
                case'+':
                    res=val1+val2;
                    break;
                case'-':
                    res=val1-val2;
                    break;
                case'*':
                    res=val1*val2;
                    break;
                case'/':
                    res=val1/val2;
                    break;
            }
            push(res);
        }
    }
    return pop();
}

int main(){
    char exp[max];
```

```
    scanf("%s",exp);
    int result=eval(exp);
    printf("%d\n",result);
}
```

Status : Correct ***Marks : 10/10***