Week-4 UE20CS207 DSLAB

- Name : P K Navin Shrinivas
- SRN : PES2UG20CS237
- Section : D
- Batch : 2

Lab problem 1 : Infix to postfix , postfix evaluvation

Code :

main.c

```
#include"1_1.h"

int prec(char element)
{
    if(element=='^')
        return 3;
    else if(element=='*' || element=='/')
        return 2;
    else
        return 1;
}

void inpo(char* stack , char* input , int* top)
{
    for(int i=0;i<strlen(input);i++)
    {
        if((input[i]>='a' && input[i]<='z') || (input[i]>='A' && input[i]<='Z'))
        {
            printf("%c",*(input+i));
        }
        else if(input[i]=='(')
            stackpush(stack , '(' , top);
        else if(input[i]==')')
        {
            while(stackpeek(stack , top)!='(')
            {
                printf("%c",stackpeek(stack , top));
                stackpop(stack , top);
            }
            stackpop(stack ,top);
        }
        else if(*top==-1 || prec(input[i])>prec(stackpeek(stack , top)))
        {
            stackpush(stack , input[i],top);
        }
        else{
            while(*top!=-1 && prec(input[i])<=prec(stackpeek(stack ,top)))
            {
                printf("%c",stackpeek(stack, top));
                stackpop(stack , top);
            }
            stackpush(stack,input[i],top);
        }
    }
    while(*top!=-1)
    {
        printf("%c",stackpeek(stack , top));
        stackpop(stack,top);
    }
}

void poeval(char* stack , char* input , int* top)
{
```

```
    for(int i=0;i<strlen(input);i++)
    {
        if((int)input[i]-48>=0 && (int)input[i]-48<=9)
            stackpush(stack,input[i],top);
        else{
            int temp=0;
            switch(input[i])
            {
                case '+':
                {
                    temp=(int)stackpeek(stack,top)-48;
                    stackpop(stack,top);
                    temp+=(int)stackpeek(stack,top)-48;
                    stackpop(stack,top);
                    stackpush(stack,(char)(temp+48),top);
                    break;
                }
                case '-':
                {
                    int copy=(int)stackpeek(stack,top)-48;
                    stackpop(stack,top);
                    int copy2=(int)stackpeek(stack,top)-48;
                    temp=copy2-copy;
                    stackpop(stack,top);
                    stackpush(stack,(char)(temp+48),top);
                    break;
                }
                case '*':
                {
                    temp=(int)stackpeek(stack,top)-48;
                    stackpop(stack,top);
                    temp*=(int)stackpeek(stack,top)-48;
                    stackpop(stack,top);
                    stackpush(stack,(char)(temp+48),top);
                    break;
                }
                case '/':
                {
                    temp=(int)stackpeek(stack,top)-48;
                    stackpop(stack,top);
                    temp/=(int)stackpeek(stack,top)-48;
                    stackpop(stack,top);
                    stackpush(stack,(char)(temp+48),top);
                    break;
                }
            }
        }
    }
}


int main(){
    char stack[STACKSIZE];
    int top=-1;
    printf("Enter infix express ion : ");
    char input[100];
    scanf("%s",input);
    inpo(stack , input , &top);
    printf("\n");
    printf("Enter valid postfix to eval : ");
    char input2[100];char stack2[STACKSIZE];int top2=-1;
    scanf("%s",input2);
    poeval(stack2 , input2 , &top2);
    printf("Value of above expression : %d\n",(int)stackpeek(stack2,&top2)-48);

}
```

1_1.h

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

```c
#include<string.h>

#define STACKSIZE 100
void stackpush(char* stack , char element , int* top);
char stackpeek(char* stack , int* top);
void stackpop(char* stack , int* top);
```

1_1.c

```c
#include "1_1.h"

void stackpush(char* stack , char element , int* top)
{
    if(*top==STACKSIZE-1)
    {
       printf("Stack is full");
       return;
    }
    else{
        *top+=1;
        *(stack+*top)=element;
    }
}
char stackpeek(char* stack , int* top)
{
    return *(stack+*top);
}
void stackpop(char* stack , int* top)
{
    *top-=1;
    return;
}
```

Screenshots :