Assignment No. 2

Name: Bhavin Ratansing Patil

Roll No.: 26 SEDA

Write a program to convert a given Prefix expression into its equivalent Infix expression and evaluate it using stack.

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

    void push(char);
    char pop();
    void printStack();
    int checkOperand(char);

    //global variable declaration
    char stack[20] = {'\0'}; //declaring array with initializing with null
value.
    int start=-1;
    char character;

    int main(){

        char prefix[20]={'\0'};
        char element;
        printf("\nEnter Prefix Expression: ");
        scanf("%s",prefix);
        int i = strlen(prefix)-1;
        while(i>=0)
        {
            element = prefix[i];
            if(checkOperand(element))
            {
                push(element);
                push(' ');
                printf("\n element: %c Pushed to Stack",element);
            }
            else
            {
                printf("\nOperator: %c",element);
                for(int j = start-1; j>=0; j--)
                {
                    if(stack[j]==' ')
                    {
                        stack[j] = element;
```

```
                    break;
                }
            }
        }
        i--;
        printStack();
        printf("\n_____________________________");
    }
    printf("\n Entered Prefix Expression is : %s",prefix);
    printf("\n Infix Expression will be : %s",strrev(stack));

    return 0;
}
void push(char element){
    if(start==19)
    {
        printf("Stack is Full");
    }
    else
    {
        start++;
        stack[start]=element;
    }
}

char pop()
{
    if(start==-1)
    {
        printf("Stack is Empty");
    }
    else
    {
        character=stack[start];
        start--;
    }
    return character;
}

void printStack()
{
    for(int i=0;i<=start;i++)
    {
        printf("\ns[%d] : %c ",i,stack[i]);
    }
}

int checkOperand(char element)
```

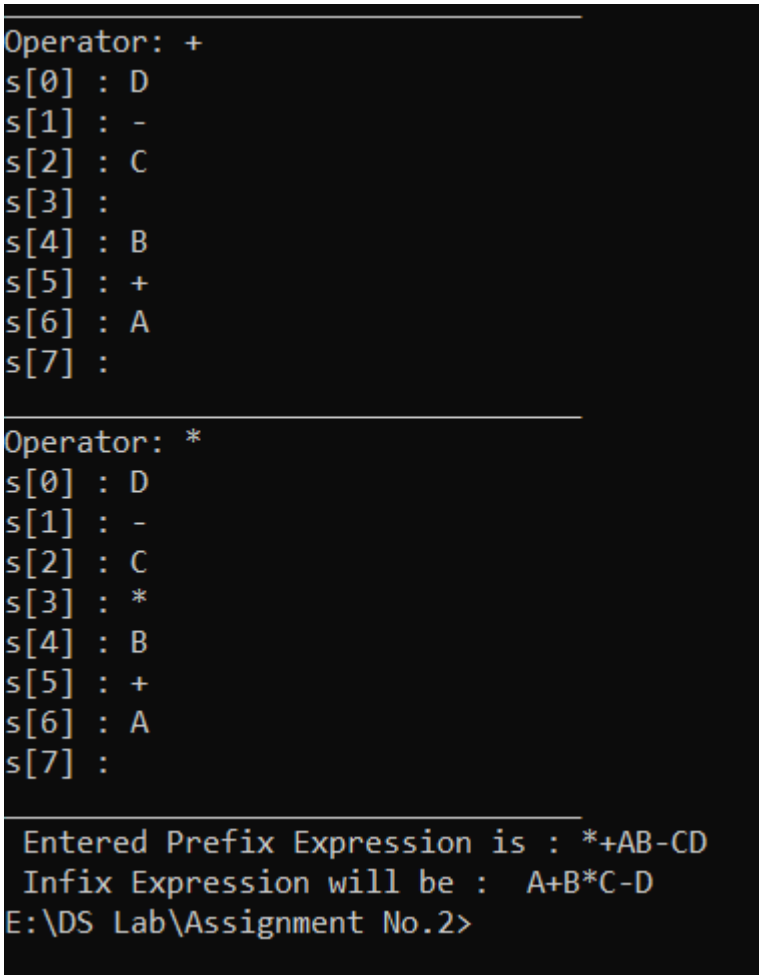
```
    {
        if((element>='A' && element<='Z')||(element>='a' && element<='z'))
        {
            return 1;
        }
        else
        {
            return 0;
        }
    }
```

Output:

```
E:\DS Lab\Assignment No.2>AssignmentNo2

Enter Prefix Expression: *+AB-CD

 element: D Pushed to Stack
s[0] : D
s[1] :
_____________________________________
 element: C Pushed to Stack
s[0] : D
s[1] :
s[2] : C
s[3] :
_____________________________________
Operator: -
s[0] : D
s[1] : -
s[2] : C
s[3] :
_____________________________________
 element: B Pushed to Stack
s[0] : D
s[1] : -
s[2] : C
s[3] :
s[4] : B
s[5] :
_____________________________________
 element: A Pushed to Stack
s[0] : D
s[1] : -
s[2] : C
s[3] :
s[4] : B
s[5] :
s[6] : A
s[7] :
_____________________________________
Operator: +
s[0] : D
s[1] : -
s[2] : C
s[3] :
s[4] : B
s[5] : +
s[6] : A
s[7] :
```


Operator: +
s[0] : D
s[1] : -
s[2] : C
s[3] :
s[4] : B
s[5] : +
s[6] : A
s[7] :
Operator: *
s[0] : D
s[1] : -
s[2] : C
s[3] : *
s[4] : B
s[5] : +
s[6] : A
s[7] :
Entered Prefix Expression is : *+AB-CD
Infix Expression will be : A+B*C-D
E:\DS Lab\Assignment No.2>