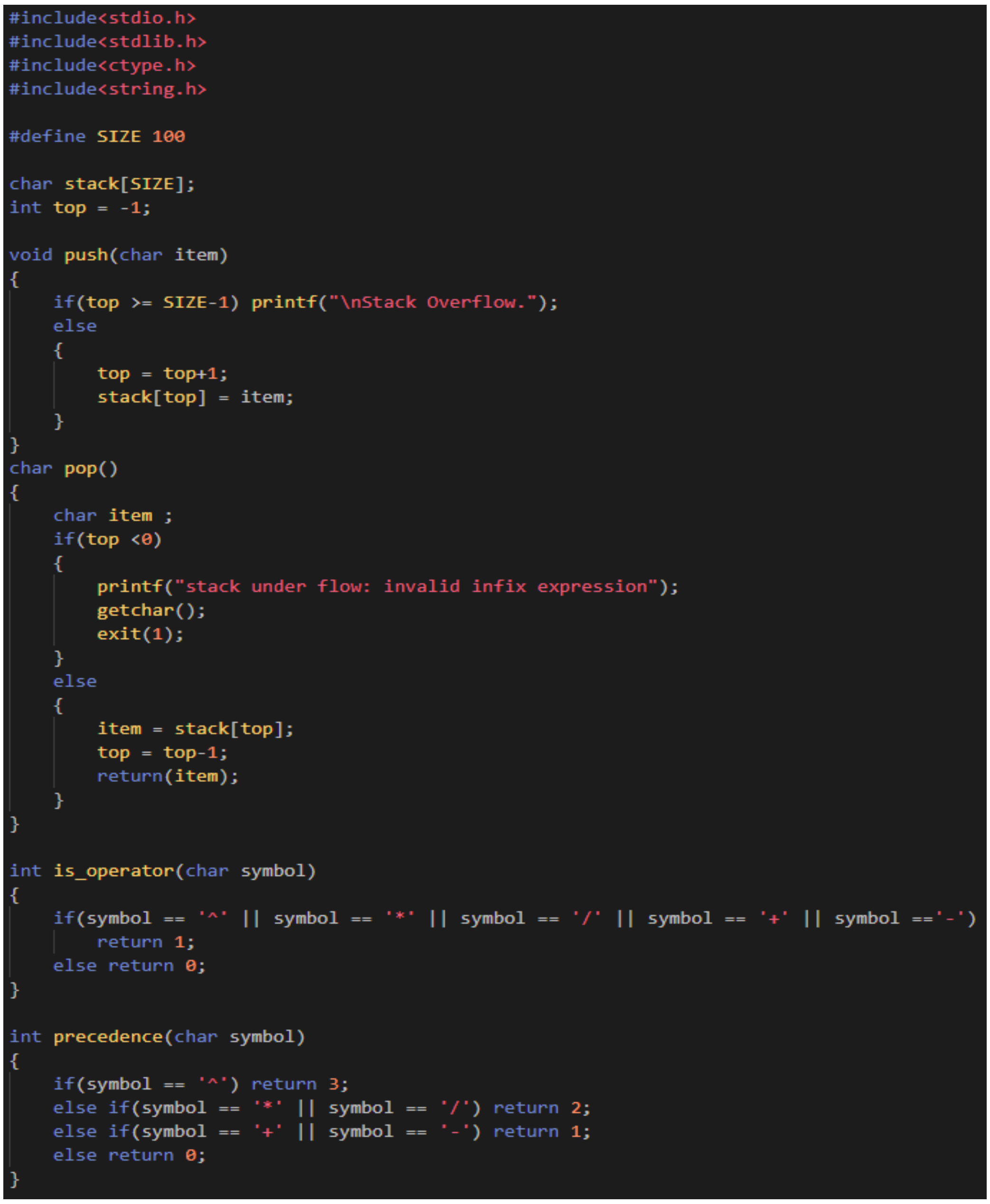
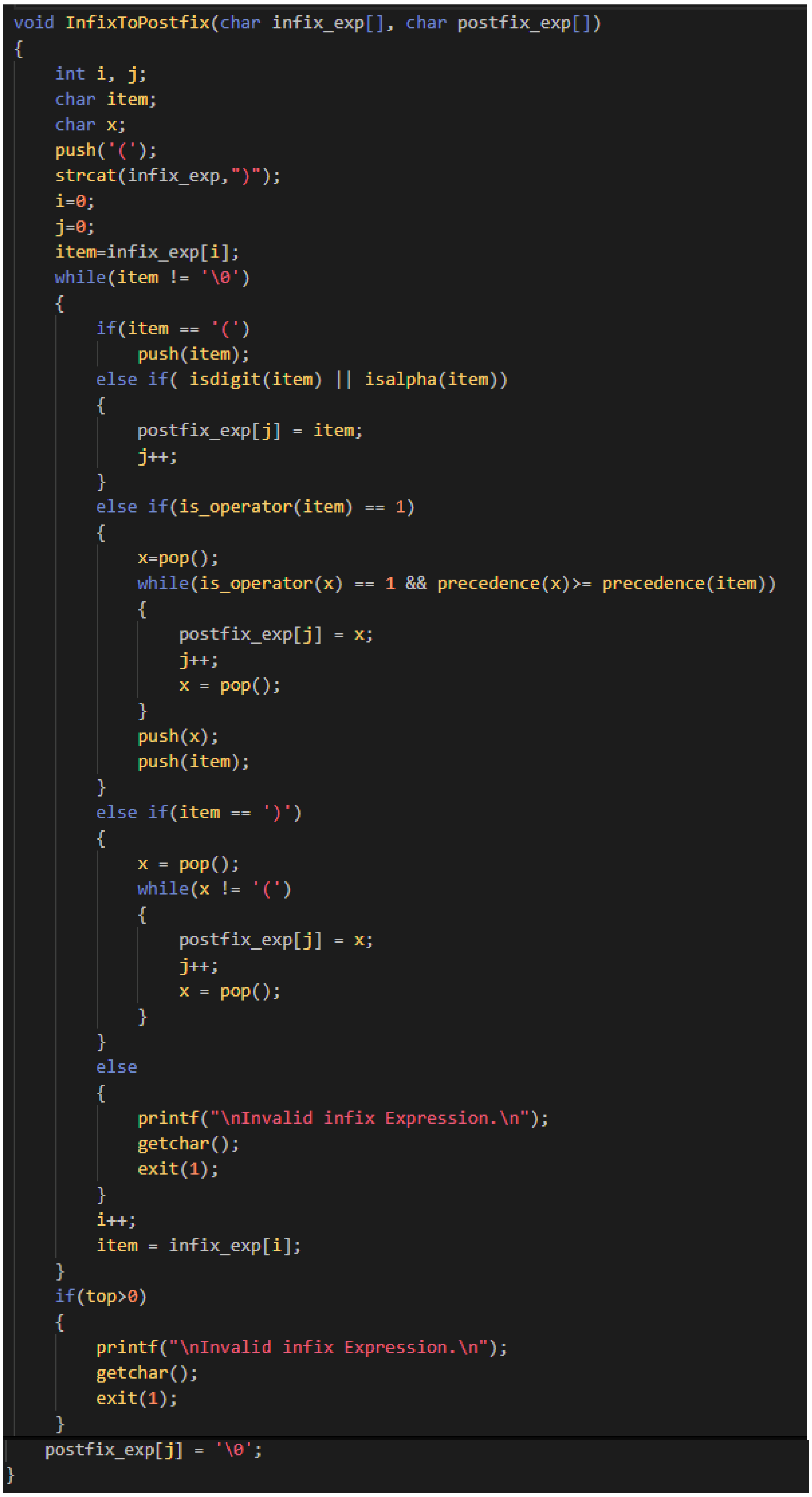
DS LAB SEM3
Name : H M Mythreya
SRN : PES2UG20CS130
Section : C
Week 4 : Stacks

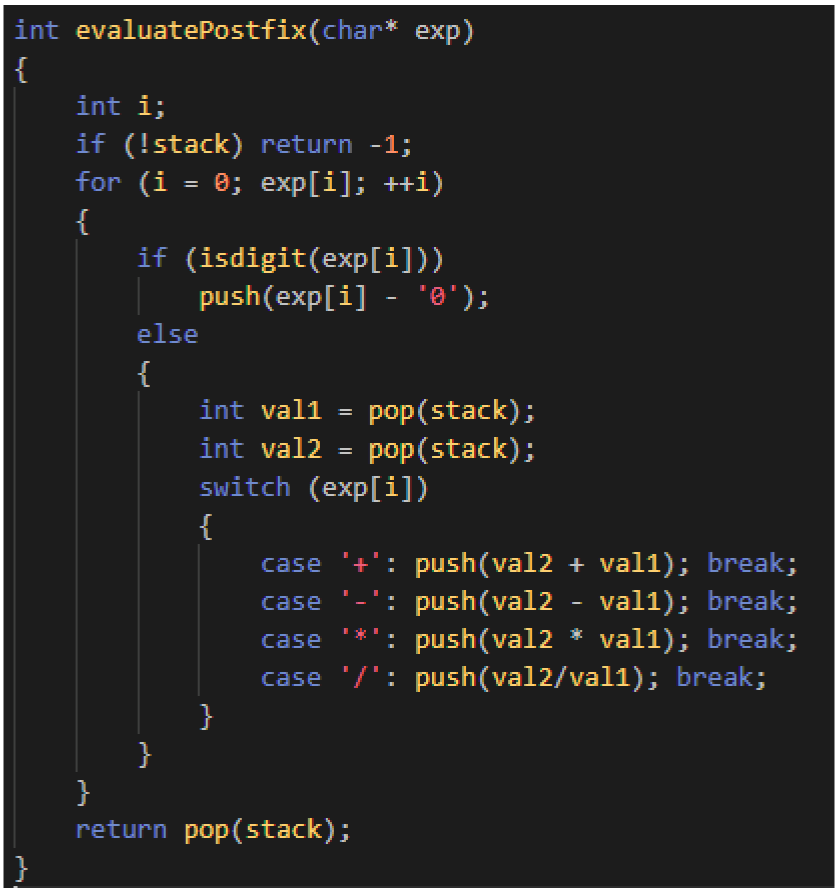
Infix to postfix, and evaluation
Code:



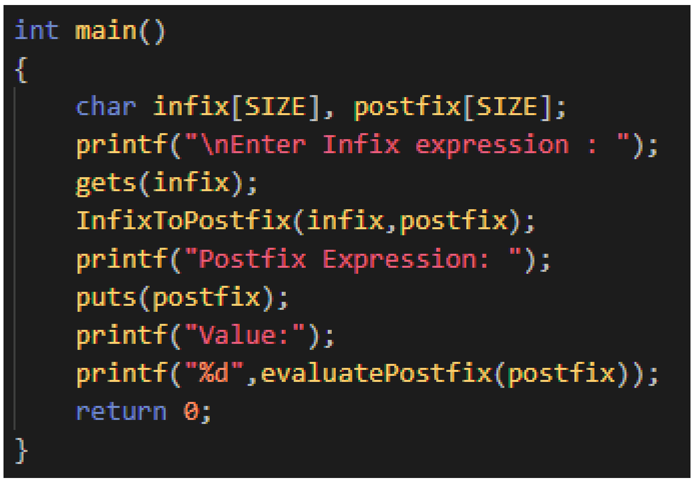
```c
#include<stdio.h>
#include<stdlib.h>
#include<ctype.h>
#include<string.h>

#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;
}
```



```c
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);
    }
    postfix_exp[j] = '\0';
}
```

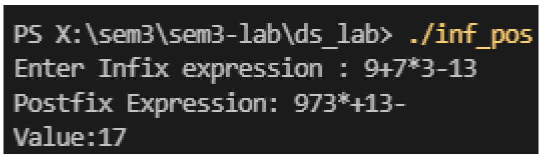


```c
int evaluatePostfix(char* exp)
{
    int i;
    if (!stack) return -1;
    for (i = 0; exp[i]; ++i)
    {
        if (isdigit(exp[i]))
            push(exp[i] - '0');
        else
        {
            int val1 = pop(stack);
            int val2 = pop(stack);
            switch (exp[i])
            {
                case '+': push(val2 + val1); break;
                case '-': push(val2 - val1); break;
                case '*': push(val2 * val1); break;
                case '/': push(val2/val1); break;
            }
        }
    }
    return pop(stack);
}
```



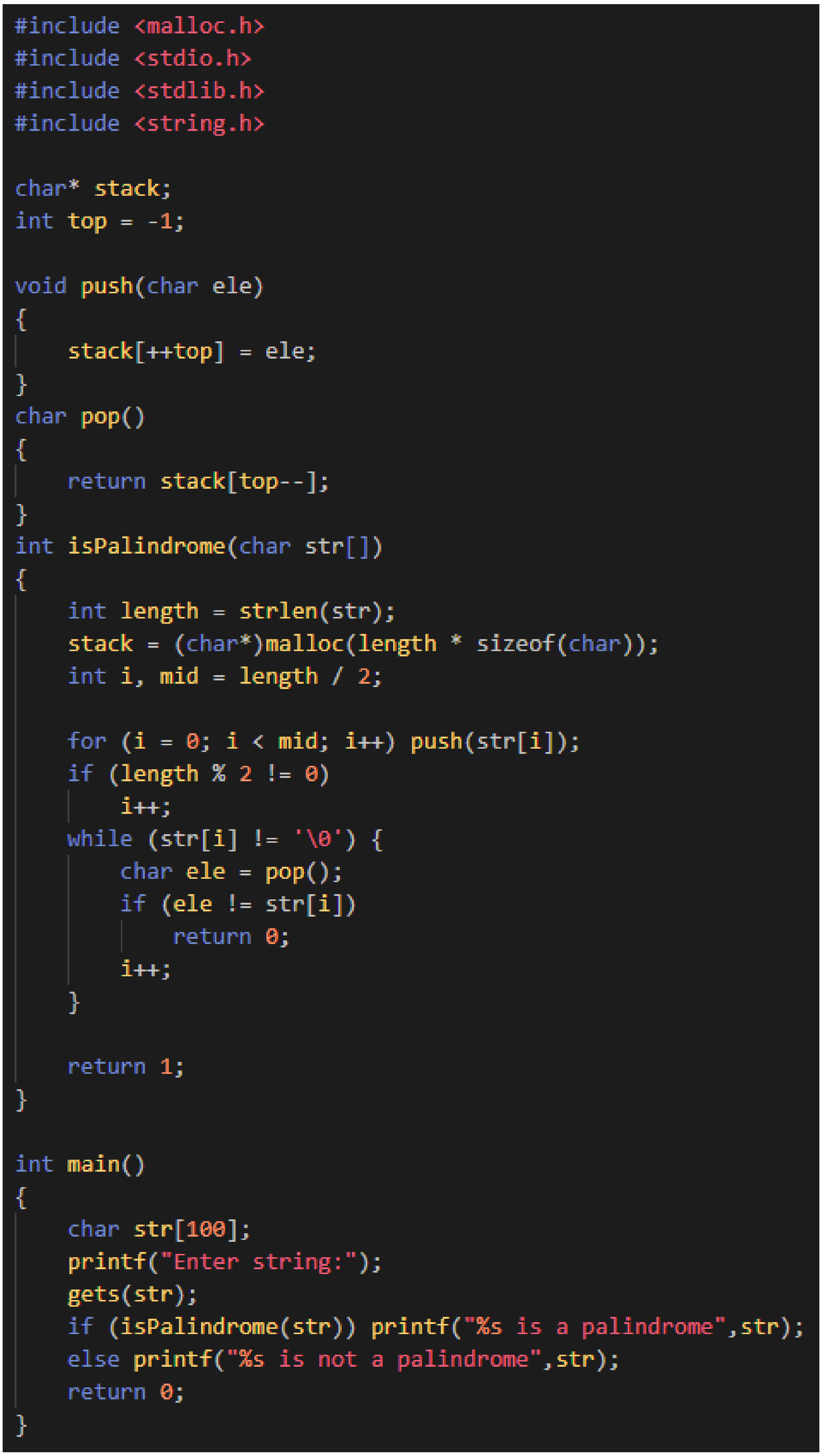
```c
int main()
{
    char infix[SIZE], postfix[SIZE];
    printf("\nEnter Infix expression : ");
    gets(infix);
    InfixToPostfix(infix,postfix);
    printf("Postfix Expression: ");
    puts(postfix);
    printf("Value:");
    printf("%d",evaluatePostfix(postfix));
    return 0;
}
```

Output:



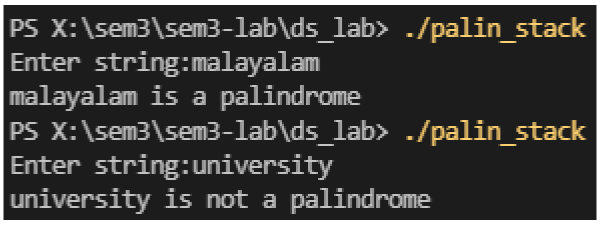
```
PS X:\sem3\sem3-lab\ds_lab> ./inf_pos
Enter Infix expression : 9+7*3-13
Postfix Expression: 973*+13-
Value:17
```

Check if string is palindrome using stack



```c
#include <malloc.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

char* stack;
int top = -1;

void push(char ele)
{
    stack[++top] = ele;
}
char pop()
{
    return stack[top--];
}
int isPalindrome(char str[])
{
    int length = strlen(str);
    stack = (char*)malloc(length * sizeof(char));
    int i, mid = length / 2;

    for (i = 0; i < mid; i++) push(str[i]);
    if (length % 2 != 0)
        i++;
    while (str[i] != '\0') {
        char ele = pop();
        if (ele != str[i])
            return 0;
        i++;
    }

    return 1;
}

int main()
{
    char str[100];
    printf("Enter string:");
    gets(str);
    if (isPalindrome(str)) printf("%s is a palindrome",str);
    else printf("%s is not a palindrome",str);
    return 0;
}
```

Output:



```
PS X:\sem3\sem3-lab\ds_lab> ./palin_stack
Enter string:malayalam
malayalam is a palindrome
PS X:\sem3\sem3-lab\ds_lab> ./palin_stack
Enter string:university
university is not a palindrome
```