Balanced parantheses using stack

Code:

#include <stdio.h>

#define MAX 100

int st[MAX],top=-1;

void push(int d){

    st[++top]=d;

}

char pop(){

    int item;

    if(top!=-1){

        item=st[top--];

        return item;

    }

}

int main(){

    char exp[MAX],\*e;

    printf("\nEnter expression:");

    scanf("%s",exp);

    e=exp;

    while(\*e!='\0'){

        if(\*e=='(')

            push(\*e);

        else if(\*e==')'){

            while(st[top]!='('){

                pop();

            }

            pop();

        }

        if(\*e=='{')

            push(\*e);

        else if(\*e=='}'){

            while(st[top]!='{'){

                pop();

            }

            pop();

        }

        if(\*e=='[')

            push(\*e);

        else if(\*e==']'){

            while(st[top]!='['){

                pop();

            }

            pop();

        }

        e++;

    }

    //If stack is empty that means parentheses are balanced

    if(top==-1){

        printf("\nBalanced parentheses");

    }

    else{

        printf("\nNot Balanced parentheses");

    }

}

Output:

Enter expression:(a+{b/c})

Balanced parentheses

Enter expression:(a+{b/c}

Not Balanced parentheses