//BALANCED PARANTHESIS

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

#define MAX 30
int top=-1;
int stack[MAX];

void push(char);
char pop();
int match(char a,char b);
int check(char []);

int main()
{
	char exp[MAX];
	int valid;
	printf("Enter an algebraic expression : ");
	gets(exp);
	valid=check(exp);
	if(valid==1)
		printf("Valid expression\n");
	else
		printf("Invalid expression\n");

		return 0;

}
int check(char exp[] )
{
	int i;
	char temp;
	for(i=0;i<strlen(exp);i++)
	{
		if(exp[i]=='(' || exp[i]=='{' || exp[i]=='[')
			push(exp[i]);
		if(exp[i]==')' || exp[i]=='}' || exp[i]==']')
			if(top==-1)	/*stack empty*/
			{
				printf("Right parentheses are more than left parentheses\n");
				return 0;
			}
			else
			{
				temp=pop();
				if(!match(temp, exp[i]))
				{
					printf("Mismatched parentheses are : ");
					printf("%c and %c\n",temp,exp[i]);
					return 0;
				}
			}
	}
	if(top==-1) /*stack empty*/
	{
		printf("Balanced Parentheses\n");
		return 1;
	}
	else
	{
		printf("Left parentheses more than right parentheses\n");
		return 0;
	}
}/*End of main()*/
int match(char a,char b)
```

```
{
        if(a=='[' && b==']')
                return 1;
        if(a=='{' && b=='}')
                return 1;
        if(a=='(' && b==')')
                return 1;
        return 0;
}/*End of match()*/

void push(char item)
{
        if(top==(MAX-1))
        {
                printf("Stack Overflow\n");
                return;
        }
        top=top+1;
        stack[top]=item;
}/*End of push()*/

char pop()
{
        if(top==-1)
        {
                printf("Stack Underflow\n");
                exit(1);
        }
        return(stack[top--]);
}/*End of pop()*/
```

// OUTPUT

```
/*

  Enter an algebraic expression : (((5+7)*6)/2)

Balanced Parentheses
Valid expression


*/
```