**Practical no 9**

**#include <iostream>**

**Using namespace std;**

**#define size 10**

**Class stackexp**

**{**

**Int top;**

**Char stk[size];**

**Public:**

**Stackexp()**

**{**

**Top=-1;**

**}**

**Void push(char);**

**Char pop();**

**Int isfull();**

**Int isempty();**

**};**

**Void stackexp::push(char x)**

**{**

**Top=top+1;**

**Stk[top]=x;**

**}**

**Char stackexp::pop()**

**{**

**Char s;**

**S=stk[top];**

**Top=top-1;**

**Return s;**

**}**

**Int stackexp::isfull()**

**{**

**If(top==size)**

**Return 1;**

**Else**

**Return 0;**

**}**

**Int stackexp::isempty()**

**{**

**If(top==-1)**

**Return 1;**

**Else**

**Return 0;**

**}**

**Int main()**

**{**

**Stackexp s1;**

**Char exp[20],ch;**

**Int i=0;**

**Cout << “\n\t!! Parenthesis Checker..!!!!” << endl; // prints !!!Hello World!!!**

**Cout<<”\nEnter the expression to check whether it is in well form or not : “;**

**Cin>>exp;**

**If((exp[0]==’)’)||(exp[0]==’]’)||(exp[0]==’}’))**

**{**

**Cout<<”\n Invalid Expression…..\n”;**

**Return 0;**

**}**

**Else**

**{**

**While(exp[i]!=’\0’)**

**{**

**Ch=exp[i];**

**Switch(ch)**

**{**

**Case ‘(‘:s1.push(ch);break;**

**Case ‘[‘:s1.push(ch);break;**

**Case ‘{‘:s1.push(ch);break;**

**Case ‘)’:s1.pop();break;**

**Case ‘]’:s1.pop();break;**

**Case ‘}’:s1.pop();break;**

**}**

**I=i+1;**

**}**

**}**

**If(s1.isempty())**

**{**

**Cout<<”\nExpression is well parenthesised…\n”;**

**}**

**Else**

**{**

**Cout<<”\nSorry !!! Invalid Expression or not in well parenthesized….\n”;**

**}**

**Return 0;**

**}**

**Output:**

**Exp: ((a+b)\*[c-d])**

**!! Parenthesis Checker..!!!!**

**Enter the expression to check whether it is in well form or not : ((a+b)\*[c-d])**

**Expression is well**

**Parenthes8zed….**

**Exp: (a+b]\*{c/d)**

**!! Parenthesis Checker..!!!!**

**Enter the expression to check whether it is in well form or not : (a+b]\*{c/d)**

**Sorry !!! Invalid Expression or not in well parenthesized….**