**Modern Education Society’s  
College of Engineering, Pune**

|  |  |
| --- | --- |
| **NAME OF STUDENT:** Prathamesh Kalyan Sable | **CLASS:** SE Comp 1 |
| **SEMESTER/YEAR:** Sem-3 / 2022-23 | **ROLL NO:** 015 |
| **DATE OF PERFORMANCE:**  / /2022 | **DATE OF SUBMISSION:** / /2022 |
| **EXAMINED BY:** Prof. N.S. Gore | **EXPERIMENT NO: DSL D-26** |

**TITLE: PERFORM VARIOUS OPERATION ON STACK TO CHECK WELL PARENTHESIZED EXPRESSION**

**PROBLEM STATEMENT:** In any language program mostly syntax error occurs due to unbalancing delimiter such as (),{},[]. Write C++ program using stack to check whether given expression is well parenthesized or not.

**OBJECTIVES:  
 1.** To understand structure of stack.  
 **2.** To understand How to Create, Display and perform various operation on stack.

**OUTCOME:** 1. To use effective and efficient data structure in solving Computer Engineering   
 domain problem.  
 2. To analyze the problem to apply suitable algorithm and data structure.  
 3. To discriminate the usage of various structure in approaching problem solution.

**PRE-REQUISITES:  
 1.** Knowledge of C++ Programming  
 2. Knowledge of stack.

**APPARATUS:**

Computer Machine, c++ compiler installed, Code Editor, etc.

**QUESTIONS:**1. Explain the types of stack.  
2. Write down the application of stack.

**SOURCE CODE:**

#include <iostream>

#define max 15

using namespace std;

class stack {

    char list[max];

    public:

    int top;

    stack() {

        top = -1;

    }

    bool isempty() {

        return (top <= -1) ? true : false;

    }

    bool isfull() {

        return (top == max - 1) ? true : false;

    }

    void push(char elmt) {

        if (not isfull()) {

            top++;

            list[top] = elmt;

        }

    }

    char pop() {

        if (not isempty()) {

            top--;

            return list[top + 1];

        }

    }

};

bool checkParenthesis(string exp) {

    stack list;

    for (int i = 0; i < exp.size(); i++) {

        // for opening parenthesis add its closing parenthesis

        if (exp[i] == '{') {

            list.push('}');

        } else if (exp[i] == '[') {

            list.push(']');

        } else if (exp[i] == '(') {

            list.push(')');

        } else if ((exp[i] == '}') or (exp[i] == ']') or (exp[i] == ')')) {

            // check if closing parenthesis is in stack

            if (exp[i] != list.pop()) {

                return false;  // not valid

                break;

            }

        }

    }

    if (not list.isempty()) {

        return false;  // not valid

    }

    return true;  // if all runs fine valid

}

int main() {

    string exp;

    cout << "\* Maximun Parinthesis Allowed are 15 \*" << endl;

    cout << "Enter the expression:";

    cin >> exp;

    bool result = checkParenthesis(exp);

    if (result) {

        cout << "Expression is valid " << endl;

    } else {

        cout << "Expression is not valid " << endl;

    }

    return 0;

}

**OUTPUT:**

