**Foundation University**

**School of Science and Technology**



**Data Structure Lab Report:2**

**Name: Wazzaif Ali**

**Roll No:095**

**Department of Engineering Technology**

## Exercises:

1. Write a program to take numbers from user and stored into stack. Pop the values from stack and show only even values on screen and show highest marks on the screen.
2. Write a program to input an infix expression into a string variable. Convert the expression into post-fix notation by using a stack.

## Solution:

#include <iostream>

#include <stack>

using namespace std;

int main() {

stack<int> marksStack;

int n, value;

cout << "How many marks do you want to enter? ";

cin >> n;

cout << "Enter the marks:\n";

for (int i = 0; i < n; i++) {

cin >> value;

marksStack.push(value);

}

int highest = -1;

cout << "\nEven values popped from the stack:\n";

while (!marksStack.empty()) {

int topValue = marksStack.top();

marksStack.pop();

if (topValue > highest) {

highest = topValue;

}

if (topValue % 2 == 0) {

cout << topValue << " ";

}

}

cout << "\n\nHighest marks: " << highest << endl;

return 0;

}

**Answer:2**

#include <iostream>

#include <stack>

#include <string>

using namespace std;

int precedence(char op) {

if (op == '+' || op == '-') return 1;

if (op == '\*' || op == '/') return 2;

return 0;

}

string infixToPostfix(string infix) {

stack<char> s;

string postfix = "";

for (int i = 0; i < infix.length(); i++) {

char ch = infix[i];

if ((ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z') || (ch >= '0' && ch <= '9')) {

postfix += ch;

}

else if (ch == '(') {

s.push(ch);

}

else if (ch == ')') {

while (!s.empty() && s.top() != '(') {

postfix += s.top();

s.pop();

}

if (!s.empty()) s.pop();

}

else {

while (!s.empty() && precedence(s.top()) >= precedence(ch)) {

postfix += s.top();

s.pop();

}

s.push(ch);

}

}

while (!s.empty()) {

postfix += s.top();

s.pop();

}

return postfix;

}

int main() {

string infix;

cout << "Enter infix expression: ";

cin >> infix;

string postfix = infixToPostfix(infix);

cout << "Postfix expression: " << postfix << endl;

return 0;

}

**Output:**

