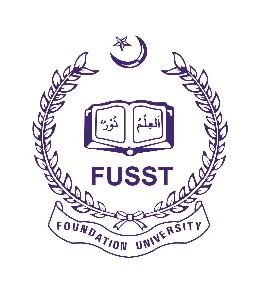
**Department of Engineering Technology**

**Foundation University Islamabad**

**School of Science and Technolog**

Data Structure

**NAME: Muhammad Anas**

**ROLL NO: 072**

**Lab 2**

**Topic : Introduction to Stack.**

**Objective : Stack Implementation**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Performance** | | | **Lab Report** | | |
| **Description** | **Total Marks** | **Marks Obtained** | **Description** | **Total Marks** | **Marks**  **Obtained** |
| **Problem Understanding** | **5** |  | **Code Writing** | **5** |  |
| **Communication** | **5** |  |  |  |  |
| **Total Marks obtained** | | |  | | |

**Remarks (if any): ………………………………….**

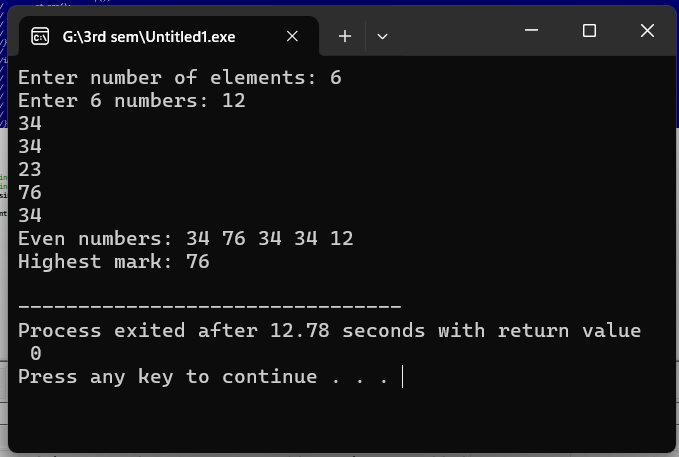
**Name & Signature of faculty: …………………………………**

Exercises: LAB 02

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.**

**CODE:**

1. #include <iostream>
2. #include <stack>
3. using namespace std;
4. int main() {
5. stack<int> st;
6. int n, val;
7. cout << "Enter number of elements: ";
8. cin >> n;
9. cout << "Enter " << n << " numbers: ";
10. for (int i = 0; i < n; i++) {
11. cin >> val;
12. st.push(val);
13. }
14. int highest = st.top();
15. cout << "Even numbers: ";
16. while (!st.empty()) {
17. int topVal = st.top();
18. st.pop();
19. if (topVal % 2 == 0) {
20. cout << topVal << " ";
21. }
22. if (topVal > highest) {
23. highest = topVal;
24. }
25. }
26. cout << "\nHighest mark: " << highest << endl;
27. return 0;
28. }

**OUTPUT :** 

**2. Write a program to input an infix expression into a string variable. Convert the expression into post-fix notation by using a stack.**

**CODE:**

1. #include <iostream>
2. #include <stack>
3. #include <string>
4. using namespace std;
5. int priority(char c) {
6. if (c == '+' || c == '-') return 1;
7. if (c == '\*' || c == '/') return 2;
8. if (c == '^') return 3;
9. return 0;
10. }
11. string infixToPostfix(string infix) {
12. stack<char> st;
13. string result = "";
14. for (int i = 0; i < infix.length(); i++) {
15. char c = infix[i];
16. if (isalnum(c)) {
17. result += c;
18. }
19. else if (c == '(') {
20. st.push(c);
21. }
22. else if (c == ')') {
23. while (!st.empty() && st.top() != '(') {
    1. result += st.top();
    2. st.pop();
24. }
25. st.pop();
26. }
27. else {
28. while (!st.empty() && priority(st.top()) >= priority(c)) {
    1. result += st.top();
    2. st.pop();
29. }
30. st.push(c);
31. }
32. }
33. while (!st.empty()) {
34. result += st.top();
35. st.pop();
36. }
37. return result;
38. }
39. int main() {
40. string exp;
41. cout << "Enter infix expression: ";
42. cin >> exp;
43. cout << "Postfix expression: " << infixToPostfix(exp) << endl;
44. return 0;
45. }

**OUTPUT:**

