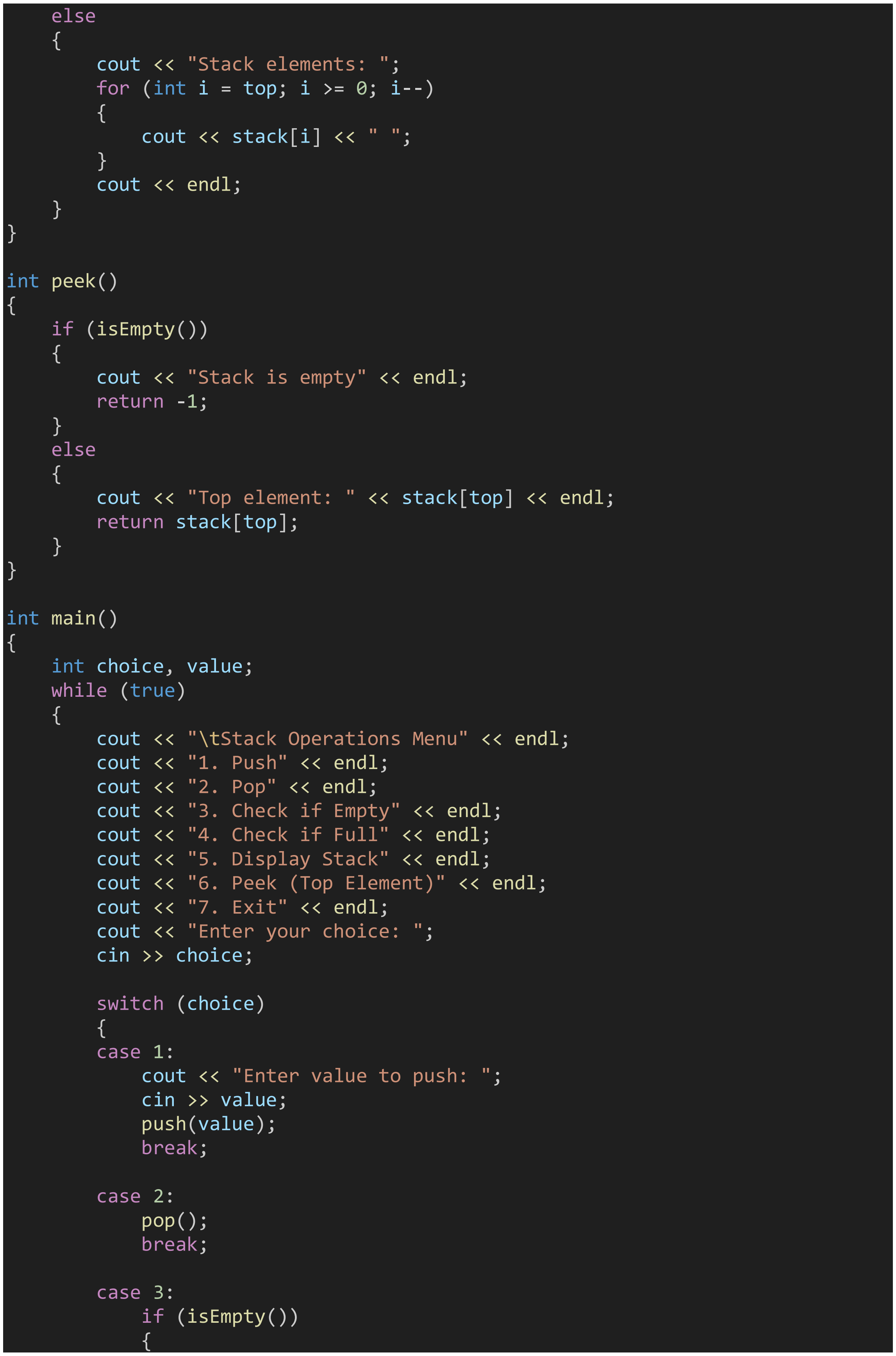
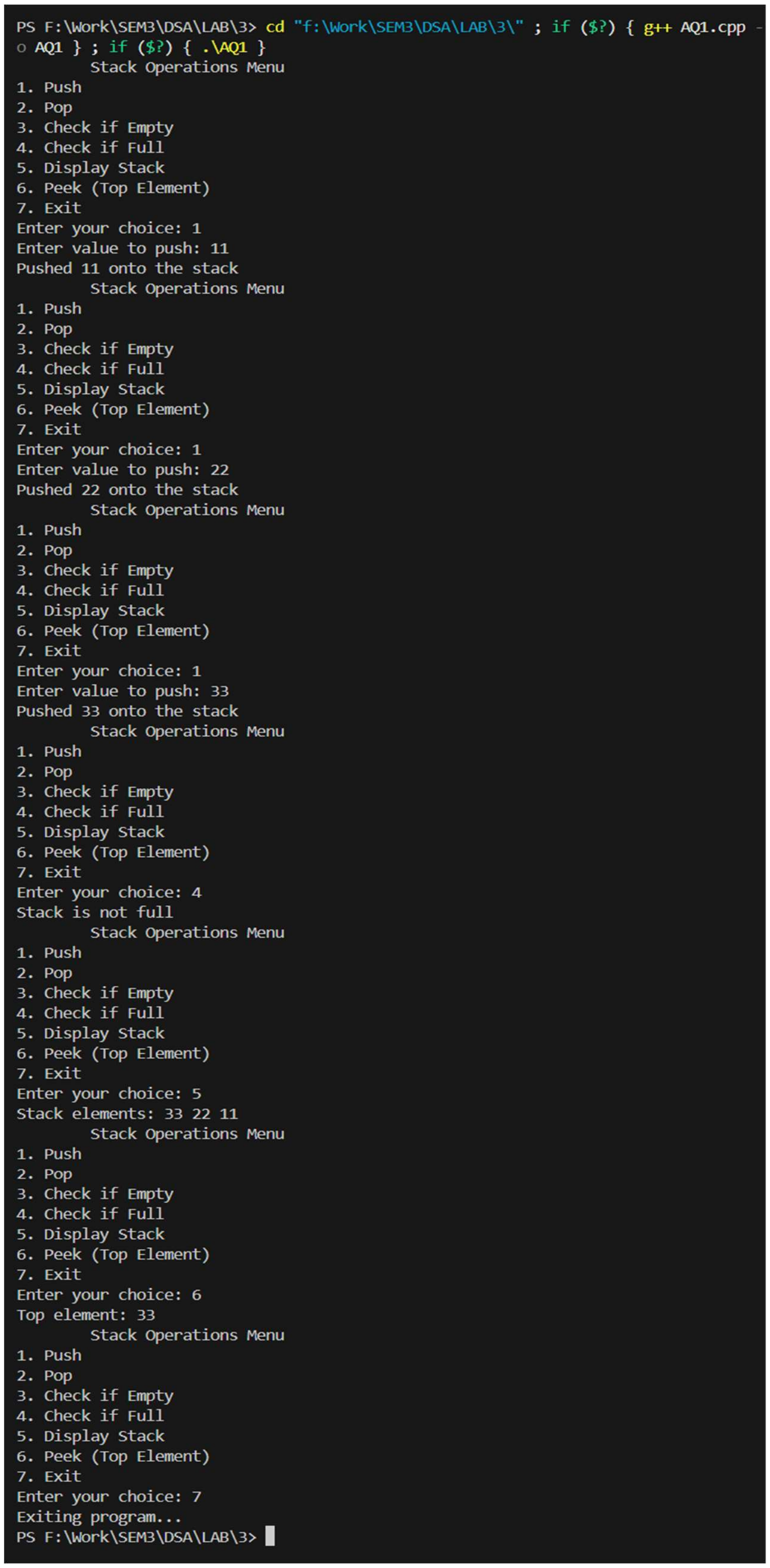
AQ1

```cpp
#include <iostream>
using namespace std;

const int MAX_SIZE = 100;
int stack[MAX_SIZE];
int top = -1;

void push(int value)
{
    if (top >= MAX_SIZE - 1)
    {
        cout << "Stack Overflow! Cannot push " << value << endl;
    }
    else
    {
        top++;
        stack[top] = value;
        cout << "Pushed " << value << " onto the stack" << endl;
    }
}

int pop()
{
    if (top < 0)
    {
        cout << "Stack Underflow! Stack is empty" << endl;
        return -1;
    }
    else
    {
        int value = stack[top];
        top--;
        cout << "Popped " << value << " from the stack" << endl;
        return value;
    }
}

bool isEmpty()
{
    return (top < 0);
}

bool isFull()
{
    return (top >= MAX_SIZE - 1);
}

void display()
{
    if (isEmpty())
    {
        cout << "Stack is empty" << endl;
    }
```



```cpp
    else
    {
        cout << "Stack elements: ";
        for (int i = top; i >= 0; i--)
        {
            cout << stack[i] << " ";
        }
        cout << endl;
    }
}

int peek()
{
    if (isEmpty())
    {
        cout << "Stack is empty" << endl;
        return -1;
    }
    else
    {
        cout << "Top element: " << stack[top] << endl;
        return stack[top];
    }
}

int main()
{
    int choice, value;
    while (true)
    {
        cout << "\tStack Operations Menu" << endl;
        cout << "1. Push" << endl;
        cout << "2. Pop" << endl;
        cout << "3. Check if Empty" << endl;
        cout << "4. Check if Full" << endl;
        cout << "5. Display Stack" << endl;
        cout << "6. Peek (Top Element)" << endl;
        cout << "7. Exit" << endl;
        cout << "Enter your choice: ";
        cin >> choice;

        switch (choice)
        {
        case 1:
            cout << "Enter value to push: ";
            cin >> value;
            push(value);
            break;

        case 2:
            pop();
            break;

        case 3:
            if (isEmpty())
            {
                cout << "Stack is empty" << endl;
            }
            else
            {
                cout << "Stack is not empty" << endl;
            }
            break;

        case 4:
            if (isFull())
            {
                cout << "Stack is full" << endl;
            }
            else
            {
                cout << "Stack is not full" << endl;
            }
            break;

        case 5:
            display();
            break;

        case 6:
            peek();
            break;

        case 7:
            cout << "Exiting program..." << endl;
            return 0;

        default:
            cout << "Invalid choice! Please try again." << endl;
        }
    }
    return 0;
}
```

Output:

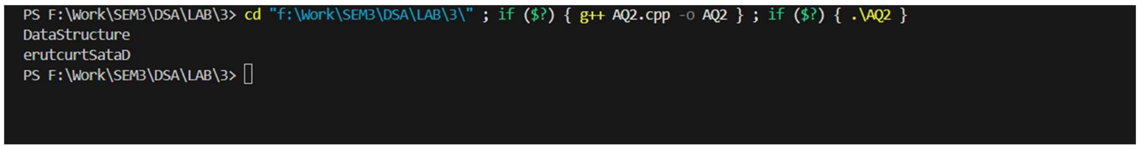


```
PS F:\Work\SEM3\DSA\LAB\3> cd "f:\Work\SEM3\DSA\LAB\3\" ; if ($?) { g++ AQ1.cpp -
o AQ1 } ; if ($?) { .\AQ1 }
        Stack Operations Menu
1. Push
2. Pop
3. Check if Empty
4. Check if Full
5. Display Stack
6. Peek (Top Element)
7. Exit
Enter your choice: 1
Enter value to push: 11
Pushed 11 onto the stack
        Stack Operations Menu
1. Push
2. Pop
3. Check if Empty
4. Check if Full
5. Display Stack
6. Peek (Top Element)
7. Exit
Enter your choice: 1
Enter value to push: 22
Pushed 22 onto the stack
        Stack Operations Menu
1. Push
2. Pop
3. Check if Empty
4. Check if Full
5. Display Stack
6. Peek (Top Element)
7. Exit
Enter your choice: 1
Enter value to push: 33
Pushed 33 onto the stack
        Stack Operations Menu
1. Push
2. Pop
3. Check if Empty
4. Check if Full
5. Display Stack
6. Peek (Top Element)
7. Exit
Enter your choice: 4
Stack is not full
        Stack Operations Menu
1. Push
2. Pop
3. Check if Empty
4. Check if Full
5. Display Stack
6. Peek (Top Element)
7. Exit
Enter your choice: 5
Stack elements: 33 22 11
        Stack Operations Menu
1. Push
2. Pop
3. Check if Empty
4. Check if Full
5. Display Stack
6. Peek (Top Element)
7. Exit
Enter your choice: 6
Top element: 33
        Stack Operations Menu
1. Push
2. Pop
3. Check if Empty
4. Check if Full
5. Display Stack
6. Peek (Top Element)
7. Exit
Enter your choice: 7
Exiting program...
PS F:\Work\SEM3\DSA\LAB\3>
```

AQ2

```cpp
#include <iostream>
#include <stack>
using namespace std;

int main() {
    string s = "DataStructure";
    stack<char> st;
    for (char c : s)
        st.push(c);
    string res;
    while (!st.empty()) {
        res += st.top();
        st.pop();
    }
    cout << s << endl;
    cout << res << endl;
    return 0;
}
```

Output:

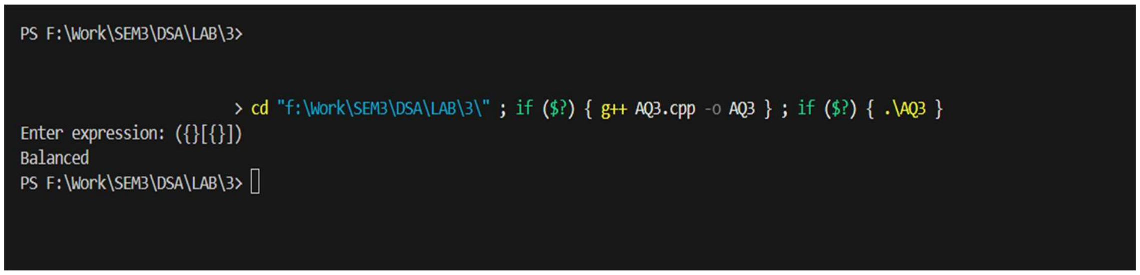


```
PS F:\Work\SEM3\DSA\LAB\3> cd "f:\Work\SEM3\DSA\LAB\3\" ; if ($?) { g++ AQ2.cpp -o AQ2 } ; if ($?) { .\AQ2 }
DataStructure
erutcurtSataD
PS F:\Work\SEM3\DSA\LAB\3> 
```

AQ3

```cpp
#include <iostream>
#include <stack>
using namespace std;
int main() {
    string s;
    cout << "Enter expression: ";
    cin >> s;

    stack<char> st;
    bool ch = true;
    for (char c : s)
    {
        if (c == '(' || c == '{' || c == '[')
        {
            st.push(c);
        }
        else if (c == ')' || c == '}' || c == ']')
        {
            if (st.empty())
            {
                ch = false;
                break;
            }
            char t = st.top();
            st.pop();
            if ((c == ')' && t != '(') || (c == '}' && t != '{') || (c == ']'
&& t != '['))
            {
                ch = false;
                break;
            }
        }
    }
    if (!st.empty()) ch = false;
    if (ch)
    {
        cout << "Balanced";
    }
    else
    {
        cout << "Not Balanced";
    }
    return 0;
}
```

Output:



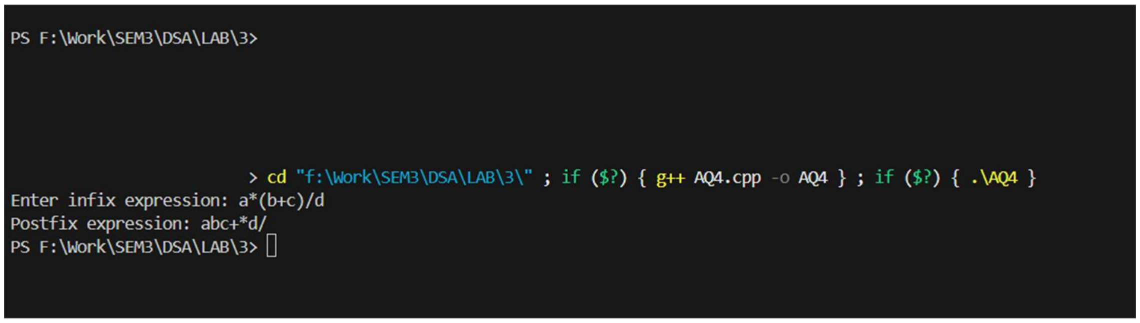
```
PS F:\Work\SEM3\DSA\LAB\3>

                          > cd "f:\Work\SEM3\DSA\LAB\3\" ; if ($?) { g++ AQ3.cpp -o AQ3 } ; if ($?) { .\AQ3 }
Enter expression: ({}[{}])
Balanced
PS F:\Work\SEM3\DSA\LAB\3>
```

AQ4

```
#include <iostream>
#include <stack>
#include <string>
using namespace std;

int main() {
    string s;
    cout << "Enter infix expression: ";
    cin >> s;
    stack<char> st;
    string res;
    for (int i = 0; i < s.length(); i++) {
        char c = s[i];

        if ((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z') || (c >= '0' && c
<= '9'))
            res += c;
        else if (c == '(')
            st.push('(');
        else if (c == ')') {
            while (st.top() != '(') {
                res += st.top();
                st.pop();
            }
            st.pop();
        }
        else {
            int prec_c;
            if (c == '^')
                prec_c = 3;
            else if (c == '/' || c == '*')
                prec_c = 2;
            else if (c == '+' || c == '-')
                prec_c = 1;
            else
```

```cpp
                prec_c = -1;

            while (!st.empty()) {
                char top = st.top();
                int prec_top;
                if (top == '^')
                    prec_top = 3;
                else if (top == '/' || top == '*')
                    prec_top = 2;
                else if (top == '+' || top == '-')
                    prec_top = 1;
                else
                    prec_top = -1;

                if (prec_c <= prec_top) {
                    res += st.top();
                    st.pop();
                } else {
                    break;
                }
            }
            st.push(c);
        }
    }

    while (!st.empty()) {
        res += st.top();
        st.pop();
    }

    cout << "Postfix expression: " << res << endl;
    return 0;
}
```

Output:

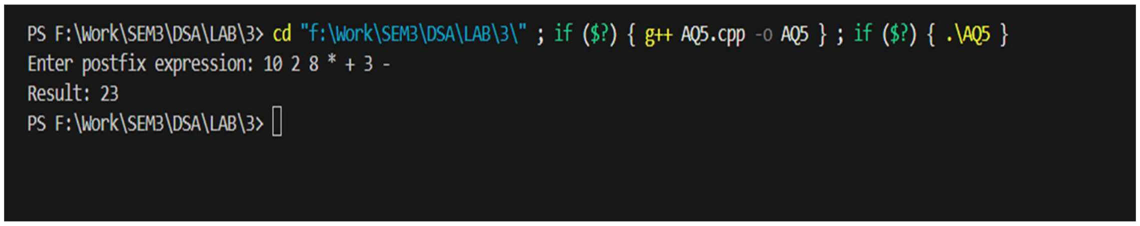


```
PS F:\Work\SEM3\DSA\LAB\3>



                  > cd "f:\Work\SEM3\DSA\LAB\3\" ; if ($?) { g++ AQ4.cpp -o AQ4 } ; if ($?) { .\AQ4 }
Enter infix expression: a*(b+c)/d
Postfix expression: abc+*d/
PS F:\Work\SEM3\DSA\LAB\3> 
```

AQ5

```cpp
#include <iostream>
#include <stack>
#include <string>
using namespace std;
int main()
{
    string input;
    cout << "Enter postfix expression: ";
    getline(cin, input);
    stack<int> st;
    string token = "";

    for (int i=0;i<=input.length();i++)
    {
        if (i==input.length() || input[i]==' ')
        {
            if (!token.empty())
            {
                if (isdigit(token[0]) || (token.size()>1 && token[0]=='-'))
                {
                    int num = 0;
                    int sign = 1;
                    int start = 0;
                    if (token[0]=='-')
                    {
                        sign = -1;
                        start = 1;
                    }
                    for (int j=start;j<token.length(); j++)
                    {
                        num=num*10+(token[j]-'0');
                    }
                    st.push(sign*num);
                }
                else
                {
                    int val1 = st.top();
                    st.pop();
                    int val2 = st.top();
                    st.pop();
                    if (token=="+")
                    {
                        st.push(val2 + val1);
                    }
                    else if (token=="-")
                    {
                        st.push(val2 - val1);
                    }
                    else if (token=="*")
                    {
                        st.push(val2 * val1);
                    }
                    else if (token=="/")
                    {
                        st.push(val2 / val1);
                    }
                }
                token = "";
            }
        }
        else if (input[i]!=' ')
        {
            token+=input[i];
        }
    }

    cout << "Result: " << st.top() << endl;
    return 0;
}
```

Output:



```
PS F:\Work\SEM3\DSA\LAB\3> cd "f:\Work\SEM3\DSA\LAB\3\" ; if ($?) { g++ AQ5.cpp -o AQ5 } ; if ($?) { .\AQ5 }
Enter postfix expression: 10 2 8 * + 3 -
Result: 23
PS F:\Work\SEM3\DSA\LAB\3> 
```