Problem: Lab Assessment (Stack)

Code:

stackas.cpp > main()

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

const int MAX = 100;

int stack[MAX];
int top = -1;

void push(int value) {
    if (top >= MAX - 1) {
        cout << "Stack Overflow" << endl;
        return;
    }
    stack[++top] = value;
    cout << "Push " << value << " → stack: [";
    for (int i = 0; i <= top; ++i) {
        cout << stack[i];
        if (i < top) cout << ", ";
    }
    cout << "]" << endl;
}

int pop() {
    if (top == -1) {
        cout << "Stack Underflow" << endl;
        return -1;
    }
    int popped = stack[top--];
    cout << "Pop " << popped << " → stack: [";
    for (int i = 0; i <= top; ++i) {
        cout << stack[i];
        if (i < top) cout << ", ";
    }
    cout << "]" << endl;
    return popped;
}

```

stackas.cpp > main()

```cpp
bool isValidUnloadSequence(int loaded[], int unloaded[], int n) {
    top = -1;

    int unloadIndex = 0;

    for (int i = 0; i < n; ++i) {
        push(loaded[i]);

        while (top >= 0 && stack[top] == unloaded[unloadIndex]) {
            pop();
            unloadIndex++;
        }
    }

    return (top == -1);
}

int main() {

    int n = 5;
    int loaded[n] ;
    int unloaded[n] ;

    cout << "Enter the load Secuence: " ;
    for(int i=0; i<n ; i++){
        cin >> loaded[i] ;
    }
    cout << "Enter the unload Secuence: " ;
    for(int i=0; i<n ; i++){
        cin >> unloaded[i] ;
    }

    cout << "\n--- Simulating Stack Operations ---\n\n";
    bool result = isValidUnloadSequence(loaded, unloaded, n);

    cout << "\nResult: ";
    if (result)
        cout << "Unloading sequence is VALID.\n";
    else
        cout << "Unloading sequence is NOT valid.\n";

    return 0;
}

```

Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
Active code page: 65001

D:\GitHub002\03 Third Semester\CSE 2104_Data Structures Lab\
 Third Semester\CSE 2104_Data Structures Lab\Lab Report\7th

d:\GitHub002\03 Third Semester\CSE 2104_Data Structures Lab\
xe"
Enter the load Secuence: 1 2 3 4 5
Enter the unload Secuence: 4 5 3 2 1

--- Simulating Stack Operations ---

Push 1 → stack: [1]
Push 2 → stack: [1, 2]
Push 3 → stack: [1, 2, 3]
Push 4 → stack: [1, 2, 3, 4]
Pop 4 → stack: [1, 2, 3]
Push 5 → stack: [1, 2, 3, 5]
Pop 5 → stack: [1, 2, 3]
Pop 3 → stack: [1, 2]
Pop 2 → stack: [1]
Pop 1 → stack: []

Result: Unloading sequence is VALID.

d:\GitHub002\03 Third Semester\CSE 2104_Data Structures Lab\
```