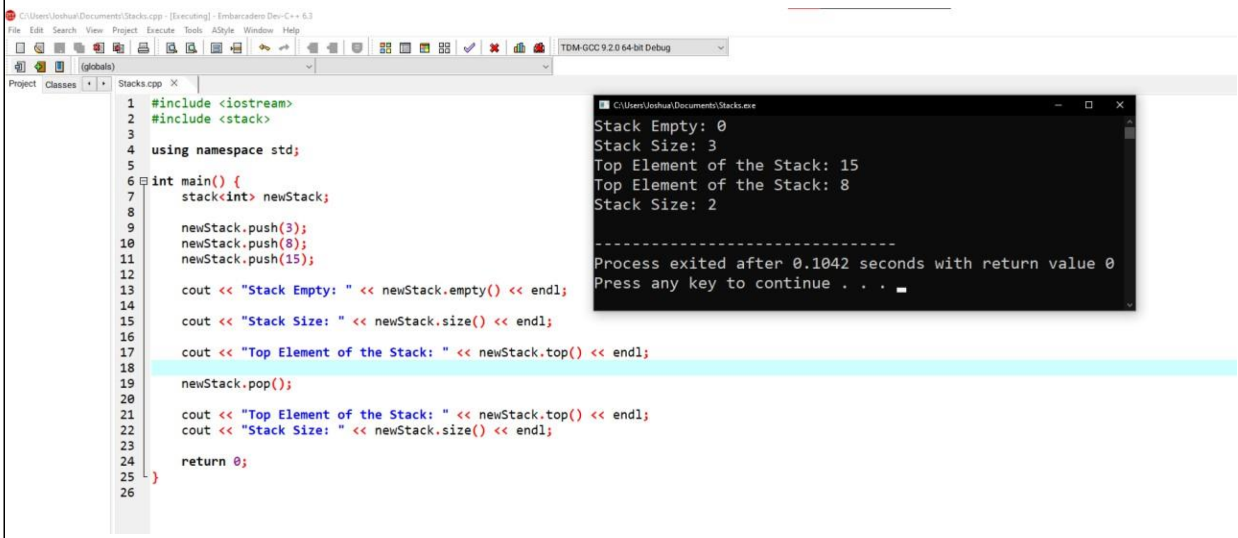
Activity No. <4>	
<STACKS>	
Course Code: CPE010	**Program:** Computer Engineering
Course Title: Data Structures and Algorithms	**Date Performed: 8/25/25**
Section: CPE21S4	**Date Submitted: 8/25/25**
Name(s): Quioyo, Angelo M.	**Instructor: Engr. Jimlord Quejado**

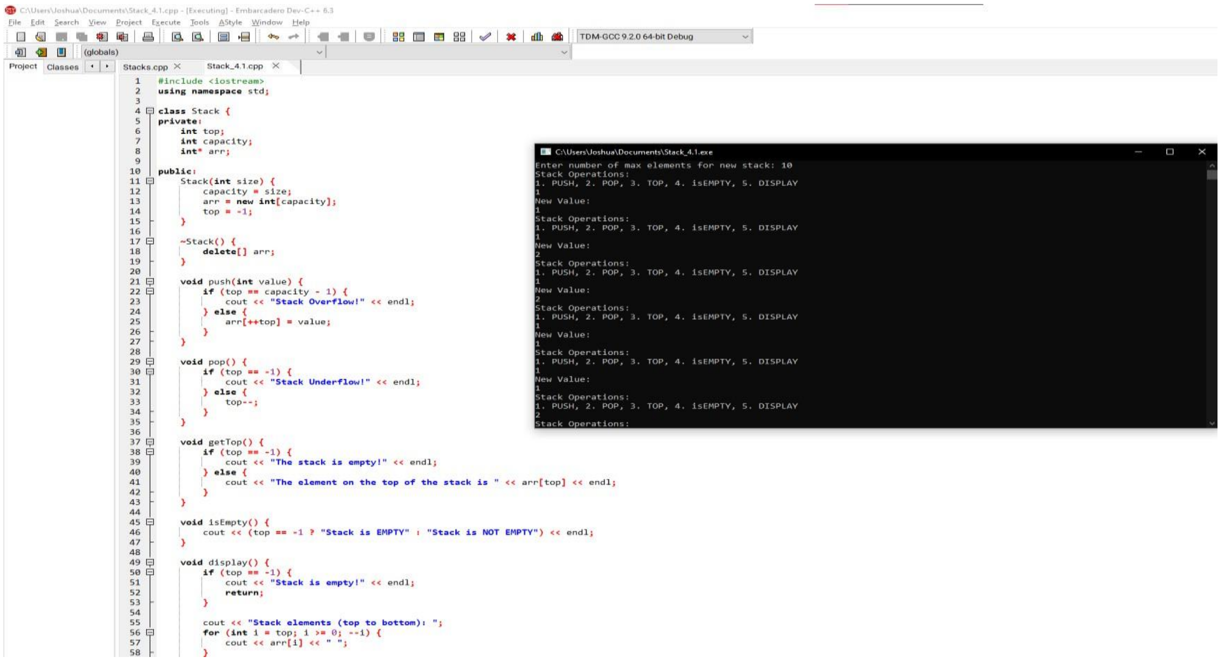
6. Output:



```cpp
#include <iostream>
#include <stack>

using namespace std;

int main() {
    stack<int> newStack;

    newStack.push(3);
    newStack.push(8);
    newStack.push(15);

    cout << "Stack Empty: " << newStack.empty() << endl;

    cout << "Stack Size: " << newStack.size() << endl;

    cout << "Top Element of the Stack: " << newStack.top() << endl;

    newStack.pop();

    cout << "Top Element of the Stack: " << newStack.top() << endl;
    cout << "Stack Size: " << newStack.size() << endl;

    return 0;
}
```

```
Stack Empty: 0
Stack Size: 3
Top Element of the Stack: 15
Top Element of the Stack: 8
Stack Size: 2

--------------------------------
Process exited after 0.1042 seconds with return value 0
Press any key to continue . . .
```

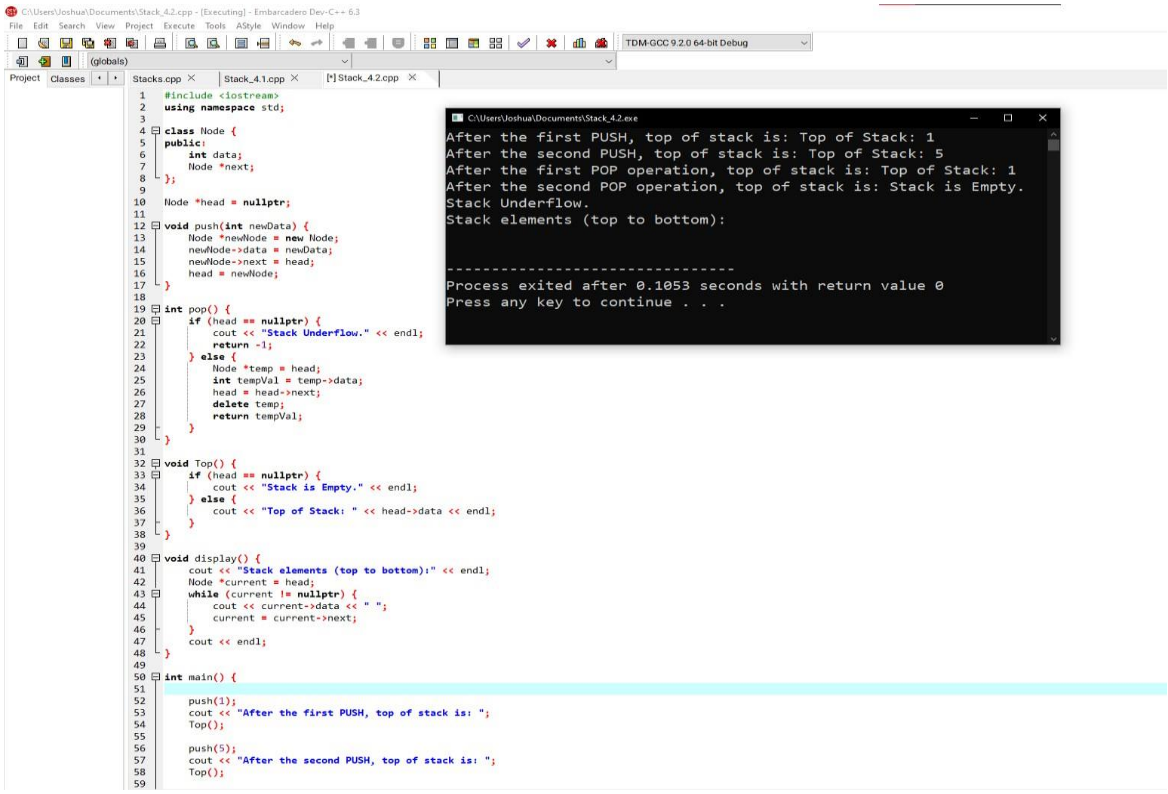
Table 4.1



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

class Stack {
private:
    int top;
    int capacity;
    int* arr;

public:
    Stack(int size) {
        capacity = size;
        arr = new int[capacity];
        top = -1;
    }

    ~Stack() {
        delete[] arr;
    }

    void push(int value) {
        if (top == capacity - 1) {
            cout << "Stack Overflow!" << endl;
        } else {
            arr[++top] = value;
        }
    }

    void pop() {
        if (top == -1) {
            cout << "Stack Underflow!" << endl;
        } else {
            top--;
        }
    }

    void getTop() {
        if (top == -1) {
            cout << "The stack is empty!" << endl;
        } else {
            cout << "The element on the top of the stack is " << arr[top] << endl;
        }
    }

    void isEmpty() {
        cout << (top == -1 ? "Stack is EMPTY" : "Stack is NOT EMPTY") << endl;
    }

    void display() {
        if (top == -1) {
            cout << "Stack is empty!" << endl;
            return;
        }

        cout << "Stack elements (top to bottom): ";
        for (int i = top; i >= 0; --i) {
            cout << arr[i] << " ";
        }
```

```
Enter number of max elements for new stack: 10
Stack Operations:
1. PUSH, 2. POP, 3. TOP, 4. isEMPTY, 5. DISPLAY
1
New Value:
1
Stack Operations:
1. PUSH, 2. POP, 3. TOP, 4. isEMPTY, 5. DISPLAY
1
New Value:
2
Stack Operations:
1. PUSH, 2. POP, 3. TOP, 4. isEMPTY, 5. DISPLAY
1
New Value:
2
Stack Operations:
1. PUSH, 2. POP, 3. TOP, 4. isEMPTY, 5. DISPLAY
1
New Value:
1
Stack Operations:
1. PUSH, 2. POP, 3. TOP, 4. isEMPTY, 5. DISPLAY
1
New Value:
1
Stack Operations:
1. PUSH, 2. POP, 3. TOP, 4. isEMPTY, 5. DISPLAY
2
Stack Operations:
```

Table 4.2



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

class Node {
public:
    int data;
    Node *next;
};

Node *head = nullptr;

void push(int newData) {
    Node *newNode = new Node;
    newNode->data = newData;
    newNode->next = head;
    head = newNode;
}

int pop() {
    if (head == nullptr) {
        cout << "Stack Underflow." << endl;
        return -1;
    } else {
        Node *temp = head;
        int tempVal = temp->data;
        head = head->next;
        delete temp;
        return tempVal;
    }
}

void Top() {
    if (head == nullptr) {
        cout << "Stack is Empty." << endl;
    } else {
        cout << "Top of Stack: " << head->data << endl;
    }
}

void display() {
    cout << "Stack elements (top to bottom):" << endl;
    Node *current = head;
    while (current != nullptr) {
        cout << current->data << " ";
        current = current->next;
    }
    cout << endl;
}

int main() {

    push(1);
    cout << "After the first PUSH, top of stack is: ";
    Top();

    push(5);
    cout << "After the second PUSH, top of stack is: ";
    Top();

```

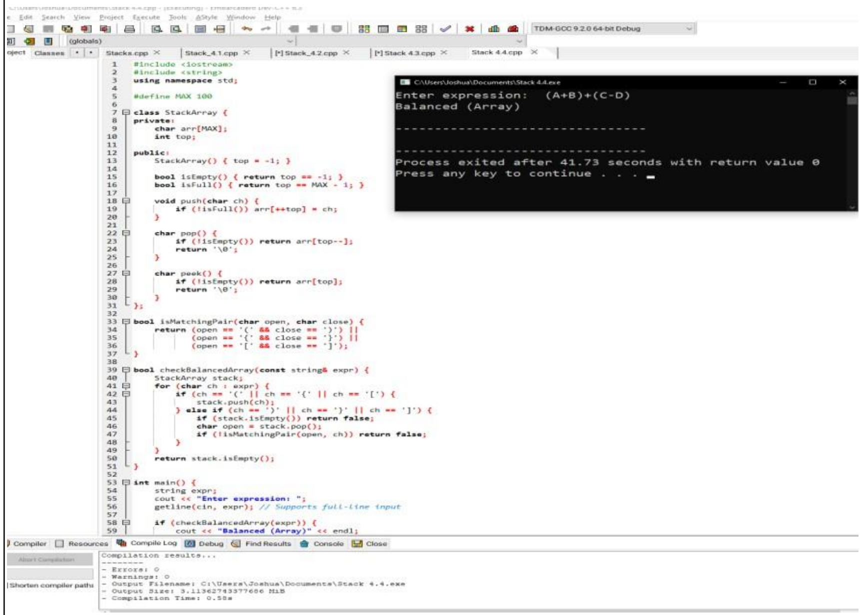
```
After the first PUSH, top of stack is: Top of Stack: 1
After the second PUSH, top of stack is: Top of Stack: 5
After the first POP operation, top of stack is: Top of Stack: 1
After the second POP operation, top of stack is: Stack is Empty.
Stack Underflow.
Stack elements (top to bottom):


--------------------------------
Process exited after 0.1053 seconds with return value 0
Press any key to continue . . .
```

7. Supplementary Activity:

ILO C: SOLVE PROBLEMS USING AN IMPLEMENTATION OF STACK:

Table 4.3

a. Stack Using Arrays

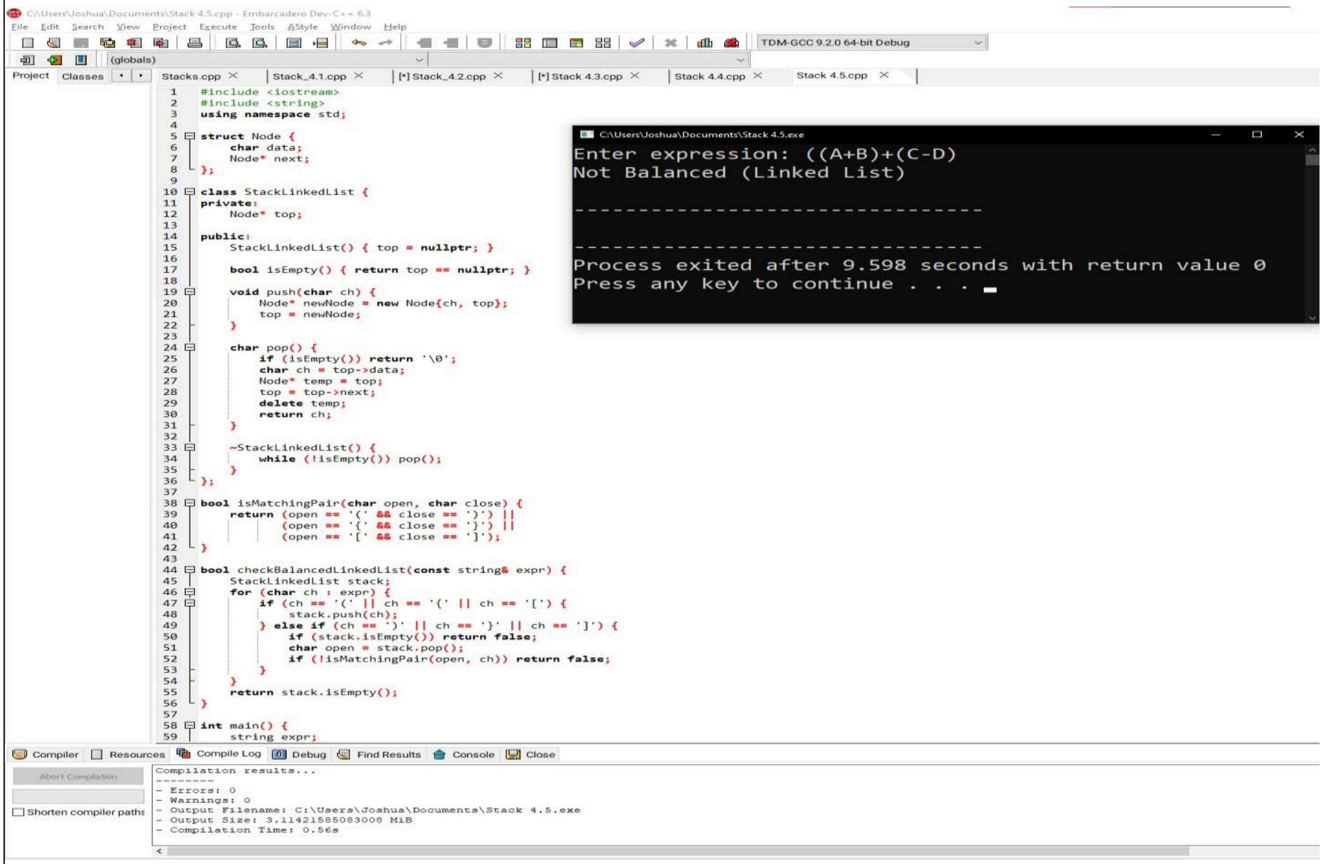


```cpp
#include <iostream>
#include <string>
using namespace std;

#define MAX 100

class StackArray {
private:
    char arr[MAX];
    int top;

public:
    StackArray() { top = -1; }

    bool isEmpty() { return top == -1; }
    bool isFull() { return top == MAX - 1; }

    void push(char ch) {
        if (!isFull()) arr[++top] = ch;
    }

    char pop() {
        if (!isEmpty()) return arr[top--];
        return '\0';
    }

    char peek() {
        if (!isEmpty()) return arr[top];
        return '\0';
    }
};

bool isMatchingPair(char open, char close) {
    return (open == '(' && close == ')') ||
           (open == '{' && close == '}') ||
           (open == '[' && close == ']');
}

bool checkBalancedArray(const string& expr) {
    StackArray stack;
    for (char ch : expr) {
        if (ch == '(' || ch == '{' || ch == '[') {
            stack.push(ch);
        } else if (ch == ')' || ch == '}' || ch == ']') {
            if (stack.isEmpty()) return false;
            char open = stack.pop();
            if (!isMatchingPair(open, ch)) return false;
        }
    }
    return stack.isEmpty();
}

int main() {
    string expr;
    cout << "Enter expression: ";
    getline(cin, expr); // Supports full-line input

    if (checkBalancedArray(expr)) {
        cout << "Balanced (Array)" << endl;
```

```
Enter expression:  (A+B)+(C-D)
Balanced (Array)

--------------------------------

--------------------------------
Process exited after 41.73 seconds with return value 0
Press any key to continue . . .
```

```
Compilation results...
--------
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\Joshua\Documents\Stack 4.4.exe
- Output Size: 3.11362743377686 MiB
- Compilation Time: 0.58s
```

b. Stack Using Linked Lists


C:\Users\Joshua\Documents\Stack 4.5.cpp - Embarcadero Dev-C++ 6.3
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 9.2.0 64-bit Debug
(globals)
Project Classes
Stacks.cpp
Stack_4.1.cpp
[*] Stack_4.2.cpp
[*] Stack 4.3.cpp
Stack 4.4.cpp
Stack 4.5.cpp
C:\Users\Joshua\Documents\Stack 4.5.exe
Compiler Resources Compile Log Debug Find Results Console Close
Abort Compilation
Shorten compiler paths

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

struct Node {
    char data;
    Node* next;
};

class StackLinkedList {
private:
    Node* top;

public:
    StackLinkedList() { top = nullptr; }

    bool isEmpty() { return top == nullptr; }

    void push(char ch) {
        Node* newNode = new Node{ch, top};
        top = newNode;
    }

    char pop() {
        if (isEmpty()) return '\0';
        char ch = top->data;
        Node* temp = top;
        top = top->next;
        delete temp;
        return ch;
    }

    ~StackLinkedList() {
        while (!isEmpty()) pop();
    }
};

bool isMatchingPair(char open, char close) {
    return (open == '(' && close == ')') ||
           (open == '{' && close == '}') ||
           (open == '[' && close == ']');
}

bool checkBalancedLinkedList(const string& expr) {
    StackLinkedList stack;
    for (char ch : expr) {
        if (ch == '(' || ch == '{' || ch == '[') {
            stack.push(ch);
        } else if (ch == ')' || ch == '}' || ch == ']') {
            if (stack.isEmpty()) return false;
            char open = stack.pop();
            if (!isMatchingPair(open, ch)) return false;
        }
    }
    return stack.isEmpty();
}

int main() {
    string expr;
```

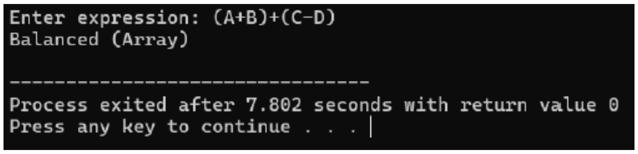
```
Enter expression: ((A+B)+(C-D)
Not Balanced (Linked List)

--------------------------------

--------------------------------
Process exited after 9.598 seconds with return value 0
Press any key to continue . . .
```

```
Compilation results...
--------
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\Joshua\Documents\Stack 4.5.exe
- Output Size: 3.11421585083008 MiB
- Compilation Time: 0.56s
```

Self-Checking:

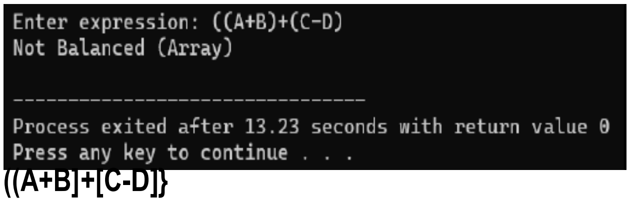
Expression:

(A+B)+(C-D)



```
Enter expression: (A+B)+(C-D)
Balanced (Array)

--------------------------------
Process exited after 7.802 seconds with return value 0
Press any key to continue . . .
```

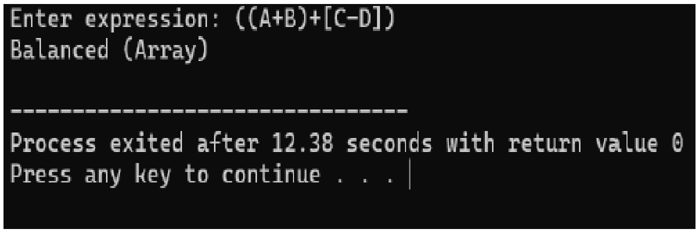
((A+B)+(C-D)



```
Enter expression: ((A+B)+(C-D)
Not Balanced (Array)

--------------------------------
Process exited after 13.23 seconds with return value 0
Press any key to continue . . .
```

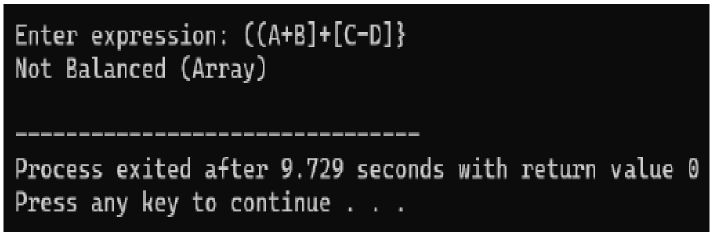
((A+B]+[C-D]}

((A+B)+[C-D])



```
Enter expression: ((A+B)+[C-D])
Balanced (Array)

--------------------------------
Process exited after 12.38 seconds with return value 0
Press any key to continue . . .
```

((A+B]+[C-D]}



```
Enter expression: ((A+B]+[C-D]}
Not Balanced (Array)

--------------------------------
Process exited after 9.729 seconds with return value 0
Press any key to continue . . .
```

8. Conclusion:

During this activity, I developed a thorough and full understanding of the operations on a stack - that is, push, pop, top, and isempty. Implementing these functions really helped me understand the Last In, First Out (LIFO) concept which is the core concept of how stacks function. I found a systematic and organized approach to processing the state of the stack and that all of the operations on the stack were correct and efficient. I am more confident than ever with the core concepts related to the stack, but I realize I have much to learn, especially in terms of further optimizing my code for readability and performance. This experience has given me a solid foundation to continue to explore more complex data structures.

9. Assessment Rubric