Assignment 3

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

const int MAX_CAPACITY = 100;
int stackArr[MAX_CAPACITY];
int topIndex = -1;

void push(int element) {
    if (topIndex >= MAX_CAPACITY - 1) {
        cout << "[!] Stack Overflow! Cannot insert " << element << endl;
    } else {
        stackArr[++topIndex] = element;
        cout << "[+] Inserted: " << element << endl;
    }
}

int pop() {
    if (topIndex < 0) {
        cout << "[!] Stack Underflow! Nothing to pop.\n";
        return -1;
    } else {
```

```
        int poppedValue = stackArr[topIndex--];
        cout << "[-] Removed: " << poppedValue << endl;
        return poppedValue;
    }
}
bool isEmpty() {
    return topIndex < 0;
}

bool isFull() {
    return topIndex >= MAX_CAPACITY - 1;
}

int peek() {
    if (isEmpty()) {
        cout << "[!] Stack is empty, no top element.\n";
        return -1;
    } else {
        cout << "[*] Current top: " << stackArr[topIndex] << endl;
        return stackArr[topIndex];
    }
}
```

```
void printStack() {
    if (isEmpty()) {
        cout << "[*] Stack is currently empty.\n";
    } else {
        cout << ">>> Stack (Top to Bottom): ";
        for (int i = topIndex; i >= 0; --i) {
            cout << stackArr[i] << " ";
        }
        cout << endl;
    }
}

int main() {
    int choice, item;

    while (true) {
        cout << "\n STACK MENU: \n";
        cout << "1. Push element\n";
        cout << "2. Pop element\n";
        cout << "3. Check if Empty\n";
        cout << "4. Check if Full\n";
        cout << "5. Display Stack\n";
        cout << "6. Peek at Top\n";
```

```
    cout << "7. Exit\n";
    cout << "Select option: ";
    cin >> choice;

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

        case 2:
            pop();
            break;

        case 3:
            cout << (isEmpty() ? "Stack is empty.\n" : "Stack is not
empty.\n");
            break;

        case 4:
            cout << (isFull() ? "Stack is full.\n" : "Stack has space.\n");
            break;
```

```
        case 5:
            printStack();
            break;

        case 6:
            peek();
            break;

        case 7:
            cout << "Exiting the program. \n";
            return 0;

        default:
            cout << "[!] Invalid selection. Try again.\n";
    }
  }

  return 0;
}
```

‘

```
 STACK MENU:
1. Push element
2. Pop element
3. Check if Empty
4. Check if Full
5. Display Stack
6. Peek at Top
7. Exit
Select option: 1
Enter value to add:
4
[+] Inserted: 4

 STACK MENU:
1. Push element
2. Pop element
3. Check if Empty
4. Check if Full
5. Display Stack
6. Peek at Top
7. Exit
Select option: 1
Enter value to add: 67
[+] Inserted: 67

 STACK MENU:
1. Push element
2. Pop element
3. Check if Empty
4. Check if Full
5. Display Stack
6. Peek at Top
7. Exit
Select option: 5
>>> Stack (Top to Bottom): 67 4

 STACK MENU:
1. Push element
```

```
 STACK MENU:
1. Push element
2. Pop element
3. Check if Empty
4. Check if Full
5. Display Stack
6. Peek at Top
7. Exit
Select option: 7
Exiting the program.
```

AQ2

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

int main() {
    char input[200];
    stack<char> stk;

    cout << "Type a string to reverse: ";
    cin >> input;

    int idx = 0;
    while (input[idx] != '\0') {
        stk.push(input[idx]);
        idx++;
    }

    string reversedText;
```

```
    while (!stk.empty()) {
        reversedText += stk.top();
        stk.pop();
    }

    cout << "Original text: " << input << endl;
    cout << "Reversed form: " << reversedText << endl;

    return 0;
}
```

```
Type a string to reverse: Hello
Original text: Hello
Reversed form: olleH
```

AQ3

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

int main() {
    string expr;
    cout << "Input your expression: ";
```

```
cin >> expr;

stack<char> charStack;
bool balanced = true;

for (char ch : expr) {
    if (ch == '(' || ch == '{' || ch == '[') {
        charStack.push(ch);
    }
    else if (ch == ')' || ch == '}' || ch == ']') {
        if (charStack.empty()) {
            balanced = false;
            break;
        }
        char topChar = charStack.top();
        charStack.pop();
        if ((ch == ')' && topChar != '(') ||
            (ch == '}' && topChar != '{') ||
            (ch == ']' && topChar != '[')) {
            balanced = false;
            break;
```

```
            }
        }
    }

    if (!charStack.empty())
        balanced = false;

    if (balanced)
        cout << "Expression is balanced\n";
    else
        cout << "Expression is not balanced\n";

    return 0;
}
```

```
Input your expression: 5*(3+9)
Expression is balanced
```

AQ4

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

int getPrecedence(char op) {
    if (op == '+' || op == '-') return 1;
    if (op == '*' || op == '/') return 2;
    if (op == '^') return 3;
    return 0;
}

int main() {
    string infixInput;
    cout << "Input infix expression: ";
    cin >> infixInput;

    stack<char> operators;
    string postfixResult;
```

```
    for (size_t i = 0; i < infixInput.length(); i++) {
        char ch = infixInput[i];

        if ((ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z') || (ch
>= '0' && ch <= '9')) {
            postfixResult += ch;
        }
        else if (ch == '(') {
            operators.push(ch);
        }
        else if (ch == ')') {
            while (!operators.empty() && operators.top() != '(') {
                postfixResult += operators.top();
                operators.pop();
            }
            if (!operators.empty()) operators.pop();
        }
        else {
            int prec_ch = getPrecedence(ch);
```

```
            while (!operators.empty() &&
getPrecedence(operators.top()) >= prec_ch) {
                postfixResult += operators.top();
                operators.pop();
            }
            operators.push(ch);
        }
    }

    while (!operators.empty()) {
        postfixResult += operators.top();
        operators.pop();
    }

    cout << "Converted postfix: " << postfixResult << "\n";
    return 0;
}
```

```
Input infix expression: 5*(3+9)
Converted postfix: 539+*
```

AQ5

```
#include <iostream>
#include <stack>
#include <string>
using namespace std;
int main() {
    string postfixExp;
    cout << "Enter postfix expression: ";
    cin >> postfixExp;
    stack<int> s;
    for (int i = 0; i < postfixExp.length(); i++) {
        char ch = postfixExp[i];
        if (ch >= '0' && ch <= '9') {
            s.push(ch - '0');
        }
        else {
            int b = s.top();
            s.pop();
            int a = s.top();
            s.pop();
```

```
            if (ch == '+') s.push(a + b);
            else if (ch == '-') s.push(a - b);
            else if (ch == '*') s.push(a * b);
            else if (ch == '/') s.push(a / b);
        }
    }
    cout << "Result: " << s.top() << endl;
    return 0;
}
```

```
Enter postfix expression: 539+*
Result: 60
```