**NAME : MALIHA SHAHID**

**ROLL NO:SU92-BSSEM-S24-071**

**LAB #10**

**Stack with Array & Linklist**

1. **With Array; Push, Pop, Display**

#include <iostream>

using namespace std;

class StackArray {

private:

int topIndex;

int stackElements[100];

int capacity;

public:

StackArray(int maxSize = 100) {

topIndex = -1;

capacity = maxSize;

}

void push(int value) {

if (topIndex == capacity - 1) {

cout << "Stack Overflow\n";

return;

}

stackElements[++topIndex] = value;

}

void pop() {

if (topIndex == -1) {

cout << "Stack Underflow\n";

return;

}

topIndex--;

}

void display() {

if (topIndex == -1) {

cout << "Stack is empty\n";

return;

}

cout << "Stack (Top -> Bottom): ";

for (int i = topIndex; i >= 0; i--) {

cout << stackElements[i] << " ";

}

cout << "\n";

}

};

int main() {

StackArray myStack;

myStack.push(10);

myStack.push(20);

myStack.push(30);

myStack.display();

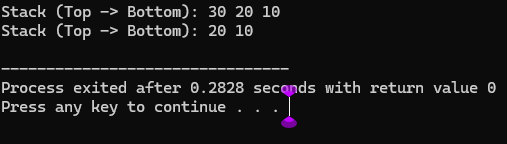
myStack.pop();

myStack.display();

return 0;

}

**OUTPUT**

****

**EXPLANATION**

1. This code defines a class **StackArray** to create a stack using an array.
2. The stack can store up to 100 elements by default**.**
3. The **push** function adds a new value to the next available position in the stack.
4. The **pop** function decreases the topIndex, effectively removing the top element from the stack.
5. **Display** shows all stack elements from top to bottom.
6. In main, the stack is used by adding and removing elements, then displaying it.

1. **With Linkedlist; Push, Pop, Display**

#include <iostream>

using namespace std;

class StackNode {

public:

int value;

StackNode\* nextNode;

StackNode(int val) {

value = val;

nextNode = NULL;

}

};

class LinkedStack {

private:

StackNode\* topNode;

public:

LinkedStack() {

topNode = NULL;

}

void push(int val) {

StackNode\* newElement = new StackNode(val);

newElement->nextNode = topNode;

topNode = newElement;

}

void pop() {

if (topNode == NULL) {

cout << "Stack Underflow\n";

return;

}

StackNode\* tempNode = topNode;

topNode = topNode->nextNode;

delete tempNode;

}

void display() {

if (topNode == NULL) {

cout << "Stack is empty\n";

return;

}

StackNode\* current = topNode;

cout << "Stack (Top -> Bottom): ";

while (current != NULL) {

cout << current->value << " ";

current = current->nextNode;

}

cout << "\n";

}

};

int main() {

LinkedStack myStack;

myStack.push(100);

myStack.push(200);

myStack.push(300);

myStack.display();

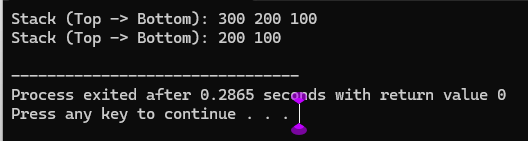
myStack.pop();

myStack.display();

return 0;

}

**OUTPUT**

****

**EXPLANATION**

1. This code defines **a stack using linked list** (called LinkedStack), where each element is a StackNode.
2. Each node stores a value and a pointer to the next node in the stack.
3. The **push** function adds a new node to the top of the stack..
4. The **pop** function removes the top node and frees memory; it shows "Stack Underflow" if the stack is empty.
5. The **display** function prints all values from top to bottom of the stack.
6. In main, values 100, 200, and 300 are pushed, one is popped, and the stack is displayed before and after.