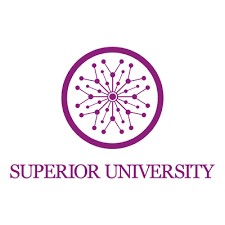
**TASK NO 10**



**M ZUHAIB ANWAR**

**SU92-BSSEM-S24-005**

**SE-3A**

**DSA LAB**

//Array (stack)

#include <iostream>

using namespace std;

const int MAX\_SIZE = 100;

class StackArray{

private:

int stack[MAX\_SIZE];

int top;

public:

StackArray()

{

top = -1;

}

void push(int item)

{

if (top >= MAX\_SIZE - 1)

{

cout << "It is Overflow and cannot push." << endl;

return;

}

stack[++top] = item;

cout << "Push " << item << " to stack" << endl;

}

int pop()

{

if (isEmpty())

{

cout << "It is Underflow and cannot pop." << endl;

return -1;

}

int item = stack[top--];

cout << "Pop " << item << " from stack" << endl;

return item;

}

void display() {

if (isEmpty())

{

cout << "It is empty" << endl;

return;

}

cout << "Current Stack (top to bottom): ";

for (int i = top; i >= 0; i--)

{

cout << stack[i] << " ";

}

cout << endl;

}

bool isEmpty() {

return top == -1;

}

};

int main() {

StackArray st;

st.push(10);

st.push(20);

st.push(30);

st.display();

st.pop();

st.display();

st.pop();

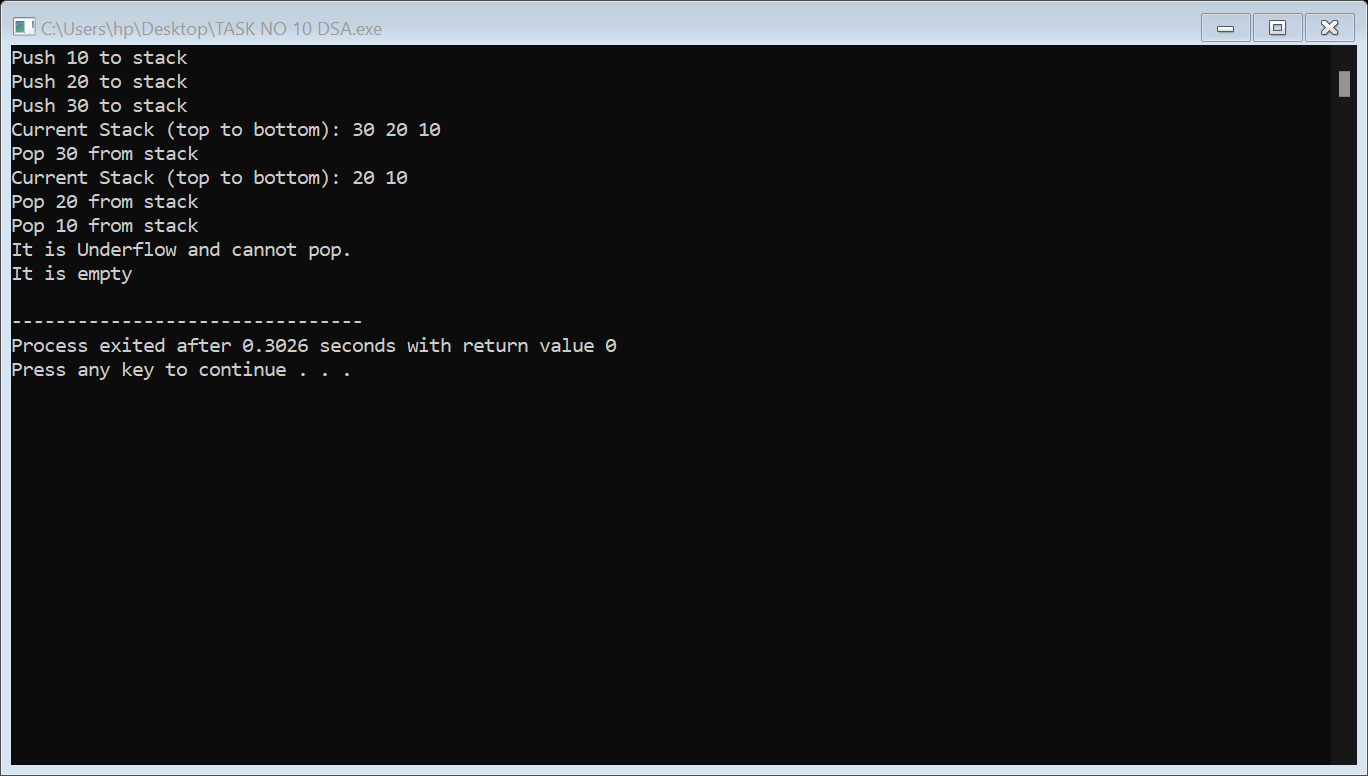
st.pop();

st.pop();

st.display();

return 0;

}



//Linked List (stack)

#include <iostream>

using namespace std;

class Node

{

public:

int data;

Node\* next;

Node(int val) {

data = val;

next = nullptr;

}

};

class LinkedListStack

{

private:

Node\* top;

public:

LinkedListStack()

{

top = nullptr;

}

void push(int item)

{

Node\* newNode = new Node(item);

newNode->next = top;

top = newNode;

cout << "Push" << item << " to stack" << endl;

}

int pop()

{

if (isEmpty())

{

cout << "It is Underflow and cannot pop." << endl;

return -1;

}

Node\* temp = top;

int item = top->data;

top = top->next;

delete temp;

cout << "Pop" << item << " from stack" << endl;

return item;

}

void display()

{

if (isEmpty())

{

cout << "It is empty" << endl;

return;

}

cout << "Current Stack (top to bottom): ";

Node\* current = top;

while (current != nullptr)

{

cout << current->data << " ";

current = current->next;

}

cout << endl;

}

bool isEmpty()

{

return top == nullptr;

}

~LinkedListStack()

{

while (!isEmpty())

{

pop();

}

}

};

int main()

{

LinkedListStack st;

st.push(10);

st.push(20);

st.push(30);

st.display();

st.pop();

st.display();

st.pop();

st.pop();

st.pop();

st.display();

return 0;

}

