TASK 1: #include<iostream>

#include<stack>

using namespace std;

struct node

{

int data;

node\* next;

};

class ArrayStack

{

int top;

public:

int a[100];

ArrayStack()

{

top = -1;

}

bool push(int x)

{

if (top > (100 - 1))

{

cout << "Stack Overflow";

return false;

}

else

{

a[++top] = x;

cout << x << " pushed into stack\n";

return true;

}

}

int pop()

{

if (top < 0)

{

cout << "Stack Underflow";

return 0;

}

else

{

int x = a[top--];

return x;

}

}

int peek()

{

if (top < 0)

{

cout << "Stack is Empty";

return 0;

}

else

{

int x = a[top];

return x;

}

}

bool isEmpty()

{

return (top < 0);

}

};

class linkedliststack

{

private:

node\* top;

public:

linkedliststack()

{

top = NULL;

}

void push(int data)

{

node\* temp=new node;

if (!temp)

{

cout << "Stack Overflow";

}

else

{

temp->data = data;

temp->next = top;

top=temp;

}

}

int isEmpty()

{

return top == NULL;

}

int peek()

{

if (!isEmpty())

return top->data;

else

exit(1);

}

void pop()

{

struct node\* temp;

if (top == NULL)

{

cout << "Stack Underflow" << endl;

}

else

{

temp = top;

top = top->next;

temp->next = NULL;

delete temp;

}

}

void display()

{

struct node\* temp;

if (top == NULL)

{

cout << "Stack Underflow";

}

else

{

temp = top;

while (temp != NULL)

{

cout << temp->data << " Pushed into the stack"<<endl;

temp = temp->next;

}

}

}

};

int main()

{

ArrayStack as;

linkedliststack lls;

int op;

cout << "Press 1 for the stack in Array\nPress 2 for linklist stack ";

cin >> op;

if (op == 1)

{

as.push(100);

as.push(500);

as.push(300);

cout << as.pop();

cout << " Popped from stack" << endl;

cout << "Elements present in stack : ";

while (!as.isEmpty())

{

cout << as.peek() << "\t";

as.pop();

}

}

else if (op == 2)

{

lls.push(10);

lls.push(20);

lls.push(30);

lls.push(40);

lls.display();

cout << "Top element is " << lls.peek() << endl;

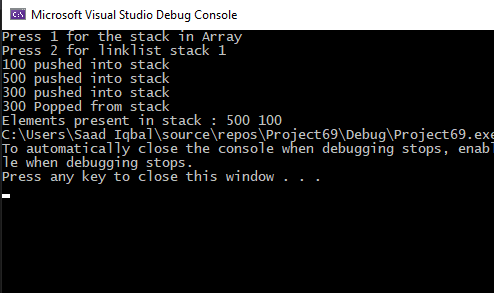
lls.pop();

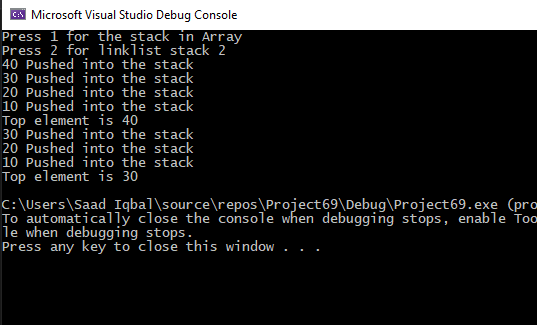
lls.display();

cout << "Top element is "<< lls.peek() << endl;

}

}

****

****