|  |  |
| --- | --- |
| download | COMSATS University Islamabad, Vehari Campus Department ofComputer Science |

**Class: BCS-SP22**

**Subject: Data Structures and Algorithms-Lab Instructor: Yasmeen Jana Reg. No: SP22-BCS-016 Name: Esha Rehman**

**Email:** [**yasmeenjana@cuivehari.edu.pk**](mailto:yasmeenjana@cuivehari.edu.pk)

**Question:2:**

**Implement stack using array.**

**Code:**

#include <iostream>

using namespace std;

int stack[100], n=100, top=-1;

void push(int val) {

if(top>n-1)

cout<<"Stack Overflow"<<endl;

else {

top++;

stack[top]=val;

}

}

void pop() {

if(top<=-1)

cout<<"Stack Underflow"<<endl;

else {

cout<<"The popped element is "<< stack[top] <<endl;

top--;

}

}

void display() {

if(top>=0) {

cout<<"Stack elements are:";

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

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

cout<<endl;

} else

cout<<"Stack is empty";

}

int main() {

int ch, val;

cout<<"1) Push in stack"<<endl;

cout<<"2) Pop from stack"<<endl;

cout<<"3) Display stack"<<endl;

cout<<"4) Exit"<<endl;

do {

cout<<"Enter choice: "<<endl;

cin>>ch;

switch(ch) {

case 1: {

cout<<"Enter value to be pushed:"<<endl;

cin>>val;

push(val);

break;

}

case 2: {

pop();

break;

}

case 3: {

display();

break;

}

case 4: {

cout<<"Exit"<<endl;

break;

}

default: {

cout<<"Invalid Choice"<<endl;

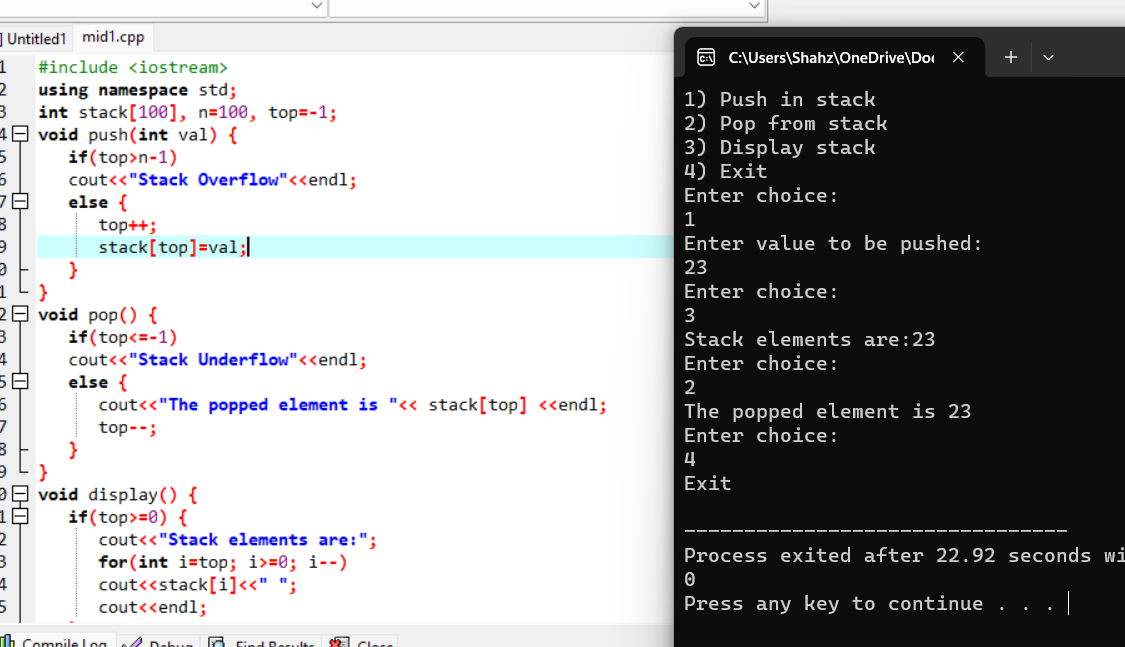
}

}

}while(ch!=4);

return 0;

}

**Output: **

**Question:1:**

#include <iostream>

using namespace std;

class Node {

public:

int data;

Node\* next;

Node(int value) : data(value), next(NULL) {}

};

// Singly Linked List class

class LinkedList {

public:

Node\* head;

LinkedList() : head(NULL) {}

// Function to insert a new node at the end of the list

void insert(int value) {

Node\* newNode = new Node(value);

if (head == NULL) {

head = newNode;

} else {

Node\* temp = head;

while (temp->next != NULL) {

temp = temp->next;

}

temp->next = newNode;

}

}

// Function to check if the linked list is a palindrome

bool SLL\_SCND\_FUNCTION\_PALINDROME() {

if (head == NULL) {

return true; // An empty list is considered a palindrome

}

// Use a stack to store the first half of the linked list

Node\* slow = head;

Node\* fast = head;

Node\* prev = NULL;

while (fast != NULL && fast->next != NULL) {

fast = fast->next->next;

// Reverse the first half of the list while traversing

Node\* nextNode = slow->next;

slow->next = prev;

prev = slow;

slow = nextNode;

}

// If the total number of nodes is odd, move slow one step forward

if (fast != NULL) {

slow = slow->next;

}

// Compare the first half with the second half

while (slow != NULL) {

if (prev->data != slow->data) {

return false;

}

slow = slow->next;

prev = prev->next;

}

return true;

}

};

int main() {

LinkedList list;

list.insert(1);

list.insert(2);

list.insert(2);

list.insert(1);

if (list.SLL\_SCND\_FUNCTION\_PALINDROME()) {

cout << "The linked list is a palindrome." << endl;

} else {

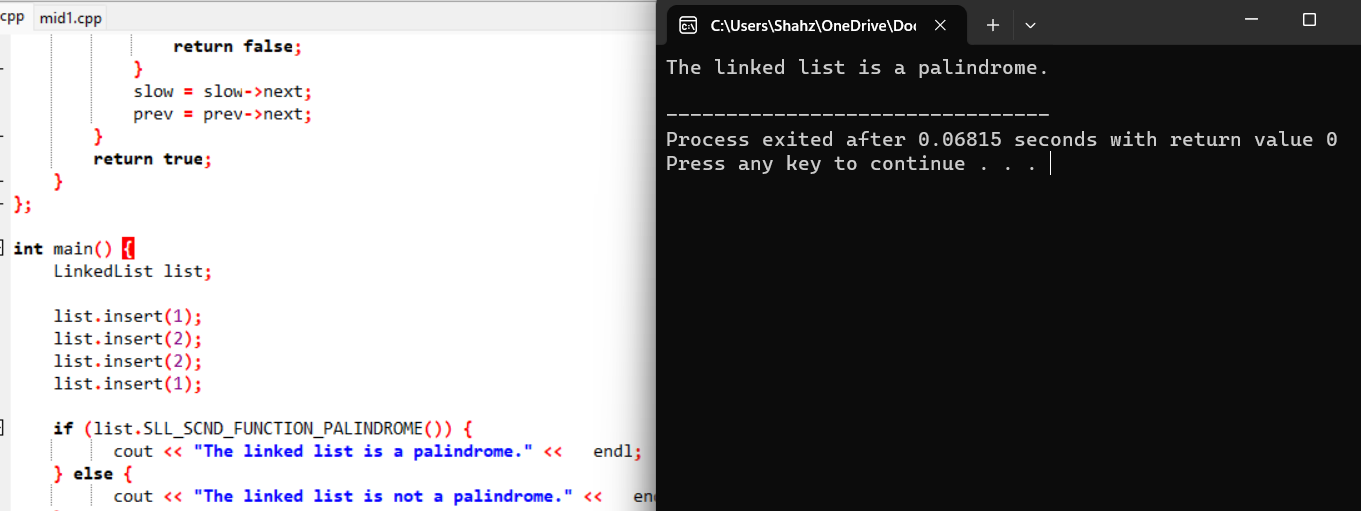
cout << "The linked list is not a palindrome." << endl;

}

return 0;

}

**Output:**

****