

**COMSATS University Islamabad,**

**Vehari Campus**

**Name: Naeem Sajjad**

**Registration Number: SP22-BCS-050**

**Department: Computer Science**

**Course: DSA Lab**

**Instructor: Mam. Yasmeen Jana**

**Exam: Mid term**

**Section: A**

**Mid**

**Question 1**

#include <iostream>

using namespace std;

class Node {

public:

int data;

Node\* next;

Node(int val) {

data = val;

next = NULL;

}

};

class LinkedList {

public:

Node\* head;

LinkedList() {

head = NULL;

}

void insert\_beg(int n) {

Node\* newNode = new Node(n);

if (head == NULL) {

head = newNode;

} else {

newNode->next = head;

head = newNode;

}

}

void display() {

Node\* current = head;

while (current != NULL) {

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

current = current->next;

}

cout << endl;

}

bool isPalindrome() {

if (head == NULL) {

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

}

// Create a custom stack-like data structure using pointers

Node\* slow = head;

Node\* fast = head;

Node\* stack = NULL;

// Push elements from the first half onto the custom stack

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

Node\* newNode = new Node(slow->data);

newNode->next = stack;

stack = newNode;

slow = slow->next;

fast = fast->next->next;

}

// If the list has an odd number of elements, skip the middle element

if (fast != NULL) {

slow = slow->next;

}

// Compare the second half with the elements popped from the custom stack

while (slow != NULL) {

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

return false; // Not a palindrome

}

Node\* temp = stack;

stack = stack->next;

delete temp;

slow = slow->next;

}

return true; // It's a palindrome

}

};

int main() {

LinkedList list;

list.insert\_beg(1);

list.insert\_beg(2);

list.insert\_beg(3);

list.insert\_beg(2);

list.insert\_beg(1);

cout << "Linked List Contents:" << endl;

list.display();

if (list.isPalindrome()) {

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

} else {

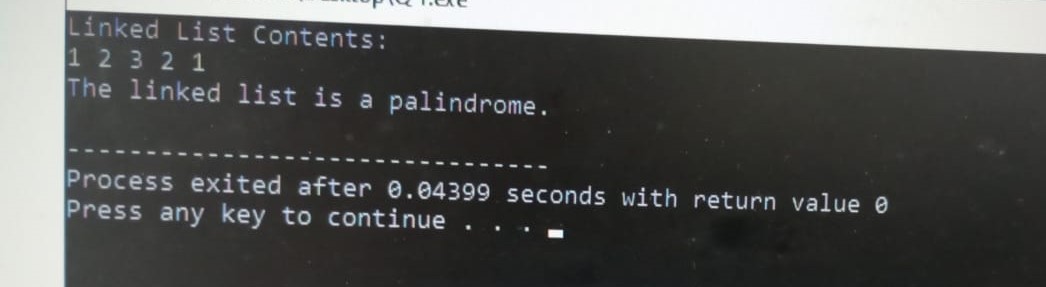
cout << "The linked list is not a palindrome." << endl;

}

return 0;

}

**Output**



**Question 2**

#include <iostream>

using namespace std;

class Stack{

private:

int\* arr;

int capacity;

int top;

public:

Stack(int size) {

capacity = size;

arr = new int[size];

top = -1;

}

void push(int val){

if(top<capacity-1){

top++;

arr[top]=val;

}

else{

cout<<"Stack is overflow"<<endl;

}

}

void pop(){

if(top>=0){

cout<<"poped value is "<<arr[top]<<endl;

top--;

}

else{

cout<<"Stack is underflow"<<endl;

}

}

void display(){

if(top==-1){

cout<<"Stack is empty"<<endl;

}

else{

cout<<"\nStack elements are ";

for(int i=0; i<=top; i++){

cout<<arr[i]<<" ";

}

cout<<endl<<endl;

}

}

};

int main(){

Stack s(5);

s.push(1);

s.push(2);

s.push(3);

s.push(6);

s.push(9);

s.display();

s.pop();

s.pop();

s.pop();

s.display();

return 0;

}

**Output**

