|  |  |
| --- | --- |
| download | COMSATS University Islamabad, Vehari Campus Department of Computer Science |

**Class: BSCS-SP22-4B Date: 23 Oct 2023**

**Subject: Data Structure & Algorithm Lab Instructor: Yasmeen Jana Max Marks: 25 Reg. No: SP22-BCS-075**

**Max Time: 90 Minutes**

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

## Activity 1:

Write a C++ code to create a singly linked list using "SLL()" function and Remove duplicates from an unsorted linked list as RemoveDup() function and display linked list with unique values. **(15)**

For Example:

Input: linked list = 12->11->12->21->41->43->21

Output: 12->11->21->41->43.



Hint:

Use two loops, Outer loop is used to pick the elements one by one and the Inner loop compares the picked element with the rest of the elements.

## Activity 2:

Write a C++ code to create a Queue using a linked list. The code should contain functions for Enqueue(), Dequeue(), and Display(). **(10)**

## Program no 1:

## Code:

#include <iostream>

class Node {

public:

int data;

Node\* next;

Node(int value) : data(value), next(0) {}

};

class LinkedList {

public:

Node\* head;

LinkedList() : head() {}

// Function to add a new node to the end of the linked list

void addNode(int value) {

Node\* newNode = new Node(value);

if (head == 0) {

head = newNode;

} else {

Node\* current = head;

while (current->next != 0) {

current = current->next;

}

current->next = newNode;

}

}

// Function to remove duplicates from the linked list

void removeDuplicates() {

Node\* current = head;

while (current != 0) {

Node\* runner = current;

while (runner->next != 0) {

if (runner->next->data == current->data) {

Node\* temp = runner->next;

runner->next = runner->next->next;

delete temp;

} else {

runner = runner->next;

}

}

current = current->next;

}

}

// Function to display the linked list with unique values

void displayList() {

Node\* current = head;

while (current != 0) {

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

current = current->next;

}

std::cout << std::endl;

}

};

int main() {

LinkedList list;

// Adding elements to the linked list

list.addNode(1);

list.addNode(2);

list.addNode(3);

list.addNode(2);

list.addNode(4);

list.addNode(1);

list.addNode(1);

std::cout << "Original Linked List: ";

list.displayList();

// Removing duplicates

list.removeDuplicates();

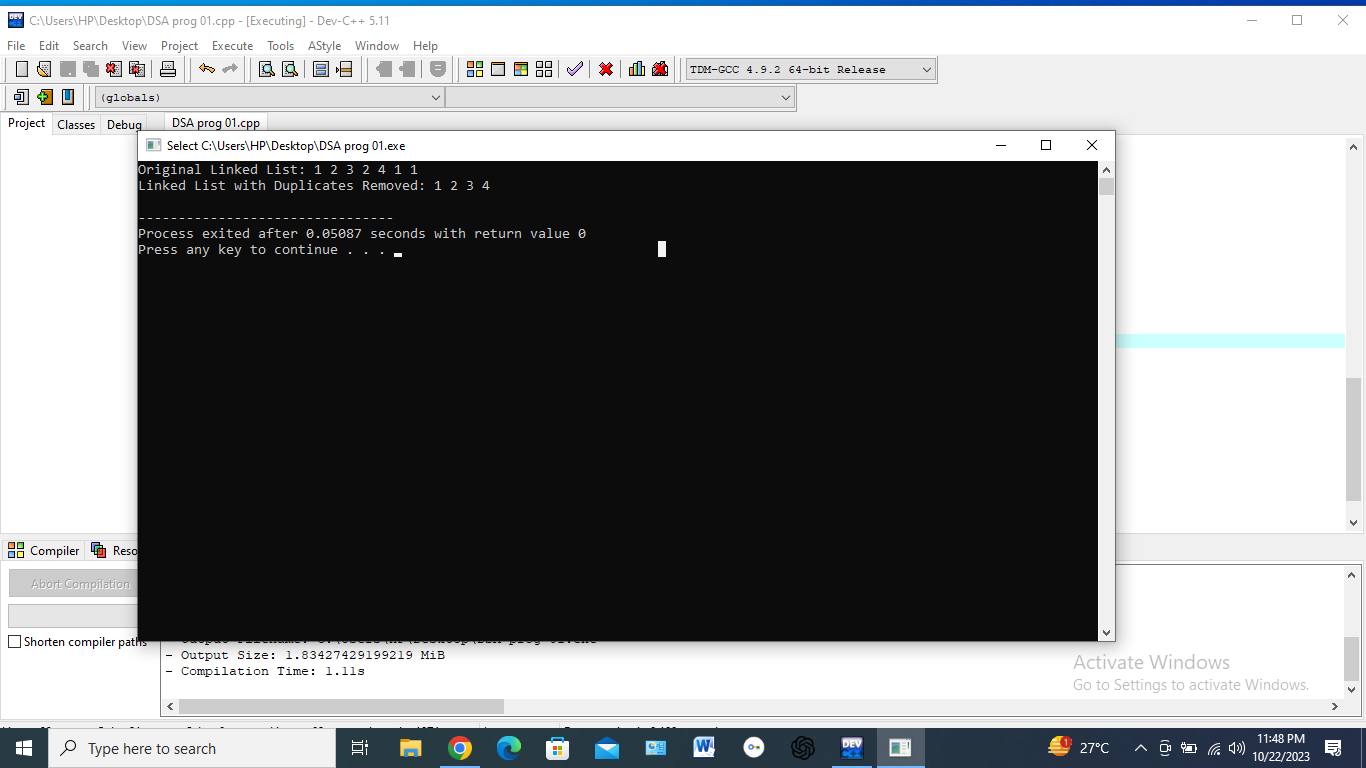
std::cout << "Linked List with Duplicates Removed: ";

list.displayList();

return 0;

}

**Output:**

****

Program no 2:

Code:

#include<iostream>

using namespace std;

class Node {

private:

int data;

Node \*next;

public:

Node \*front,\*rear=NULL;

void enqueue(int x){

Node \*p=new Node;

p->data=x;

p->next=NULL;

if(front==NULL || rear==NULL){

front=p;

rear=p;

cout<<"\nThe inserted element in queue is: \n"<<rear->data;

}

else{

rear->next=p;

rear=p;

cout<<"\nThe inserted element in queue is: \n"<<rear->data;

}

}

void dequeue(){

Node \*d=new Node();

d=front;

if(d==NULL)

{

cout<<"\nEmpty queue";

}

else{

//if(d==NULL)

cout<<"\nThe dequeue elements is: ";

cout<<d->data;

front=front->next;

delete d;

d=NULL;

}

}

void display() {

Node \*temp = front;

cout << "\nThe queue elements are: ";

if (temp == NULL) {

cout << "empty";

}

while (temp != NULL) {

cout << temp->data << " ";

temp = temp->next;

}

}

};

int main(){

Node i;

i.enqueue(1);

i.enqueue(2);

i.display();

i.dequeue();

i.display();

i.enqueue(3);

i.enqueue(4);

i.display();

i.dequeue();

i.dequeue();

i.dequeue();

i.dequeue();

i.display();

return 0;

}

Output:

