|  |  |
| --- | --- |
| 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: SP22BCS-129**

**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.

### **Solution**:

#include <iostream>

using namespace std;

struct Node {

int data;

Node\* next;

Node(int value) {

data=value;

next=NULL;}

};

void insert(Node\*& head, int data) {

Node\* newNode = new Node(data);

if (head == NULL) {

head = newNode;

} else {

Node\* current = head;

while (current->next != NULL) {

current = current->next;

}

current->next = newNode;

}

}

void RemoveDup(Node\* head) {

if (head == NULL) {

return;

}

Node\* current = head;

while (current != NULL) {

Node\* runner = current;

while (runner->next != NULL) {

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

Node\* temp = runner->next;

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

delete temp;

} else {

runner = runner->next;

}

}

current = current->next;

}

}

void display(Node\* head) {

Node\* current = head;

while (current != NULL) {

cout << current->data;

if (current->next != NULL) {

cout << " -> ";

}

current = current->next;

}

cout <<endl;

}

int main() {

Node\* head = NULL;

insert(head, 12);

insert(head, 11);

insert(head, 12);

insert(head, 21);

insert(head, 41);

insert(head, 43);

insert(head, 21);

cout << "Original Linked List: ";

display(head);

RemoveDup(head);

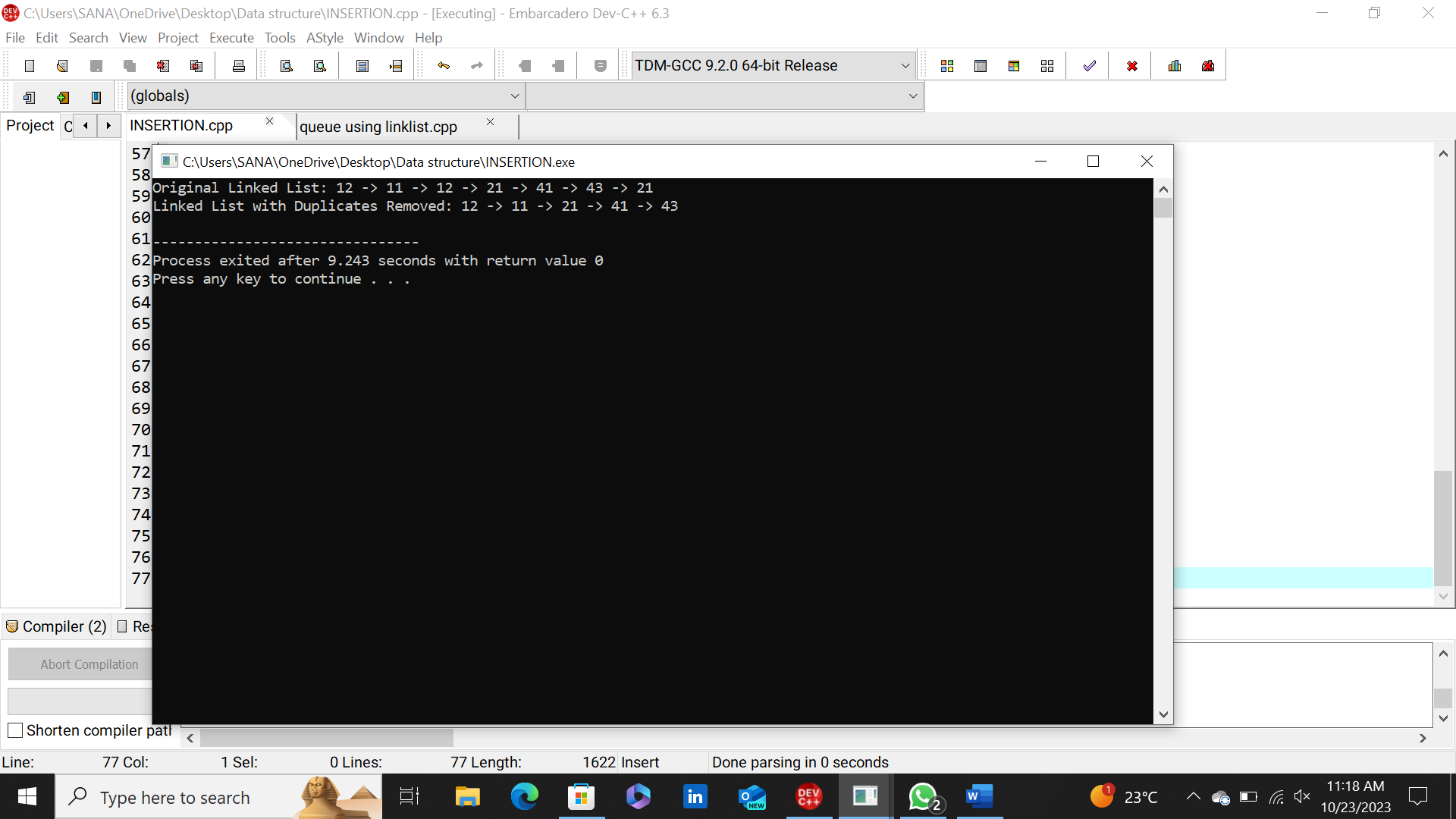
cout << "Linked List with Duplicates Removed: ";

display(head);

return 0;

}

### **Output:**



## 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)**

### **Solution:**

#include <iostream>

using namespace std;

class Node {

public:

int data;

Node\* next;

Node(int val) : data(val), next(nullptr) {}

};

class Queue {

private:

Node\* front;

Node\* rear;

public:

Queue() : front(nullptr), rear(nullptr) {}

// Enqueue a value into the queue

void enqueue(int val) {

Node\* newNode = new Node(val);

if (!front) {

front = rear = newNode;

}

else {

rear->next = newNode;

rear = newNode;

}

cout << "The inserted element in the queue is: " << rear->data << endl;

}

// Dequeue an element from the queue

void dequeue() {

if (!front) {

cout << "Empty queue" << endl;

return;

}

Node\* temp = front;

cout << "Dequeued element is: " << temp->data << endl;

front = front->next;

delete temp;

}

// Display the elements in the queue

void display() {

cout << "Queue elements are: ";

Node\* temp = front;

while (temp) {

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

temp = temp->next;

}

cout << endl;

}

};

int main() {

Queue myQueue;

myQueue.enqueue(1);

myQueue.enqueue(2);

myQueue.display();

myQueue.dequeue();

myQueue.display();

myQueue.enqueue(3);

myQueue.enqueue(4);

myQueue.display();

myQueue.dequeue();

myQueue.dequeue();

myQueue.dequeue();

myQueue.dequeue();

myQueue.display();

return 0;

}

### **Output:**

