|  |  |
| --- | --- |
| 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:**

**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)**

Solutions:

**Activity No 01:**

**Code:**

**#include<iostream>**

**using namespace std;**

**class Node{**

**private:**

**int data;**

**Node \*next;**

**public:**

**Node \*head;**

**Node(){**

**head=NULL;**

**}**

**void insert\_beg(int n){**

**if(head==NULL){**

**head=new Node();**

**head->data=n;**

**head->next=NULL;**

**}**

**else{**

**Node \*ptr;**

**ptr=new Node();**

**ptr->next=head;**

**ptr->data=n;**

**head=ptr;**

**}**

**}**

**void insert\_end(int n){**

**if(head==NULL){**

**head=new Node;**

**head->data=n;**

**head->next=NULL;**

**}**

**else{**

**Node \*ptr, \*p;**

**ptr=head;**

**while(ptr->next!=NULL){**

**ptr=ptr->next;**

**}**

**p= new Node();**

**p->data=n;**

**p->next=NULL;**

**ptr->next=p;**

**}**

**}**

**void del\_beg(){**

**if(head==NULL){**

**cout<<"List is empty"<<endl;**

**}**

**else{**

**Node \*ptr;**

**ptr=head;**

**head=ptr->next;**

**delete ptr;**

**ptr=NULL;**

**}**

**}**

**void del\_end(){**

**if(head==NULL){**

**cout<<"list is empty"<<endl;**

**}**

**else{**

**Node \*p1,\*p2;**

**p1=head;**

**while(p1->next!=NULL){**

**p2=p1;**

**p1=p1->next;**

**}**

**p2->next=NULL;**

**delete p1;**

**p1=NULL;**

**}**

**}**

**void display(){**

**if(head==NULL){**

**cout<<"There is no list "<<endl;**

**}**

**else{**

**Node \*ptr;**

**ptr=head;**

**cout<<"The linked list is: "<<endl;**

**while(ptr!=NULL){**

**cout<<ptr->data<<" ";**

**ptr=ptr->next;**

**}**

**cout<<endl;**

**}**

**}**

**void remove\_duplicates() {**

**if (head == NULL || head->next == NULL) {**

**cout<<"the list is empty or there is only one element"<<endl;**

**return;**

**}**

**Node\* current = head;**

**while (current != NULL) {**

**Node\* runner = current;**

**while (runner->next != NULL) {**

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

**// Duplicate element found, remove it**

**Node\* temp = runner->next;**

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

**delete temp;**

**} else {**

**runner = runner->next;**

**}**

**}**

**current = current->next;**

**}**

**}**

**};**

**int main(){**

**Node n;**

**n.insert\_beg(1);**

**n.insert\_beg(2);**

**n.insert\_beg(3);**

**n.insert\_beg(2);**

**n.insert\_beg(1);**

**n.display();**

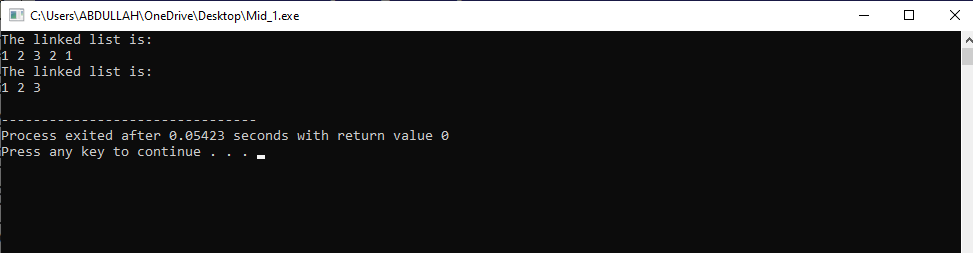
**n.remove\_duplicates();**

**n.display();**

**return 0;**

**}**

Output:



**Activity No 02:**

**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{

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

cout<<front->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 n;

n.enqueue(1);

n.enqueue(2);

n.enqueue(3);

n.dequeue();

n.display();

return 0;

}

Output:

