



## 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-092**

**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.

A screenshot of a terminal window with a black background and yellow text. It shows two lines of output: "Original Linked List: 1 2 3 2 4 1 1" and "Linked List with Duplicates Removed: 1 2 3 4".

```
Original Linked List: 1 2 3 2 4 1 1
Linked List with Duplicates Removed: 1 2 3 4
```

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.

### **Program**

```
#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;
        }
    }
    current = current->next;
}
}

```

```

        } 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

The screenshot displays a C++ IDE with the following components:

- Source Code (Mid\_1.cpp):**

```

87 cout<<"The linked list is: "<<endl;
88 while(ptr!=NULL){
89     cout<<ptr->data<<" ";
90     ptr=ptr->next;
91 }
92 cout<<endl;
93
94
95
96 void remove_duplicates(Node* head)
97 {
98     if (head == NULL || head->next == NULL)
99         return;
100
101     Node* current = head;
102     while (current != NULL)
103     {
104         Node* runner = current->next;
105         while (runner != NULL)

```
- Output Window:**

```

The linked list is:
1 2 3 2 1
The linked list is:
1 2 3
Process exited after 0.18 seconds with return value 0
Press any key to continue . . .

```
- Compiler Window:**

```

Compilation results...
- Errors: 0
- Warnings: 0
- Output Filename: E:\Lab DSA\Mid_1.exe
- Output Size: 1.8339786529541 MiB
- Compilation Time: 2.42s

```

### **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**

```
#include<iostream>
using namespace std;
class Node{
    private:
        int data;
        Node *next;
    public:
        Node *front,*rear=NULL;

    void Enqueue(int n){
        Node *p;
        p=new Node();
        p->data=n;
        p->next=NULL;

        if(front==NULL || rear==NULL){
            front=p;
            rear=p;
            cout<<"The Inserted elements in Queue is : "<<rear->data<<endl;
        }
        else{
            rear->next=p;
            rear=p;
            cout<<"The inserted element in Queue is: "<<rear->data<<endl;
        }
    }

    void Dequeue(){
        Node *ptr;
        ptr=new Node();
        ptr=front;
        if(ptr==NULL)
        {
            cout<<"Empty queue"<<endl;
        }
    }
}
```

```

else{
    if(ptr==NULL)
    { cout<<"The Dequeue elements is: "<<endl;
      cout<<ptr->data;
      front=front->next;
      delete ptr;
      ptr=NULL;
    }

    else{
        cout<<"The Dequeue elements is: ";
        cout<<front->data;
        front=front->next;
        delete ptr;
        ptr=NULL;
    }
}

void Display(){
    Node *temp;
    temp=front;
    cout<<" The Queue Elemwnts are :";
    if(temp==NULL){
        cout<<"The Queue is an Empty : ";
    }
    while(temp!=NULL){
        cout<<temp->data<<" ";
        temp=temp->next;
    }
}

};

int main (){
    Node en;
    en.Enqueue(9);
    en.Enqueue(8);
    en.Enqueue(6);
    en.Enqueue(9);

```

```

    en.Display();
    en.Dequeue();
    en.Display();
    return 0;
}

```

## OUTPUT

The screenshot displays a C++ IDE with the source code for a queue implemented using a linked list. The code defines a `Node` class with `data` and `next` attributes, and an `Enqueue` function that adds elements to the queue. The output window shows the execution results, including the insertion of elements 19, 8, and 9, and the subsequent dequeuing of element 9.

```

1 #include<iostream>
2 using namespace std;
3 class Node{
4     private:
5         int data;
6         Node *next;
7     public:
8         Node *front,*rear=NULL;
9
10    void Enqueue(int n){
11        Node *p;
12        p=new Node();
13        p->data=n;
14        p->next=NULL;
15
16        if(front==NULL ||
17            rear==NULL){
18            front=p;
19            rear=p;
20            cout<<"The Ins

```

Output:

```

The Inserted elements in Queue is :9
The Inserted element in Queue is: 8
The Inserted element in Queue is: 9
The Queue Elements are :9 8 6 9 The Dequeue elements is: 9 The Queue Elements are :8 6 9
.....
Process exited after 0.2426 seconds with return value 0
Press any key to continue . . .

```

Compilation results...

```

- Errors: 0
- Warnings: 0
- Output Filename: E:\LaB DSA\Queue.using.linklist.exe
- Output Size: 1.83443355560303 MiB
- Compilation Time: 2.00s

```

Line 79, Column 14, Selection: 0, Lines: 79, Length: 1228, Insert, Done parsing in 0.015 seconds