**LAB ASSIGNMENT**

Name – Yashwant Patidar

Scholar Number – 191112243

Section – CSE\_2

Ques1. Write a program to create a single linked list and choose the operation to be formed:

i. Insertion:

a) at the front

b) at the end

c) at any other position

ii. Deletion:

a) at the front

b) at the end

c) at any other position

iii. Print the information at each node.

iv. Search for an element.

v. Count number of elements in the list.

🡪

#include<iostream>

#include<stdlib.h>

using namespace std;

class Node {

public :

int key;

int data;

Node \*next;

Node()

{

key = 0;

data = 0;

next = NULL;

}

Node (int k , int d)

{

key = k;

data = d;

}

};

class LinkedList

{

public :

Node \*head;

LinkedList()

{

head = NULL;

}

LinkedList (Node \*n)

{

head = n;

}

Node\* node\_exist (int k)

{

Node\* temp = NULL;

Node\* ptr = head;

while (ptr != NULL)

{

if(ptr->key == k)

{

temp = ptr;

}

ptr = ptr->next;

}

return temp;

}

void append (Node \*n)

{

if(node\_exist(n->key) != NULL)

{

cout<<"Node Already Exist with key : "<<n->key<<" ; Append another Node with a diff key..!!"<<endl;

}

else

{

{

head = n;

cout<<"Node Inserted at the End...!!"<<endl;

}

else

{

Node \*ptr = head;

while (ptr->next != NULL)

{

ptr = ptr->next;

}

ptr->next = n;

cout<<"Node Inserted at the End...!!"<<endl;

}

}

}

void prepend (Node \*n)

{

if(node\_exist(n->key) != NULL)

{

cout<<"Node Already Exist with key : "<<n->key<<" ; Insert another Node with a diff key..!!"<<endl;

}

else

{

n->next = head;

head = n;

cout<<"Node Inserted at the Front...!!"<<endl;

}

}

void insert (int k , Node \*n)

{

Node \*ptr = node\_exist(k);

if( ptr == NULL)

{

cout<<"No Node Exists with key "<<k<<endl;

}

else

{

if (node\_exist(n->key) != NULL)

{

cout<<"Node ALREADY Exists with key : "<<n->key<<" ; Insert Another Node With Diff Key..!!"<<endl;

}

else

{

n->next = ptr->next;

ptr->next = n;

cout<<"Node Inserted..!!"<<endl;

}

}

}

void Delete (int k)

{

if(head == NULL)

{

cout<<"Linked List is EMPTY ; Can't Delete..!!"<<endl;

}

else if (head != NULL)

{

if (head->key == k) //deletion of 1st Node

{

Node \*first = head;

head = head->next;

delete first;

cout<<"Node Deleted with Key : "<<k<<endl;

}

else

{

Node \*temp = NULL;

Node \*prev = head;

Node \*current = head->next;

while (current != NULL)

{

if(current->key == k)

{

temp = current;

current = NULL;

}

else

{

prev = prev->next;

current = current->next;

}

}

if (temp != NULL)

{

prev->next = temp->next;

delete temp;

cout<<"Node Deleted with Key : "<<k<<endl;

}

else

{

cout<<"Node doesn't Exist with key : "<<k<<endl;

}

}

}

}

void search\_node (int k)

{

Node \*ptr = node\_exist(k);

if(ptr != NULL)

{

//ptr is not NULL means the node is found and its ptr returned

cout<<"Node Found...!!"<<endl;

}

else

{

cout<<"Node doesn't Exist with the given key "<<k<<endl;

}

}

void print ()

{

if (head == NULL)

{

cout<<"EMPTY List..!! Can't Print Anything...!!"<<endl;

}

else

{

cout<<endl<<" LINKED LIST : "<<endl;

cout<<" ------------"<<endl;

Node \*temp = head;

while (temp != NULL)

{

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

temp = temp->next;

}

}

cout<<"NULL"<<endl;

}

//to count No. of nodes

void count ()

{

int count = 0;

Node\* ptr = head;

while (ptr != NULL)

{

count++;

ptr= ptr->next;

}

cout<<"No. Of Elements in The List : "<<count<<endl;

}

};

int main ()

{

LinkedList s;

int op , key1 , k1 , data1;

do{

cout<<"\nWhat Operation Do You Wanna Perform ? Select Option . Enter 0 to Exit . "<<endl

<<"1) Insert Node at the Front"<<endl

<<"2) Insert Node at the End"<<endl

<<"3) Insert Node After Any Node "<<endl

<<"4) Delete Node "<<endl

<<"5) Search Node "<<endl

<<"6) Print List "<<endl

<<"7) Count Nodes "<<endl

<<"8) Clear Screen "<<endl;

cin>>op;

Node \*n1 = new Node;

switch (op)

{

case 0 :

cout<<"Exiting..."<<endl;

exit (0);

case 1 :

cout<<"Enter Key & Data of Node To Be Inserted at Front : ";

cin>>key1;

cin>>data1;

n1->key = key1;

n1->data = data1;

s.prepend(n1);

break;

case 2 :

cout<<"Enter Key & Data of Node To Be Inserted at End : ";

cin>>key1>>data1;

n1->key = key1;

n1->data = data1;

s.append(n1);

break;

case 3 :

cout<<"Enter Key of Existing Node After Which You Wanna insert the Node : ";

cin>>k1;

cout<<"Enter Key & Data of New Node : ";

cin>>key1>>data1;

n1->key = key1;

n1->data = data1;

s.insert(k1 , n1);

break;

case 4 :

cout<<"Enter The Key of the Node to Delete : ";

cin>>k1;

s.Delete(k1);

break;

case 5 :

cout<<"Search Node By Key. Enter Key Of Node To Be Searched : ";

cin>>key1;

s.search\_node(key1);

break;

case 6 :

s.print();

cout<<endl;

break;

case 7 :

s.count();

cout<<endl;

break;

case 8 :

system ("cls");

break;

default :

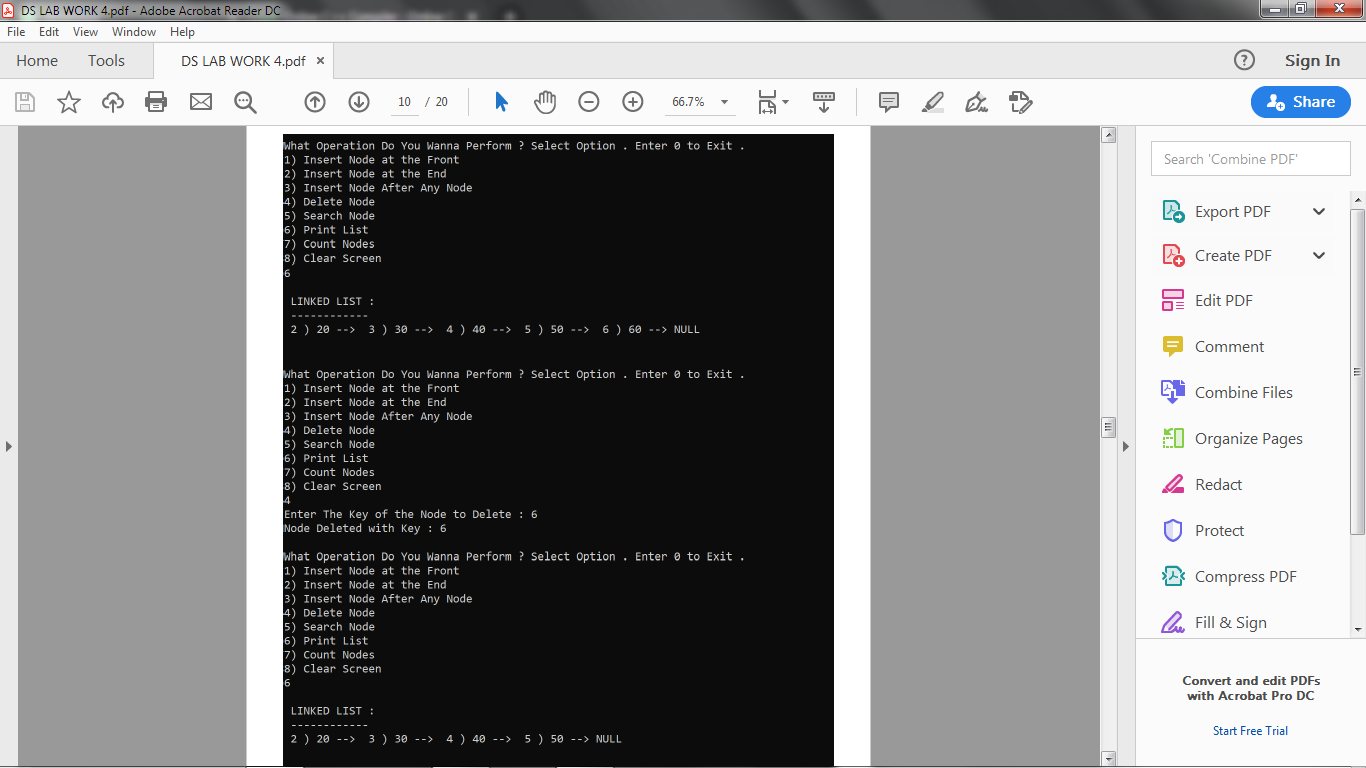
cout<<"Wrong Choice...!!"<<endl;

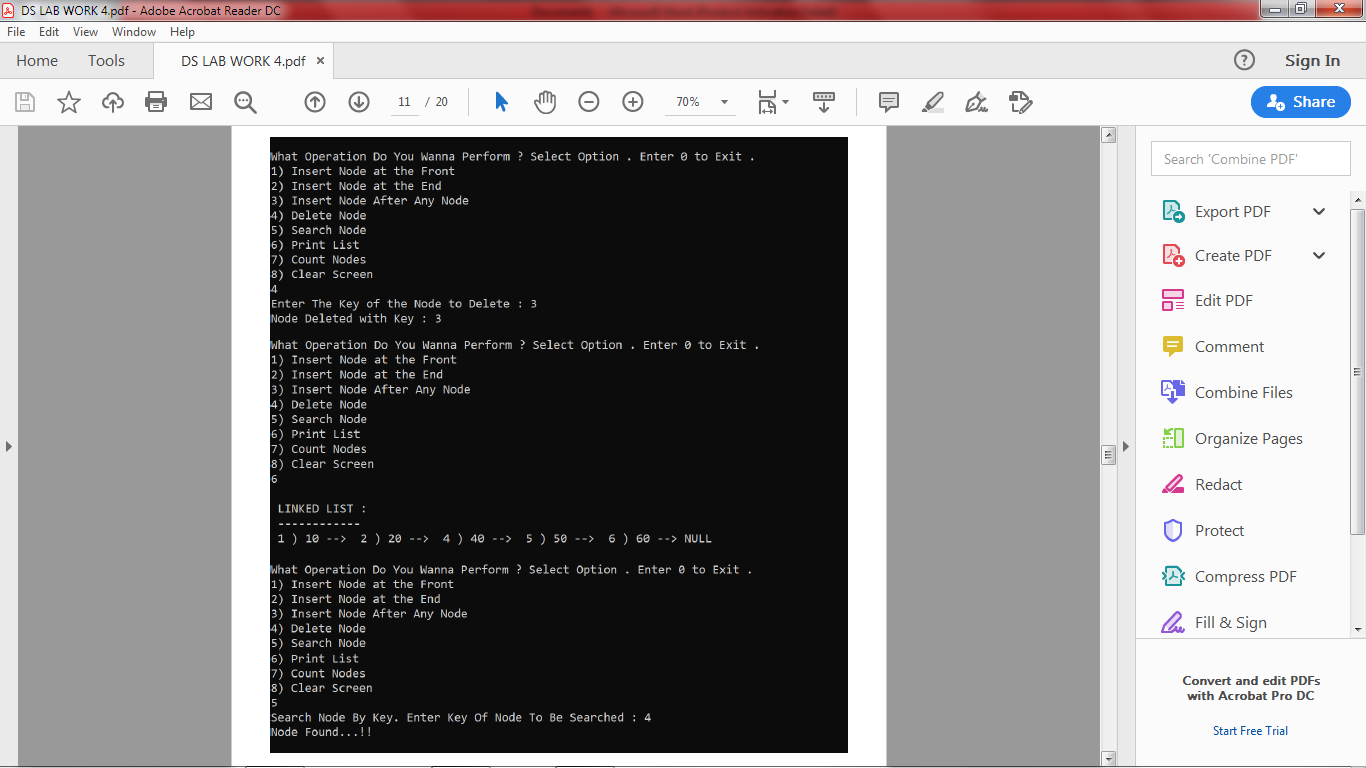
}

} while (op != 0);

return 0;

}





Ques2. Write a program for finding the middle element in the linked list.

🡪

#include<bits/stdc++.h>

using namespace std;

struct node {

int data;

node\* link;

} \*head;

void insert (int data , int n)

{

node\* temp1 = new node;

temp1->data = data;

temp1->link = NULL;

if(n == 1)

{

temp1->link = head;

head = temp1;

return ;

}

node\* temp2 = head; int i ;

for(i=0 ; i<n-2 ; i++)

{

temp2 = temp2->link;

}

temp1->link = temp2->link;

temp2->link = temp1;

}

void print ()

{

node\* ptr = head;

while (ptr != NULL)

{

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

ptr = ptr->link;

}

cout<<"NULL"<<endl;

}

void search\_mid ()

{

struct node \*slow\_ptr = head;

struct node \*fast\_ptr = head;

if (head!=NULL)

{

while (fast\_ptr != NULL && fast\_ptr->link != NULL)

{

fast\_ptr = fast\_ptr->link->link;

slow\_ptr = slow\_ptr->link;

}

cout<<"Middle Element = "<<slow\_ptr->data<<endl;

}

}

int main ()

{

head = NULL;

int n , i ; char ch ;

do {

cout<<"Enter position and Data to Inserted : ";

cin>>i>>n;

insert (n,i);

cout<<"Wanna Enter More (y/n) : ";

cin>>ch;

} while (ch == 'y');

cout<<endl;

cout<<"Linked List : "<<endl

<<"------------- "<<endl;

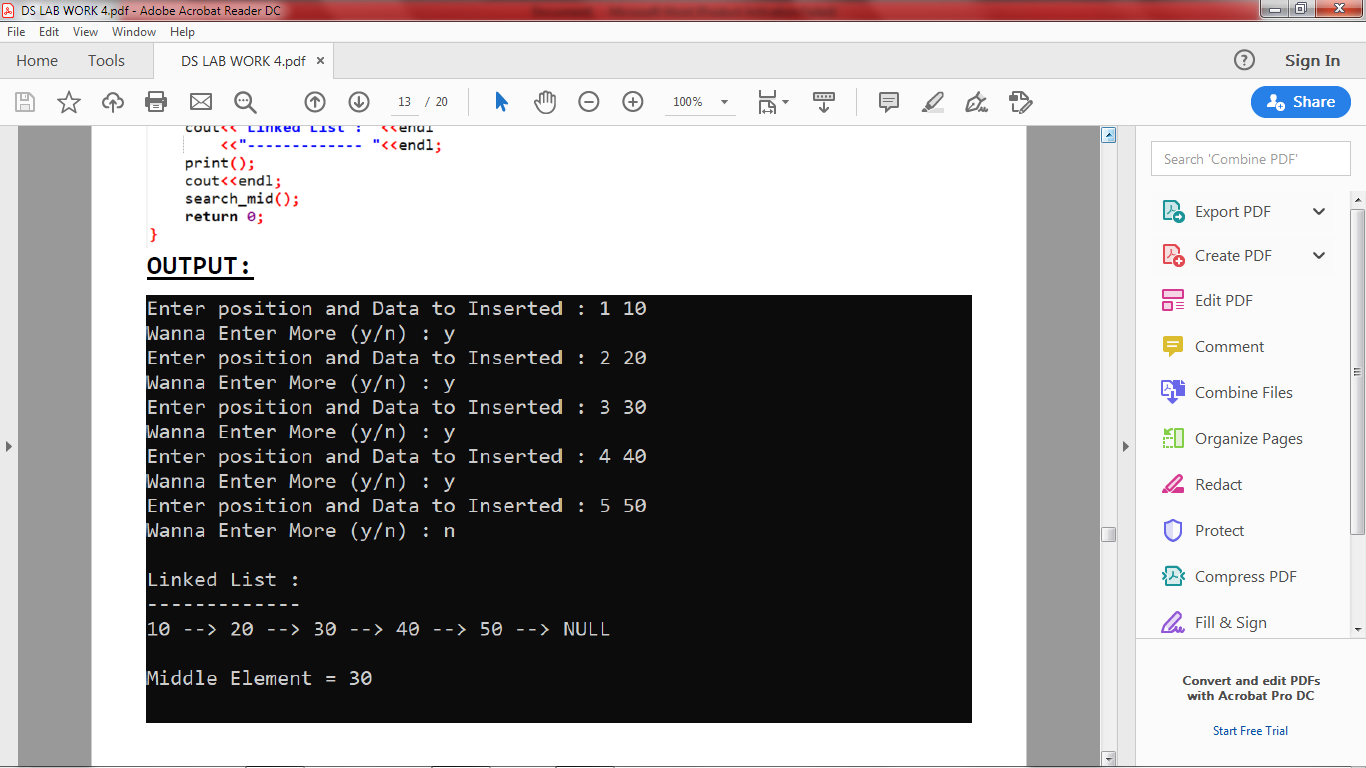
print();

cout<<endl;

search\_mid();

return 0;

}



Ques3. Write a program for reversing the linked list.

🡪

#include<iostream>

#include<stdlib.h>

using namespace std;

struct Node {

int data;

Node \*next;

};

Node\* reverse (Node\* head) //to reverse the list...

{

Node \*current , \*prev , \*next;

current = head;

prev = NULL;

while (current != NULL)

{

next = current->next;

current->next = prev;

prev = current;

current = next;

}

head = prev;

return head;

}

Node\* Insert (Node \*head , int data) //to insert data in the list...

{

Node \*temp = new Node;

temp->data = data;

temp->next = NULL;

if (head == NULL)

head = temp;

else

{

Node \*t = head;

while (t->next != NULL)

t = t->next;

t->next = temp;

}

return head;

}

void print (Node \*head) //to print the list

{

while (head != NULL)

{

cout<<" "<<head->data<<" --> ";

head = head->next;

}

cout<<"NULL"<<endl;

}

int main ()

{

Node \*head = NULL;

int n ; char ch;

do {

cout<<" Enter Data To Fill the List : ";

cin>>n;

head = Insert (head , n);

cout<<endl<<" Wanna Enter More : (y/n) ? ";

cin>>ch;

} while (ch == 'y');

cout<<endl;

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

print (head);

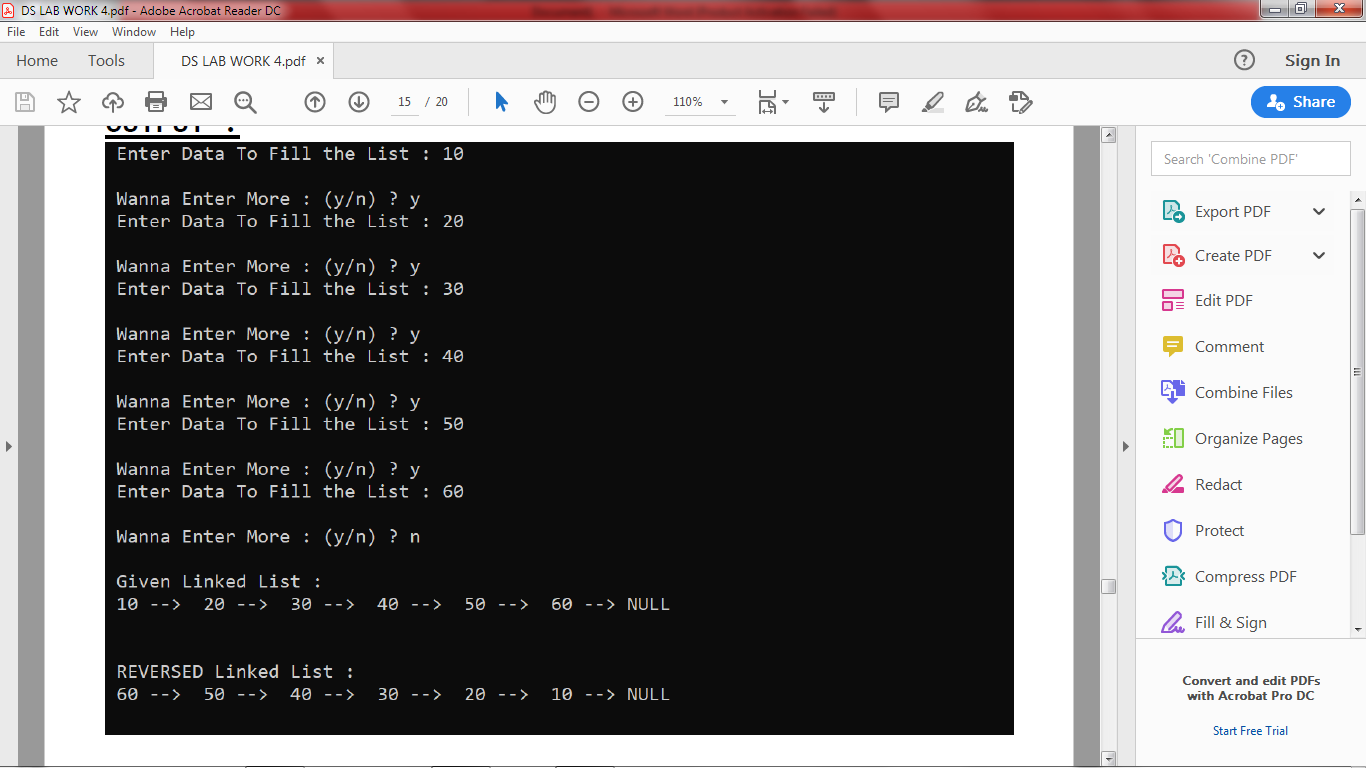
head = reverse (head);

cout<<endl<<endl;

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

print (head);

return 0;



Ques4. Write a program to concatenate two singly linked lists.

🡪

#include<bits/stdc++.h>

using namespace std;

struct Node {

int data;

Node \*next;

};

Node\* Insert (Node \*head , int data) //to insert data in the list...

{

Node \*temp = new Node;

temp->data = data;

temp->next = NULL;

if (head == NULL)

head = temp;

else

{

Node \*t = head;

while (t->next != NULL)

t = t->next;

t->next = temp;

}

return head;

}

void print (Node \*head) //to print the list

{

while (head != NULL)

{

cout<<" "<<head->data<<" --> ";

head = head->next;

}

cout<<"NULL"<<endl;

}

int main ()

{

Node \*head = NULL;

Node \*head\_2 = NULL;

int n , n2 ; char ch , ch2;

do {

cout<<" Enter Data To Fill the List 1 : ";

cin>>n;

head = Insert (head , n);

cout<<endl<<" Wanna Enter More : (y/n) ? ";

cin>>ch;

} while (ch == 'y');

cout<<endl;

do {

cout<<" Enter Data To Fill the List 2 : ";

cin>>n2;

head\_2 = Insert (head\_2 , n2);

cout<<endl<<" Wanna Enter More : (y/n) ? ";

cin>>ch2;

} while (ch2 == 'y');

cout<<endl<<endl;

cout<<" List 1 : "<<endl

<<" -------- "<<endl;

print (head);

Node\* ptr = head;

while (ptr->next != NULL)

ptr = ptr->next;

ptr->next = head\_2;

//delete (head2);

Node\* new\_head = head;

cout<<endl;

cout<<" List 2 : "<<endl

<<" -------- "<<endl;

print (head\_2);

cout<<endl<<endl;

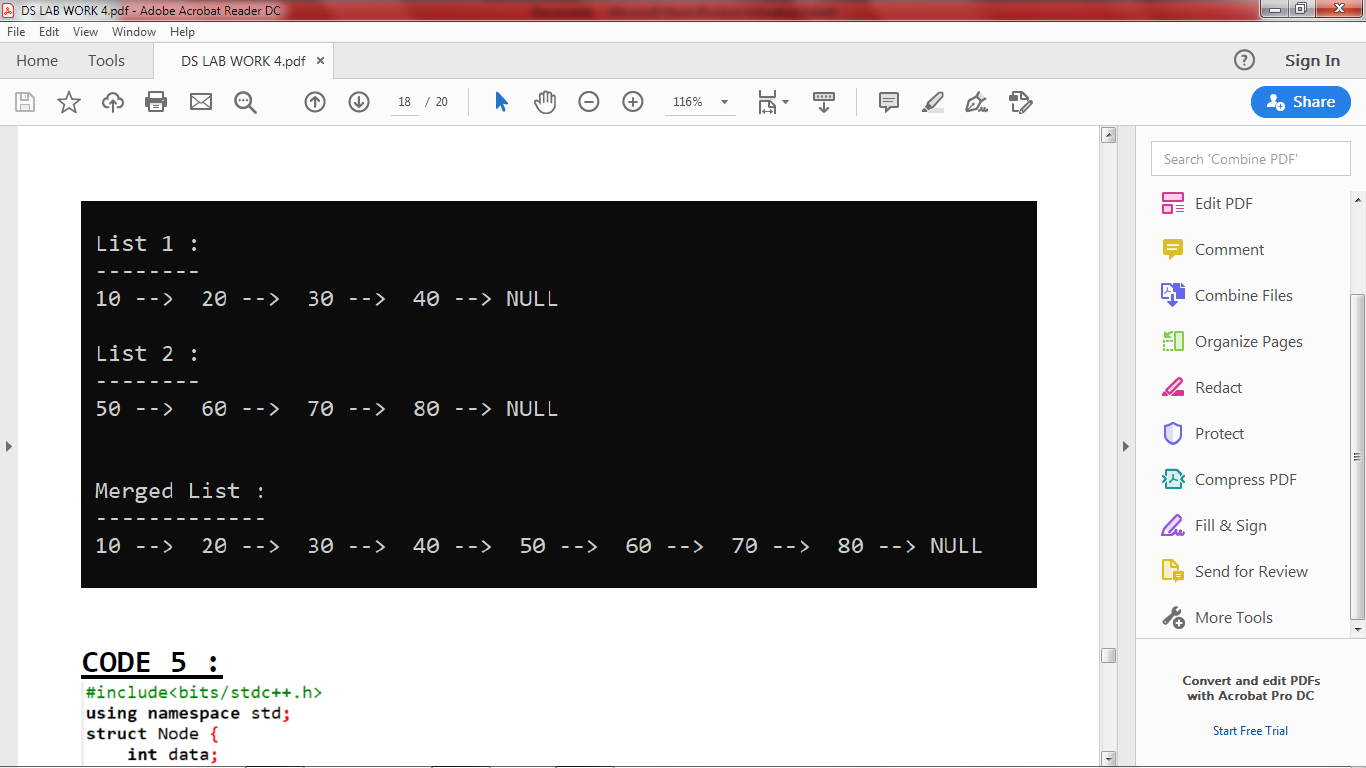
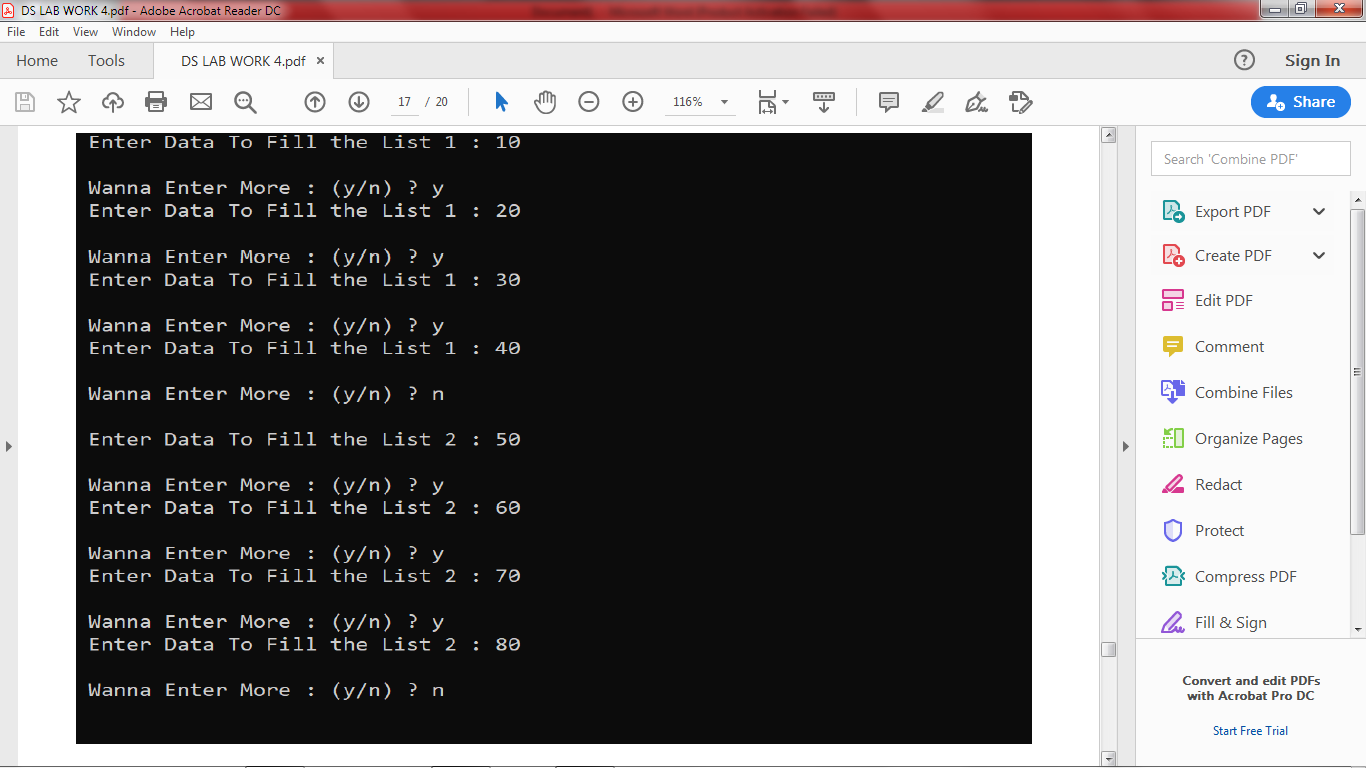
cout<<" Merged List : "<<endl

<<" ------------- "<<endl;

print(new\_head);

return 0

}



Ques5. Write a program to copy a singly linked list into another list by duplicating the content of

each node into the newly allocated node.

🡪

#include<bits/stdc++.h>

using namespace std;

struct Node {

int data;

Node \*next;

};

Node\* Insert (Node \*head , int data) //to insert data in the list...

{

Node \*temp = new Node;

temp->data = data;

temp->next = NULL;

if (head == NULL)

head = temp;

else

{

Node \*t = head;

while ( != NULL)

t = t->next;

t->next = temp;

}

return head;

}

void print (Node \*head) //to print the list

{

while (head != NULL)

{

cout<<" "<<head->data<<" --> ";

head = head->next;

}

cout<<"NULL"<<endl;

}

int main ()

{

Node \*head = NULL;

int n ; char ch;

do {

cout<<" Enter Data To Fill the List : ";

cin>>n;

head = Insert (head , n);

cout<<endl<<" Wanna Enter More : (y/n) ? ";

cin>>ch;

} while (ch == 'y');

cout<<endl;

Node \*ptr = head;

Node \*head2 = new Node;

Node \*ptr1 = head2;

ptr1->data = NULL;

Node \*new\_node = new Node;

while (ptr != NULL)

{

Node \*new\_node = new Node;

new\_node->data = ptr->data;

ptr1->next = new\_node;

new\_node->next = NULL;

ptr1 = new\_node;

ptr = ptr->next;

}

cout<<endl<<endl;

cout<<" Original Linked List 1 : "<<endl

<<" ------------------------ "<<endl;

print(head);

cout<<endl<<endl;

cout<<" Copied Linked List 2 : "<<endl

<<" --------------------- "<<endl;

print(head2->next);

return 0;

}

