**TITLE OF LAB: (IMPLEMENTATION OF LISTS USING LINK LIST)**

**LAB REPORT NO.07**



**Spring 2022**

**CSE-210L Data Structures and Algorithm Lab**

Submitted by

Name:: **Safi Ullah Khan**

Registration No. **20PWCSE1943**

Class Section: **B**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

Student Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Submitted to:

**Dr. Khurram Shehzad Khattak**

(Friday, July 29th, 2022)

**Department of Computer Systems Engineering**

**University of Engineering and Technology, Peshawar**

**OBJECTIVES OF THE LAB**

‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐

In this lab, we will learn about the data structure linked lists and their useful operations.

‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐‐

## **Task 01**

Implement singly linked list with following operations:

1. Traverse
2. Delete
3. Print
4. Add

**Source Code:**

**#include<iostream> using namespace std;**

**struct node**

**{**

**int data;**

**node \*next;**

**}; node \*head;**

**void Insert(int data,int n)**

**{**

**node \*temp1=new node();**

**temp1->data=data;**

**temp1->next=NULL;**

**if(n==1)**

**{**

**temp1->next=head;**

**head=temp1;**

**return;**

**}**

**node \*temp2=head; for(int i=0;i<n-2;i++)**

**{**

**temp2=temp2->next;**

**}**

**temp1->next=temp2->next;**

**temp2->next=temp1;**

**} void Print()**

**{**

**node \*temp=head;**

**while(temp!=NULL)**

**{**

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

**temp=temp->next;**

**}**

**cout<<endl;**

**}**

**void Delete(int n)**

**{**

**node\* temp1=head;**

**if(n==1)**

**{**

**head=temp1->next;//head now points to second node**

**delete temp1;**

**return;**

**}**

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

**{**

**temp1=temp1->next;**

**}**

**node\* temp2=temp1->next;**

**temp1->next=temp2->next;**

**delete temp2;**

**}**

**int main()**

**{**

**head=NULL;**

**int data,nth; int n;**

**int check; do**

**{**

**cout<<"\t\t\tPLease Enter One of Them\n\n\t\t\t1: Insert: \n\t\t\t2: Delete:**

**\n\t\t\t3: Print\n\t\t\t0: For Exit\n"; cin>>check;**

**cout<<"\n\t\t\tRemember Please Enter Integer Type Data as Input\n";**  **switch(check)**

**{**

**case 1:**

**cout<<"Enter Data: ";**   **cin>>data;**

**cout<<"Enter Position: ";**   **cin>>nth; Insert(data,nth);break;**

**case 2:**

**cout<<"Enter Position: ";**

**cin>>n; Delete(n);**

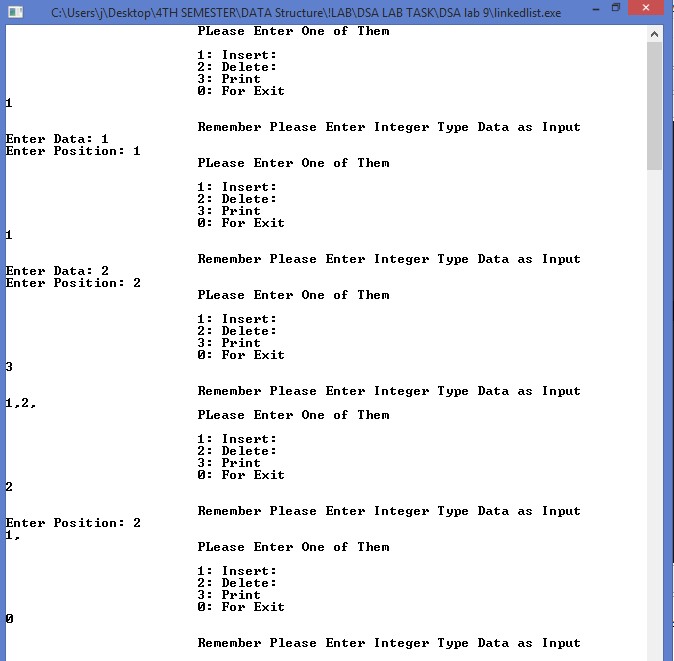
**case 3: Print();**

**}**

**}while(check!=0); cout<<endl;**

**cout<<"I have done my work "; return 0;**

**}**



## **Task 02**

Implement a single linked list with following operations

1. Insert at start of the list (preappend data)
2. Insertion at end (append data)
3. Insertion at nth location
4. Deletion from start of the list
5. Deletion the end of the list
6. Insertion from nth location
7. Search for a key
8. Update list
9. Empty list
10. Calculate size of the list

**Source Code:**

**#include<iostream>**

**#include<stdio.h>**

**#include<stdlib.h> using namespace std;**

**struct node**

**{**

**int data;**

**node \*next;**

**}; node \*head;**

**void Insert(int data,int n)**

**{**

**node \*temp1=new node();**

**temp1->data=data;**

**temp1->next=NULL;**

**if(n==1)**

**{**

**temp1->next=head;**

**head=temp1;**

**return;**

**}**

**node \*temp2=head;**

**{**

**Print();**

case 1:

cout<<”Enter Data:”;

**cin>>data;**

**cout<<"Enter Position: "; cin>>nth;**

**Insert(data,nth);**

**break; case 2:**

**cout<<"Enter Data: "; cin>>data;**

**insert\_at\_start(data);**

**break; case 3:**

**cout<<"Enter Data: "; cin>>data; insert\_at\_end(data);**

**break; case 4:**

**deletestart();**

**break; case 5:**

**deleteEnd();**

**break; case 6:**

**cout<<"Enter Position: ";**

**cin>>n; Delete(n); case 7:**

**{while(check!=0); cout<<endl; cout<<"I have done my work! "; return 0;}**

**Output (Compilation, Debugging & Testing):**

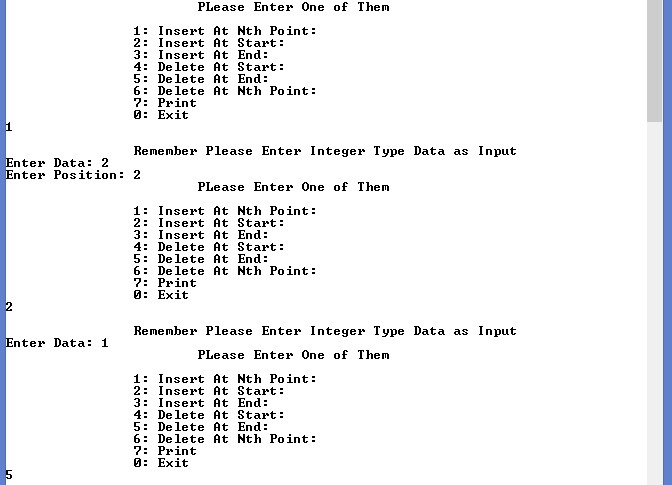
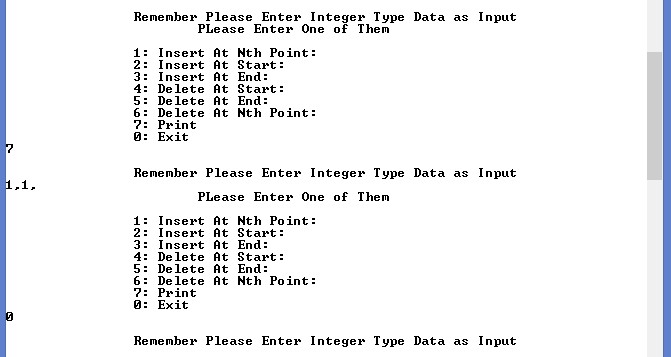
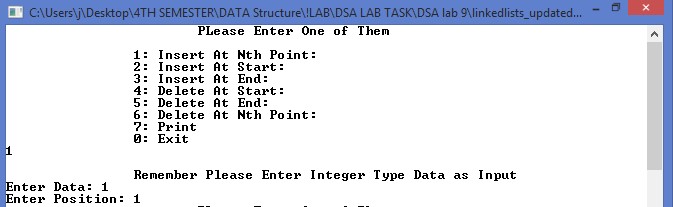
Page

**8**

of

**1**

**5**



## **Task 03**

Implement doubly linked list and perform operations mentioned on it from Task 02

**Source Code:**

**#include<iostream> using namespace std;**

**int insertdata(int x); void display();**

**void deleteint(int x); void reversel();**

**int searchint(int x);**

**int compare\_fn(int a,int b)**

**{ if(a>b)**

**return 1; else if(b>a)**

**return -1;**

**}**

**int compare\_no=1;**

**struct node**

**{**

**int data;**

**node \*prev;**

**node \*next;**

**};**

**node \*top = NULL;**

**int main()**

**{**

**int ch,d,y;**

**char ans='y'; while(ans=='y')**

**{ cout<<"\n\t 1.Insert 2. Delete 3.Reverse 4.EXIT\nEnter Choice : ";**

**cin>>ch; if(ch==1)**

**{**

**cout<<"Enter An Element To be inserted : "; cin>>d;**

**d=insertdata(d);**

**display();**

**}**

**else if(ch==2)**

**{**

**cout<<"Enter Element To Be Deleted : ";**

**node \*b=insertele->next; node \*N =insertele;**

**n->prev=insertele; n->next=b; insertele->next=n;**

**if(b!=NULL)**

**b->prev=n;**

**}**

**} void display()**

**{ cout<<"Element In The Linked List Are : ";**

**node \*disp=top;**

**while(disp!=NULL)**

**{**

**cout<<" "<<disp->data; if(disp->next==NULL)**

**{**

**break;**

**} disp=disp->next;**

**}**

**}**

**void deleteint(int x)**

**{**

**node \*del=top;**

**if(del->data == x)**

**{**

**if(del->next==NULL && del->prev==NULL)**

**{**

**top=NULL; return;**

**}**

**del->next->prev=NULL;**

**top=del->next;**

**} else {**

**node \*delsuc=del->next;**

**if(del==NULL){**

**cout<<"\nElement Not Found\n"; return;**

**}**

**if(delsuc==NULL) {**

**cout<<"\nElement Not Found\n";**   **return; }**

**while(delsuc->data != x)**

**{ del=del->next; delsuc=delsuc->next;**

**if(del==NULL)**

**deleteint(d);**

**display();**

**}**

**else if(ch==3) reversel(); else**

**return 0;**

**}**

**return 0;**

**}**

**int searchint(int x)**

**{**

**int count=0; node \*searchele=top;**

**while( searchele!=NULL)**

**{**

**if(compare\_fn(x,searchele->data)==compare\_no)**

**{**

**searchele=searchele->next;**

**count+=1; }**

**else**

**break; }**

**return count; }**

**int insertdata(int x)**

**{**

**if(top==NULL)**

**{**

**top=new node; top->data=x; top->next=NULL;**

**top->prev=NULL;**

**}**

**else if(compare\_fn(top->data ,x)==compare\_no)**

**{**

**node \*n=new node;**

**n->data=x; n->next=top; n->prev=NULL; top->prev=n;**

**top=n;}**

**else**

**{**

**int c=searchint(x);**

**node \*insertele=top;**

**for(int i=0;i<c-1;i++)**

**insertele=insertele->next;**

**node \*n=new node;**

**n->data=x;**

**{ cout<<"\nElement Not Found\n";**

**return;**

**}**   **if(delsuc==NULL)**

**{**   **cout<<"\nElement Not Found\n"; return;**

**}**

**}**

**del->next=delsuc->next;**

**if(delsuc->next!=NULL)**

**delsuc->next->prev=del;**

**}**

**}**

**void reversel()**

**{**

**node \*a=top;**

**node \*b,\*c,\*d;**

**while(a!=NULL)**

**{**

**d=a;**

**c=a->next;**

**b=a->prev;**

**a->prev=a->next;**

**a->next=b;**

**a=c;**

**}**

**top=d;**

**cout<<"After Reversing the linked list";**

**display();**

**compare\_no\*=-1;**

**}**

