Assignment : Lab Report

Course Code : CSE-1316

Course Title : Data Structure

Submitted to

Debojyoti Biswas

Lecture

Department of Computer Science and Engineering

Leading University, Sylhet

Submitted by

Name : Naeem khan

Student ID : 2012020105

Section : 4C

Batch : 53

Date of Submisson : 27/06/21

Tasks -1 :

Title :

Write a program that uses functions to perform the following operations on singly linked list.
i) Creation ii) Insertion iii) Deletion iv) Traversal.

problem Analysis :

This program will be perform the following operation on singly linked list :

(i) Creation
(ii) Insertion
(iii) Deletion
(iv) Traversal

Based on problem, it is required to set the input of value, data, position. The program will need a value other than -1 to create the node. To perform the insertion and deletion, the program will need to take input the position of the singly linked list. In addition, the program will also need the value insert,

while traversing, it will display the node data. So, the input is value, position and data. Node data will be displayed, it should be the output variable.

Input variable	processing variable/calculation	output variables	necessary header files/ functions/macros
value(int) data(int) position(int)	sizeof, malloc	node data (int)	stdio.h stdlib.h free() scanf() & printf() for formatted i/o creation(), insertion() Deletion(), traversal()

problem :-

```
# include <stdio.h>
# include<stdlib.h>

typedef struct node *
{   int node;
    struct node * next;
} node;
```

```
Node *head = NULL, *tail = NULL;

int count = 0;
Node * creation()
{ int value;
printf("In Enter some integer value for create
      a linked list and (-1) for stop Linked list\n");
scanf("%d", &value);
      head = NULL;
      tail = NULL;
while (value != -1)
  { Node * new_node = (node*)malloc(sizeof(node));
    New_node -> data = value;
    new_node -> next = NULL;
  if (head == NULL)
    { head = new_node;
      tail = new_node; }
  else { tail -> next = new_node;
      tail = new_node;
    } scanf("%d", &value);
  } return head;
  }
```

```
int totalNode (Node *q)
  { while(q != NULL)
  { count++;
    q = q -> next;
  } return count;
  }
Node *Insertion()
  { Node * avail;
    int i=1, position=1;
printf("\n Enter the position where we want to
    insert Node = ");
scanf("%d", & position);
if (position <0 || position > count)
 } printf("\n Invalid position ");
 }
else if (position == 1)
{  Node * new_node = (Node*) malloc(sizeof(Node));
 printf("\n Enter New node value = ");
  scanf("%d", & new_node -> data);
```

```
new_node -> next = head;
    head = new_node;
    Count++;
  }
  else
    { Node * new node = (Node*) malloc (sizeof (node));
      avail = head;
    while (i< position-1)
      { avail = avail -> next
          i++;
        }
    printf ("\n Enter new node value =");
    Scanf("%d", & new-node -> data);
      avail -> next = avail -> next;
       avail -> next = new_node;
        Count++; }
        return head; }
  Node *Deletion ()
  {
    Node *avail, * next_node;
    int i=1, position = 0;
```

```
printf("\n Enter the position where we want to
        remove node = 4);
Scanf ("%d", & position);
        avail = head;
  if( position < 0 || position > count)
   { printf("\n Invalid position in");}
else { while (i< position -1)
   { avail = avail -> next;
     i++; }
  next_node = avail -> next
   avail-> next = next_node ->next;
   free (nen_node); } return head; }
void p(Node *q)
 {while (q != NULL)
 { printf("%d, q-> data);
     q = q-> next; }}
 int main()
 { Node * i = creation ();
    totalNode (i);
   if (i == NULL)
   { printf("\n Link is NULL \n"); }
else { i = Insertion ();
     P(i);
   i = Deletion ();
     P(i); }
 return 0; }
```

Task no 2

Title :-

Write a program that uses function to perform the following operations on doubly linked list.

i) Creation ii) Insertion (iii) Deletion iv) Traversal.

problem ~~Ana~~ Analysis :-

This program will be perform the following operation on doubly linked list :

i) Creation ii) Insertion (iii) Deletion iv) Traversal.

Based on problem, it is required to get the input of value other then -1 to create the node. To perform the insertion and deletion, the program will need to take input the position, ~~the program will~~ of the doubly linked list. In addition, the program will also need the ~~take input~~ value insert, while traversing, it will display the node data.

So, the input is value, position and data,

Node data will be displayed, it should be the output variable.

Input Variable	processing variable/ function	output variables	necessary header files / function / macros
Value(int) data (int) position(int)	sizeof malloc	Node data (int)	stdio.h stdlib.h free () Scanf() & printf() for Formatted i/o. creation () Traversal() Insertion () totalNode() Deletion ()

problem :

```
# include <stdio.h>
# include < stdio.h>
  typedef stdlib.h>
  typedef struct node
  { int data ;
    Struct node *prev ;
    Struct node *Next ; } Node ;
  Node *head = NULL, *tail = NULL ;
  Node *creation ()
  { int value ;
```

```
printf("\nEnter Some integer value for create a
    linked list and (-1) for Stop linked list.\n");
Scanf("%d", &value);
while (value != -1)
{ Node *NewNode = (Node*) malloc(sizeof(Node));
   NewNode -> data = value;
   NewNode -> prev = NULL;
   newNode -> next = NULL;
if ( head == = NULL)
     { head = newNode;
       newNode -> prev = tail;
       tail = newNode; }
   Scanf ("%d", &value); }
       return head; }
int countNode = 0;
 int Count = 0;
 Node *total Node (Node *a)
{ a = head
   ~~while (a)~~ while (a != NULL) { count++;
     a = a -> next; }
printf("\n Total Node in the list = %d", Count); }
  Node *Insertion ()
```

~~printf("\n Total Node in the list = %d", co~~

```
{ int position;
  printf("\n Enter the position where we want
          to insert new node = ");
  scanf("%d", &position);
  Node *new_node = (Node*)malloc(sizeof(Node));
  printf("\n Enter node value = ");
  scanf("%d", &new_node->data);
    new_node->prev = NULL;
    new_node->next = NULL;
      Node *temp;
      temp = head;
  if(position<0 || position > count)
    { printf("\n Invalid position"); }
  else if (position == 1)
      { if (head == NULL)
        { head = tail = new_node;
            count++; }
  else { int i=1;
      while (i< position-1)
        { temp = temp->next;
            i++; }
    new_node->prev = temp;
    new_node->next = temp->next;
```

```
temp->next = new_node;
new_node->next->prev = new_node; }
    return head; }
Node *Deletion()
    {int position, i=1
    Node *temp;
 print("\n Enter the position where we want
        to delete = ");
 Scanf( "%d", &position)
     temp = head;
 while( i<position)
        { temp = temp->next;
            i++; }
   temp->prev->next = temp->next;
   temp->next->prev = temp->prev;
      Free(temp); }
void P(Node *a)
   { a = head;
   while (a != NULL)
       { print ("%d ", a->data);
           a = a->next; }
   }
   int main()
   { Node *h = creation();
      h = total Node(h)
   if (h == NULL)
```

```
} printf("\n The list is empty");}
else { h = Insertion();
       p(h);
       h = Deletion();
       p(h);
     }
return 0;
}
```

Tasks-3

Title :-

Write a program that uses function to perform the following operations on Circular linked List.

i) Creation ii) Insertion iii) Deletion iv) Traversal.

problem Analysis :

This program will be perform the following operation on circular linked list.

i) Creation ii) Insertion iii) Deletion iv) Traversal Based on program, it is required to get the input of value, data and position. The program will need a value other than -1 to create the node. To ~~pre~~ perform the insertion and deletion, the program will need do take input the position of the Circular linked list. In addition will also need the value insert, while traversing, it will display the node data.

Input variable	processing variables/calculation	output variable	Necessary header file/ functions/macros
value(int) Data(int) position(int)	sizeof, malloc	node data (int)	stdio.h stdlib.h Create List () total Node () Insert Given position Deletion (), Traversal ()

problem:

```
# include <stdio.h>
# include < stdlib.h>
  typedef struct node
  { int data;
    struct node *next;
    } Node;
  Node *head = NULL, *tail = NULL;
    Node * createList ()
  { int value;
  printf("\n Enter some integer and (-1) for stop
                linked list \n");
   Scanf("%d", & value);
    while (value != -1)
```

```
{ Node * new_node = (Node*) malloc (sizeof (Node));
    new_node -> data = value;
    new_nod -> next = NULL;
    if ( head == NULL)
      { head = tail = new_node; }
else
    { tail -> next = new_node;
      tail = new_node; }
    tail -> next = head;
    scanf ("%d", & value); }
        return head; }
int count = 0;
Node * totalNode()
{ Node * avail;
    avail = head;
while (avail -> next != head)
    { count++;
      printf ("\n total Node = %d", count); }
Node *Insert Given position ()
{ Node *new_node = (Node*) malloc (sizeof (Node));
  int position = 0, i = 1;
printf ("\n Enter position where we want to
        insert new node = ");
```

```
scanf("%d", &position);
if(position<0 || position>count)
 { printf("\n Invalid position"); }
else if (position==1);
 {printf("\n Enter the value of Node = ");
   scanf("%d", &new_node->data);
   Scanf("%d", &new_node->data);
      new_node->next = NULL;
   if (tail==NULL)
     { tail = new_node;
       tail=>next = new_node; }
   else { new_node->next = head; }
       tail->next = new_node;
        head = new_node; }
     Count++;
    }
else {printf("\n Enter the value of Node=");
     scanf("%d", &new_node->data);
       new_node->next = NULL;
       Node *temp = tail->next;
     while (i<position-1)
      { temp = temp->next;
          i++; }
```

```
new_node -> next = temp -> next;
    Temp -> next = new_node; }
        return head; }
Node * Deleteposition ()
{ Node * nextNode, * currentNode;
    int position = 0, i = 1;
    currentNode = tail -> next;
    printf ("\n Enter position = ");
    Scanf ("%d", & position);
  if (position < 0 || position > count)
   { printf ("\n Invalid position"); }
   else if (position == 1)
      { Node * temp;
        temp = head;
    if (tail == 0)
     { printf ("\n It underflow"); }
    else if (temp -> next = temp)
       { tail = 0;
         free (temp); }
    else
       { temp = head;
         head = head -> next;
```

```
tail -> next = head;
    free (temp); }
  }
else
    { while (i< position -1)
    { currentNode = currentNode -> next;
        i++; }
     nextNode = currentNode -> next;
    current Node -> next = next Node -> Next;
      free (nextNode); }
        return head; }
void Traverse (Node *avail)
      { if (head == 0)
        {printf ("\n The data is off "); }
  else
    { avail = head;
     while (avail -> next != head)
    { printf ("%d ", avail -> data);
          avail = avail -> next;
      printf ("%d", avail -> data); }
        }
```

```
int main()
{ node *y = createList();
   y = totalNode();
   y = InsertGivenposition();
   Traverse(y);
  y = Delete position();
   Traverse(y);
    return 0;
   }
```

Tasks 4(i):

Title: Write a program that implement stack using Array.

problem Analysis: ~~This pro~~

This program will be perform the fallowing operation

(i) Insertion
(ii) Deletion
(iii) Traversal

Based on the problem, it is required to get input of value and choice. The program should display the stack ~~and~~ top. The input variable ~~is~~ should be value and choice.

Input Variable	processing variable/calculation	output variables	necessary header files/ functions/ macros
value(int) Choice(int)	Top(int)	Top(int)	stdio.h stdlib.h push(), pop() display(), scanf & printf formatted i/o.

problem :-

```
#include<stdio.h>
#include<stdlib.h>
#define N 10
int stack[N];
int top = -1;
void push()
{ int value;
  printf("\n\t Enter value = ");
    scanf("%d", &value);
  if(top == N-1)
    { printf("\n\t overflow");
    }
  else { top++;
    stack[top] = value;
    }
  }
void pop()
    {int item;
     if(top==-1)
    { printf("\n\t underflow");
    }
```

```
else { item = stack[top];
       top--;
       printf("\n\t popped item = %d", item);
     }
}
void display ()
{ int i;
  printf("\n Enter stack elements = ");
     For(i=top; i>0; i--)
     { printf("%d", stack[i]); }
     }
int main()
{ int choice;
  printf("\n 1- Insert Stack");
  printf("\n 2 - Delete Stack");
  printf("\n 3- Display ");
  printf("\n 4- Exit\n");
while (1)
   { printf("\n Enter your choice = ");
     Scanf("%d", &choice);
    Switch (choice)
```

```
else { item = stack [top];
       top--;
printf ("\n\t popped item = %d ", item);
     }
Void display ()
  {int i;
   printf ("\n Enter stack elements = ");
       for ( i = top; i>=0; i--)
         { printf ("%d ", stack[i]); }
       }
int main()
   { int choice;
     printf ("\n 1- Insert stack ");
     printf ("\n 2- Delete stack ");
     printf ("\n 3 - Display ");
     printf ("\n 4 - Exit\n ");
while (1)
   { printf ("\n Enter your choice = ");
     scanf ("%d", &choice);
      switch (choice)
```

```
2 case 1:
        push();
            break;
    case 2:
        pop();
            break;
    case 3:
            display();
                break;
    case 4:
        printf("The program is over\n");
                Exit(0);
                break;
    default:
        printf("Invalid choice, please try
                again\n"); }
    }
        return 0;
    }
```

Tasks-4.(ii):

Title :- Write a program that implement stack (its operation using Linked List (pointers).

problem Analysis :

This program will be perform the following operations.

(i) Insertion.
(ii) Deletion.
(iii) Traversal.

Base on program, it is required to get input of value and choice. The program display the Stack Top. The input variable should be Value and Choice.

Input variable	processing variable Calculation	output variables	necessary header files/ function/macros
value(int) choice(int)	Top(int) sizeof, malloc	Top(int)	stdio.h stdlib.h push(), pop() display(), scanf() & printf formatted i/o.

problem:

```
# include <stdio.h>
# include <stdlib.h>
typedef struct node
    {int node;
     struct node * Link;
    } Node;
Node * top = 0;
void posh()
    {int value;
      print ("\n\t Enter value = ");
      Scanf ("%d", &value);
Node *new_node = (Node*) malloc (sizeof (Node));
new_node -> data = value;
new_node -> link = top;
top = new_node; }
void pop ()
    { Node * temp;
      temp = top
     if ( top == 0)
     { printf ("\n\t underflow");
       }
```

```
else
    {printf("\n\t popped data = %d ", top->data);

        top = top -> link;
        free (temp); }}
Void display ()
    { Node *a;
        a = top;
if (top == 0)
    { printf("\n\t Stack empty "); }
else { print("\n Enter Stack element = ");
    While (a != 0)
        {printf("%d ", a -> data);
            a = a -> link; }}}
int main ()
    {int choice;
    printf("\n 1- Insert Stack ");
    printf("\n 2- Delete stack ");
    printf("\n 3 - Display ");
    printf("\n 4 - Exit\n");
```

```
while (1)
{
    printf ("\n Enter your choice :");
    scanf ("%d", &choice);
    switch (choice)
    {
        case 1:
            push ();
            break;
        case 2:
            pop ();
            break;
        case 3:
            display ();
            break;
        case 4:
            printf ("The program is over\n");
            Exit (0);
            break;
        default:
            printf ("Invalid choice, please
                    try again.\n"); }
    }
    return 0;
}
```

Tasks - 5(i)

Title :- Write a program that implement Queue (its operations) using Array.

problem Analysis :-

This program will be perform the following operation

(i) Insertion.
(ii) Deletion.
(iii) Traversal.

Based on the problem, it is required to get input of value and choice. The program should display the Queue Front and rear the input variable should be value and choice.

Input variable	processing variable calculation	output variable	necessary header files/ function/macros
value(int) choice(int)	Front(int) rear(int)	Front(int) rear(int)	stdio.h stdlib.h push(), pop(), display(), scent() & printf() formatted i/o.

problem :

```
# include <stdio.h>
# include <stdlib.h>
# define N 10
int queue[N];
  int front = -1
  int rare = -1
 void push ()
   {int value;
   printf("\n\t Enter value = ");
    scanf("%d", & value);
    if (rare == N-1)
        { printf("\n overflow "); }
  else if (front == -1 && rare == -1)
        {front = rare = 0;
          queue[rare] = value; }}
void pop()
  { int item;
   if (front == -1 && rare == -1)
    { printf("\n\t underflow"); }
```

```
else { printf ("\n\t popped value queue=%d",
                    queue[front]);
       front++ }}
void display ()
   { int i;
   if (front == -1 && rare == -1)
     { printf ("\n\t queue is Empty"); }
else { printf ("\n Enter stack Eelements = ");
   for ( i=front; i<rare+1; i++)
     { printf ("%d ", queue[i]); }}}
  int main ()
     { int choice;
     printf ("\n 1- Insert Queue");
     printf ("\n 2- Delete Queue");
     printf ("\n 3- Display ");
     printf ("\n 4 - Exit \n");
while (1) {
   printf ("\n Enter your choice = ");
     scanf ("%d", & choice);
     switch (choice)
     {
```

```
case 1:
    push();
    break;
case 2:
    pop();
    break;
case 3:
    display();
    break;
case 4:
    printf("The program is over\n");
    exit(0);
    break;
default:
    printf("Invalid choice, please try again.\n");
    }
  }
  return 0;
}
```

Tasks -5(ii)

Title :- Write a program that implement queue (it's operation) using linked list.

problem Analysis :-

This program will be perform the following operation (i) Insertion, ii) Deletion iii) Traversal
Based on the problem, it is required to get input of the problem, it is required to get input of value and choice. The program should display the queue ~~Rear~~ front rear. The input variable should be value and choice.

Input variables	processing variable/calculation	output variable	necessary header files/ function/ macros
value(int) choice(int)	sizeof, malloc front (int) rear (int)	front(int) rear(int)	stdio.h stdlib.h push(), pop() display(), scanf() & printf() formatted i/o.

problem:

```
#include<stdio.h>
#include<stdlib.h>
  typedef struct node
 { int data;
  Struct node *link;
  } Node;
  Node *front=0, *rare=0;
  void push()
  { Node *new_node = (Node*)malloc(sizeof(Node));
  printf("\n\t Enter value = ");
   Scanf("%d", &new_node->data);
     new_node->link = NULL;
   if (front=0 && rare==0)
     { front = rare = new_node; }
     else
      { rare-> link = new_node;
        rare = new_node; }
      }
   void pop()
  { Node * temp;
```

```
if (front == 0 && rare == 0)
  { printf ("\n\t underflow");
  else
    { printf ("%d", front -> data);
      front = front -> link;
      free (temp); }
    }
Void display ()
  { Node *temp;
    if (front == 0 && rare == 0)
    { print ("\n\t overflow"); }
  else
    { temp = front;
      while (temp != 0)
      { printf ("%d", temp -> data);
        temp = temp -> link; }
      }
int main ()
  { int choice;
      printf ("\n\t 1- Insert Queue");
      printf ("\n\t 2- Delete Queue");
      printf ("\n\t 3 = Display");
      printf ("\n 4 - Exit \n");
```

```
while(1){
    printf("\n Enter choice = ");
    Scanf("%d", &choice);
    switch (choice)
    { case 1:
            push();
            break;
      case 2:
            pop();
            break;
      case 3:
            display();
            break;
      case 4:
            printf("The program is over \n");

            Exit(0);
            break;
    default:
      printf("Invalid choice, please try
                                again \n")
                                    }
    }return 0;
    }
```

Tasks-6(i) :

Title :

Wright a program that implement circular queue using arrays.

problem Analysis :

This program will be perform the fallowing operations

i) Insert ii) Delete iii) Traverse.

Based on the problem. it is required to get input of choice and item. the program should display the queue front and rare. The input variable should be item and choice.

Input variable	processing variable ~~&~~ calculation	output variables	necessary header files / function / ~~unformatted~~ macros
choice(int) Item(int)	front (int) rare (int)	~~structure~~ [] ~~(int)~~ enqueue- con[] (int)	stdio.h stdlib.h display() insert() delete(), Scanf() printf formatted i/o.

problem :-

```
# include <stdio.h>
# include <stdlib.h>
# define max 10
  int equeue_arr[MAX];
  int front = -1;
  int rear = -1;
void display ();
  void insert (int item);
   int del ();
int main ()
 { int choice, item;
 while (1)
     { printf ("\n 1. Insert \n");
       printf (" 2. Delete \n");
       printf (" 3. Display");
       printf (" 4. Exit \n");
printf ("\n Enter your choice : ");
 scanf ("%d", &choice);
```

```
switch (choice)
    { case 1 :
        printf("\n Enter the element for
                        insertion: ");
      scanf ("%d", &item);
        insert (item);
        break;
      case 2:
        printf("\n Element deleted is : %d", del(d));
          break;
      case 3:
          display ();
          break;
      case 4 :
          exit (0);
    default:
        printf ("\n Wrong choice \n");
      }
      return 0;
      }
void insert (int item)
  { if ((front == 0 && rear == max-1) ||
    (front == rear+1))
```

```
{ printf("\n queue overflow \n");
    return; }
if (front == -1) front = 0;
if (rear == max-1) rear = 0;
else
    rear = rear+1;
    equeue_arr[rear] = item;
}
int del()
{ int item;
if (front == -1)
    { printf("\n queue underflow \n");
        Exit(1); }
    item = equeue_arr[front];
if (front == rear)
    { front = -1; rear = -1; }
else if (front == max-1) front = 0;
else front = front+1;
    return item; }
```

```
void display ()
{int i;
  if (front == -1)
  { printf ("\n Queue is empty \n");
     return; }
  printf ("\n Queue element : \n");
        i = front;
  if (front <= rear)
     { while (i<=rear)
       {while (i<=rear)
         printf ("%d", equeue_arr[i++]);
         }
else { while (i<=max-1)
       printf ("%d", equeue_arr[i++]);
          i=0;
       while (i<=rear)
             printf ("%d", equeue_arr[i++]);}
       printf ("\n");
       }
```

Tasks 6 (ii) = (a)

Title :- Weight a program that uses both recursive and non-recursive function to perform the following searching operations for key value in a given list of integers : Linear search.

problem Analysis :-

This program will be perform the following operation on the Linear search.

Based on problem, it is required to get the input ~~Analysis~~ data on linear search Analysis uses both recursive and non-recusive.

Input variable	processing variable / calculation	output variables	necessary header files / function / macros
result (int)	~~sizeof(found (int)~~ sizeof, favors main (void)	answer [i]	iostream <bits / stdc++.h> cin >> << cout system()

```
# include < iostream>
# include < bits/stdc++.h>
    using namespace std;
int recursiveLinearSearch(int array[], int key, int size){
      size = size-1;
    if(size <0){
      return -1;
    } else if (array[size] == key){
      return 1;
    } else{
      return
recursiveLinearSearch(array, key, size);
  }
int main(){
   out << " Enter the size of Array : ";
   int size;
  cin >> size;
 int array[size], key, i;
 for( int j=0; j<size; j++){
     Cout << " Enter " << j << " Element : ";
     cin >> array[j];
 }
```

```
for(int a=0; a<size; a++){
  cout << "array[" << a << "]=";
  cout << array[a] << endl;
}
  cout << " Enter key To Search in Array ";
    cin >> key;
  int result;
result = recursiveLinearSearch(array, key, size--);
    if(result == 1){
    cout << " key Fount in Array ";
} else{
    cout << "key NOT Fount in Array ";
}
return 0;
}
```

```
#include <iostream>
#include <bits/stdc++.h>
using namespace std;
int search (int arr[], int n, int x)
{
  int i;
  for (i=0; i<n; i++)
    if (arr[i] == x)
      return i;
  return -1;
}
int main (void)
{ int arr[] = { 2, 3, 4, 10, 40 };
  int x = 10;
int n = sizeof (arr) / sizeof (arr[0]);
int result = search (arr, n, x);
  result == -1
    ? cout << "Element is not present in
        array"
  : cout << "Element is present at index "
  << result;
  return 0;
}
```