Apex College

BCIS Program

Affiliated to Pokhara University

late
checked
28/6/22

Data Structure & Algorithms
Lab Report
10.
Circular Linked List

Date: __-06-2022

Submitted by:
Ishwor Shrestha
Roll no.: 2018-BCIS-414

Submitted to:
Pravakar Ghimire, &
Anmol Shrestha
Apex College

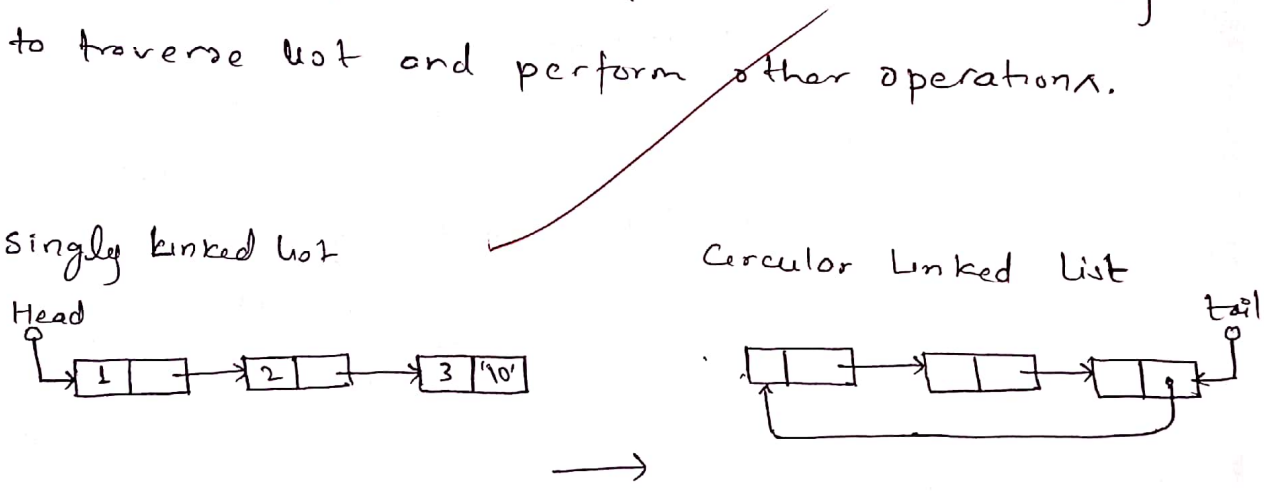
Lab 10 Circular Linked List | Objectives

- To understand about Circular Linked List
- To implement various operations of Circular Linked List.

#Introduction

Circular Linked list is a Linked List where all nodes are connected to form a circle. There is no NULL at end.

In singly linked list, we can traverse whole list to the end, if we need to go back to first or starting nodes, Then we need to repeat the process from start. But by using circular linked list we can link first node to the first node of list, which made easy to traverse list and perform other operations.



A program to perform ~~all~~ different operations in Circular Linked list.

```
#Include <stdio.h>
#include <stdlib.h>  #include <stdlib.h>
struct list{
  int data; struct list *next; };
```

```
typedef struct list node;

void insert_from_beg(node **tail, int item){
    node *temp;
    temp = (node *) malloc(sizeof(node));
    temp->data = item;
    if(*tail == NULL){
        temp->next = temp;
        *tail = temp;
    }else{
        temp->next = (*tail)->next;
        (*tail)->next = temp;
    }
}

void insert_from_end(node **tail, int item){
    node *temp;
    temp = (node *)malloc(sizeof(node));
    temp->data = item;
    if(*tail == NULL){
        temp->next = temp;
        *tail = temp;
    }else{
        temp->next = (*tail)->next;
        (*tail)->next = temp;
        *tail = temp;
    }
}

void insert_from_sp_pos(node **tail, int item, int pos){
    int i;
    node *temp, *temp1;
    temp = (node *) malloc(sizeof(node));
    temp->data = item;
    temp1 = (*tail)->next;
```

```
    for (i=1; i<pos-1; i++){
        temp1 = temp1->next;
    }
    temp->next = temp1->next;
    temp1->next = temp;
}

int del_from_beg (node **tail);
    node *temp;
    int item=-1;
    if (*tail==NULL)
        printf("List is empty.\n");
    else{
        temp = (*tail)->next;
        (*tail)->next = temp->next;
        item = temp->data;
        free(temp);
        }
        return item;
}

int del_from_end (node **tail){
    node *temp, *temp1;
    int item=-1;
    if (*tail==NULL)
        printf("List is empty.\n");
    else if ((*tail)->next==*tail){
        temp = *tail;
        item = temp->data;
        free(temp);
        *tail=NULL;
    }
    else{
        temp1 = (*tail)->next;
```

```
        while (temp->next != *tail)
            temp1 = temp1->next;

        temp1->next = (*tail)->next;
        temp = *tail;
        *tail = temp1;
        itemp = temp->data;
        free(temp);
    }
    return itemp;
}

int del_from_sp_pos(node **tail, int pos){
    int i, itemp=-1;
    node *temp, *temp1;
    temp = (*tail)->next;
    for(i=1; i<pos; i++){
        temp1 = temp;
        temp = temp->next);
    }
    temp1->next = temp->next;
    item = temp->data;
    free(temp);
    return item;
}

void display(node **tail){
    node *temp;
    temp = (*tail)->next;
    printf("The elements in the list:\n");
    do{
        printf("%d", temp->data);
        temp = temp->next;
```

```
    }while (temp!=*tail);
    printf("\n");
  }
}

bool search (node **tail, int item){
    node *current;
    current = (*tail)->next;
    printf("%d", *(current));
    if(*tail != NULL){
      do{
        if(current->data == item)
          return true;
        current = current->next;
      }while(current != *tail);
      return false;
  }

int main(){
    node *tail = NULL;
    int item, choice, pos, item;
    while(1){
      printf("\n1. Insert from Beginning. -
      printf("Enter your choice: \n");
      scanf("%d", &choice);
      switch(choice){
        case 1:
            printf("Enter the data: \n");
            scanf("%d", &item);
```

```
        insert_from_beg(&tail, item);
        scanf("%d", &item); break;
    case 2:
        printf("Enter the data: ");
        scanf("%d", &item);
        insert_from_end(&tail, item);
        break;
    case 3:
        printf("Enter the data: \n");
        scanf("%d", &item);
        printf("Enter the position: \n");
        scanf("%d", &pos);
        insert_from_sp_pos(&tail, item, pos);
        break;
    case 4:
        item = del_from_beg(&tail);
        printf("%d is deleted \n", item);
        break;
    case 5:
        item = del_from_end(&tail);
        printf("%d is deleted \n", item);
        break;
    case 6:
        printf("Enter the position to be deleted");
        scanf("%d", &pos);
        del_from_sp_pos(&tail, pos);
        printf("%d is deleted \n", item);
        break;
```

```
case 7:
    display (&tail);
    break;

case 8:
    printf("Enter value for search: \n");
    scanf("%d", &item);
    if (search (&tail; item)){
        printf("The value %d has been found. \n", item);
    }else{
        printf("The value is not found. \n")
    break;

case 9:
    exit (0);

default:
    printf ("Invalid choice");
}
return 0
}
```

Activities

We performed different operations in this lab like. insert, delete, search, display etc.

Conclusion

I learned about the implementation of circular linked list with its different operations.