Name : Mayur Jadhav	Class/Roll No. :D6AD-A /21	Grade :

Circular Linked List

Theory :

In a circular Singly linked list, the last node of the list contains a pointer to the first node of the list. We can have circular singly linked list as well as circular doubly linked list.

We traverse a circular singly linked list until we reach the same node where we started. The circular singly liked list has no beginning and no ending. There is no null value present in the next part of any of the nodes.

The following image shows a circular singly linked list.

Code :

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


struct Node{
    int data;
    struct Node* next;
};

struct Node* head = NULL;

struct Node* getNode(){
    return (struct Node*)malloc(sizeof(struct Node));
};
```

```
void PrintCll();

void insertFront(){
    int val ;
    printf("Enter value");
    scanf("%d",&val);
    struct Node* temp = getNode();
    temp -> data = val;
    if(head==NULL){
        head = temp;
        temp -> next = head;
    }
    else{
        temp -> next = head;
        struct Node* ptr = head;
        while (ptr -> next != head){
            ptr = ptr -> next;
        };
        ptr -> next = temp;
        head = temp;
    }
}
void deleteFront(){
    if(head==NULL){
        printf("Empty");
    }
    else{
        struct Node* ptr = head;
        while(ptr -> next != head){
            ptr = ptr -> next;
        };
        ptr -> next = head -> next;
        head = head->next;
    };
}


void insertEnd(){
    int val;
    printf("Enter value");
    scanf("%d",&val);
    struct Node* temp = getNode();
    temp -> data = val;
    if(head==NULL){
```

```
        head = temp;
        temp -> next = head;
    }
    else{
        struct Node* ptr = head;
        while (ptr -> next != head){
            ptr = ptr -> next;
        }
        ptr -> next = temp;
        temp-> next = head;
    }
}
void deleteEnd(){
    if(head==NULL){
        printf("Empty");
    }
    else{
        struct Node* ptr = head;
        while(ptr -> next -> next != head){
            ptr = ptr -> next;
        };
        ptr -> next = head;
    }
};

void insertBefore(){
    int val , x;
    printf("Enter value");
    scanf("%d",&val);
    printf("Before which element it should be added");
    scanf("%d",&x);

    struct Node* temp = getNode();
    temp -> data = val;



    struct Node* ptr = head;
    if ( ptr -> data == x){
        temp -> next = ptr;
        while(ptr -> next != head)
            {
                ptr = ptr -> next;
            };
```

```
        ptr -> next = temp;
    }
    else{
        while(ptr -> next -> data != x)
        {
            ptr = ptr -> next;
        };
        temp -> next = ptr -> next;
        ptr -> next = temp;
    }

}

void insertAfter(){
    int val , x;
    printf("Enter value");
    scanf("%d",&val);
    printf("After which element it should be added");
    scanf("%d",&x);

    struct Node* temp = getNode();
    temp -> data = val;


    struct Node* ptr = head;
    while (ptr -> data != x){
        ptr = ptr -> next;
    }
    if (ptr -> next == head){
        ptr -> next = temp;
        temp -> next = head;
    }
    else{
        temp -> next = ptr -> next;
        ptr -> next = temp;
    }

}
void addNode(int val){
    struct Node* temp = getNode();
    temp -> data = val;
    if(head==NULL){
        head = temp;
        temp -> next = head;
```

```
    }
    else{
        struct Node* ptr = head;
        while (ptr -> next != head){
            ptr = ptr -> next;
        }
        ptr -> next = temp;
        temp-> next = head;
    }
}
void createList(){


    int i,n;
    printf("NO of Elements : ");
    scanf("%d",&n);
    int val;
    for (i=0;i<n;i++){
        printf("Enter Element : ");
        scanf("%d",&val);
        addNode(val);
    };
}



void PrintCll(){
    struct Node* ptr = head;
    while(ptr->next != head){
        printf("%d -> ",ptr-> data);
        ptr = ptr->next;
    };
     printf("%d -> \n",ptr-> data);
}


void main(){
   int choice;
    do {
        printf("\ncircular linked list----\n");
        printf("\t\t1. create list \n\t\t2. display list\n\t\t3.insert before\n\t\t4.insert after\n\t\t5.insert
front\n\t\t6.insert end\n\t\t7.delete front\n\t\t8.delete end\n\t\t9.exit\n\t\tChoice :");

        scanf("%d",&choice);
```

```
        switch(choice){
            case 1:
                createList();
                break;
            case 2:
                PrintCll();
                break;
            case 3:
                insertBefore();
                break;
            case 4:
                insertAfter();
                break;
            case 5:
                insertFront();
                break;
            case 6:
                insertEnd();
                break;
            case 7:
                deleteFront();
                break;
            case 8:
                deleteEnd();
                break;
            case 9:
                break;
        };
    }
    while(choice<9);
}
```

Output:

```
circular linked list----
                1. create list
                2. display list
                3.insert before
                4.insert after
                5.insert front
                6.insert end
                7.delete front
                8.delete end
                9.exit
                Choice :1
NO of Elements : 4
Enter Element : 1
Enter Element : 2
Enter Element : 3
Enter Element : 4

circular linked list----
                1. create list
                2. display list
                3.insert before
                4.insert after
                5.insert front
                6.insert end
                7.delete front
                8.delete end
                9.exit
                Choice :2
1 -> 2 -> 3 -> 4 ->
```

```
circular linked list----
                1. create list
                2. display list
                3.insert before
                4.insert after
                5.insert front
                6.insert end
                7.delete front
                8.delete end
                9.exit
                Choice :8

circular linked list----
                1. create list
                2. display list
                3.insert before
                4.insert after
                5.insert front
                6.insert end
                7.delete front
                8.delete end
                9.exit
                Choice :2
1 -> 2 -> 3 ->

circular linked list----
                1. create list
                2. display list
                3.insert before
                4.insert after
                5.insert front
                6.insert end
                7.delete front
                8.delete end
                9.exit
                Choice :7
```

```
circular linked list----
                1. create list
                2. display list
                3.insert before
                4.insert after
                5.insert front
                6.insert end
                7.delete front
                8.delete end
                9.exit
                Choice :2
2 -> 3 ->

circular linked list----
                1. create list
                2. display list
                3.insert before
                4.insert after
                5.insert front
                6.insert end
                7.delete front
                8.delete end
                9.exit
                Choice :5
Enter value1

circular linked list----
                1. create list
                2. display list
                3.insert before
                4.insert after
                5.insert front
                6.insert end
                7.delete front
                8.delete end
                9.exit
```

```
                3.insert before
                4.insert after
                5.insert front
                6.insert end
                7.delete front
                8.delete end
                9.exit
                Choice :2
1 -> 2 -> 3 ->

circular linked list----
                1. create list
                2. display list
                3.insert before
                4.insert after
                5.insert front
                6.insert end
                7.delete front
                8.delete end
                9.exit
                Choice :3
Enter value9
Before which element it should be added2

circular linked list----
                1. create list
                2. display list
                3.insert before
                4.insert after
                5.insert front
                6.insert end
                7.delete front
                8.delete end
                9.exit
                Choice :2
1 -> 9 -> 2 -> 3 ->
```