Experiment No.7
Aim: Implementation of Circular Linked List ADT
Name: Nikhil Kamalaji Shingade
Roll no: 57
Date of Performance:
Date of Submission:

Experiment No. 7: Circular Linked List Operations

Aim: Implementation of Circular Linked List ADT

Objective:

In circular linked list last node is connected to first node. On other hand circular linked list can be used to implement traversal along web pages.

Theory:

In a circular linked list, the last node contains a pointer to the first node of the list. We can have a circular singly linked list as well as a circular doubly linked list. While traversing a circular linked list, we can begin at any node and traverse the list in any one direction, forward or backward, until we reach the same node where we started. Thus, a circular linked list has no beginning and no ending.

Inserting a New Node in a Circular Linked List Case
1: The new node is inserted at the beginning.
Case 2: The new node is inserted at the end.

Deleting a Node from a Circular Linked List Case
1: The first node is deleted.
Case 2: The last node is deleted.
Insertion and Deletion after or before a given node is same as singly linked list.

Algorithm

Algorithm to insert a new node at the beginning

```
Step 1: IF AVAIL = NULL
            Write OVERFLOW
            Go to Step 9 [END OF IF]
Step 2: SET NEW_NODE = AVAIL
Step 3: SET AVAIL = AVAIL    NEXT
Step 4: SET NEW_NODE-->DATA = VAL Step
5: SET PTR=START
```

```
Repeat Step 6 while PTR NEXT != START
Step 6: SET PTR = PTR NEXT [END OF LOOP]
Step 7: SET NEW_NODE--> NEXT= START
Step 8: SET PTR-->NEXT = START
Step 9: SET START = NEW_NODE
Step 10: EXIT
```

Algorithm to insert a new node at the end

```
Step 1: IF AVAIL = NULL
        Write OVERFLOW
        Go to Step 11 [END OF IF]
Step 2: SET NEW_NODE = AVAIL
Step 3: SET AVAIL = AVAIL--> NEXT
Step 4: SET NEW_NODE -->DATA = VAL
Step 5: SET NEW_NODE-->NEXT = START
Step 6: SET PTR = START
Step 7: Repeat Step 8 while PTR--> NEXT != START
Step 8: SET PTR = PTR -->NEXT [END OF LOOP]
Step 9: SET PTR -->NEXT = NEW_NODE
Step 10: EXIT
```

Algorithm to delete the first node

```
Step 1: IF START = NULL
        Write UNDERFLOW
        Go to Step 6 [END OF IF]
Step 2: SET PTR = START
Step 3: Repeat Step 4 while PTR--> NEXT != START
Step 4: SET PTR = PTR -->NEXT [END OF LOOP]
Step 4: SET PTR   NEXT = START -->NEXT
Step 5: FREE START
Step 6: EXIT
```

Algorithm to delete the last node

```
Step 1: IF START = NULL
            Write UNDERFLOW
            Go to Step 7 [END OF IF]
Step 2: SET PTR = START [END OF LOOP]
Step 3: Repeat Step 4 and Step 5 while PTR -->NEXT != START
Step 4: SET PREPTR = PTR
Step 5: SET PTR = PTR -->NEXT
Step 6: SET PREPTR-->NEXT = START
Step 7: FREE PTR
Step 8: EXIT
```

Code:

```
#include<stdio.h>
#include<malloc.h>
struct node
{
 int data;
 struct node *next;
};
 struct node *head=NULL;
void add_at_begin()
{
 struct node *temp;
 struct node *p;
 int num;
 temp=(struct node*)malloc(sizeof(struct node));
 printf("Enter the number to perform add at begin:");
 scanf("%d",&num);
 if(head==NULL)
  {
 temp->data=num;
 temp->next=temp;
 head=temp;
 }
 else
 {
 temp->data=num;
 temp->next=head;
 p=head;
 while(p->next!=head)
 {
 p=p->next;
 }
 p->next=temp;
 head=temp;
}

}
void add_at_end()
{
 struct node *temp,*p;
 int num;
 temp=(struct node*)malloc(sizeof(struct node));
 printf("Enter the number to perform add at END:");
 scanf("%d",&num);
 temp->data=num;
 temp->next=head;
 if(head==NULL)
 {
 temp->data=num;
 temp->next=temp;
 head=temp;
 }
 else
 {
 p=head;
 while(p->next!=head)
 {
 p=p->next;
 }
 p->next=temp;
temp->next=head;
}
}
void add_in_between()
{
 int pos, num,i;
 struct node*temp,*p;
 temp=(struct node*)malloc(sizeof(struct node));
 printf("\nEnter the number to perform add in between:");
 scanf("%d",&num);
 temp->data=num;
 printf("Enter the position");
 scanf("%d",&pos);
 p=head;
 for(i=1;i<=pos-1;i++)
 {
 p=p->next;
 }
 temp->next=p->next;
 p->next=temp;
}

void search()
{
 struct node*p=head;
 int r=0;
 int num;
 printf("\nEnter the number to be search:");
 scanf("\n%d",&num);
do
{
 if(p->data==num)
 {
 r=1;
 break;
 }
 p=p->next;
 } while(p!=head);

 if(r==0)
 {
 printf("Number is not present");
 }
 else
 {
 printf("Number is present");
 }
}
void delete_begin()
{
 struct node *p,*q;
 p=head;
q=head;
while(p->next!=head)
{
p=p->next;
}
 head=head->next;
```

```
p->next=head;
 q->next=NULL;
}
void delete_end()
{
 struct node*p,*q;
 p=head;
 while(p->next!=head)
  {
 q=p;
 p=p->next;
  }
 q->next=head;
p->next=NULL;

}
void delete_in_between()
{
 struct node *p, *q;
 int pos,i;
 p=head;
 printf("\nEnter the position from which you want to delete number:");
 scanf("\n%d",&pos);
 for(i=1;i<=pos;i++)
  {
 q=p;
 p=p->next;
  }
 q->next=p->next;
 p->next=NULL;

}
void display()
{
 struct node *p;
 p=head;
 printf("Element of linked list:");
 while(p->next!=head)
  {
 printf("\n%d",p->data);
 p=p->next;
  }
printf("\n%d",p->data);

}
void main()
{
 add_at_begin();
 add_at_begin();
 add_at_begin();
 add_at_begin();
 display();
 add_at_end();
 display();
 add_in_between();
 display();
 search();
 delete_in_between(head);
 display(head);
 delete_begin();
 display();
 delete_end();
 display();
}
```

Output:

```
/tmp/86lEnBgwmc.o
Enter the number to perform add at begin:20
Enter the number to perform add at begin:30
Enter the number to perform add at begin:50
Enter the number to perform add at begin:60
Element of linked list:
60
50
30
20Enter the number to perform add at END:30
Element of linked list:
60
50
30
20
30
Enter the number to perform add in between:70
Enter the position3
Element of linked list:
60
50
30
70
20
30
Enter the number to be search:40
Number is not present
Enter the position from which you want to delete number:2
Element of linked list:
60
```

```
50
70
20
30Element of linked list:
50
70
20
30Element of linked list:
50
70
20

=== Code Exited With Errors ===
```

Conclusion:

Write an example of insertion and deletion in the circular linked list while traversing the web pages?

Ans:

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

// Define the structure for a web page node
struct WebPage {
    char url[100];
    struct WebPage* next;
};

// Function to create a new web page node
struct WebPage* createWebPage(char* url) {
    struct WebPage* newPage = (struct WebPage*)malloc(sizeof(struct WebPage));
    strcpy(newPage->url, url);
    newPage->next = NULL;
    return newPage;
}

// Function to insert a web page into the circular linked list
void insertPage(struct WebPage** last, char* url) {
    struct WebPage* newPage = createWebPage(url);
    if (*last == NULL) {
        *last = newPage;
        (*last)->next = *last;
    } else {
        newPage->next = (*last)->next;
        (*last)->next = newPage;
        *last = newPage;
    }
    printf("Inserted: %s\n", url);
}

// Function to delete the web page from the circular linked list
void deletePage(struct WebPage** last, char* url) {
    if (*last == NULL) {
        printf("List is empty.\n");
        return;
    }

    struct WebPage *current = (*last)->next, *prev = *last;
    while (current != *last) {
        if (strcmp(current->url, url) == 0) {
            prev->next = current->next;
            free(current);
            printf("Deleted: %s\n", url);
            return;
        }
        prev = current;
        current = current->next;
    }

    if (strcmp((*last)->url, url) == 0) {
        prev->next = (*last)->next;
        free(*last);
        *last = prev;
        printf("Deleted: %s\n", url);
        return;
    }

    printf("Page not found: %s\n", url);
}

// Function to display the web pages in the circular linked list
void displayPages(struct WebPage* last) {
    if (last == NULL) {
        printf("List is empty.\n");
        return;
    }

    struct WebPage* current = last->next;
    do {
        printf("%s -> ", current->url);
        current = current->next;
    } while (current != last->next);
    printf("\n");
```

```
}

int main() {
    struct WebPage* last = NULL;

    // Simulate visiting web pages
    insertPage(&last, "google.com");
    insertPage(&last, "stackoverflow.com");
    insertPage(&last, "github.com");
    insertPage(&last, "linkedin.com");

    printf("Web pages visited:\n");
    displayPages(last);

    // Simulate deleting a web page
    deletePage(&last, "stackoverflow.com");

    printf("Web pages after deletion:\n");
    displayPages(last);

    return 0;
}
```