September 27, 2021

IMPLEMENTATION OF SINGLY CIRCULAR LINK LIST

—

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

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

struct node *temp,*tail=0;
struct node *head=0;

//Function Declaractions
void CreateCll();       //create Circular Link list
void DisplayCll();      //Dislpay circular Link List
void InsAtBeg();        //Insert At Begning
void InsAtPos();        //Insert At Given position
void InsAfterPos();     //Insert After Given Position
void InsAtEnd();        //Insert At End
void DelAtBeg();        //Delete At Begning
void DelAtPos();        //Delete at Given position
void DelAfterPos();     //Delete After Given Position
void DelAtEnd();        //Deleta At end
void GetLen();          //Gets The Length Of Circular Link list
void ReverseCll();      //Reverse The circular Link  list

int main(){
        int choice;
        while(choice!=13){
                printf("\n\n***MAIN MENU***\n\n");
```

```
30.             printf("1. Create Circular Link List\n");
31.             printf("2. Insert At Begning\n");
32.             printf("3. Insert at Given Position\n");
33.             printf("4. Insert After Given Position\n");
34.             printf("5. Insert At End\n");
35.             printf("6. Delete At Begning\n");
36.             printf("7. Delete At Given Position\n");
37.             printf("8. Delete After Given Position\n");
38.             printf("9. Delete At End\n");
39.             printf("10. Find The Length Of The List\n");
40.             printf("11. Reverse The List\n");
41.             printf("12. Display Link List\n");
42.             printf("13. Exit\n");
43.             printf("\nEnter choice ");
44.             scanf("%d",&choice);
45.             switch(choice){
46.                     case 1:
47.                             CreateCll();
48.                             break;
49.                     case 2:
50.                             InsAtBeg();
51.                             break;
52.                     case 3:
53.                             InsAtPos();
54.                             break;
55.                     case 4:
56.                             InsAfterPos();
57.                             break;
58.                     case 5:
59.                             InsAtEnd();
60.                             break;
61.                     case 6:
62.                             DelAtBeg();
63.                             break;
64.                     case 7:
65.                             DelAtPos();
66.                             break;
67.                     case 8:
68.                             DelAfterPos();
69.                             break;
70.                     case 9:
71.                             DelAtEnd();
72.                     case 10:
73.                             GetLen();
74.                             break;
```

```
                        case 11:
                                ReverseCll();
                                break;
                        case 12:
                                DisplayCll();
                                break;
                        case 13:
                                exit(0);
                                break;
                        default:
                                printf("\n Enter Valid Choice\n");
                }
        }
        return 0;
}
/* //Created With Both Head And Tail
void CreateCll(){
        struct node *newnode;
        int choice=1;
        while(choice){
                newnode=(struct node*)malloc(sizeof(struct node));
                printf("\nEnter Data ");
                scanf("%d",&newnode->data);
                newnode->next=0;
                if(head==0){
                        head=tail=newnode;
                        //tail->next=head;
                }
                else{
                        tail->next=newnode;
                        tail=newnode;
                        //tail->next=head;
                }
                tail->next=head;
                printf("\n0 To Exit 1 To Add");
                scanf("%d",&choice);
        }
}
*/

/* This Function Will Create Circular List Only By Tail Pointer */
void CreateCll(){
        struct node *newnode;
        int choice=1;
        while(choice){
```

```
120.                 newnode=(struct node*)malloc(sizeof(struct node));
121.                 printf("\nEnter Data ");
122.                 scanf("%d",&newnode->data);
123.                 newnode->next=0;
124.                 if(tail==0){
125.                         tail=newnode;
126.                         tail->next=newnode;
127.                 }
128.                 else{
129.                         newnode->next=tail->next;
130.                         tail->next=newnode;
131.                         tail=newnode;
132.                 }
133.                 printf("\n0 To Exit 1 To Add ");
134.                 scanf("%d",&choice);
135.         }
136. }
137.
138. void DisplayCll(){
139.         //temp=head;
140.         temp=tail->next;  /*Only if List Is Created Only With Tail Pointer */
141.         printf("Displaing Circular Link List\n");
142.         do{
143.                 printf("%d ",temp->data);
144.                 temp=temp->next;
145.         }while(temp!=tail->next);
146. }
147.
148.
149. //To Insert At Begning
150. void InsAtBeg(){
151.         struct node *newnode;
152.         if(tail==0)     {
153.                 printf("\nCreate A List Firt\n");
154.         }
155.         else{
156.                 temp=tail->next;
157.                 newnode=(struct node*)malloc(sizeof(struct node));
158.                 printf("Enter Data ");
159.                 scanf("%d",&newnode->data);
160.                 newnode->next=temp;
161.                 tail->next=newnode;
162.                 printf("\nNode Created Successfully");
163.         }
164. }
```

```
165.
166. //To insert At Given Position
167. void InsAtPos(){
168.         int pos,i;
169.         struct node *newnode;
170.         if(tail==0){
171.                 printf("\nCreate A List Firt\n");
172.         }
173.         else{
174.                 temp=tail->next;
175.                 printf("Enter position ");
176.                 scanf("%d",&pos);
177.                 if(pos==1){
178.                         printf("You've Chosen Wrong Option\n");
179.                         printf("if you want to insert at begning then choose '2'
180.                 }
181.                 else{
182.                 newnode=(struct node*)malloc(sizeof(struct node));
183.                 printf("Enter Data ");
184.                 scanf("%d",&newnode->data);
185.                         for(i=1;i<pos-1;i++){
186.                                 temp=temp->next;
187.                         }
188.                         newnode->next=temp->next;
189.                         temp->next=newnode;
190.                         printf("Node Created Successfully\n");
191.                 }
192.         }
193. }
194.
195. //To insert Atfrt Given position
196. void InsAfterPos(){
197.         int pos,i;
198.         struct node *newnode;
199.         if(tail==0){
200.                 printf("\nCreate A List First");
201.         }
202.         else{
203.                 printf("Enter Position ");
204.                 scanf("%d",&pos);
205.                 if(pos==1){
206.                         printf("You've Chosen Wrong Option\n");
207.                         printf("if you want to insert at begning then choose '2'
208.                 }
209.                 else{
```

```
                        temp=tail->next;
                        newnode=(struct node*)malloc(sizeof(struct node));
                        printf("Enter Data ");
                        scanf("%d",&newnode->data);
                        for(i=1;i<pos;i++){
                                temp=temp->next;
                        }
                        newnode->next=temp->next;
                        temp->next=newnode;
                        printf("\n Node created successfully\n");
                }
        }
}

//To insert at end
void InsAtEnd(){
        struct node *newnode;
        if(tail==0){
                printf("Create List Firt");
        }
        else{
                newnode=(struct node*)malloc(sizeof(struct node));
                printf("Enter Data ");
                scanf("%d",&newnode->data);
                newnode->next=tail->next;
                tail->next=newnode;
                tail=newnode;
                printf("\nNode Created Successfully\n");
        }
}

//Delete At begning
void DelAtBeg(){
        if(tail==0){
                printf("Create List First\n");
        }
        else{
        temp=tail->next;
        tail->next=temp->next;
        free(temp);
        printf("\nNode Deleted Successfully\n");
        }
}

//Delete At Given position
```

```
255. void DelAtPos(){
256.         int pos,i;
257.         struct node *del;
258.         if(tail==0){
259.                 printf("You can't Delete Node Without Creating It\n");
260.         }
261.         else{
262.                 printf("Enter Position ");
263.                 scanf("%d",&pos);
264.                 if(pos==1){
265.                         DelAtBeg();
266.                 }
267.                 else{
268.                         temp=tail->next;
269.                         for(i=1;i<pos-1;i++){
270.                         temp=temp->next;
271.                         }
272.                         del=temp->next;
273.                         temp->next=del->next;
274.                         free(del);
275.                         printf("Node Deleted Successfully");
276.                 }
277.         }
278. }
279.
280. //Delete after given Position
281. void DelAfterPos(){
282.         int pos,i;
283.         struct node *del;
284.         if(tail==0){
285.                 printf("You can't Create Node Without Creating It\n");
286.         }
287.         else{
288.                 printf("Enter Position ");
289.                 scanf("%d",&pos);
290.                 if(pos==1){
291.                         DelAtBeg();
292.                 }
293.                 else{
294.                         temp=tail->next;
295.                         for(i=1;i<pos;i++){
296.                         temp=temp->next;
297.                         }
298.                         del=temp->next;
299.                         temp->next=del->next;
```

```
                        free(del);
                        printf("Node Deleted Successfully");
                }
        }
}

//Delete At end
void DelAtEnd(){
        struct node *prev;
        if(tail==0){
                printf("You can't delete Node Without Creating ");
        }
        else{
                temp=tail->next;
                while(temp->next!=tail->next){
                        prev=temp;
                        temp=temp->next;
                }
                prev->next=tail->next;
                tail=prev;
                free(temp);
                printf("\n node deleted success fully\n");
        }
}

//To Get The Length of Circular Link list
void GetLen(){
        int count=1;
        temp=tail->next;
        while(temp->next!=tail->next){
                temp=temp->next;
                count++;
        }
        printf("Lenghth Of List Is %d ",count);
}

//To Reverse Circular Link List
void ReverseCll(){
struct node *current,*prev,*nextnode;
        current=tail->next;
        nextnode=current->next;
        if(tail==0){
                printf("List is Empty");
        }
        else if(tail->next==tail){
```

```
345.                DisplayCll();
346.        }
347.        else{
348.                while(current!=tail){
349.                        prev=current;
350.                        current=nextnode;
351.                        nextnode=current->next;
352.                        current->next=prev;
353.                }
354.                nextnode->next=tail;
355.                tail=nextnode;
356.                printf("Reversed Successfully\n");
357.        }
358. }
359.
360. //This Is All About Singly Circular Link List
361. //Created By SMIT
```

Share

Labels: circular link list, implementation of Singly circular Link List

COMMENTS

Enter your comment...

POPULAR POSTS