

## PRACTICAL NO : 8

**AIM** :- Implement a Circular Single Linked List and Perform the operation : Create, Traverse, Insert\_beg, Insert\_end, Delete\_beg, Delete\_end using Menu Driver Program.

### PROGRAM :

```
#include<stdio.h>

#include<stdlib.h>

struct node{

    int data;

    struct node*next;

};

struct node *s, *p, *q, *a, *t;

void create(){

    printf("Creating the Circular Linked List(CLL).\nEnter data for the First node:\t");

    p = (struct node *)malloc(sizeof(struct node));

    scanf("%d", &p -> data);

    p -> next = p;

    s = p;

}

void Traverse(){

    printf("\nTraversing the linked list:\t");

    t = s;

    do{

        printf("%d\t", t -> data);

        t = t -> next;

    } while (t != s);

}
```

## PRACTICAL NO : 8

```
}
```

```
void Insert_Beg()
```

```
{ printf("\nInserting node at beggining.\nEnter the data:\t");
```

```
  p = (struct node *)malloc(sizeof(struct node));
```

```
  scanf("%d", &(p -> data));
```

```
  if(s == NULL){
```

```
    p -> next = p;
```

```
    s = p;
```

```
  }else{
```

```
    t = s;
```

```
    while (t -> next != s)
```

```
    {
```

```
      t = t-> next;
```

```
    }
```

```
    p->next = s;
```

```
    t->next = p;
```

```
    s = p;
```

```
  }
```

```
}
```

```
void Insert_End()
```

```
{
```

```
  t = s;
```

## PRACTICAL NO : 8

```
while(t -> next != s){
```

```
    t = t -> next;
```

```
}
```

```
p = (struct node*)malloc(sizeof(struct node));
```

```
printf("\nEnter data of last node:\t");
```

```
scanf("%d", &(p -> data));
```

```
p -> next = s;
```

```
t -> next = p;
```

```
}
```

```
void Delete_Beg()
```

```
{ printf("\nDeleting the node at beggining..\n");
```

```
if (s == NULL)
```

```
{
```

```
    printf("The linked list is empty..\n");
```

```
}
```

```
t = s;
```

```
while(t -> next != s){
```

```
    t = t -> next;
```

```
}
```

```
q = s -> next;
```

```
t -> next = q;
```

```
free(s);
```

```
s = q;
```

```
}
```

## PRACTICAL NO : 8

```
void Delete_End()
{ printf("\nDeleteing the node at end..");

  t = s;

  while(t -> next != s){

    q = t;

    t = t-> next;

  }

  q -> next = s;

  free(t);
}

int main(){

  int choice;

  printf("\nCHOICES\n1.Create\t2.Traverse\t3.Insert_Beg\n4.Dlete_Beg\t5.Insert_end\t6.
  Delte_End\t7.Exit");

  do{

    printf("\nEnter valid choice.\n");

    scanf("%d", &choice);

    switch (choice)

    {

    case 1:

      create();

      break;

    case 2:

      Traverse();

      break;
```

## PRACTICAL NO : 8

case 3:

Insert\_Beg();

break;

case 4:

Delete\_Beg();

break;

case 5:

Insert\_End();

break;

case 6:

Delete\_End();

break;

case 7:

printf("Exit the program..");

break;

default:

printf("\nPlease enter a valid choice\n");

break;

}

}while(choice != 7);

}

# PRACTICAL NO : 8

## OUTPUT

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS C:\Users\230212\Desktop\DSA_programs> cd "c:\Users\230212\Desktop\DSA_programs"
PS C:\Users\230212\Desktop\DSA_programs> cd "c:\Users\230212\Desktop\DSA_programs\" ; if ($?) { gcc practical7.c -o practical7 } ; if ($?) { .\practical7 }

CHOICES
1.Create      2.Traverse    3.Insert_Beg  4.Dlete_Beg  5.Insert_end  6.Delte_End  7.Exit
Enter choice.
1
Creating the Circular Linked List(CLL).
Enter data for the First node: 30

Enter choice.
3

Inserting node at beggining.
Enter the data: 20

Enter choice.
5

Enter data of last node:      40

Enter choice.
2

Traversing the linked list:    20    30    40
Enter choice.
4

Deleting the node at beggining..

Enter choice.
6

Deleteing the node at end..
Enter choice.
2

Traversing the linked list:    30
Enter choice.
7
Exit the program..
PS C:\Users\230212\Desktop\DSA_programs> |
```

GITHUB LINK : <https://github.com/AmolNagargoje04/Data-Structure-practical>