Objective: To implement circular doubly linked list in C language.

Code:

```c
#include <stdio.h>
#include <stdlib.h>
struct node
{ int info;
  struct node *front, *back;
} *start, *last, *temp, *flag, *p, *q;
void insert ()
{ p = (struct node *) malloc (sizeof (struct node));
  if (p == NULL)
  { printf ("\nFailed to allocate memory");
    return;
  }
  printf ("\nEnter the info part: ");
  scanf ("%d", & p->info);
  p->front = NULL;
  p->back = NULL;
  if (start == NULL)
  { start = p;
    last = p;
    start->back = start->front;
    start->front = start->back;
  }
  else
  { last->front = p;
    p->back = last;
    last = last->front;
    last->front = start;
    start->back = last;
  }
}
```

JIET GROUP OF INSTITUTIONS

Student Name .... Roll No.: ....

Experiment No. ....

Date ....

```
void ainsert ().
{  int ch, n;
   printf ("\n Enter the number after which you
          wish to add data: ");
   scanf ("%d", & ch);
   printf (" \n Enter the info part: ");
   scanf ("%d", & n);
   p = (struct node *) malloc (sizeof (struct node));
   if (p == NULL)
   {  printf ("\n Failed to allocate memory");
      return;
   }

   p -> info = n;
   p -> front = NULL;
   p -> back = NULL;
   temp = start;
   flag = start -> front;
   while (temp -> info != ch)
   {  temp = temp -> front;
      flag = flag -> front;
   }
   temp -> front = p;
   p -> back = temp;
   p -> front = flag;
   flag -> back = p;
   if (flag -> info == start -> info)
          last = last -> front;
   q = p;
```

JIET GROUP OF INSTITUTIONS

Student Name .... Roll No.: ....

Experiment No. .... Date ....

```c
while (q-> back != last)
{ printf("%d->", q->info);
    q = q -> back;
}
    printf("%d-> ", q->info);
    printf("\b\b\b.    \n");
}
void delete ()
{  int x;
    printf("\n Enter the data to be deleted: ");
    scanf("%d", &x);
    temp = start -> front;
    flag = start;
    if (flag -> info == x)
    {   start = start -> front;
        start -> back = last;
        last -> front = start;
        printf("%d\n", start -> info);
        flag -> front = NULL;
        flag -> back = NULL;
        free (flag);
        return;
    }
    while (temp -> info != x)
    { temp = temp -> front;
        flag = flag -> front;
    }
}
```

```c
if (temp->front == start)
{   last = flag;
    q = temp->front;
    flag->front = q;
    q->back = flag;
    temp->front = NULL;
    temp->back = NULL;
    free (temp);
    return;
}

q = temp->front;
flag->front = q;
q->back = flag;
temp->front = NULL;
temp->back = NULL;
free (temp);
}

void display ()
{   temp = start;
    while (temp->front != start)
    { printf("%d -> ", temp->info);
        temp = temp->front;
    }
    printf("%d -> ", temp->info);
    printf("\b\b\b  \n");
}
```

```c
int main ().
{   int ch;
    start = NULL;
    temp = NULL;
    flag = NULL;
    p = NULL;
    q = NULL;
                        CIRCULAR
    printf("\t\t\t\t DOUBLY LINKED LIST \n");
    printf("1). Insert \n");
    printf("2) Insert after a data \n");
    printf("3). Display \n");
    printf("4) Delete \n");
    printf("5) Exit \n");
    printf("6)
    printf("Enter your choice: ");
    scanf("%d", &ch);
    do
    {   switch (ch)
        {   case 1: insert();
                    break;
            case 2: printf("\n\n");
                    ainsert();
                    break;
            case 3: printf("\n\n");
                    display();
                    break;
            case 4: delete();
                    break;
            case 5: break;
        }
}
```

```c
        printf("\t\t\t\t CIRCULAR DOUBLY LINKED LIST \n");
        printf("1) Insert \n");
        printf("2) Insert after a data \n");
        printf("3) Display \n");
        printf("4) Delete \n");
        printf("5) Exit \n");
        printf("Enter your choice: ");
        scanf("%d", &ch);
    } while(ch != 5);
    free(start);
    free(flag);
    free(p);
    free(q);
    return 0;
}
```