**Collage : Vishwakarma Institute of Technology**

**Course Name : Data Structure in C**

**Name : Vedika Vikas Sontakke**

**Roll no : 37**

**PRN NO 12220206**

Assignment 4 : Write a Program to implement circular double ended queue where user can add and remove the elements from bot front and rear of the queue.

Program :

#include<stdio.h>

#include<stdlib.h>

#define SIZE 5

int dequeue[SIZE];

int front = -1;

int rear = -1;

void enqueue\_front()

{

    int data;

    printf("enter a data : ");

    scanf("%d",&data);

    if((front==0 && rear==SIZE-1) || (front == rear+1)) printf("overflow\n");

    else if(front==-1 && rear==-1)

    {

        // dequeue is empty

        front = rear = 0;

        dequeue[front] = data;

    }else if(front == 0)

    {

        front = SIZE-1;

        dequeue[front] = data;

    }

    else

    {

        front = front-1;

        dequeue[front] = data;

    }

}

void enqueue\_rear()

{

    int data;

    printf("enter a data : ");

    scanf("%d",&data);

    if((front==0 && rear==SIZE-1) || (front == rear+1)) printf("overflow\n");

    else if(front==-1 && rear==-1)

    {

        // dequeue is empty

         rear = 0;

        dequeue[rear] = data;

    }else if(rear == SIZE-1)

    {

        rear = 0;

        dequeue[rear] = data;

    }

    else

    {

        rear = rear+1;

        dequeue[rear] = data;

    }

}

void dequeue\_front()

{

    if(front == -1 && rear == -1) printf("underflow\n");

    else if(front == rear )

    {

        // single element in a list

        printf("delelted element is %d ", dequeue[front]);

        front = rear = -1;

    }else if (front == SIZE-1)

    {

        printf("delelted element is %d ", dequeue[front]);

        front = 0;

    }else

    {

        printf("delelted element is %d ", dequeue[front]);

        front++;

    }

}

void dequeue\_rear()

{

    if(front == -1 && rear == -1) printf("underflow\n");

    else if(front == rear )

    {

        // single element in a list

        printf("delelted element is %d ", dequeue[rear]);

        front = rear = -1;

    }else if (rear == 0)

    {

        printf("delelted element is %d ", dequeue[rear]);

        rear = SIZE-1;

    }else

    {

        printf("delelted element is %d ", dequeue[rear]);

        rear--;

    }

}

void get\_front()

{

    if(front == -1 && rear == -1) printf("underflow\n");

    else printf("front end value is %d \n",dequeue[front]);

}

void get\_rear()

{

    if(front == -1 && rear == -1) printf("underflow\n");

    else printf("rear end value is %d \n",dequeue[rear]);

}

void display()

{

       if(front == -1 && rear == -1) printf("underflow\n");

       else

       {

         printf("elements in a array are : \n ");

         int i=front;

         while(i!=rear)

         {

            printf(" %d ",dequeue[i]);

            i = (i+1)%SIZE;

         }

         printf(" %d ", dequeue[rear]);

       }

}

int main()

{

    int choice;

    printf("1.Insert element at front end \n");

    printf("2.Insert element at rear end \n");

    printf("3.Delete element from front end \n");

    printf("4.Delete element from rear end \n");

    printf("5.Display element at front end \n");

    printf("6.Display element at rear end \n");

    printf("7.Display all elements of queue \n");

    printf("8.Quit \n");

    while (1)

    {

        printf("\nEnter your choice : ");

        scanf("%d", &choice);

        switch (choice)

        {

            case 1:

            enqueue\_front();

            break;

            case 2:

            enqueue\_rear();

            break;

            case 3:

            dequeue\_front();

            break;

            case 4:

            dequeue\_rear();

            break;

            case 5:

            get\_front();

            break;

            case 6:

            get\_rear();

            break;

            case 7:

            display();

            break;

            case 8:

            exit(1);

            default:

            printf("Wrong choice \n");

        }

    }

}

Output :

