**Question 1 .**

**Write a program implementing insert, delete and display operation of Circular Queue**

**Solution ;**

#include<stdio.h>

#include<stdlib.h>

#define MAX 10

int cqueue\_arr[MAX];

int front=-1;

int rear=-1;

void display( );

void insert(int item);

int del();

int peek();

int isEmpty();

int isFull();

int main()

{

int choice,item;

while(1)

{

printf("\n1.Insert\n");

printf("2.Delete\n");

printf("3.Peek\n");

printf("4.Display\n");

printf("5.Quit\n");

printf("\nEnter your choice : ");

scanf("%d",&choice);

switch(choice)

{

case 1 :

printf("\nInput the element for insertion : ");

scanf("%d",&item);

insert(item);

break;

case 2 :

printf("\nElement deleted is : %d\n",del());

break;

case 3:

printf("\nElement at the front is : %d\n",peek());

break;

case 4:

display();

break;

case 5:

exit(1);

default:

printf("\nWrong choice\n");

}/\*End of switch\*/

}/\*End of while \*/

return 0;

}/\*End of main()\*/

void insert(int item)

{

if( isFull() )

{

printf("\nQueue Overflow\n");

return;

}

if(front == -1 )

front=0;

if(rear==MAX-1)/\*rear is at last position of queue\*/

rear=0;

else

rear=rear+1;

cqueue\_arr[rear]=item ;

}/\*End of insert()\*/

int del()

{

int item;

if( isEmpty() )

{

printf("\nQueue Underflow\n");

exit(1);

}

item=cqueue\_arr[front];

if(front==rear) /\* queue has only one element \*/

{

front=-1;

rear=-1;

}

else if(front==MAX-1)

front=0;

else

front=front+1;

return item;

}/\*End of del() \*/

int isEmpty()

{

if(front==-1)

return 1;

else

return 0;

}/\*End of isEmpty()\*/

int isFull()

{

if((front==0 && rear==MAX-1) || (front==rear+1))

return 1;

else

return 0;

}/\*End of isFull()\*/

int peek()

{

if( isEmpty() )

{

printf("\nQueue Underflow\n");

exit(1);

}

return cqueue\_arr[front];

}/\*End of peek()\*/

void display()

{

int i;

if(isEmpty())

{

printf("\nQueue is empty\n");

return;

}

printf("\nQueue elements :\n");

i=front;

if( front<=rear )

{

while(i<=rear)

printf("%d ",cqueue\_arr[i++]);

}

else

{

while(i<=MAX-1)

printf("%d ",cqueue\_arr[i++]);

i=0;

while(i<=rear)

printf("%d ",cqueue\_arr[i++]);

}

printf("\n");

}/\*End of display() \*/

**Question 3.**

**Implement push, pop and find the minimum element in a stack in O(1) time complexity**

**Solution;**

#include<stdio.h>

#include<conio.h>

struct MyStack

{

    stack<int> s;

    int minEle;

    // Prints minimum element of MyStack

    void getMin()

    {

        if (s.empty())

            print("Stack is empty\n");

        // variable minEle stores the minimum element

        // in the stack.

        else

            printf(“%d%,"Minimum Element in the stack is: ",minEle,"\n");

    }

    // Prints top element of MyStack

    void peek()

    {

        if (s.empty())

        {

            printf( "Stack is empty ");

            return;

        }

        int t = s.top(); // Top element.

        Printf( "Top Most Element is: ");

        // If t < minEle means minEle stores

        // value of t.

        (t < minEle)? Printf(minEle): printf(t);

    }

    // Remove the top element from MyStack

    void pop()

    {

        if (s.empty())

        {

            Printf( "Stack is empty\n");

            return;

        }

        Printf("Top Most Element Removed: ");

        int t = s.top();

        s.pop();

        // Minimum will change as the minimum element

        // of the stack is being removed.

        if (t < minEle)

        {

            Printf(“%d”,minEle"\n");

            minEle = 2\*minEle - t;

        }

        else

            printf(“%d”,t"\n");

    }

    // Removes top element from MyStack

    void push(int x)

    {

        // Insert new number into the stack

        if (s.empty())

        {

            minEle = x;

            s.push(x);

            printf(“%d”,"Number Inserted: ", x , "\n");

            return;

        }

        // If new number is less than minEle

        if (x < minEle)

        {

            s.push(2\*x - minEle);

            minEle = x;

        }

        else

           s.push(x);

        printf(“%d”,"Number Inserted: " x,"\n");

    }

};

// Driver Code

int main()

{

    MyStack s;

    s.push(3);

    s.push(5);

    s.getMin();

    s.push(2);

    s.push(1);

    s.getMin();

    s.pop();

    s.getMin();

    s.pop();

    s.peek();

    return 0;

}