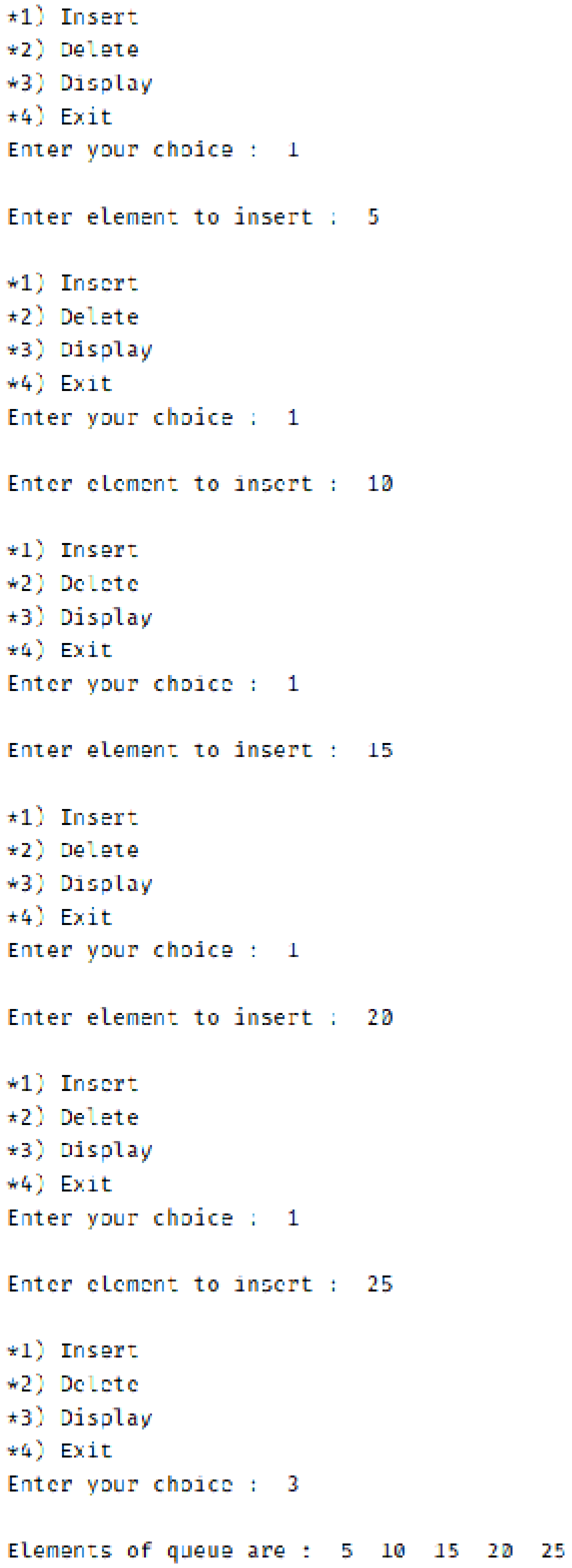
Implementation of circular queue using array.

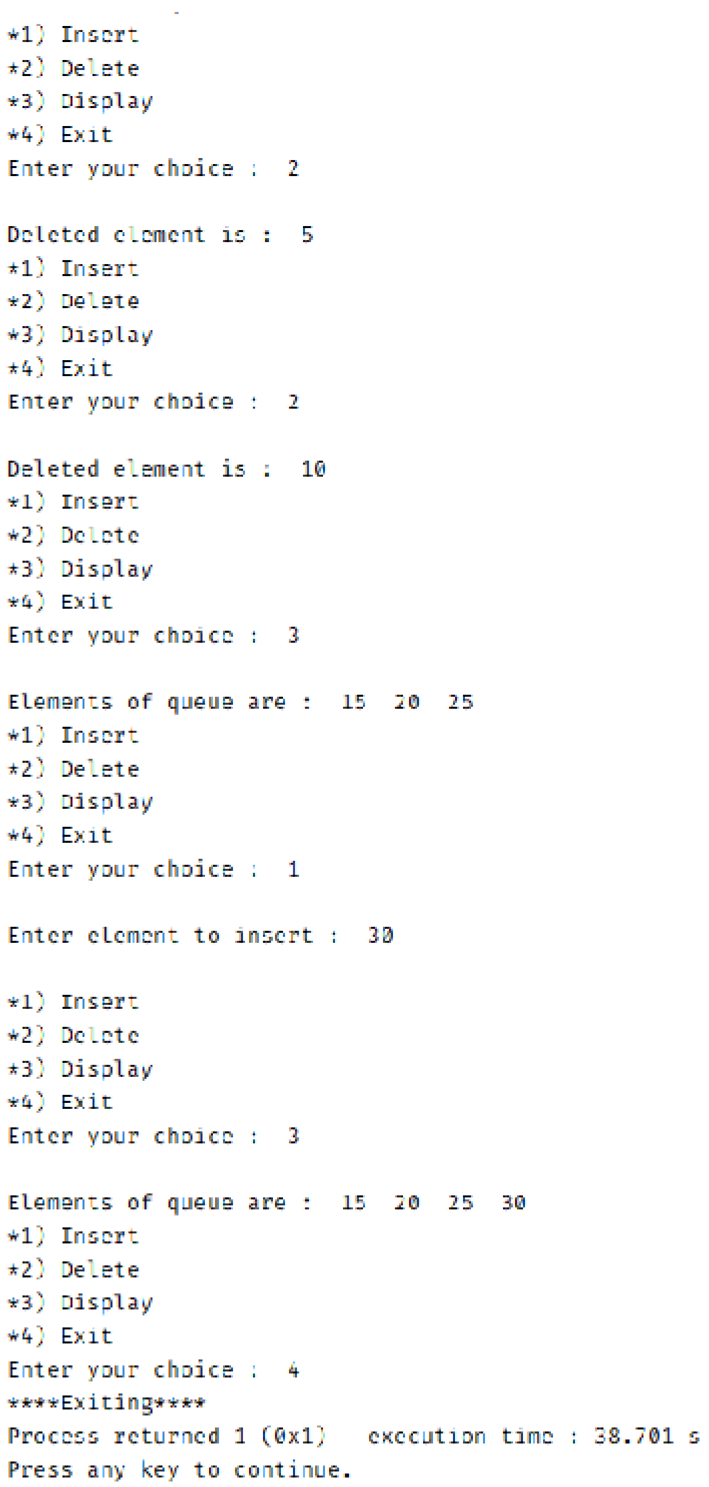
```
//code

//Implementation of circular queue
#include <stdio.h>
#include <stdlib.h>
#define MAX 5

//queue
int queue[MAX];
int rear = -1;
int front = -1;

//function prototype
void insert(int );
void del(int* deleted);
void display();

int main() {
    int choice, item, deleted;
    while(1) {
        printf("\n*1) Insert ");
        printf("\n*2) Delete ");
        printf("\n*3) Display ");
        printf("\n*4) Exit ");
        printf("\nEnter your choice :  ");
        scanf("%d", &choice);

        switch(choice) {
            case 1:
                printf("\nEnter element to insert :  ");
                scanf("%d", &item);
                insert(item);
                break;
            case 2:
                del(&deleted);
                printf("\nDeleted element is :  %d", deleted);
                break;
            case 3:
                printf("\nElements of queue are :  ");
                display();
                break;
            case 4:
                printf("****Exiting****");
                exit(1);
            default :
                printf("\nInvalid option");
        }
    }
}
```

```
void insert(int item) {
    if((front==0 && rear==MAX-1) || (front==rear+1)) {
        printf("\nOVERFLOW");
        return;
    }
    if(front==-1 && rear==-1) { //Empty queue
        front++;
        rear++;
    } else if(rear==MAX-1) { //if insertion is at last space, and 0th position is empty
        rear = 0;
    } else { //normal condition
        rear++;
    }
    queue[rear] = item;
}

void del(int* deleted) {
     if (front==-1) {
        printf("\nUNDERFLOW");
        return;
     }
     *deleted = queue[front];
     if(front == rear) { //if only remaining element is deleted
        front = -1;
        rear = -1;
     } else {
         if(front == MAX-1) { //when element is the last space is deleted and 0th position is not empty
            front = 0;
         } else { //normal condition
            front++;
         }
     }
}

void display() {
    int i;
    if(front == -1) {
        printf("UNDERFLOW");
        return;
    }
    for(i=front ; i!=rear ; i=((i+1)%MAX)) {
        printf("%d  ", queue[i]);
    }
    printf("%d", queue[rear]);
}
```

//output



```
*1) Insert
*2) Delete
*3) Display
*4) Exit
Enter your choice :  1

Enter element to insert :  5

*1) Insert
*2) Delete
*3) Display
*4) Exit
Enter your choice :  1

Enter element to insert :  10

*1) Insert
*2) Delete
*3) Display
*4) Exit
Enter your choice :  1

Enter element to insert :  15

*1) Insert
*2) Delete
*3) Display
*4) Exit
Enter your choice :  1

Enter element to insert :  20

*1) Insert
*2) Delete
*3) Display
*4) Exit
Enter your choice :  1

Enter element to insert :  25

*1) Insert
*2) Delete
*3) Display
*4) Exit
Enter your choice :  3

Elements of queue are :  5  10  15  20  25
```



```
*1) Insert
*2) Delete
*3) Display
*4) Exit
Enter your choice :  2

Deleted element is :  5
*1) Insert
*2) Delete
*3) Display
*4) Exit
Enter your choice :  2

Deleted element is :  10
*1) Insert
*2) Delete
*3) Display
*4) Exit
Enter your choice :  3

Elements of queue are :  15  20  25
*1) Insert
*2) Delete
*3) Display
*4) Exit
Enter your choice :  1

Enter element to insert :  30

*1) Insert
*2) Delete
*3) Display
*4) Exit
Enter your choice :  3

Elements of queue are :  15  20  25  30
*1) Insert
*2) Delete
*3) Display
*4) Exit
Enter your choice :  4
****Exiting****
Process returned 1 (0x1)   execution time : 38.701 s
Press any key to continue.
```