```c
#include<stdio.h>
#include<stdlib.h>
#define SIZE 3
int front = -1, rear = -1, queue[SIZE];

void insert(int n);
int delete_element();
void display();

int main()
{
    int opt, n, val;
    while(1)
    {
        printf("\nenter your choice to perform linear queue operation:\n");
        printf("enter:\n1.inserting element\n2.deleting element\n3.displaying queue elements\n4.exit\n");
        scanf("%d", &opt);
        switch(opt)
        {
            case 1: printf("enter the element to insert:");
                    scanf("%d", &n);
                    insert(n);
                    break;
            case 2: val = delete_element();
                    if(val != 0)
                    {
                        printf("deleted element is %d\n", val);
                    }
                    break;
            case 3: display();
                    break;
            case 4: exit(0);
        }
    }
    return 0;
}

void insert(int n)
{
    if(rear == SIZE - 1)
```

D:\1BM23CS245\linear_queue\queue.c

```c
void insert(int n)
{
    if(rear == SIZE - 1)
    {
        printf("overflow");
        return;
    }

    if (front == -1 && rear == -1)
    {
        front = 0;
        rear = 0;
    }
    else{
        rear = rear +1;
    }
    queue[rear] = n;
    printf("inserted element = %d\n", queue[rear]);
}

int delete_element()
{
    if(front == -1 && rear == -1)
    {
        printf("underflow");
        return 0;
    }
    int val = queue[front];
    if (front == rear)
    {
        front = -1;
        rear = -1;
    }
    else
    {
        front = front + 1;
    }
    return val;
}
```

```c
        }
        queue[rear] = n;
        printf("inserted element = %d\n", queue[rear]);
    }

    int delete_element()
    {
        if(front == -1 && rear == -1)
        {
            printf("underflow");
            return 0;
        }
        int val = queue[front];
        if (front == rear)
        {
            front = -1;
            rear = -1;
        }
        else
        {
            front = front + 1;
        }
        return val;
    }
    void display()
    {
        if(front == -1 && rear == -1)
        {
            printf("queue is empty");
            return;
        }
        printf("Queue:\n");
        for (int i = front; i <= rear; i++)
        {
            printf("%d\t", queue[i]);
        }
    }
```

```
enter your choice to perform linear queue operation:
enter:
1.inserting element
2.deleting element
3.displaying queue elements
4.exit
2
underflow
enter your choice to perform linear queue operation:
enter:
1.inserting element
2.deleting element
3.displaying queue elements
4.exit
3
queue is empty
enter your choice to perform linear queue operation:
enter:
1.inserting element
2.deleting element
3.displaying queue elements
4.exit
1
enter the element to insert:1
inserted element = 1

enter your choice to perform linear queue operation:
enter:
1.inserting element
2.deleting element
3.displaying queue elements
4.exit
1
enter the element to insert:2
inserted element = 2

enter your choice to perform linear queue operation:
enter:
1.inserting element
2.deleting element
3.displaying queue elements
4.exit
3
Queue:
1       2
enter your choice to perform linear queue operation:
enter:
1.inserting element
2.deleting element
3.displaying queue elements
4.exit
2
deleted element is 1

enter your choice to perform linear queue operation:
enter:
1.inserting element
2.deleting element
3.displaying queue elements
4.exit
2
deleted element is 2
```

```
D:\1BM23CS245\linear_queue\bin\Debug\linear_queue.exe

enter:
1.inserting element
2.deleting element
3.displaying queue elements
4.exit
2
deleted element is 1

enter your choice to perform linear queue operation:
enter:
1.inserting element
2.deleting element
3.displaying queue elements
4.exit
2
deleted element is 2

enter your choice to perform linear queue operation:
enter:
1.inserting element
2.deleting element
3.displaying queue elements
4.exit
3
queue is empty
enter your choice to perform linear queue operation:
enter:
1.inserting element
2.deleting element
3.displaying queue elements
4.exit
1
enter the element to insert:4
inserted element = 4

enter your choice to perform linear queue operation:
enter:
1.inserting element
2.deleting element
3.displaying queue elements
4.exit
1
enter the element to insert:5
inserted element = 5

enter your choice to perform linear queue operation:
enter:
1.inserting element
2.deleting element
3.displaying queue elements
4.exit
3
Queue:
4       5
enter your choice to perform linear queue operation:
enter:
1.inserting element
2.deleting element
3.displaying queue elements
4.exit
```