Lab work on QUEUE sequence

If the sequence of operations – DEQUEUE, ENQUEUE (m), ENQUEUE (e), DEQUEUE, ENQUEUE (h), ENQUEUE (e), ENQUEUE (d), DEQUEUE, ENQUEUE (i), ENQUEUE (a), DEQUEUE are performed on a Queue, therefore find out the REAR and FRONT value [Capacity of Queue is 3. (Circular Queue + simple queue)

1

CIRCULAR QUEUE

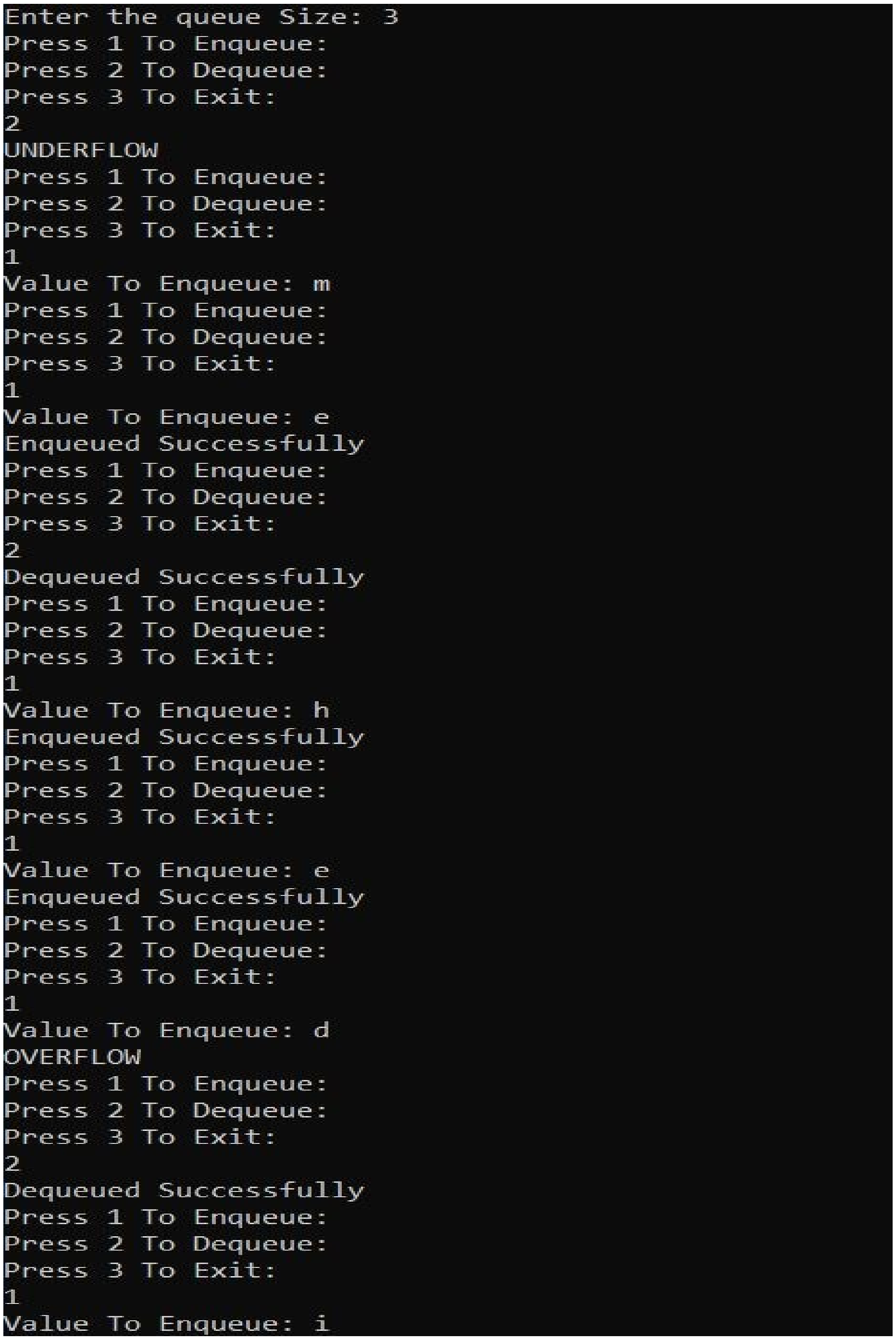
```
#include<stdio.h>

int main()

{

   int rear=-1,front=-1,count=0,i,n,j;

   char value;

   printf(“Enter the queue Size: “);

   scanf(“%d”, &n);

   char arr[n];

   while(1)

   {

      printf(“Press 1 To Enqueue: \n”);

      printf(“Press 2 To Dequeue: \n”);

      printf(“Press 3 To Exit: \n”);
```

```
scanf(“%d”, & i);
if(i==3)
{
    printf(“Final Outcome Of Queue= “);
    for(j=0; j<n; j++)
    {
        printf(“ %c “,arr[j]);
    }
    printf(“\nRear= %d\n”,rear);
    printf(“Front= %d\n”,front);
    break;
}
else if(i==1)
{
    printf(“Value To Enqueue: “);
    getchar();
    scanf(“%c”,& value);
    if(count==n)
    {
        printf(“OVERFLOW\n”);
    }
    else
    {
        if(rear==-1)
        {
            rear=0;
            front=0;
            arr[rear]=value;
            count++;
        }
        else
        {
            if(rear==n-1)
            {
                rear=0;
                arr[rear]=value;
                count++;
            }
            else
            {
                rear++;
                arr[rear]=value;
                count++;
            }
            printf("Enqueued Successfully\n");
        }
    }
}
```

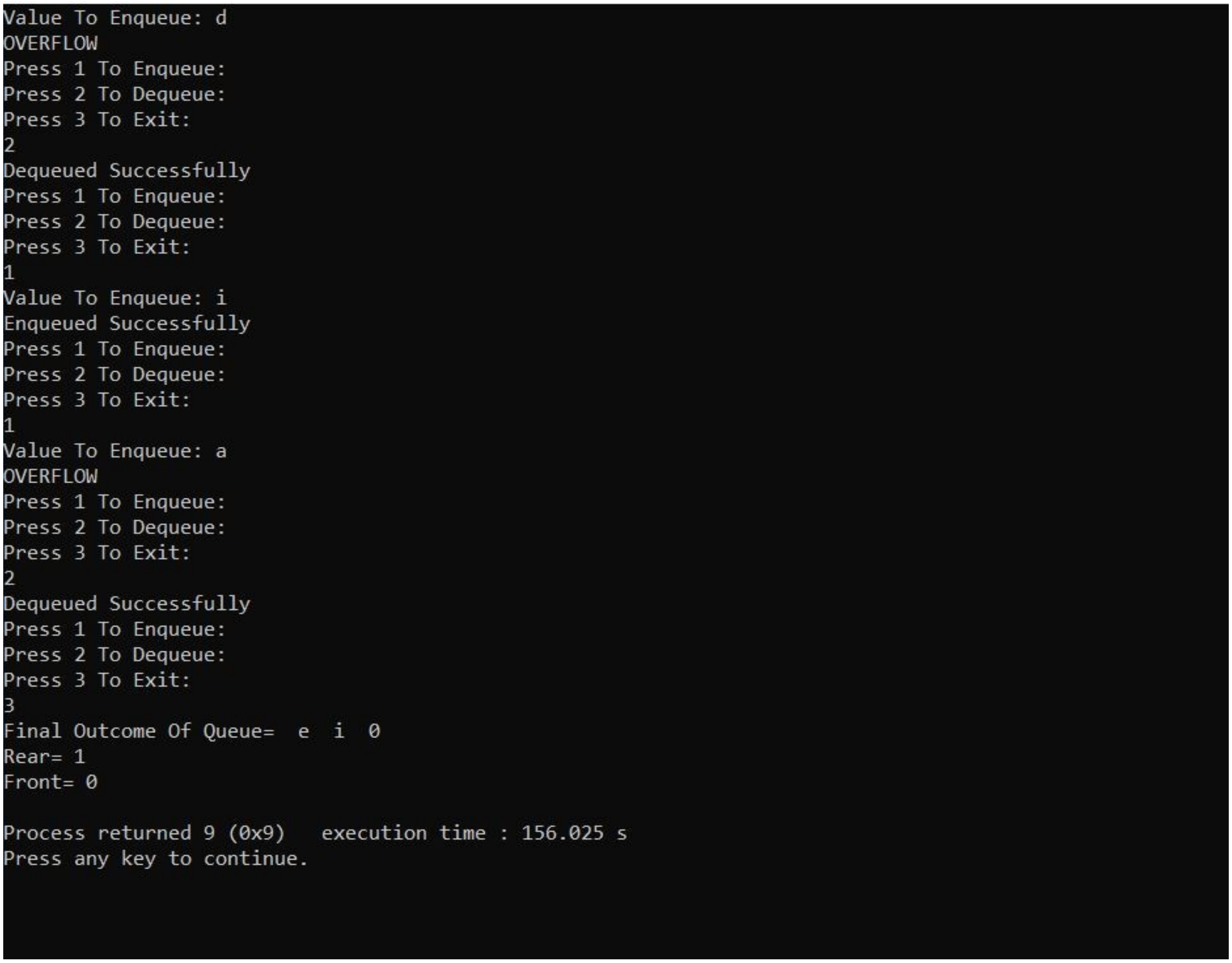
```
else
{
    if(count==0)
    {
        printf(“UNDERFLOW\n”);
    }
    else
    {
        arr[front]= 48;
        count--;
        if(count==0)
        {
            front=-1;
            rear=-1;
        }
        else
        {
            if(front==n-1)
            {
                front=0;
            }
            else
            {
                front++;
```

```
                    }
                }
                printf(“Dequeued Successfully\n”);
            }
        }
    }

}
```

Final outcome



```
Enter the queue Size: 3
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
2
UNDERFLOW
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
1
Value To Enqueue: m
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
1
Value To Enqueue: e
Enqueued Successfully
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
2
Dequeued Successfully
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
1
Value To Enqueue: h
Enqueued Successfully
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
1
Value To Enqueue: e
Enqueued Successfully
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
1
Value To Enqueue: d
OVERFLOW
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
2
Dequeued Successfully
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
1
Value To Enqueue: i
```



```
Value To Enqueue: d
OVERFLOW
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
2
Dequeued Successfully
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
1
Value To Enqueue: i
Enqueued Successfully
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
1
Value To Enqueue: a
OVERFLOW
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
2
Dequeued Successfully
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
3
Final Outcome Of Queue=  e  i  0
Rear= 1
Front= 0

Process returned 9 (0x9)   execution time : 156.025 s
Press any key to continue.
```

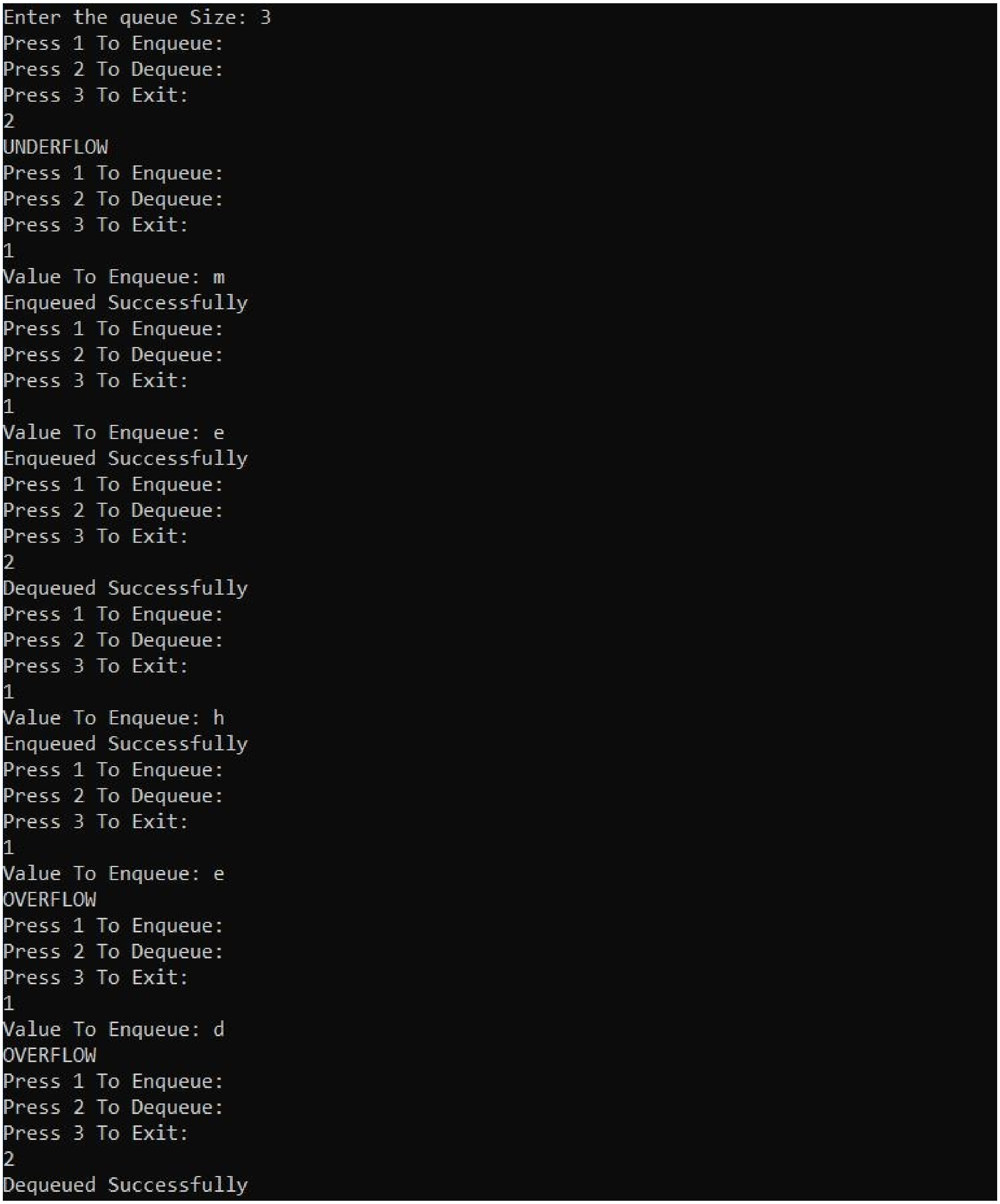
2

SIMPLE QUEUE

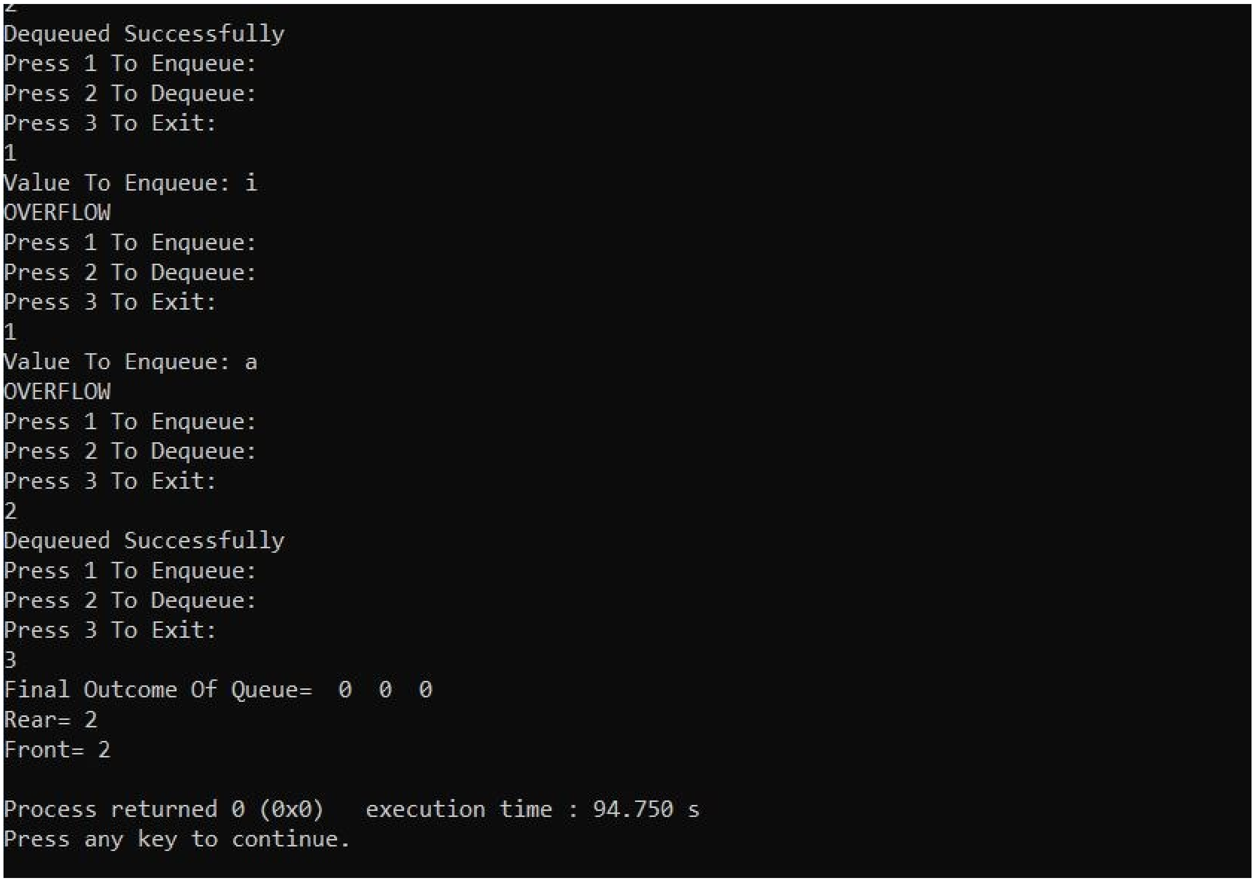
```
#include<stdio.h>

int main()

{

   int rear=-1,front=-1,i,n,j;

   char value;

   printf("Enter the queue Size: ");
```

```
scanf("%d", &n);
char arr[n];
while(1)
{
    printf("Press 1 To Enqueue: \n");
    printf("Press 2 To Dequeue: \n");
    printf("Press 3 To Exit: \n");
    scanf("%d", & i);
    if(i==3)
    {
        printf("Final Outcome Of Queue= ");
        for(j=0; j<n; j++)
        {
            printf(" %c ",arr[j]);
        }
        printf("\nRear= %d\n",rear);
        printf("Front= %d\n",front);
        break;
    }
    else if(i==1)
    {
        printf("Value To Enqueue: ");
        getchar();
        scanf("%c",& value);
        if(rear==n-1)
        {
            printf("OVERFLOW\n");
        }
        else
        {
            if(rear==-1)
            {
                rear=0;
                front=0;
                arr[rear]=value;
            }

                else
                {
                    rear++;
                    arr[rear]=value;
                }
                printf("Enqueued Successfully\n");
            }
        }

else
    {
        if(front==-1)
        {
            printf("UNDERFLOW\n");
        }
        else
        {
           arr[front]= 48;
            if(front==n-1)
                front=n-1;
            else
            front++;

            printf("Dequeued Successfully\n");
            }

        }
}
}
```

Final outcome



```
Enter the queue Size: 3
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
2
UNDERFLOW
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
1
Value To Enqueue: m
Enqueued Successfully
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
1
Value To Enqueue: e
Enqueued Successfully
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
2
Dequeued Successfully
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
1
Value To Enqueue: h
Enqueued Successfully
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
1
Value To Enqueue: e
OVERFLOW
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
1
Value To Enqueue: d
OVERFLOW
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
2
Dequeued Successfully
```



```
2
Dequeued Successfully
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
1
Value To Enqueue: i
OVERFLOW
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
1
Value To Enqueue: a
OVERFLOW
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
2
Dequeued Successfully
Press 1 To Enqueue:
Press 2 To Dequeue:
Press 3 To Exit:
3
Final Outcome Of Queue=  0  0  0
Rear= 2
Front= 2

Process returned 0 (0x0)   execution time : 94.750 s
Press any key to continue.
```