

İSİM: ABDOU VALERİO FOMA KENFACK

BÖLÜM: BİLGİSAYAR MÜHENDİSLİĞİ

SINIF: 2

KUYRUK ÖDEVİ

Banka kuyruğu örneğini ele alıp Banka işlemlerini kendi içerisinde üç öncelik grubuna ayırınız. Her yeni gelen kişiyi iş seçimine göre otomatik olarak ait olduğu grubun en sonuna ekleyiniz (enqueue). Her eleman alımında (dequeue) ise kuyruktaki en yüksek seviyeli grubun ilk elemanını alınız. Bunu hem dizi mantığı hem de Linked list mantığı ile yaparak tek Main içinde fonksiyonları çağırıp çalıştırınız.

KOD

```
public class QueueNode
{
    public string Name;
    public int ID;
    public QueueNode Next;

    public QueueNode (int id, string name)
    {
        ID = id;
        Name = name;
        Next = null;
    }
}
public class QueueLinklist
{
    private QueueNode[] heads;
    private QueueNode[] tails;
    int PRIORITY;
    public QueueLinklist(int priority)
    {
        PRIORITY = priority;
        heads = new QueueNode[PRIORITY];
        tails = new QueueNode[PRIORITY];
    }
    public void enqueue(int number, string name,int group)
    {
        int index = group - 1;
        QueueNode newNode = new QueueNode (number,name);
        if(heads[index] == null)
```

```

        {
            heads[index] = newNode;
            tails[index] = newNode;
        }
        else
        {
            tails[index].Next = newNode;
            tails[index] = newNode;
        }
        Console.WriteLine($"{number} numaralı {name} müşterisi başarıyla eklendi");
    }
    public void dequeue()
    {
        for(int i = 0; i < PRIORITY; i++)
        {
            if( heads[i] != null)
            {
                QueueNode dequeueNode = heads[i];
                heads[i] = heads[i].Next;
                if( heads[i] == null)
                {
                    tails[i] = null;
                }
                Console.WriteLine($"Öncelik {i+1} : {dequeueNode.ID} numaralı
{dequeueNode.Name} müşterisi başarıyla kaldırıldı");
                return;
            }
        }
        Console.WriteLine("Kuyruk boş çıkacak müşteri yoktur");
    }
}
public class CustomerInfo
{
    public int ID;
    public string Name;
}
public class QueueArray
{
    private int[] on;
    private int[] arka;
    private int[] elemanSayisi;
    private int PRIORITY;
    private int MAX;
    private CustomerInfo[,] T;

    public QueueArray(int priority, int max)
    {
        PRIORITY = priority;
        MAX = max;
        on = new int[PRIORITY];
        arka = new int[PRIORITY];
        elemanSayisi = new int[PRIORITY];
        T = new CustomerInfo[PRIORITY,MAX];
    }

    public void enqueue(int id, string name, int group)
    {

```

```

        int index = group - 1;
        if (elemanSayisi[index] == MAX)
        {
            Console.WriteLine("Kuyruk doludur");
            return;
        }
        T[index, arka[index] ] = new CustomerInfo { ID = id, Name = name };
        arka[index]++;
        if (arka[index] == MAX)
        {
            arka[index] = 0;
        }
        elemanSayisi[index]++;
        Console.WriteLine($"Öncelik {group} :{id} numaralı {name} müşterisi
başarıyla eklendi");
    }
    public void dequeue()
    {
        for(int i = 0; i < PRIORITY; i++)
        {
            if (elemanSayisi[i] != 0)
            {
                int id = -1;
                id = on[i];
                on[i]++;
                if (on[i] == MAX)
                {
                    on[i] = 0;
                }
                string isim = T[i, id].Name;
                T[i, id] = null;
                elemanSayisi[i]--;
                Console.WriteLine($" {isim} müşterisi başarıyla kaldırıldı");
                return;
            }
        }
        Console.WriteLine("Kuyruk boş çıkacak müşteri yoktur");
    }
}
class Program
{
    static void Main()
    {
        Console.WriteLine("Bağlantılı listeye dayalı bir kuyruk\n");
        QueueLinklist queueLinklist = new QueueLinklist(3);
        queueLinklist.enqueue(2508, "Yusuf", 2);
        queueLinklist.enqueue(2511, "Valerio", 2);
        queueLinklist.enqueue(2579, "Abdoulah", 3);
        queueLinklist.enqueue(2568, "Foma", 1);
        Console.WriteLine();
        queueLinklist.dequeue(); // En yüksek öncelik (grup 1)
        queueLinklist.dequeue(); // Sonraki grup 2 ögesi
        queueLinklist.dequeue(); // Başka grup 2
        queueLinklist.dequeue(); // Grup 3
        queueLinklist.dequeue(); // Boş
    }
}

```

```
Console.WriteLine("*****");
");
```

```
    Console.WriteLine("Dize tabanlı bir kuyruk\n");
    QueueArray queueArray = new QueueArray(3,10);
    queueArray.enqueue(2408, "Yunus", 2);
    queueArray.enqueue(2411, "Gul", 1);
    queueArray.enqueue(2479, "Yildiz", 3);
    queueArray.enqueue(2468, "Ahmed", 1);
    Console.WriteLine();
    queueArray.dequeue(); // En yüksek öncelik (grup 1)
    queueArray.dequeue(); // Başka grup 1 ögesi
    queueArray.dequeue(); // Sonraki grup 2
    queueArray.dequeue(); // Grup 3
    queueArray.dequeue(); // Boş
}
}
```

ÇIKTI

```
Microsoft Visual Studio Debug Console

2508 numarali Yusuf müsterisi basariyla eklendi
2511 numarali Valerio müsterisi basariyla eklendi
2579 numarali Abdoulah müsterisi basariyla eklendi
2568 numarali Foma müsterisi basariyla eklendi

Öncelik 1 : 2568 numarali Foma müsterisi basariyla kaldırildi
Öncelik 2 : 2508 numarali Yusuf müsterisi basariyla kaldırildi
Öncelik 2 : 2511 numarali Valerio müsterisi basariyla kaldırildi
Öncelik 3 : 2579 numarali Abdoulah müsterisi basariyla kaldırildi
Kuyruk bos çıkacak müşteri yoktur
*****
Dize tabanlı bir kuyruk

Öncelik 2 :2408 numarali Yunus müsterisi basariyla eklendi
Öncelik 1 :2411 numarali Gul müsterisi basariyla eklendi
Öncelik 3 :2479 numarali Yildiz müsterisi basariyla eklendi
Öncelik 1 :2468 numarali Ahmed müsterisi basariyla eklendi

Gul müsterisi basariyla kaldırildi
Ahmed müsterisi basariyla kaldırildi
Yunus müsterisi basariyla kaldırildi
Yildiz müsterisi basariyla kaldırildi
Kuyruk bos çıkacak müşteri yoktur
```