

**EGE UNIVERSITY**

**FACULTY OF ENGINEERING**

**COMPUTER ENGINEERING DEPARTMENT**

**204 DATA STRUCTURES (3+1)**

**2021–2022 FALL SEMESTER**

**PROJECT-2 REPORT**

**(List, Stack, Queue, PQ – Priority Queue Data Structures)**

**DELIVERY DATE**

04/01/2022

**PREPARED BY**

05200000030, Osman Bora Yılmazmış

İçindekiler

[1.a Bileşik Veri Yapısı için Ön Çalışma 2](#_Toc91168778)

[1.b Bileşik Veri Yapısı Kodlama ve Çalıştırma 2](#_Toc91168779)

[1.b.1 Kaynak Kod 2](#_Toc91168780)

[1.b.2 Ekran görüntüleri 2](#_Toc91168781)

[1.b.3 Veri Yapıları ve Açıklama 2](#_Toc91168782)

[1.c Bileşik Veri Yapısı Bilgi Çıkarma 2](#_Toc91168783)

[1.c.1 Kaynak Kod 2](#_Toc91168784)

[1.c.2 Ekran görüntüleri 2](#_Toc91168785)

[2.a Yığıt 2](#_Toc91168786)

[2.a.1 Kaynak Kod 2](#_Toc91168787)

[2.a.2 Ekran görüntüleri 2](#_Toc91168788)

[2.b Kuyruk 2](#_Toc91168789)

[2.b.1 Kaynak Kod 2](#_Toc91168790)

[2.b.2 Ekran görüntüleri 2](#_Toc91168791)

[3.a Öncelikli Kuyruk Oluşturma 2](#_Toc91168792)

[3.a.1 Kaynak Kod 2](#_Toc91168793)

[3.a.2 Ekran görüntüleri 2](#_Toc91168794)

[3.b ArrayList ve Dizi altyapılarının karşılaştırılması 2](#_Toc91168795)

[4.a Öncelikli Kuyruk Güncelleme 3](#_Toc91168796)

[4.b Ortalama İşlem Tamamlama Süresi 3](#_Toc91168797)

[4.b.1 Kaynak Kod 3](#_Toc91168798)

[4.b.2 Ekran görüntüleri 3](#_Toc91168799)

[4.c Öncelikli Kuyruk Tartışma 3](#_Toc91168800)

[5. Özdeğerlendirme Tablosu 3](#_Toc91168801)

LİSTE, YIĞIT, KUYRUK ve ÖNCELİKLİ KUYRUK VERİ YAPILARI

//The platform, version, and programming language used

## 1.a Bileşik Veri Yapısı için Ön Çalışma

//1-a maddesi için kurşunkalemle A4 kağıda yazılmış algoritmanın görüntüsünü ekleyiniz

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

## 1.b Bileşik Veri Yapısı Kodlama ve Çalıştırma

### 1.b.1 Kaynak Kod

//1-b maddesi için yazmış olduğunuz kod parçasını buraya ekleyiniz

static Random Rastgele = new Random();

static string[] yemeklerim = {"nohutlu tavuk", "dana rosto", "spagetti ", "yoğurtlu makarna","bulgur pilavı", "pilav","salçalı tavuk","hatay dürüm","izmir kumrusu","küfte","mantı","hotdog" ,"Sucuklu Şahanem"};

static void Main(string[] args)

{

Console.BackgroundColor = ConsoleColor.White;

Console.Clear();

Console.ForegroundColor = ConsoleColor.Black;

Console.WriteLine("Merhaba");

//ARRAY BÖLÜMÜ

string[] MahalleAdı = { "Özkanlar", "Evka 3", "Atatürk","Erzene", "Kazımdirik", "Mevlana", "Doğanlar", "Ergene" };

int[] TeslimatSayısı = { 5, 2, 7, 2, 7, 3,0, 1 };

ArrayList Moto\_Kurye = new ArrayList();

for (int i = 0; MahalleAdı.Length> i; i++)

{

MahalleSınıfı Mahalle = new MahalleSınıfı(MahalleAdı[i]);

for(int q = 0; TeslimatSayısı[i]> q; q++)

{

string rastgeleyemek = yemeklerim[Rastgele.Next(0, yemeklerim.Length)];

int rastgeleyemeksayisi = Rastgele.Next(1,10);

TeslimatSinif Teslimat = new TeslimatSinif(rastgeleyemek,rastgeleyemeksayisi);

Mahalle.teslimatlar.Add(Teslimat);

}

Moto\_Kurye.Add(Mahalle);

}

### 1.b.2 Ekran görüntüleri

//1-b maddesi için üretilen konsol/ekran görüntüsünü buraya ekleyiniz

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

### 1.b.3 Veri Yapıları ve Açıklama

//1-b maddesi için kullanmış olduğunuz veri yapılarını burada listeleyip kısaca açıklayınız,

Main metodunun içine Mahalle adlarını tutan liste ve teslimat sayılarını tutan liste oluşturulur. Daha sonra ArrayList tipinde Moto\_Kurye listesi oluşturulur.

Sonra iç içe for döngüsünün ilk for’unda MahalleSınıfı tipinde mahallelerimiz her döngüde listemize atılır daha sonra 2. For döngümüzde önce rastgele yemek oluşturulur. Daha sonra rastgele yemek sayisi oluşturulup , TeslimatSınıfı tipinde Teslimat değişkenine oluşturulan bilgiler atanır daha sonra oluşan bu teslimat mahallemizn içerisine gönderilir.

En sonunda Moto\_Kurye arrayine add metoduyla Mahalle gönderilir ve bu da bütün teslimat bilgilerine sahiptir.

## 1.c Bileşik Veri Yapısı Bilgi Çıkarma

### 1.c.1 Kaynak Kod

//1c maddesi için yazmış olduğunuz kodları buraya ekleyiniz

Console.WriteLine(" ");

Console.WriteLine(" Mahalle Sayisi : " + Moto\_Kurye.Count);

int sayac = 0;

for (int i = 0; Moto\_Kurye.Count > i; i++)

{

MahalleSınıfı mahalle = (MahalleSınıfı)Moto\_Kurye[i];

sayac += mahalle.teslimatlar.Count;

}

Console.WriteLine("Teslimat Sayimiz : " + sayac);

Console.ReadLine();

### 

### 71.c.2 Ekran görüntüleri

//1-c maddesi için üretilen konsol/ekran görüntüsünü buraya ekleyiniz



## 2.a Yığıt

### 2.a.1 Kaynak Kod

//2-a maddesi için yazmış olduğunuz kodları buraya ekleyiniz

//STACK BÖLÜMÜ

Console.WriteLine(" ");

Console.WriteLine("STACKLE OLUSTURULAN BÖLÜM");

Stack<MahalleSınıfı> Yığıt = new Stack<MahalleSınıfı>();

for (int i = 0; Moto\_Kurye.Count > i; i++)

{

Yığıt.Push((MahalleSınıfı)Moto\_Kurye[i]);

}

while(Yığıt.Count != 0)

{

MahalleSınıfı mahalle = Yığıt.Pop();

Console.WriteLine(" ");

Console.WriteLine(" Mahallenin Adi : " + mahalle.MahalleAdi);

for ( int i = 0; mahalle.teslimatlar.Count > i; i++)

{

TeslimatSinif teslimat = mahalle.teslimatlar[i];

Console.WriteLine("gönderilen yemek ve sayisi: " + teslimat.yemek + " " + teslimat.yemeksayisi);

}

}

### 2.a.2 Ekran görüntüleri

//2-a maddesi için üretilen konsol/ekran görüntüsünü buraya ekleyiniz

metin, tablo içeren bir resim

Açıklama otomatik olarak oluşturuldu

## 2.b Kuyruk

### 2.b.1 Kaynak Kod

//2-b maddesi için yazmış olduğunuz kodları buraya ekleyiniz

//KUYRUK BÖLÜMÜ

Console.WriteLine(" ");

Console.WriteLine("KUYRUKLA OLUSTURULAN BÖLÜM");

Queue<MahalleSınıfı> Kuyruk = new Queue<MahalleSınıfı>();

for (int i = 0; Moto\_Kurye.Count > i; i++)

{

Kuyruk.Enqueue((MahalleSınıfı)Moto\_Kurye[i]);

}

while (Kuyruk.Count != 0)

{

MahalleSınıfı mahalle = Kuyruk.Dequeue();

Console.WriteLine(" ");

Console.WriteLine(" Mahallenin Adi : " + mahalle.MahalleAdi);

for (int i = 0; mahalle.teslimatlar.Count > i; i++)

{

TeslimatSinif teslimat = mahalle.teslimatlar[i];

Console.WriteLine("gönderilen yemek ve sayisi: " + teslimat.yemek + " " + teslimat.yemeksayisi);

}

}

### 2.b.2 Ekran görüntüleri

//2-b maddesi için üretilen konsol/ekran görüntüsünü buraya ekleyiniz

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

## 3.a Öncelikli Kuyruk Oluşturma

### 3.a.1 Kaynak Kod

//3-a maddesi için yazmış olduğunuz kodları buraya ekleyiniz

// PQ BÖLÜMÜ

Console.WriteLine(" ");

Console.WriteLine("PQ OLUSTURULAN BÖLÜM");

ÖncelikliKuyruk OncelikliKuyruk = new ÖncelikliKuyruk();

for (int i = 0; Moto\_Kurye.Count > i; i++)

{

OncelikliKuyruk.ElemanEkle((MahalleSınıfı)Moto\_Kurye[i]);

}

while (!OncelikliKuyruk.BosMu())

{

MahalleSınıfı mahalle = OncelikliKuyruk.ElemanSil();

Console.WriteLine(" ");

Console.WriteLine(" Mahallenin Adi : " + mahalle.MahalleAdi);

for (int i = 0; mahalle.teslimatlar.Count > i; i++)

{

TeslimatSinif teslimat = mahalle.teslimatlar[i];

Console.WriteLine("gönderilen yemek ve sayisi: " + teslimat.yemek + " " + teslimat.yemeksayisi);

}

}

### 3.a.2 Ekran görüntüleri

//3-a maddesi için üretilen konsol/ekran görüntüsünü buraya ekleyiniz

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

## 3.b ArrayList ve Dizi altyapılarının karşılaştırılması

//3-b maddesi için açıklama

Projemizde dizi kullanılırsa eğer diziye eleman ekleme ve çıkarma daha zor hale gelir ve liste kullanılması bu durumu daha kolay hale getirir. Bununla beraber cok fazla mahalle varsa liste kullanımı diziye göre daha avantajlı olur .

## 4.a Öncelikli Kuyruk Güncelleme

//4-a maddesi için oluşturduğunuz kodu buraya ekleyiniz

//KUYRUK BÖLÜMÜ

class Kuyruk

{

List<int> müşteriler = new List<int>();

public Kuyruk() { }

public Kuyruk (int[] dizi)

{

for (int i = 0; dizi.Length > i; i++)

{

müşteriler.Add((int)dizi[i]);

}

}

public bool BosMu()

{

if (müşteriler.Count == 0)

{

return true;

}

return false;

}

public void ElemanEkle(int Müsteri)

{

müşteriler.Add(Müsteri);

}

public int ElemanSil()

{

int min = müşteriler[0];

müşteriler.Remove(min);

return min;

}

}

//PQ BÖLÜMÜ

class ÖncelikliKuyruk

{

List<int> müşteriler = new List<int>();

public ÖncelikliKuyruk() { }

public ÖncelikliKuyruk(int [] dizi)

{

for (int i = 0; dizi.Length > i; i++)

{

müşteriler.Add((int)dizi[i]);

}

}

public bool BosMu()

{

if (müşteriler.Count == 0)

{

return true;

}

return false;

}

public void ElemanEkle(int Müsteri)

{

müşteriler.Add(Müsteri);

}

public int ElemanSil()

{

int min = müşteriler[0];

for (int i = 0; müşteriler.Count > i; i++)

{

if (müşteriler[i] < min)

{

min = müşteriler[i];

}

}

müşteriler.Remove(min);

return min;

}

}

## 4.b Ortalama İşlem Tamamlama Süresi

### 4.b.1 Kaynak Kod

//4-b maddesi için yazmış olduğunuz kodları buraya ekleyiniz

static void Main(string[] args)

{

Console.BackgroundColor = ConsoleColor.White;

Console.Clear();

Console.ForegroundColor = ConsoleColor.Black;

Console.WriteLine("Merhaba");

Console.WriteLine("PQ Bölümünde Gerçekleşen çözüm");

Console.WriteLine(" ");

int [] sepetlerim = { 6, 7, 2, 1, 12, 5, 3, 7, 4, 2 };

ÖncelikliKuyruk müşterilistesi = new ÖncelikliKuyruk(sepetlerim);

int üc = 3;

int vakitsayar = 0;

double ortalamBeklemeSüresi = 0;

for(int i = 0; i< sepetlerim.Length; i++)

{

int ürünsayisi = 0;

ürünsayisi = müşterilistesi.ElemanSil();

vakitsayar += üc \* ürünsayisi;

Console.WriteLine("TOPLAM HARCANAN VAKİT : " + vakitsayar);

ortalamBeklemeSüresi += vakitsayar;

}

Console.WriteLine("ORTALAMA BEKLEME SURESİ : " + (ortalamBeklemeSüresi/ sepetlerim.Length));

Console.ReadLine();

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

Console.WriteLine(" ");

Console.WriteLine("Kuyruk Bölümünde gerçekleşen çözüm");

Kuyruk müşterilistesi2 = new Kuyruk(sepetlerim);

double ortalamBeklemeSüresi2 = 0;

int vakitsayar2 = 0;

for (int i = 0; i < sepetlerim.Length; i++)

{

int ürünsayisi = 0;

ürünsayisi = müşterilistesi2.ElemanSil();

vakitsayar2 += üc \* ürünsayisi;

Console.WriteLine("TOPLAM HARCANAN VAKİT : " + vakitsayar2);

ortalamBeklemeSüresi2 += vakitsayar2;

}

Console.WriteLine("ORTALAMA BEKLEME SURESİ : " + (ortalamBeklemeSüresi2 / sepetlerim.Length));

Console.ReadLine();

}

### 4.b.2 Ekran görüntüleri

//4-b maddesi için üretilen konsol/ekran görüntüsünü buraya ekleyiniz

tablo içeren bir resim

Açıklama otomatik olarak oluşturuldu  
4.b.3 Sözel olarak karşılaştırma

PQ da ortalama bekleme süresi kuyruk bölümündeki ortlama bekleme süresinden daha fazladır

## 4.c Öncelikli Kuyruk Tartışma

//4-c maddesi için açıklamanızı buraya yazınız

Öncelikli kuyruk kuyruk yapısından daha etkili olmasına rağmen gerçek hayatta uzun sürebilecek işlemlerde kullanılması sorunlara yol açacaktır.

Bu nedenle kısa sürecek işlemlerde öncelikli kuyruk kullanılabilecekken daha uzun süreli işlemlerde kuyruk yapısının kullanılması daha uygundur.

# 5. Özdeğerlendirme Tablosu

**Özdeğerlendirme Tablosu**

|  |  |  |  |
| --- | --- | --- | --- |
| **Proje 2 Maddeleri** | **Puan** | **Tahmini Not** | **Açıklama** |
| **1 a) A4 Ön çalışma** | **20** | **20** | **yapıldı** |
| **1 b) Kaynak kod, ekran görüntüsü, veri yapısının elemanlarının listelenmesi** | **20** | **20** | **yapıldı** |
| **1 c) Kaynak kodlar, Liste sayısı, veri yapısındaki toplam teslimat sayısı** | **5** | **5** | **yapıldı** |
| **2 a) Yığıt kaynak kod ve ekran görüntüleri** | **5** | **5** | **yapıldı** |
| **2 b) Kuyruk kaynak kod ve ekran görüntüleri** | **5** | **5** | **yapıldı** |
| **3 a) Öncelikli Kuyruk kod ve ekran görüntüleri** | **10** | **10** | **yapıldı** |
| **3 b) ArrayList ve Dizi altyapılarının karşılaştırılması** | **5** | **5** | **yapıldı** |
| **4) Kod, sonuçlar tablosu, ekran görüntüleri, karşılaştırma ve soruların cevapları.** | **20** | **20** | **yapıldı** |
| **5) Özdeğerlendirme Tablosu** | **10** | **10** | **yapıldı** |
| **Toplam** | **100** | **100** |  |

**Açıklama kısmında yapıldı, yapılmadı bilgisi ve hangi maddelerin nasıl yapıldığı (ve nelerin yapılmadığı / yapılamadığı) yazılmalıdır. Tahmini not kısmına da ilgili maddeden kaç almayı beklediğinizi yazmalısınız.**