

**EGE UNIVERSITY**

**FACULTY OF ENGINEERING**

**COMPUTER ENGINEERING DEPARTMENT**

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

**2022–2023 FALL SEMESTER**

**PROJECT-2 REPORT**

**LİSTE, YIĞIT, KUYRUK ve ÖNCELİKLİ KUYRUK VERİ YAPILARI**

**DELIVERY DATE**

29/11/2022

**PREPARED BY**

05210000291,Görkem Turkut

İçindekiler

[1) Türkiye’deki Milli Parklar Listesi 3](#_Toc118965473)

[1.a Milli Park Sınıfı Oluşturma 3](#_Toc118965474)

[1.a.1 Kodlar 3](#_Toc118965475)

[1.a.2 Açıklama 3](#_Toc118965476)

[1.b İki Elemanlı Bir Generic List Dizisi Oluşturma 3](#_Toc118965477)

[1.b.1 Kodlar 3](#_Toc118965478)

[1.c Dizideki Her Bir Listedeki Bilgilerin ve Elemanların Yüzölçümlerinin Toplamının Yazdırılması 3](#_Toc118965479)

[1.c.1 Kodlar 3](#_Toc118965480)

[1.c.2 Ekran görüntüleri 3](#_Toc118965481)

[2) YIĞIT ve KUYRUK 3](#_Toc118965482)

[2.a Milli Park Yığıtı Oluşturma 3](#_Toc118965483)

[2.a.1 Kaynak Kod 3](#_Toc118965484)

[2.a.2 Ekran Görüntüleri 3](#_Toc118965485)

[2.b Milli Park Kuyruk Veri Yapısı Oluşturma 3](#_Toc118965486)

[2.b.1 Kaynak Kod 3](#_Toc118965487)

[2.b.2 Ekran Görüntüleri 3](#_Toc118965488)

[3) Öncelikli Kuyruk 4](#_Toc118965489)

[3.a Öncelikli Kuyruk (sınıfı) Oluşturma 4](#_Toc118965490)

[3.a.1 Kaynak Kod 4](#_Toc118965491)

[3.a.2 Ekran görüntüleri 4](#_Toc118965492)

[4) Kuyruk ve Öncelikli Kuyruk 4](#_Toc118965493)

[4.a Kuyruk kullanarak süre hesaplanması 4](#_Toc118965494)

[4.a.1 Kaynak Kod 4](#_Toc118965495)

[4.a.2 Ekran görüntüleri 4](#_Toc118965496)

[4.b Öncelikli Kuyruk sınıfı kullanarak süre hesaplanması 4](#_Toc118965497)

[4.b.1 Kaynak Kod 4](#_Toc118965498)

[4.b.2 Ekran görüntüleri 4](#_Toc118965499)

[4.c Kuyruk ve Öncelikli Kuyruk Yapılarının Karşılaştırılması 4](#_Toc118965500)

[Öz değerlendirme Tablosu 5](#_Toc118965501)

# 1) Türkiye’deki Milli Parklar Listesi

IntelliJ,2022.1,Java

## 1.a Milli Park Sınıfı Oluşturma

### 1.a.1 Kodlar

public class MilliPark {

private int no;

private String milliParkAdı;

private String ilAdi;

private int yuzOlcumu;

private String tarih;

public MilliPark(int no, String milliParkAdı, String ilAdi, int yuzOlcumu, String tarih) {

this.no = no;

this.milliParkAdı = milliParkAdı;

this.ilAdi = ilAdi;

this.yuzOlcumu = yuzOlcumu;

this.tarih = tarih;

}

public int getNo() {

return no;

}

public void setNo(int no) {

this.no = no;

}

public String getMilliParkAdı() {

return milliParkAdı;

}

public void setMilliParkAdı(String milliParkAdı) {

this.milliParkAdı = milliParkAdı;

}

public String getIlAdi() {

return ilAdi;

}

public void setIlAdi(String ilAdi) {

this.ilAdi = ilAdi;

}

public int getYuzOlcumu() {

return yuzOlcumu;

}

public void setYuzOlcumu(int yuzOlcumu) {

this.yuzOlcumu = yuzOlcumu;

}

public String getTarih() {

return tarih;

}

public void setTarih(String tarih) {

this.tarih = tarih;

}

public String toString(){

return ("No:"+no+" Adı:"+milliParkAdı+" İl:"+ilAdi+" Yüz Ölçümü:"+yuzOlcumu+" Tarih:"+tarih);

}

}

### 1.a.2 Açıklama

MP numarası ve yüz ölçümü için int tipinde;ismi,tarihi,il adı için String tipinde değerler tutuldu.

## 1.b İki Elemanlı Bir Generic List Dizisi Oluşturma

### 1.b.1 Kodlar

List[] milliParklar={kucukMilliParklar,buyukMilliParklar};

## 1.c Dizideki Her Bir Listedeki Bilgilerin ve Elemanların Yüzölçümlerinin Toplamının Yazdırılması

### 1.c.1 Kodlar

public static void yazdir(List[] milliParklar,int toplam){ //Milli parkların yazdırıldığı metot.

for (int i=0;i<milliParklar.length;i++){

for (int j=0;j<milliParklar[i].size();j++){

System.*out*.println(milliParklar[i].get(j));

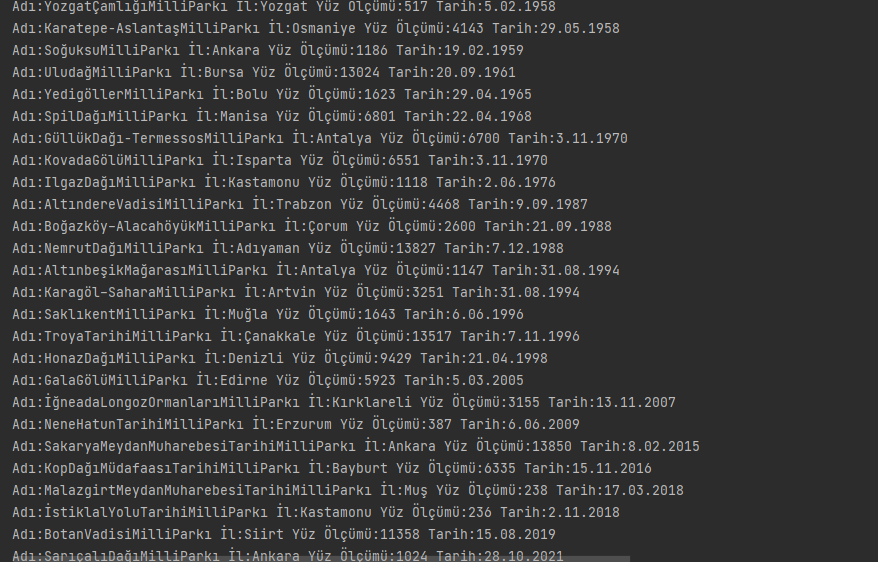
}

}

System.*out*.println("Yüzölçümleri toplamı:"+toplam);

}

### 1.c.2 Ekran görüntüleri



# 2) YIĞIT ve KUYRUK

IntelliJ,2022.1,Java

## 2.a Milli Park Yığıtı Oluşturma

### 2.a.1 Kaynak Kod

public class stack {

int kapasite;

int index; //Ekleme yapmak için kullanılan index.

MilliPark[] stack;

stack(int kapasite){

this.kapasite=kapasite;

this.stack= new MilliPark[kapasite];

index=-1;

}

public boolean isEmpty(){

return index < 0;

}

public void push(MilliPark data){ //ekleme

if (index==kapasite-1){

System.*out*.println("Yığıt dolu!");

}

else{

index++;

stack[index]=data;

}

}

public MilliPark pop(){ //silme

if (isEmpty()){

System.*out*.println("Stack boş");

return null;

}

else {

MilliPark data=stack[index];

index--;

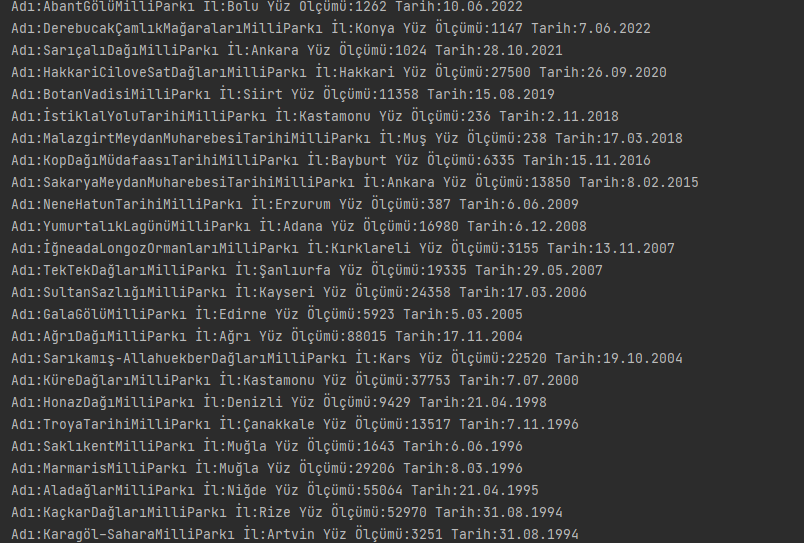
return data;

}

}

}

### 2.a.2 Ekran Görüntüleri



## 2.b Milli Park Kuyruk Veri Yapısı Oluşturma

### 2.b.1 Kaynak Kod

public class queue {

int index1; //Listeye ekleme yapmak için kullanılan index.

int index2; //Listeden silme yapmak için kullanılan index.

int kapasite;

MilliPark[] liste;

public queue(int kapasite){

this.kapasite=kapasite;

index1=-1;

index2=0;

liste=new MilliPark[kapasite];

}

public boolean isEmpty(){

return index1 < 0 || index2 == kapasite;

}

public void push(MilliPark data){

if (index1!=kapasite-1){

index1++;

liste[index1]=data;

}

else {

System.*out*.println("Overflow!");

}

}

public MilliPark pop(){

if (isEmpty()){

System.*out*.println("Underflow");

return null;

}

else {

MilliPark data=liste[index2];

index2++;

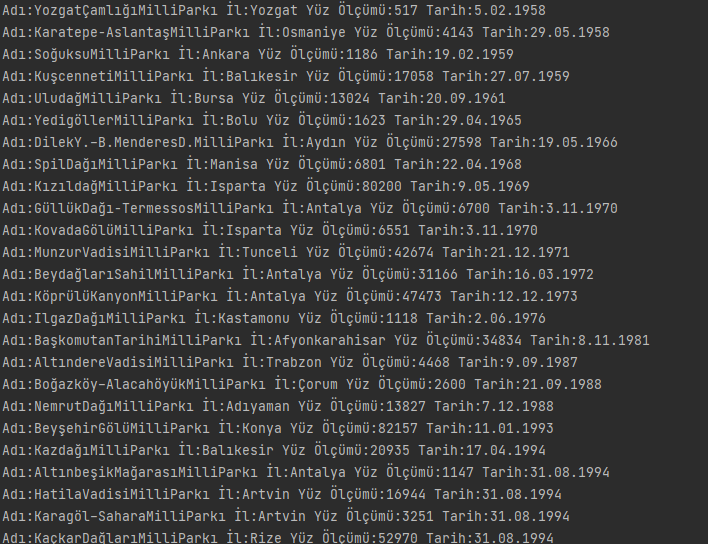
return data;

}

}

}

### 2.b.2 Ekran Görüntüleri



# 3) Öncelikli Kuyruk

## 3.a Öncelikli Kuyruk (sınıfı) Oluşturma

### 3.a.1 Kaynak Kod

import java.util.ArrayList;

import java.util.List;

public class PriorityQueue {

int size;

List<MilliPark> liste; //Generic List.

public PriorityQueue(){

size=0;

liste=new ArrayList<>();

}

public boolean isEmpty(){

return size < 1;

}

public void push(MilliPark data){

liste.add(data);

size++;

}

public MilliPark pop(){

int enKucukAlan =liste.get(0).getYuzOlcumu(); //İlk olarak en küçük mp'ye listenin başındaki elemanı atıyorız daha sonra listede gezinerek daha küçük bir eleman varsa değişkeni değiştiriyoruz.

MilliPark enKucukMilliPark=liste.get(0);

if (isEmpty()){

System.*out*.println("Underflow!");

return null;

}

else {

for (int i=0;i<size;i++){

int currentYuzOlcumu=liste.get(i).getYuzOlcumu();

if (currentYuzOlcumu<enKucukAlan){

enKucukAlan=currentYuzOlcumu;

enKucukMilliPark=liste.get(i);

}

}

liste.remove(enKucukMilliPark); //En küçük mp bulunduktan sonra listeden çıkarılıyor.

size--;

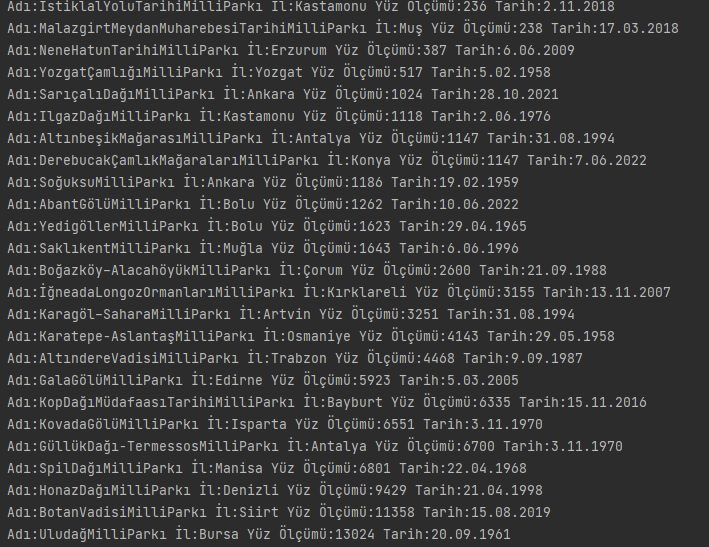
return enKucukMilliPark;

}

}

}

### 3.a.2 Ekran görüntüleri



# 4) Kuyruk ve Öncelikli Kuyruk

## 4.a Kuyruk kullanarak süre hesaplanması

### 4.a.1 Kaynak Kod

int urunAdeti=liste[index2]; //İlk sıradaki kişiden itibaren atama yapılır.

index2++;

musteriNo++;

toplamSure=toplamSure+3\*urunAdeti; //Numaralı müşteri için geçen sürenin hesaplandığı yer.

System.*out*.println(musteriNo+". müşteri:"+toplamSure);

herkesinToplamSuresi=herkesinToplamSuresi+toplamSure;

return toplamSure;

### 4.a.2 Ekran görüntüleri

## 4.b Öncelikli Kuyruk sınıfı kullanarak süre hesaplanması

### 4.b.1 Kaynak Kod

musteriNo++;

int enKucukDeger=liste.get(0); //En küçük değere ilk olarak listenin en başındaki kişiyi atıyoruz.

for (int i=0;i<liste.size();i++){ //Ardından listeyi gezerek daha en küçük değerden daha küçük bir değer varsa enKucukDeger'i değiştiriyoruz.

if (liste.get(i)<enKucukDeger){

enKucukDeger=liste.get(i);

}

}

int index=liste.indexOf(enKucukDeger); //En küçük değerin indexini buluyoruz.

liste.remove(index); //Bulunan indexe göre o müşteriyi sıradan çıkarıyoruz.

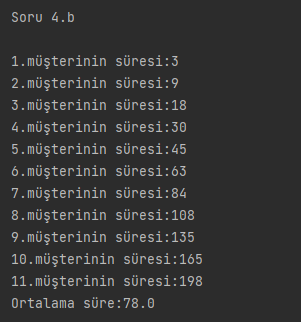
toplamSure=toplamSure+3\*enKucukDeger; //O müşteri için toplam işlem süresi.

System.*out*.println(musteriNo+".müşterinin süresi:"+toplamSure);

herkesinToplamSuresi=herkesinToplamSuresi+toplamSure;

return toplamSure;

### 4.b.2 Ekran görüntüleri



## 4.c Kuyruk ve Öncelikli Kuyruk Yapılarının Karşılaştırılması

Öncelikli kuyrukta ortalama süre daha az çıktığı gözlemlenmiştir.Bunun yanı sıra PQ da ekstradan listede arama yapıldığı için işlemler biraz daha uzuyor ve daha büyük listelerde çalışıldığında bu durum programın performansını etkileyebilir fakat normal sırada ilk giren ilk çıkar mantığıyla çalışıldığı için işlemler daha hızlı şekilde halledilebilir.

# Öz değerlendirme Tablosu

|  |  |  |  |
| --- | --- | --- | --- |
| **Proje 2 Maddeleri** | **Puan** | **Tahmini Not** | **Açıklama** |
| **1 a) Sınıf Oluşturma** | **5** | 5 | Yapıldı. |
| **1 b) Bileşik veri yapısının oluşturulması ve elemanların eklenmesi; Kaynak kod, ekran görüntüsü.** | **15** | 15 | Yapıldı |
| **1 c) Veri yapısının elemanlarının listelenmesi ve 2 adet yüzölçümü değeri bulunması; Kaynak kodlar, ekran görüntüsü.** | **10** | 10 | Yapıldı |
| **2 a) Yığıt kaynak kod ve ekran görüntüleri** | **10** | 10 | Yapıldı |
| **2 b) Kuyruk kaynak kod ve ekran görüntüleri** | **10** | 10 | Yapıldı |
| **3 ) Öncelikli Kuyruk kod ve ekran görüntüleri** | **20** | 20 | Yapıldı |
| **4) Kod, sonuçlar tablosu, ekran görüntüleri, karşılaştırma ve soruların cevapları.** | **20** | 20 | Yapıldı |
| **5) Özdeğerlendirme Tablosu** | **10** | 10 | Yapıldı |
| **Toplam** | **100** | 100 | Yapıldı |

**Açıklama kısmında yapıldı, yapılmadı bilgisi ve hangi maddelerin nasıl yapıldığı veya neden yapılamadığı kısaca yazılmalıdır. Tahmini not kısmına da ilgili maddeden kaç almayı beklediğinizi yazmalısınız.**

**Not: Raporu teslim edilmeyen projeler değerlendirmeye alınmayacaktır.**