

CSE 222 – HW7

HASAN MEN 131044009

<https://github.com/hmenn/CSE222-HW7-2016-Simulation>

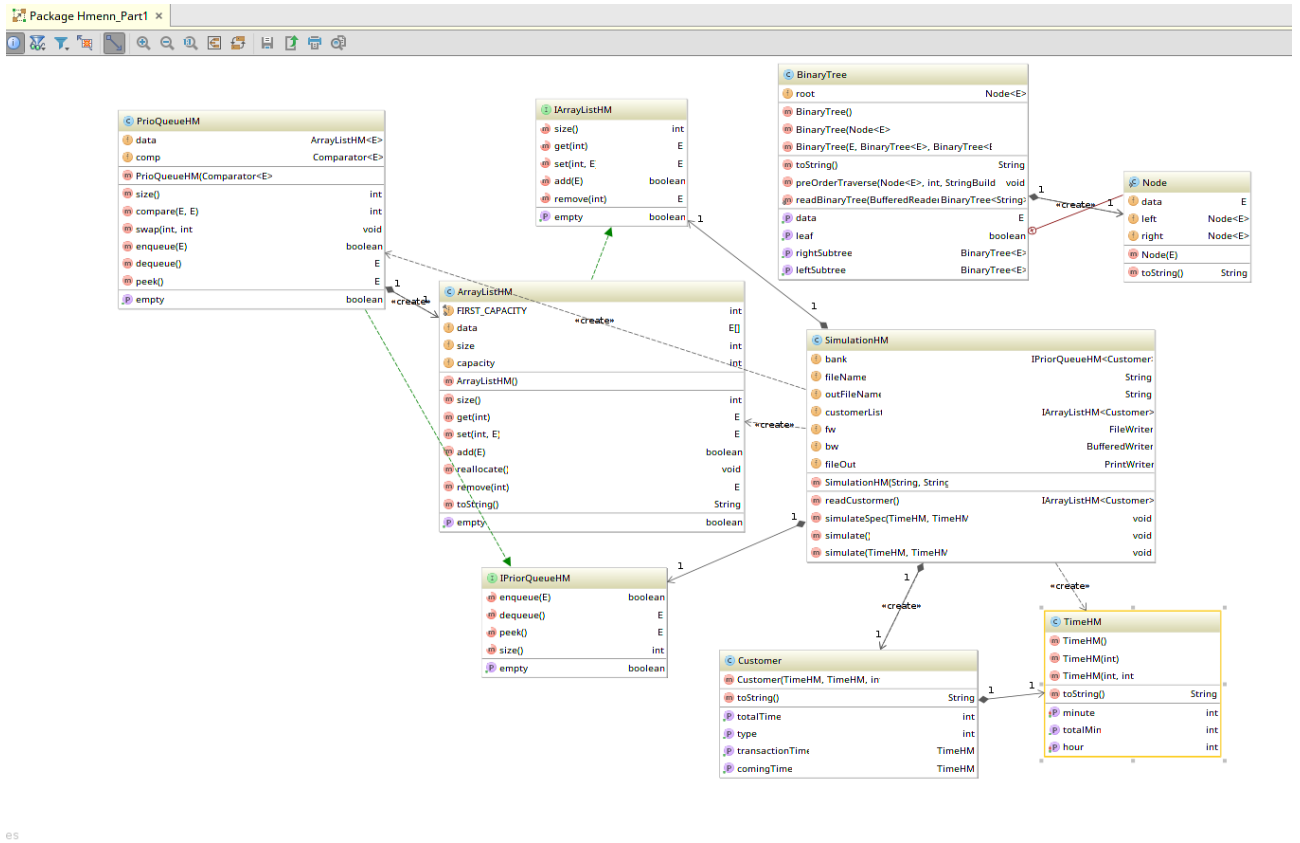
**2 Gün geç gönderim hakkımı bu ödev için kullanıyorum.

Part1:

Neler Yapılacak ?

Bir banka için küçük bir simulasyon hazırlanacak. Tek vezne üzerinden servis yapılmaktadır. Müşteri bilgileri input doyasından okunup sonuçlar ekrana yada output dosyasına ()basılacak. Sistem default olarak müşteriler bitene kadar çalışacaktır ek olarak sistem belirli aralıklarda çalıştırılabilir.

UML Diyagramı :



Nasıl Çalışır?

1. Simulasyon objemize input dosyanın path ini veririz.
2. Dosyadan müşterileri okur ve Customer Listine kaydeder.
3. Simulasyonu baslatırız.
- 4.1. Default olarak ilk elen ve son gelen musterinin saatlerini belirleyerek o aralıklarda sistem dakika arttırarak devam eder.
- 4.2 Ya da verilen süre aralığında artış yaparak simulasyona devam eder.

5. Müşteri gelince queue boş ise direk alınır.
6. İşlemi devam eden varsa queue ye eklenir.
7. Sistem müşterinin bitiş zamanı geldiğinde onu çıkarır ve yenisini alır.
8. Tüm müşteriler bitene kadar yada 4.2 deki gibi verilen süre bitene kadar devam eder.
9. Eger kuyrukta musteri kaldıysa hepsi sırayla isleme alınır ve biter.

Output1 : Default çalışma (data1.txt)

```

CSE222-HW7  src  main  java  Hmenn_Part1  TimeHM
Run  Main
/usr/lib/jvm/java-8-openjdk-amd64/bin/java ...
First Person came : [8:30] [0:12] Customer 1
Last Person came : [17:50] [0:11] Customer 3
Left Come Service Type
[8:30] [8:30] [0:12] Customer 1 come bank
[8:36] [8:36] [0:6] Customer 3 come bank
[8:42] [8:30] [0:12] Customer 1 left bank
[8:44] [8:44] [0:6] Customer 1 come bank
[8:48] [8:36] [0:6] Customer 3 left bank
[8:54] [8:54] [0:4] Customer 2 come bank
[8:54] [8:44] [0:6] Customer 1 left bank
[8:58] [8:54] [0:4] Customer 2 left bank
[9:0] [9:0] [0:11] Customer 2 come bank
[9:10] [9:10] [0:1] Customer 3 come bank
[9:11] [9:0] [0:11] Customer 2 left bank
[9:12] [9:10] [0:1] Customer 3 left bank
[9:20] [9:20] [0:5] Customer 3 come bank
[9:25] [9:20] [0:5] Customer 3 left bank
[9:28] [9:28] [0:8] Customer 3 come bank
[9:36] [9:28] [0:8] Customer 3 left bank
[9:37] [9:37] [0:9] Customer 1 come bank
[9:42] [9:42] [0:2] Customer 3 come bank
[9:46] [9:37] [0:9] Customer 1 left bank
[9:48] [9:42] [0:2] Customer 3 left bank
[9:49] [9:49] [0:7] Customer 2 come bank
[9:56] [9:56] [0:15] Customer 3 come bank
[9:56] [9:49] [0:7] Customer 2 left bank
[10:4] [10:4] [0:6] Customer 2 come bank
[10:11] [9:56] [0:15] Customer 3 left bank
[10:13] [10:13] [0:10] Customer 3 come bank
[10:17] [10:4] [0:6] Customer 2 left bank
[10:18] [10:18] [0:13] Customer 1 come bank
[10:25] [10:25] [0:2] Customer 1 come bank
[10:27] [10:13] [0:10] Customer 3 left bank
[10:29] [10:25] [0:2] Customer 1 left bank
[10:35] [10:35] [0:9] Customer 3 come bank
[10:41] [10:41] [0:5] Customer 2 come bank
[10:42] [10:18] [0:13] Customer 1 left bank
[10:47] [10:47] [0:4] Customer 3 come bank
[10:47] [10:41] [0:5] Customer 2 left bank
[10:51] [10:47] [0:4] Customer 3 left bank
[10:57] [10:57] [0:2] Customer 1 come bank
[11:0] [10:35] [0:9] Customer 3 left bank
[11:2] [10:57] [0:2] Customer 1 left bank
[11:4] [11:4] [0:13] Customer 3 come bank
[11:11] [11:11] [0:13] Customer 1 come bank
[11:17] [11:17] [0:12] Customer 1 come bank
[11:17] [11:4] [0:13] Customer 3 left bank
[11:24] [11:24] [0:13] Customer 2 come bank
[11:29] [11:17] [0:12] Customer 1 left bank
[11:34] [11:34] [0:2] Customer 3 come bank
[11:42] [11:11] [0:13] Customer 1 left bank
[11:43] [11:43] [0:14] Customer 2 come bank
[11:50] [11:50] [0:6] Customer 3 come bank
[11:55] [11:24] [0:13] Customer 2 left bank
[11:59] [11:59] [0:6] Customer 1 come bank
[12:8] [12:8] [0:12] Customer 1 come bank
[12:9] [11:43] [0:14] Customer 2 left bank
[12:17] [12:17] [0:14] Customer 1 come bank

```

Output2 Belirli zaman aralıklarında çalışma (data2.txt)

	StartTime : [0:0]		
	EndTime : [20:0]		
	Left	Come	Service Type
		[0:0]	[0:13] Customer 3 come bank
		[0:5]	[0:11] Customer 3 come bank
		[0:13]	[0:4] Customer 3 come bank
	[0:13]	[0:0]	[0:13] Customer 3 left bank
	[0:17]	[0:13]	[0:4] Customer 3 left bank
		[0:20]	[0:2] Customer 1 come bank
		[0:26]	[0:15] Customer 1 come bank
	[0:28]	[0:5]	[0:11] Customer 3 left bank
		[0:34]	[0:6] Customer 3 come bank
	[0:43]	[0:26]	[0:15] Customer 1 left bank
		[0:44]	[0:10] Customer 3 come bank
	[0:45]	[0:20]	[0:2] Customer 1 left bank
		[0:49]	[0:14] Customer 1 come bank
	[0:55]	[0:44]	[0:10] Customer 3 left bank
		[0:57]	[0:11] Customer 2 come bank
		[1:4]	[0:6] Customer 2 come bank
	[1:9]	[0:49]	[0:14] Customer 1 left bank
		[1:11]	[0:15] Customer 1 come bank
	[1:15]	[1:4]	[0:6] Customer 2 left bank
		[1:21]	[0:13] Customer 2 come bank
		[1:28]	[0:8] Customer 2 come bank
	[1:30]	[1:11]	[0:15] Customer 1 left bank
		[1:34]	[0:13] Customer 1 come bank
	[1:38]	[1:28]	[0:8] Customer 2 left bank
		[1:41]	[0:14] Customer 3 come bank
		[1:47]	[0:11] Customer 2 come bank
	[1:51]	[1:34]	[0:13] Customer 1 left bank
		[1:53]	[0:5] Customer 3 come bank
		[2:1]	[0:5] Customer 3 come bank
	[2:2]	[1:47]	[0:11] Customer 2 left bank
		[2:11]	[0:5] Customer 3 come bank
	[2:15]	[1:21]	[0:13] Customer 2 left bank
		[2:21]	[0:14] Customer 1 come bank
	[2:26]	[0:57]	[0:11] Customer 2 left bank
		[2:30]	[0:15] Customer 3 come bank
		[2:40]	[0:10] Customer 3 come bank
	[2:40]	[2:21]	[0:14] Customer 1 left bank
		[2:48]	[0:1] Customer 3 come bank
	[2:50]	[2:40]	[0:10] Customer 3 left bank
	[2:51]	[2:48]	[0:1] Customer 3 left bank
		[2:55]	[0:14] Customer 3 come bank
		[3:1]	[0:11] Customer 2 come bank
	[3:6]	[2:30]	[0:15] Customer 3 left bank
		[3:9]	[0:5] Customer 2 come bank
	[3:17]	[3:1]	[0:11] Customer 2 left bank
		[3:18]	[0:3] Customer 2 come bank
	[3:22]	[3:9]	[0:5] Customer 2 left bank
		[3:24]	[0:5] Customer 2 come bank
	[3:25]	[3:18]	[0:3] Customer 2 left bank
	[3:30]	[3:24]	[0:5] Customer 2 left bank
		[3:34]	[0:15] Customer 3 come bank
		[3:43]	[0:4] Customer 1 come bank
	[3:44]	[2:55]	[0:14] Customer 3 left bank
	[3:48]	[3:43]	[0:4] Customer 1 left bank
		[3:50]	[0:4] Customer 1 come bank

Part2

İhtiyaçlar :

GTU de asistan hocaların hem öğrenci hem academic kartları vardır. 2Li kart kullanımını önlemek ve verimlilik için bu kartlar combine edilecek. Aynı isimli kişiler olması durumu göz önüne alınacak. Aynı isimli asistan ve academic çalışanlar olabilir.

Nasıl Yapılacak-Nasıl Çalışır ?

1. Input dosyasından kişiler okunacak ve listeye kaydedilecekler.
2. Tüm kişiler tekrardan hashlenecek.
3. HashTablosunda keyler barkod numaraları valueler kişi bilgileri olacak.
4. Barcode numaralar her kişiler için olan barkod numaraları ve isimlerinin hashlenmesi ile oluşacak.
5. Stringler için standart java hashcodu, integer değerler için 17 ile carpılarak hashleme yapıldı. (* farklı hashler kullanıldıkça ortaya çıkan collisionlar aşağıda verilmiştir. 17 yada 31 iyi bir değer)
6. Oluşan yeni kartlar tekrardan aynı yöntemle hashlenerek bir 101 lik HashTablosu üzerine oturtulacaklar.
7. Tablo %75 doluluk oranına gelince rehash edilmiştir.

Hash Katsayısı seçme :

(11 iken son hashleme esnasında 3 collision var)

```
42 public int getAcademicBarcode() { return academicBarcode; }
43
44
45
46
47 @Override
48 public int hashCode() {
49     int result = name != null ? name.hashCode() : 0;
50     result = 11 * result + academicBarcode;
51     return result;
52 }
```

Run Main

```
[21:46] [14:30] [0:7] Customer 3 left bank
[21:51] [1:53] [0:5] Customer 3 left bank
[22:4] [12:42] [0:13] Customer 3 left bank
[22:7] [17:0] [0:3] Customer 3 left bank
[22:12] [19:3] [0:5] Customer 3 left bank
Read : 504 person from file.
When size : 76 Founded 1 collusion. Table rehashing...
When size : 153 Founded 0 collusion. Table rehashing...
When size : 306 Founded 3 collusion. Table rehashing...
Added 500 new person in hash table
```

(1 olarak alinınca 1-3-12 tane collision olmustur her asamada-rehash yapılıyor)

```
17 public boolean equals(Object o) {
18     if (this == o) return true;
19     if (o == null || getClass() != o.getClass()) return false;
20     if (!super.equals(o)) return false;
21     ResearchAssistant that = (ResearchAssistant) o;
22     return studentBarcode == that.studentBarcode;
23 }
24
25 @Override
26 public int hashCode() {
27     int result = super.hashCode();
28     result = 11 * result + studentBarcode;
29     return result;
30 }
31
32 @Override
33 public String toString() {
34     return "ResearchAssistant -> Name : " + getName() + " A.Barcode : " + getA
35 }
```

Run Main

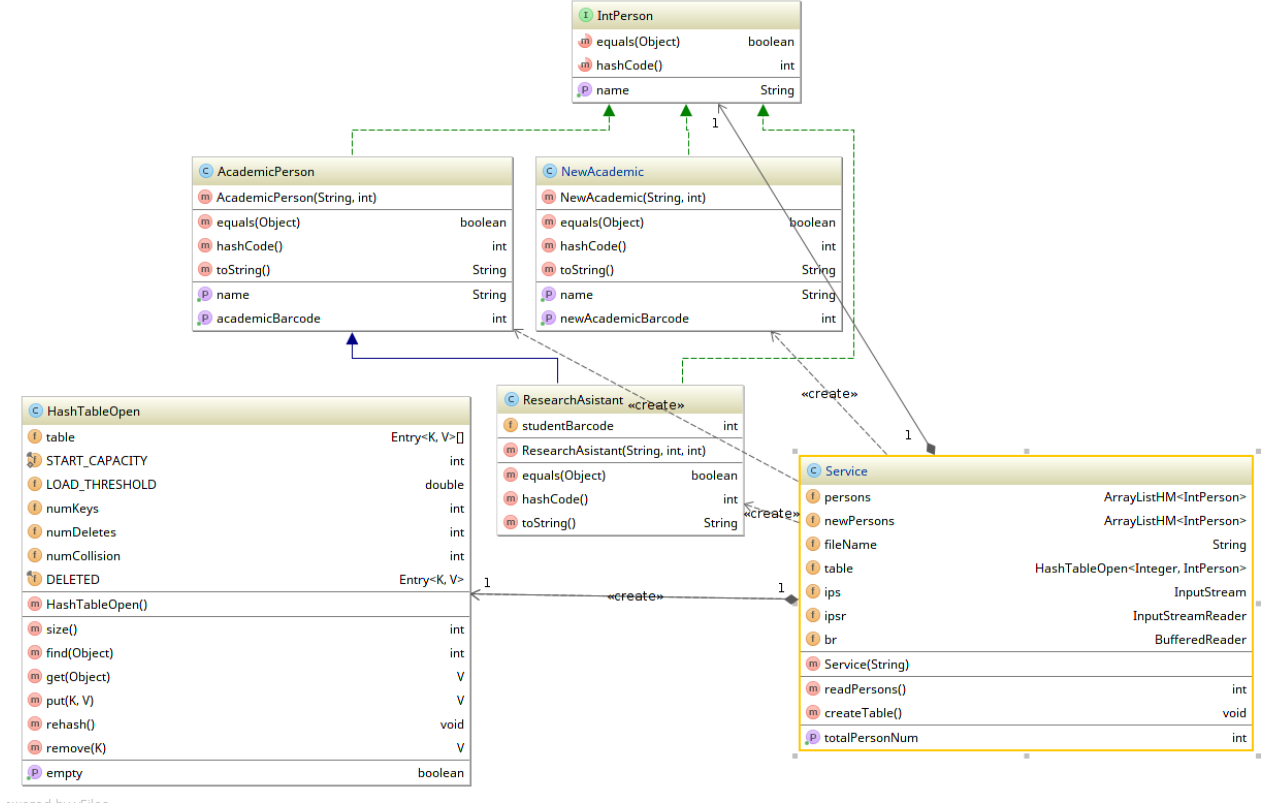
```
[20:0] [0:2] Customer 1 come bank
TAKING PERSONS OVER, HANDLING REMAIN CUSTOMERS
[20:11] [19:51] [0:6] Customer 3 left bank
[20:13] [20:0] [0:2] Customer 1 left bank
[20:28] [13:12] [0:15] Customer 3 left bank
[20:34] [17:40] [0:6] Customer 3 left bank
[20:39] [2:11] [0:5] Customer 3 left bank
[20:41] [15:3] [0:2] Customer 3 left bank
[20:51] [5:16] [0:10] Customer 3 left bank
[21:4] [14:0] [0:13] Customer 3 left bank
[21:10] [13:38] [0:6] Customer 3 left bank
[21:16] [0:34] [0:6] Customer 3 left bank
[21:26] [19:32] [0:10] Customer 3 left bank
[21:39] [15:32] [0:13] Customer 3 left bank
[21:46] [14:30] [0:7] Customer 3 left bank
[21:51] [1:53] [0:5] Customer 3 left bank
[22:4] [12:42] [0:13] Customer 3 left bank
[22:7] [17:0] [0:3] Customer 3 left bank
[22:12] [19:3] [0:5] Customer 3 left bank
Read : 504 person from file.
When size : 76 Founded 1 collusion. Table rehashing...
When size : 153 Founded 3 collusion. Table rehashing...
When size : 306 Founded 12 collusion. Table rehashing...
Added 484 new person in hash table
Process finished with exit code 0
```

(17 olarak katsayi alinınca)

```
[22:12] [19:3] [0:5] Customer 3 left bank
Read : 503 person from file.
When size : 76 Founded 0 collusion. Table rehashing...
When size : 153 Founded 0 collusion. Table rehashing...
When size : 306 Founded 0 collusion. Table rehashing...
Added 503 new person in hash table

Process finished with exit code 0
```

UML Diyagramı:



*Her asistan aynı zamanda bir academic personel olduğu için türetilmiştir.

TESTLER – JUNIT – OUTPUT DOSYALARI

1. Part1 için Customer ve PriorityQueue sınıflarının tesleri yazılmıştır.
2. Part1 için daha detaylı inceleme için ayrı ayrı outpur dosyaları yazıldı.