



**T.C.**  
**SAKARYA ÜNİVERSİTESİ**

**BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ**  
**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**  
**PROGRAMLAMA DİLLERİNİN PRENSİPLERİ ÖDEV RAPORU**

**JAVA VE C PROGRAMLARINDA RASTGELE KİŞİ ÜRETME**  
**VE KONTROL ETME**

**Grup Elemanları:**

**B181210057 - Kadir ÇELİK**

**B181210051 – Duhan UZUN**

**SAKARYA**

**Nisan, 2020**

## Programlama Dillerinin Prensipleri Dersi

### JAVA VE C PROGRAMLARINDA RASTGELE KİŞİ ÜRETME VE KONTROL ETME

Kadir Çelik<sup>a\*</sup>, Duhan Uzun<sup>b</sup>

<sup>a</sup> b181210057 1. Öğretim A Grubu

<sup>b</sup> b181210051 1. Öğretim A Grubu

---

#### Özet

Bu ödevde bizden Java ve C dillerinde rastgele kişi üreten, bu kişilerin TC kimlik numaralarını ve IMEI numaralarını kontrol eden bir program yazmamız istendi. Java kısmında içinde Rastgele, KimlikNo, Telefon, IMEINo, DosyaOkuma, Kisi ve RastgeleKisi sınıflarını bulunduran “RASTGELEKISIURET” kütüphanesini oluşturmamız gerekiyordu. Bu kütüphanede kullanılacak rastgelelik ile ilgili metotların hazır metotlar değil bizim yazdığımız metotlar olmalıydı. Java kısmında ayrıca bu kütüphaneyi kontrol eden bir test projesi yazılması isteniyordu. Ödevin C kısmında Java’da oluşturduğumuz kütüphane ve test projelerinin nesne yönelimli paradigmaya benzetilerek C dilinde yazılmalıydı. Ödevin C kısmında istenen nesne yönelimli paradigmayı “struct” ile sağladık.

© 2020 Sakarya Üniversitesi.

Bu rapor benim özgün çalışmamdır. Faydalanmış olduğum kaynakları içerisinde belirttim. Herhangi bir kopya işleminde sorumluluk bana aittir.

Anahtar Kelimeler: Java, C, rastgelelik, dosya okuma, dosya yazma, kütüphane, C dilini nesne yönelimli programlamaya benzetme

---

## 1. YAZILIM

### 1. KISIM

Projeye ilk olarak “RASTGELEKISIURET” kütüphanesindeki “Rastgele” sınıfını oluşturmakla başladık. Çünkü ödevde kullanacağım rastgelelik metotları bu sınıfı kullanacaktı. Bu sınıfta kendi rastgele sayı üretme metodumun olması gerekiyordu. Rastgeleliği sağlamak için Linear Congruential Generator (LCG) algoritmasını kullandık. Bu algoritmada bir kök ve bir sınır değeri olması gerekiyordu. Kök değerini sistemin çalışma anındaki nanosaniyesi olarak belirledik. Sınır değeri ise kullanıldığı yere göre değişkenlik gösterebiliyordu.

İkinci olarak “KimlikNo” sınıfını oluşturduk. Bu sınıf rastgele bir TC kimlik numarası üretecek ve dosyadaki TC kimlik numaralarının geçerliliğini kontrol edecekti. Bu sınıfı oluştururken

---

\* Kadir Çelik b181210057, Duhan Uzun b181210051

Mail Adresi: kadir.celik6@ogr.sakarya.edu.tr, duhan.uzun@ogr.sakarya.edu.tr

TC kimlik numarası algoritmasını arařtırdık. Kısaca algoritmaya řöyleydi: TC kimlik numarasının 10. hanesi 1, 3, 5, 7 ve 9. hanelerin toplamın 7 ile çarpılmasıyla elde edilen sonuçtan 2, 4, 6 ve 8. hanelerin toplamı çıkarıldığında ortaya çıkan sonucun 10’a bölümünden kalan sayıya eşittir. TC kimlik numarasının son hanesi ise ilk 10 hanenin toplamının 10’a bölümünden kalan sayıya eşittir. TC kimlik numarasının uzunluęu 11’den farklı olamaz ve ilk hane 0’a eşit olamaz.

Üçüncü olarak “IMEINo” sınıfını oluřturduk. Bu sınıf rastgele bir IMEI numarası üretecek ve dosyadaki IMEI numaralarının geçerlilięini kontrol edecekti. Bu sınıfı oluřtururken IMEI numarası algoritmasını arařtırdık. Kısaca algoritmaya řöyleydi: IMEI numarasını hesaplayan algoritma Luhn algoritmasıdır. Luhn algoritmasına göre IMEI numarasının son hanesi ilk 14 hane kullanılarak hesaplanır. 1, 3, 5, 7, 9, 11 ve 13. haneler toplanır. 2, 4, 6, 8, 10, 12 ve 14. haneler ikiyle çarpılır ve basamaklarına ayrılır. Elde edilen basamaklar ilk toplama eklenir. Elde edilen toplam 10’a bölünür ve son hane elde edilir. IMEI numarası 15 hane olmalıdır.

Dördüncü olarak “Telefon” sınıfını oluřturduk. Bu sınıfta Türkiye’deki telefon numaralarına göre rastgele telefon numaraları üretiliyordu. Bu sınıf içerisinde “IMEINo” sınıfını (Nesne referansı ile) barındırıyor ve üretiliyordu. Üretilen IMEI numarası burada tutuluyordu.

Beřinci olarak “DosyaOkuma” sınıfını oluřturduk. Bu sınıfta dosyadaki veriler okunuyordu ve dosyanın satır sayısı bulunuyordu.

Altıncı olarak “Kisi” sınıfını oluřturduk. Bu sınıfta üretilecek kişilerdeki bilgiler tutuluyordu. Bu bilgiler TC kimlik numarası (Nesne referansı ile), isim, soyisim, yař, telefon numarası (Nesne referansı ile) ve telefonun içinde bulunan IMEI numarasıydı.

Yedinci olarak “RastgeleKisi” sınıfını oluřturduk. Bu sınıf rastgele isim, rastgele soyisim, TC kimlik numarası (Nesne referansı ile), telefon numarası (Nesne referansı ile) ve rastgele yař üretiliyordu. Ayrıca bu sınıf “Kisi” sınıfından kalıtım alıyordu. Üretilen bu deęerleri “Kisi” sınıfına aktarıyordu.

Sekizinci olarak kütüphaneyi test eden kısım olan “RastgeleKisiDeneme” projesini oluřturduk. Bu projede “RASTGELEKISIURET” kütüphanesi kullanarak rastgele kiři ürettik ve bunları bir dosyaya yazdık. Ayrıca üretilen kişilerin TC kimlik numaralarını ve IMEI numaralarını kontrol ettirdik. Bunları yaparken kullanıcının seçim yapabilmesi için bir switch-case bloęu oluřturduk. Projeden “Çıkıř” seçeneęi seçilmedięi sürece çıkılmamaktadır.

## 2. KISIM

Bu kısımda C diline, Java dilinde yaptıklarımızı benzetmeye çalıştık. Bunu yaparken C’de kütüphane oluřturamadığımız için başlık dosyalarından ve C’de sınıf yapısı olmadığı için structlardan yararlandık. Bu kısımda kullandığımız rastgele sayı üreten fonksiyon Xorshift64 algoritmasını kullanmaktaydı. Ayrıca dosyadan veri alırken iki boyutlu dizilerden ve string veri almak için tek boyutlu char dizilerinden yararlandık. Bu kısımda da ilk kısımda olduęu gibi test kısmı vardı. Kullanıcı kiři üretebiliyor, kiřinin TC kimlik numarasını ve IMEI numarasını kontrol edebiliyordu. İstedięi zaman da programdan çıkıř yapabiliyordu.

## 2. ÇIKTILAR

### 1. KISIM

```
run:
1- Rastgele Kisi Uret
2- Uretilmis Dosya Kontrol Et
3- Cikis
1
Kac adet kisi ureteceksiniz?
1000
kisiler olusturuldu
1- Rastgele Kisi Uret
2- Uretilmis Dosya Kontrol Et
3- Cikis
2
T.C. Kimlik Kontrol
Gecerli : 1000
Gecersiz: 0

IMEI Kontrol
Gecerli : 1000
Gecersiz: 0

1- Rastgele Kisi Uret
2- Uretilmis Dosya Kontrol Et
3- Cikis
3
```

1. kısma ait örnek ekran çıktısı

## 2. KISIM

```
Seç D:\MinGW\bin\mingw32-make.exe
gcc -I ./include/ -o ./lib/Telefon.o -c ./src/Telefon.c
gcc -I ./include/ -o ./lib/DosyaOkuma.o -c ./src/DosyaOkuma.c
gcc -I ./include/ -o ./lib/Kisi.o -c ./src/Kisi.c
gcc -I ./include/ -o ./lib/RastgeleKisi.o -c ./src/RastgeleKisi.c
gcc -I ./include/ -o ./bin/RastgeleKisiDeneme ./lib/Rastgele.o ./lib/IMEINo.o ./lib/KimlikNo.o ./lib/Telefon.o ./lib/DosyaOkuma.o ./lib/Kisi.o ./lib/RastgeleKisi.o ./src/RastgeleKisiDeneme.c
./bin/RastgeleKisiDeneme
1- Rastgele Kisi Uret
2- Uretilmis Dosya Kontrol Et
3- Cikis
1
Kac adet kisi ureteceksiniz?
1000
Kisiler olusturuldu.
1- Rastgele Kisi Uret
2- Uretilmis Dosya Kontrol Et
3- Cikis
2
T.C. Kimlik Kontrol
1000 Gecerli
0 Gecersiz

IMEI Kontrol
1000 Gecerli
0 Gecersiz

1- Rastgele Kisi Uret
2- Uretilmis Dosya Kontrol Et
3- Cikis
```

### 2. kısma ait örnek ekran çıktısı

```
Kisiler.txt - Not Defteri
Dosya Düzen Biçim Görünüm Yardım
34694956398 Tyson Potts 62 05617680599 (223843879201097)
65549441288 Nathaniel Griffin 60 05593579936 (093280489386165)
16322194334 Miriam Greene 71 05512327456 (168672453565107)
17244233972 Amirah Mullins 54 05618011932 (449041300371773)
57682144300 Howard Fry 25 05369495638 (574420310635947)
38682311978 Amari Harrison 56 05615494412 (657680599934515)
60767462176 Nikolai Howard 75 05612219432 (579357993690762)
70191948504 Jessica Santos 35 05017244233 (512327456718574)
42129087654 Jenna McBride 92 05521443526 (267801193235289)
51605320574 Salma Chandler 53 05368231192 (346949563885152)
23930493520 Dalia Flynn 26 05617674621 (655494412713672)
32882578520 Fletcher Mayo 31 05019194856 (163221943263512)
70253822650 Julissa Stokes 80 05012908769 (017244233980213)
40264181770 King Campbell 25 05516053205 (576821443526143)
34578279612 Brennan Hickman 13 05330493566 (386823119238292)
45302928980 Jacqueline Bird 16 05388257857 (607674621615805)
71083512186 Amber Carney 27 05053822694 (701919485630647)
43608224560 Jayce Brady 10 05426418178 (042129087690290)
19925990228 Conrad Hancock 45 05357827965 (516053205727353)
34761480092 Megan Burch 72 05430292893 (239304935661214)
62130628660 Whitney Montgomery 43 05051211902 (328825785770012)
37403951912 Skyler Foster 67 05060822453 (702538226947679)
63150850154 Meghan Chaney 24 05599029410 (402641817809804)
37149402280 Dorian Ho 73 05376148005 (345782796574843)
41853673300 Denise Ingram 94 05611306286 (453029289392491)
6803352516 Jasmin Miles 47 05340395190 (710835121190201)
72830996600 Heather Boyd 20 05611508501 (043608224536432)
41700454724 Ashanti Chandler 21 05314940220 (199259902941031)
13409353974 Nicolas Bishop 53 05485367333 (347614800598831)
29434351854 Yareli Walton 96 05610335325 (621306286292065)
13916658616 Marie Brady 23 05399667182 (374039519067520)
```

### 2. kısımda oluşturulmuş dosya örneği

### 3. SONUÇ

Verilen bu ödev sayesinde rastgeleliğin ne kadar zor sağlandığını öğrenmiş olduk. C dilinde rastgeleliği sağlamak çok zordu. Bu ödev sayesinde Java dilinde kütüphane oluşturmayı ve bu kütüphaneyi nasıl kullanacağımızı öğrendik. C dilinde nesne yönelimli paradigmaya nasıll benzetim yapabileceğimizi öğrendik. C dilinde string olmamasının dosyadan veri alırken bizi zorlayabileceğini öğrendik. C dilinde dosya okuma, yazma ve struct yapısını daha iyi anladığımızı düşünüyoruz.

### EK BİLGİLER

- 1-Java kısmında dosyaya kişilerin verilerini yazarken sahip oldukları TC kimlik, telefon ve IMEI numaralarının daha önceden üretilip üretilmediğini kontrol ettirdik. Bu kısım “RastgeleKisiDeneme” projesinde açıklama satırı olarak bulunmaktadır.
- 2- Java kısmında dosya üretirken ve kontrol yaparken proje çok ağır çalışmaktadır.
- 3- C kısmında nadiren de olsa eksik kontrol ve hatalı dosya üretimi yapabilmektedir.

### REFERANSLAR

- ✓ [https://en.wikipedia.org/wiki/Linear\\_congruential\\_generator](https://en.wikipedia.org/wiki/Linear_congruential_generator)
- ✓ <https://en.wikipedia.org/wiki/Xorshift>
- ✓ <https://teknoseyir.com/blog/t-c-kimlik-numaralarinin-algoritmasi>
- ✓ [https://en.wikipedia.org/wiki/International\\_Mobile\\_Equipment\\_Identity](https://en.wikipedia.org/wiki/International_Mobile_Equipment_Identity)
- ✓ [https://en.wikipedia.org/wiki/Luhn\\_algorithm](https://en.wikipedia.org/wiki/Luhn_algorithm)
- ✓ [https://tr.wikipedia.org/wiki/T%C3%BCrkiye%27deki\\_telefon\\_numaralar%C4%B1](https://tr.wikipedia.org/wiki/T%C3%BCrkiye%27deki_telefon_numaralar%C4%B1)