



Döngüler



Dr. Kübra Atalay Kabasakal
Bahar 2023

Döngüler

- Döngüler diğer bütün programa dillerinde sıklıkla kullanılan **akış kontrolü (flow control)** mekanizmasının bir parçasıdır.
- Her ne kadar R vektörel elementler üzerine kurulmuş olsa da bazı durumlarda döngülerin kullanılması gerekebilir.
- Örneğin, simulasyon çalışmaları genellikle iterasyonel ve tekrar eden süreçleri içermektedir.
- Döngüler sonuç elde etmek yerine süreçteki işlemleri dikkate aldığından, simulasyon çalışmalarında kullanılır.
 - **for()**
 - **while()**
 - **repeat()**

for() Döngüsü

- **for()** döngüsü ile belirlenen sayıda işlem tekrarı yapılırken **while()** ya da **repeat()** döngülerinde bir sayaç ya da bir dizin ile kontrol sağlanarak işlemlerin tekrarlı yapılmasını sağlar.
- **for()** bir vektör, liste ya da matris içindeki her bir elemanın bir değişken yardımıyla belirlenen komutu veya kodu sırasıyla yapması için oluşturulan bir döngüdür.
- **for()** döngüsünün genel kullanımı aşağıdaki gibidir.

```
for(i in 1:10) {  
  print(i)  
}
```

```
## [1] 1  
## [1] 2  
## [1] 3  
## [1] 4  
## [1] 5  
## [1] 6
```

for() Döngüsü

- Döngüde indeks değişkeni herhangi bir nesne ile tanımlanabilir. Örneğin **i**
- Ayrıca indeks değerinin başlangıcı 1 olmak zorunda **değildir**.

```
for(i in 5:8) {  
  print(i)  
}
```

```
## [1] 5  
## [1] 6  
## [1] 7  
## [1] 8
```

for() Döngüsü

- karakter yazımında indeks **i** sadece tekrar amaçlı kullanılır.

```
for(i in 5:10){  
  print("Merhaba")  
}
```

```
## [1] "Merhaba"  
## [1] "Merhaba"  
## [1] "Merhaba"  
## [1] "Merhaba"  
## [1] "Merhaba"  
## [1] "Merhaba"
```

for() Döngüsü

- Aşağıdaki çıktıyı sağlayacak kodu yazınız.

```
## 1 + 1 = 2
## 2 + 2 = 4
## 3 + 3 = 6
## 4 + 4 = 8
## 5 + 5 = 10
## 6 + 6 = 12
## 7 + 7 = 14
## 8 + 8 = 16
## 9 + 9 = 18
## 10 + 10 = 20
```

for() Döngüsü

- Döngülerde bir değişken yeniden tanımlanacak ise mutlaka döngü öncesi o değişken tanımlanmalıdır.
- Oluşturulan bir matrisin satırlarında yer alan sayıların toplamını başka bir nesneye atama

```
(X <- cbind(a = 1:5, b=2:6))
```

```
##      a b
## [1,] 1 2
## [2,] 2 3
## [3,] 3 4
## [4,] 4 5
## [5,] 5 6
```

```
Y <- array()
for(i in 1:nrow(X)) {
  Y[i] <- X[i,1] + X[i,2]
}
Y
```

```
## [1]  3  5  7  9 11
```

for() Döngüsü

- `cat()`, `paste()` gibi fonksiyonları uzun bir döngüde, döngünün durumunu görmek için de kullanabilirsiniz .

```
islem.kontrol <- array()
for(i in 1:10){
  islem.kontrol[i] <- paste("Islem ", i, " tamamlandi", sep="")
}
islem.kontrol
```

```
## [1] "Islem 1 tamamlandi" "Islem 2 tamamlandi"
## [3] "Islem 3 tamamlandi" "Islem 4 tamamlandi"
## [5] "Islem 5 tamamlandi" "Islem 6 tamamlandi"
## [7] "Islem 7 tamamlandi" "Islem 8 tamamlandi"
## [9] "Islem 9 tamamlandi" "Islem 10 tamamlandi"
```


for() Döngüsü

- Döngülerde her zaman `i` indeksini kullanmak zorunda **değiliz**.

```
set.seed(10)
x <- sample(1:10000,100)

sayac <- 0
for (val in x) {
  if(val %% 2 == 0){
    sayac = sayac+1
  }
}
print(sayac)
```

```
## [1] 46
```

for() Döngüsü

- Her zaman işlemi tüm elemanlara uygulamak istemeyebiliriz.
- Bunun önlemek için akış kontrolü yapmak gerekir.
- Kontrol mantıksal operatörlerle ya da koşul cümleleri ile sağlanabilir.

for() Döngüsü ve Kontrol

```
for(i in 1:3){  
  if (i==2) cat("indeks cift sayidir:","\n")  
  else cat(i,"\n")  
}
```

```
## 1  
## indeks cift sayidir:  
## 3
```

for() Döngüsü ve Kontrol

```
for(i in 1:3){  
  if (i==2) {  
    cat("indeks degeri ikidir:",i,"\n")  
  }else{cat("indeks degeri iki degildir","\n")}  
}
```

```
## indeks degeri iki degildir  
## indeks degeri ikidir: 2  
## indeks degeri iki degildir
```

for() Döngüsü

- Bazen döngüler iç içe kullanılabilir 5X5 bir matrisin her bir elemanı satır ve sütun indeksleri çarpımı olsun örneğin $m[2,5]=10$ olsun.
- Bu işlemi yapmak için öncelikle boş bir matris oluşturmak lazım.

```
m2 <- matrix(0,nrow=5,ncol=5)
m2
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    0    0    0    0    0
## [2,]    0    0    0    0    0
## [3,]    0    0    0    0    0
## [4,]    0    0    0    0    0
## [5,]    0    0    0    0    0
```

- Aşağıdaki çıktıyı elde edecek kodu oluşturmaya çalışın

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    2    3    4    5
## [2,]    2    4    6    8   10
## [3,]    3    6    9   12   15
## [4,]    4    8   12   16   20
## [5,]    5   10   15   20   25
```

for() Döngüsü

- Kullanıcı tarafından belirlenen **nxn** boyutunda bir matris oluşturulsun.
- **nxn** bir matrisin her bir elemanı satır ve sütun indeksleri çarpımı olsun.
- örneğin $m[2,5]=10$ olsun.
- Eğer matrisin boyutları 10×10 'dan büyükse sadece 10 satırını yazsın eğer matrisi boyutları 10×10 'dan küçükse hepsini yazsın.
- Kullanıcı üç girdiğinde oluşacak çıktı:

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    2    4    6
## [3,]    3    6    9
```

for() Döngüsü

- Döngünün indeksi her zaman bir tam sayı olmak zorunda değildir.
- Liste, veri seti, matris de olabilir.
- **if** sadece numerik değer ve vektörlerle çalışmaz.
- Aynı zamanda veri seti, matris ve listelerle de çalışabilir.

```
d <- data.frame(a = 1:5, b=2:6)  
d
```

```
##   a b  
## 1 1 2  
## 2 2 3  
## 3 3 4  
## 4 4 5  
## 5 5 6
```

```
for(x in d) {  
  cat("sutun toplamlari:", sum(x), "\n")  
}
```

```
## sutun toplamlari: 15  
## sutun toplamlari: 20
```

for() Döngüsü

```
X <- cbind(1:10, 20:29)
X
```

```
##      [,1] [,2]
## [1,]    1  20
## [2,]    2  21
## [3,]    3  22
## [4,]    4  23
## [5,]    5  24
## [6,]    6  25
## [7,]    7  26
## [8,]    8  27
## [9,]    9  28
## [10,]   10  29
```

Aşağıdaki çıktıyı elde etmek için gerekli kodu yazınız.

```
## 1 satirdaki degerlerin carpimi 20 olarak hesaplanmistir.
## 2 satirdaki degerlerin carpimi 42 olarak hesaplanmistir.
## 3 satirdaki degerlerin carpimi 66 olarak hesaplanmistir.
## 4 satirdaki degerlerin carpimi 92 olarak hesaplanmistir.
## 5 satirdaki degerlerin carpimi 120 olarak hesaplanmistir.
## 6 satirdaki degerlerin carpimi 150 olarak hesaplanmistir.
## 7 satirdaki degerlerin carpimi 182 olarak hesaplanmistir.
## 8 satirdaki degerlerin carpimi 216 olarak hesaplanmistir.
## 9 satirdaki degerlerin carpimi 252 olarak hesaplanmistir.
```


next() ve break()

- `next()` ve `break()` fonksiyonları döngülerde kontrol mekanizmasıdır.
- Döngüyü sadece belirli bir koşulda çalıştırmak istemezseniz `next()` fonksiyonunu kullanabilirsiniz.

```
for(i in 1:6){  
  if(i==3){  
    next  
  }  
  print (i)}
```

```
## [1] 1  
## [1] 2  
## [1] 4  
## [1] 5  
## [1] 6
```

next() ve break()

- Döngüyü sadece belirli bir koşulda durdurmak isterseniz `break()` fonksiyonunu kullanabilirsiniz.

```
for(i in 1:12){  
  if(i==3){  
    break  
  }  
  print (i)}
```

```
## [1] 1
```

```
## [1] 2
```

for() Döngüsü

- Döngüler uzun zamanda çalışır.

```
# donguler uzun zamanda calisir
set.seed(853)
y<-matrix(rnorm(1000000),nrow=1000)
z<-0*y
time1<-as.numeric(Sys.time())
#loop:
time2 <- system.time(
  for(i in 1:1000){
    for(j in 1:1000){
      z[i,j]<-y[i,j]^2
    }
  })

time2
```

```
# ayni islemi dongusuz yapma
time3 <- system.time(z<-y^2)
time3
```

```
##      user  system elapsed
##          0         0         0
```

```
##      user  system elapsed
##      1.48    0.00     1.50
```

Coklu veri seti olusturma ve dışarı aktarma

- İstenilen sayıda veri seti oluşturan bir fonksiyon yazalım.
- Fonksiyonun ilk girdisi veri seti sayısı olmalı
- Kullanıcı oluşturmak istediği her bir veri seti için satır ve sütun sayısını belirleyebilirsiniz.
- Örneğin oluşturduğu ilk veri setin 5 satır, 10 sütunlu ikincisi veri seti ise 10 satır, 20 sütunlu olsun.
- Oluşturacak olan her bir veri setinin her bir sütunu standart normal dağılıma uygun olacak şekilde üretilsin.
- Oluşturulan veri setlerinden ilki "veri_1.xlsx" şeklinde, devamında ise sayı değiştirilerek dışarı aktarılсын.

for() Ödev-1

Fibonacci dizisinin elemanlari **1 1 2 3 5 8 13 21 34 55 89** ... dizinin elemanlarını **for()** döngüsü ile oluşturmaya çalışınız.

for() Ödev-2

```
set.seed(1786)
ornek<-exp(matrix(rnorm(2000),nrow=100))
index1.temp<-sample(1:100,10)
index2.temp<-sample(1:20,10)
for(i in 1:10){
  ornek[index1.temp[i],index2.temp[i]]<--1
}
head(ornek,6)
```

```
##           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
## [1,] 0.5549525 0.3247338 0.5236032 0.3821027 0.4187483 0.1588847
## [2,] 0.5671734 1.2431592 0.8812069 2.6695443 0.6984453 1.0838792
## [3,] 4.8068457 0.3449856 0.6079096 0.9194116 1.5361330 1.9082522
## [4,] 1.3509234 2.3569582 0.1931423 4.0707377 0.3527276 2.3498825
## [5,] 0.9012032 0.2310683 0.2317487 1.3809955 0.9168741 0.6237213
## [6,] 1.2331483 1.1066056 0.3546027 0.3705946 0.9002303 0.2528151
##           [,7]      [,8]      [,9]      [,10]     [,11]     [,12]
## [1,] 5.226161 2.6280057 1.2251526 0.4760966 5.2379018 1.4782655
## [2,] 1.079946 2.2087385 0.5195551 0.3757409 0.9004808 0.7409205
## [3,] 0.671977 1.5310016 0.6735007 2.2069776 0.5060078 0.7171477
## [4,] 1.198514 2.5592899 1.8205257 1.2624052 0.1524106 0.3828322
## [5,] 1.609403 1.1005990 1.0619758 2.1047783 2.7816902 1.4010878
## [6,] 3.337512 0.9799103 2.7520425 2.5407624 1.3889136 0.4346808
##           [,13]     [,14]     [,15]     [,16]     [,17]
```

for() Ödev-2

```
## [1] "Satir 1 ortalamasi 0.986111423178787"  
## [1] "Satir 2 ortalamasi 1.66440473890558"  
## [1] "Satir 3 ortalamasi 1.86445460243509"  
## [1] "Satir 4 negatif sayi icermektedir."  
## [1] "Satir 5 negatif sayi icermektedir."  
## [1] "Satir 6 ortalamasi 2.18755744815693"  
## [1] "Satir 7 ortalamasi 2.42896783600747"  
## [1] "Satir 8 ortalamasi 1.11152186047931"  
## [1] "Satir 9 ortalamasi 1.28348082027049"  
## [1] "Satir 10 ortalamasi 1.49790135754768"  
## [1] "Satir 11 ortalamasi 1.00823845594998"  
## [1] "Satir 12 ortalamasi 1.84432161490249"  
## [1] "Satir 13 ortalamasi 2.30730516248531"  
## [1] "Satir 14 ortalamasi 1.32997520232501"  
## [1] "Satir 15 ortalamasi 1.40736423997693"  
## [1] "Satir 16 ortalamasi 0.930694377568197"  
## [1] "Satir 17 ortalamasi 1.09683802891735"  
## [1] "Satir 18 ortalamasi 1.34543057465283"  
## [1] "Satir 19 ortalamasi 1.91931890408157"  
## [1] "Satir 20 ortalamasi 1.46149447129439"  
## [1] "Satir 21 ortalamasi 1.48698773010654"  
## [1] "Satir 22 ortalamasi 2.50083591324982"  
## [1] "Satir 23 ortalamasi 2.49403230671112"  
## [1] "Satir 24 ortalamasi 2.03307899444367"  
## [1] "Satir 25 ortalamasi 1.47358418101605"
```