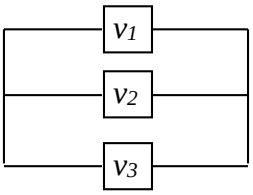
Örnek-4.1. Ördeğin Su İçmesi Olasılığı

Aşağıdaki su devresinde bulunan vanalar birbirinden bağımsız olarak çalışmakta ve çalışma olasılıklarının *p* olduğu bilinmektedir. Ördeğin su içme olasılığı nedir? Ördeğin su içmesi olasılığının 0.99'dan büyük olması için vanaların çalışması olasılığı ne olmalıdır? Teorik olarak bulunan ördeğin su içmesi olasılığını hesaplamak için başka türlü yaklaşımda bulunabilir miyiz?



Çözüm. V_1, V_2, V_3 ile vanaların çalışması olayı ve *O* ile ördeğin su içmesi olayı gösterilirse, $O = V_1 \cup V_2 \cup V_3$ olmak üzere

$$P(O) = P(V_1 \cup V_2 \cup V_3) = 3p - 3p^2 + p^3$$

olarak bulunur. Örneğin, *p*=1/2 için, *P(O)*=7/8=0.875 olarak hesaplanır. Şimdi, ördeğin su içmesi olasılığının 0.99'dan büyük olması için vanaların çalışması olasılığını bulalım.

$$P(O) = P(V_1 \cup V_2 \cup V_3) = 3p - 3p^2 + p^3 \geq 0.99$$

olacak şekilde *p* değerinin bulunması gerekmektedir. Bu amaçla bu denklem sisteminin çözülmesi gerekir.

$$f(x) = 3x - 3x^2 + x^3 - 0.99$$

fonksiyonunun kökünün bulunması istenilen değeri verecektir. Bu fonksiyonun kökünü de daha önceden bildiğimiz

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} = x_n - \frac{3x_n - 3x_n^2 + x_n^3 - 0.99}{3x_n^2 - 6x_n + 3}$$

yineleme bağıntısı ile (Newton-Raphson yöntemi) kolaylıkla çözebiliriz. Bu amaçla aşağıdaki program çalıştırılırsa, *p* değeri yaklaşık olarak 0.7845 bulunur.

```
INPUT "X0 BAŞLANGIÇ DEĞERİNİ GİRİNİZ=", x0
INPUT " ADIM SAYISINI GİRİNİZ=", N
FOR I = 1 TO N
   pay = x0 ^ 3 - 3 * x0 ^ 2 + 3 * x0 - .99
   payda = 3 * x0 ^ 2 - 6 * x0 + 3
   x1 = x0 - pay / payda
   PRINT I, x1
   IF ABS(x1 - x0) <= .0000001 THEN EXIT FOR
   x0 = x1
NEXT I
PRINT "YAKLAŞIK KÖK=", x1
```

Bir olay, süreç veya sistemle ilgili bir özelliğin veya davranışın model üzerinde gözlenmesine simülasyon (simulation) denir. "Simulation" - taklit, benzetim anlamına gelen bir sözcüktür. Matematiksel modellerde, analitik veya nümerik (sayısal) bir çözüm bulunamadığında simülasyona başvurulduğunu hatırlatalım ve optimal bir sonuç yerine, değişik koşullar altında yapılan denemelerle bir takım "gözlem" sonuçlarının elde edildiğini belirtelim. Simülasyon sonucunda gerçek olay, süreç veya sistemle ilgili "model üzerinde yapılan deneyler" ile bazı gözlem değerlerinin elde edildiğini unutmayalım.

Ördeğin su içmesi problemine yeniden dönelim ve ördeğin su içmesi olasılığına simülasyon yaparak yaklaşımda bulunup bulunamayacağımızı düşünelim. V_1, V_2, V_3 vanalarının çalışmasını 1 ile çalışmamasını da 0 ile gösterelim. Aşağıdaki

1 0 0, 1 1 0, 1 0 0, 0 1 1, 0 0 1, 0 0 1, 0 1 0, 0 1 0, 1 1 0, 0 1 1, 0 0 0, 0 0 1, 0 0 1, 0 1 0, 0 1 0

gibi durumlar olabilecektir, bunlardan en az birisi 1 ise ördek su içecektir. Acaba bu 0 ve 1 sayılarını "rasgele" nasıl üretebiliriz. İkinci bölümde kısaca bahsedildiği gibi Basic programında bulunan RND komutu ile

0.7055 0.5334 0.5795 0.2896 0.3019 0.7747 0.0140 0.7607 0.8145 0.7090 0.0454 0.4140 0.8626
0.7905 0.3735 0.9620 0.8714 0.0562 0.9496 0.3640 0.5249 0.7671 0.0535 0.5925 0.4687 0.2982
0.6227 0.6478 0.2638 0.2793 0.8298 0.8246 0.5892 0.9861 0.9110 0.2269 0.6951 0.9800 0.2439
0.5339 0.1064 0.9994 0.6762 0.0157 0.5752 0.1001 0.1030 0.7989 0.2845 0.0456 0.2958 0.3820
0.3010 0.9486 0.9798 0.4014 0.2783 0.1604 0.1628 0.6466 0.4101 0.4128 0.7127 0.3262 0.6332
0.2076 0.1860 0.5834 0.0807 0.4580 0.9057 0.2614 0.7852 0.3789 0.2897 0.9194 0.6317 0.6276
0.4285 0.0980 0.5610 0.6945 0.9137 0.8348 0.0226 0.5434 0.9162 0.4303 0.6779 0.5025 0.5137
0.4630 0.3535 0.4048 0.2697 0.0556 0.2438 0.9791 0.0609 0.3903

gibi (0, 1) aralığından rasgele sayılar üretmektedir. Şimdilik bu rasgele sayıların nasıl üretildiğine değinmeyeceğiz. Eğer bu rasgele sayılar (0, 1) aralığında hemen-hemen düzgün (aynı sayıda) dağılıyorlar ise bu sayıları kullanarak bu 0 ve 1 sayılarını oluşturmamız olanaklı olacaktır. Bu özelliklerin sağlanıp sağlanmadığını görmek için aşağıdaki program yardımıyla çıkan sonuçlara bakalım. *N*=10000 tane (0, 1) arasında RND ile sayı üretelim ve bu aralığı 10 eşit parçaya böldüğümüzde her bir aralığa kaç tane düştüğünü gözleyelim.

```
INPUT "ÜRETME SAYISI =", N
DIM ARA(10)
FOR I = 1 TO N
   a = INT(10 * RND) + 1
   ARA(a) = ARA(a) + 1
NEXT I
FOR J = 1 TO 10
   PRINT J; ".ARALIK"; "(."; J - 1; ","; J / 10; ")"; ARA(J); " TANE"
NEXT J
```

Program çalıştırıldığında çıktı aşağıdaki gibi olacaktır.

1 .ARALIK(. 0 , .1) 10034 TANE
2 .ARALIK(. 1 , .2) 9817 TANE
3 .ARALIK(. 2 , .3) 10051 TANE
4 .ARALIK(. 3 , .4) 9973 TANE
5 .ARALIK(. 4 , .5) 10045 TANE
6 .ARALIK(. 5 , .6) 10046 TANE
7 .ARALIK(. 6 , .7) 10080 TANE
8 .ARALIK(. 7 , .8) 9939 TANE
9 .ARALIK(. 8 , .9) 9988 TANE
10 .ARALIK(. 9 , 1) 10027 TANE

Sonuçlar incelendiğinde her bir aralığa hemen-hemen aynı sayıda düştüğünü görmekteyiz. 0 ve 1 sayılarını vanaların çalışma olasılıklarını kullanarak RND ile üretilen sayılar yardımıyla elde etmemiz artık olanaklı hale gelmektedir. Eğer RND ile üretilen sayı *p* değerinden küçükse 1 değilse 0 değerini alacak şekilde bir kural geliştirirsek 1 ve 0 dizisini rasgele oluşturmuş oluruz. Yukarıdaki RND ile üretilen sayılara bu kuralı uygularsak

0 0 1 1 0 1 0 0 0 1 1 0 0 1 0 0 1 0 1 0 0 1 0 1 1 0 0 1 1 0 0 0 0 0 1 0 0 1 0 1
0 0 1 0 1 1 0 1 1 1 1 1 0 0 1 1 1 1 0 1 1 0 1 0 1 1 0 1 1 0 1 0 1 1 0 0 0 1 1 0
0 0 0 1 0 0 1 0 0 0 1 1 1 1 1 1 1 0 1 1

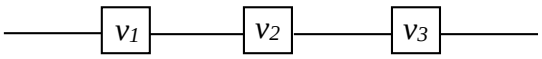
elde ederiz. Bu sayıları 3'erli düşünüp topladığımızda eğer 0'dan büyükse ördek su içer değilse içmez diyebiliriz. *N* denemede ördeğin kaç kez su içtiğini bulup deneme sayısına böldüğümüzde ördeğin su içme olasılığı ile ilgili bir yaklaşımda bulunmuş oluruz.

```
'ORDEGIN SU ICMESI OLASILIGI
INPUT "DENEME SAYISI=", N
INPUT "VANALARIN ÇALIŞMA OLASILIĞI="; P
INPUT "PARALEL SATIR SAYISI="; SATIR
DIM VANA(SATIR)
FOR I = 1 TO N
    T = 0
    FOR J = 1 TO SATIR
        IF RND < P THEN VANA(J) = 1 ELSE VANA(J) = 0
            'PRINT VANA(J)
            T = T + VANA(J)
    NEXT J
    'PRINT
    IF T > 0 THEN BASARI = BASARI + 1
NEXT I
OLAS = BASARI / N
PRINT "YAKLAŞIK OLASILIK="; OLAS
```

Program çalıştırıldığında

```
DENEME SAYISI=1000
VANALARIN ÇALIŞMA OLASILIĞI=? .5
PARALEL SATIR SAYISI=? 3
YAKLAŞIK OLASILIK= .882
```

gibi bir sonuç çıkacaktır ki yaklaşık olasılık, teorik olarak bulunan (0.875) sonuca yakın bir değerdir. Bu düşüncelerle bazı olasılık problemlerinin teorik çözümleri zor olduğunda veya olanaksız olduğunda simülasyon ile yaklaşımlarda bulunabiliriz. Benzer bir problemi



için yapacak program aşağıda verilmiştir.

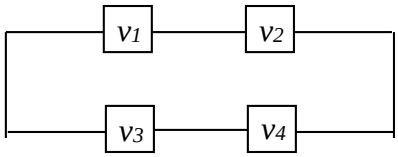
```
'ORDEGIN SU ICMESI OLASILIGI VANALAR SERİ BAGLI
INPUT "DENEME SAYISI=", N
INPUT "VANALARIN ÇALIŞMA OLASILIĞI="; p
INPUT "SERI VANA (SUTUN) SAYISI="; SUTUN
DIM VANA(SUTUN)
FOR I = 1 TO N
   T = 1
   FOR J = 1 TO SUTUN
      IF RND < p THEN VANA(J) = 1 ELSE VANA(J) = 0
         'PRINT VANA(J)
         T = T * VANA(J)
   NEXT J
   'PRINT
   IF T = 1 THEN BASARI = BASARI + 1
NEXT I
OLAS = BASARI / N
PRINT "YAKLASIK OLASILIK="; OLAS
```

Program çalıştırıldığında

```
DENEME SAYISI=1000
VANALARIN ÇALIŞMA OLASILIĞI=? .5
SERI VANA (SUTUN) SAYISI=? 2
YAKLAŞIK OLASILIK= .267
```

olarak bulunur ki teorik olasılık 0.25 değerine yakındır.

Benzer şekilde,



seri bağlanmış vanalar paralel olarak birleştirilirse bu problemin simülasyonunu yapmak için ordek2 ve ordek3 programları birleştirilebilir. Bu amaçla yazılan program aşağıda verilmiştir.

```
INPUT "DENEME SAYISI=", DENEME
INPUT "SATIR SAYISI=", N
INPUT "SUTUN SAYISI=", M
INPUT "VANANIN CALISMA OLASILIGI=", P
BASARI = 0
DIM V(N, M)
FOR k = 1 TO DENEME
   'SAYILARIN URETILMESI
   FOR I = 1 TO N
      FOR J = 1 TO M
         IF RND < P THEN
            V(I, J) = 1
         ELSE
            V(I, J) = 0
         END IF
   NEXT J, I
   'DEGERLERIN YAZDIRILMASI
   'FOR I = 1 TO N
      'FOR J = 1 TO M
         'PRINT V(I, J);
      'NEXT J
      'PRINT
   'NEXT I
   'PRINT "...."
   'ORDEGIN SU ICME SAYISININ BULUNMASI
   TOP = 0
   FOR I = 1 TO N
      CARP = 1
      FOR J = 1 TO M
         CARP = CARP * V(I, J)
      NEXT J
```

```
        TOP = TOP + CARP
      NEXT I
      IF TOP > 0 THEN
        BASARI = BASARI + 1
        'PRINT "ICTI"
        ELSE
        'PRINT "ICMEDI"
      END IF
      'INPUT TUS
    NEXT k
    OLAS = BASARI / DENEME
    PRINT "YAKLASIK OLASILIK=", OLAS
```

Program çalıştırıldığında,

```
DENEME SAYISI=10000
SATIR SAYISI=2
SUTUN SAYISI=2
VANANIN CALISMA OLASILIGI=.5
YAKLASIK OLASILIK=          .4453
```

gibi bir sonuç çıkacaktır. Seri bağlanan vana sayısını artırdığımızda ördeğin su içme olasılığının aynı olasılık değeri için düşmesini bekleriz. Bunun için seri bağlı vana sayısını 4 yapıp programı çalıştırdığımızda

```
DENEME SAYISI=10000
SATIR SAYISI=2
SUTUN SAYISI=4
VANANIN CALISMA OLASILIGI=.5
YAKLASIK OLASILIK=          .1212
```

sonucunu gözlemleriz ve bu sonucun beklentimizdeki gibi olduğunu görürüz.

Örnek-4.2. Boncuk Karıştırma Problemi

Boş bir satranç tahtasının ilk 4 satırına mavi, son 4 satırına da kırmızı boncuklar yerleştirilsin. Boncuklar *n* kez rasgele karıştırıldığında ilk 4 satırda yer alan mavi ve kırmızı boncuk sayısını veren programı yazalım.

Algoritma adımları aşağıdaki gibi tasarlanabilir.

A1. Boncuklar yerleştirilir

Satranç tahtası 8x8'lik bir matris gibi düşünülebilir. Mavi boncuklar 0 ile, kırmızı boncuklar 1 ile temsil edilirse, matrisin ilk 4 satırı 0 değerleri ile, son 4 satırı da 1 değerleri ile doldurulur. Matris tanımlandığında tüm elemanları 0 olacağından son 4 satırı 1 değerleri ile doldurmak yetecektir.

```
FOR i = 5 TO 8
  FOR j = 1 TO 8
```

```
        m(i, j) = 1
      NEXT j
   NEXT i
```

A2. Boncuklar karıştırılır

Matrisin herhangi bir satırı ve herhangi bir sütunu rasgele seçilir, böylelikle herhangi bir boncuk rasgele seçilmiş olur. Yine başka bir satır ve sütun rasgele seçilirse herhangi bir boncuk daha rasgele seçilmiş olur. Bu rasgele seçilen boncuklar yer değiştirilirse karıştırma işlemi bir kez gerçekleşmiş olur.

```
sat1 = INT(RND * 8) + 1
sut1 = INT(RND * 8) + 1
sat2 = INT(RND * 8) + 1
sut2 = INT(RND * 8) + 1
SWAP m(sat1, sut1), m(sat2, sut2)
```

A3. ilk 4 satırdaki kırmızı ve mavi boncukların sayısı hesaplanır

```
mavi1 = 0
kirmizi1 = 0
FOR i = 1 TO 4
    FOR j = 1 TO 8
        IF m(i, j) = 0 THEN mavi1 = mavi1+1 ELSE kirmizi1=kirmizi1+1
    NEXT j
NEXT i
```

İlk 4 satırda bulunan mavi sayısını hesaplamak diğerlerinin de sayısını verecektir. İlk 4 satırdaki mavi boncuk sayısı yani 1'lerin sayısı bir dosyaya (OPEN "o", #1, "c:\ist106\boncuk.txt") yazdırılır.

```
PRINT #1, mavi1
```

Program aşağıda verilmiştir.

```
'Boncuk karıştırma probleminin simülasyonu
OPEN "o", #1, "c:\ist106\boncuk.txt"
INPUT "karıştırma sayısı="; n
DIM m(8, 8)
FOR i = 5 TO 8
   FOR j = 1 TO 8
      m(i, j) = 1
   NEXT j
NEXT i
FOR i = 1 TO 8
   FOR j = 1 TO 8
      PRINT m(i, j);
```

```
    NEXT j
    PRINT
  NEXT i
  INPUT tus
  FOR s = 1 TO n
    sat1 = INT(RND * 8) + 1
    sut1 = INT(RND * 8) + 1
    sat2 = INT(RND * 8) + 1
    sut2 = INT(RND * 8) + 1
    SWAP m(sat1, sut1), m(sat2, sut2)


    'karıştırdıktan sonra matris tekrar yazdırılıyor
    FOR i = 1 TO 8
      FOR j = 1 TO 8
        PRINT m(i, j);
      NEXT j
      PRINT
    NEXT i
    mavi1 = 0
    kirmizi1 = 0
    FOR i = 1 TO 4
      FOR j = 1 TO 8
       IF m(i, j) = 0 THEN mavi1 = mavi1 + 1 ELSE kirmizi1 = kirmizi1 + 1
      NEXT j
    NEXT i
    LOCATE 10, 10: PRINT "1.kısım mavi sayısı="; mavi1
    LOCATE 12, 10: PRINT "1.kısım kırmızı sayısıi="; kirmizi1
    LOCATE 14, 10: PRINT "2.kısım mavi sayısı="; 32 - mavi1
    LOCATE 16, 10: PRINT "2.kısım kırmızı sayısı="; 32 - kirmizi1
    PRINT #1, mavi1
  NEXT s
  CLOSE #1
```

Program çalıştırılıp *n*=500 değeri girilirse, programın sonucu aşağıdaki gibi olacaktır. En son yazdırılan matriste mavi ve kırmızı boncukların nerede bulunduğu görülmektedir.

```
 0  1  0  1  0  1  1  0
 1  1  1  0  1  0  0  0
 1  1  1  1  0  0  1  0
 1  0  1  1  0  0  0  1
 1  0  1  0  1  0  1  1
```

1 1 0 0 0 1 1 0

0 1 1 0 1 1 0 0

0 0 1 0 0 1 0 0

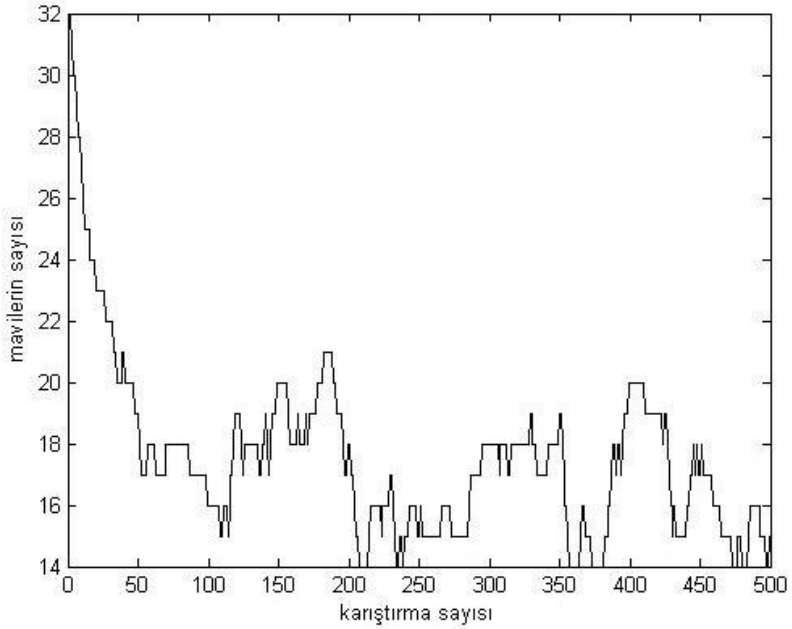
1.kısım mavi sayısı= 15

1.kısım kırmızı sayısı= 17

2.kısım mavi sayısı= 17

2.kısım kırmızı sayısı= 15

Ayrıca boncuk.txt dosyasında bulunan sayıların grafiği Şekil 4.1.'deki gibi çizdirilir. Şekil 4.1.'den görülebileceği gibi bu sayılar 16'nın etrafında salınmaktadır. Yani karıştırma belirli bir sayıya ulaştığında hemen- hemen kırmızı ve mavi boncukların sayısı birbirine yakın çıkmaktadır.



Şekil 4.1. boncuk.txt dosyasında bulunan sayıların grafiği.