BÖLÜM 6

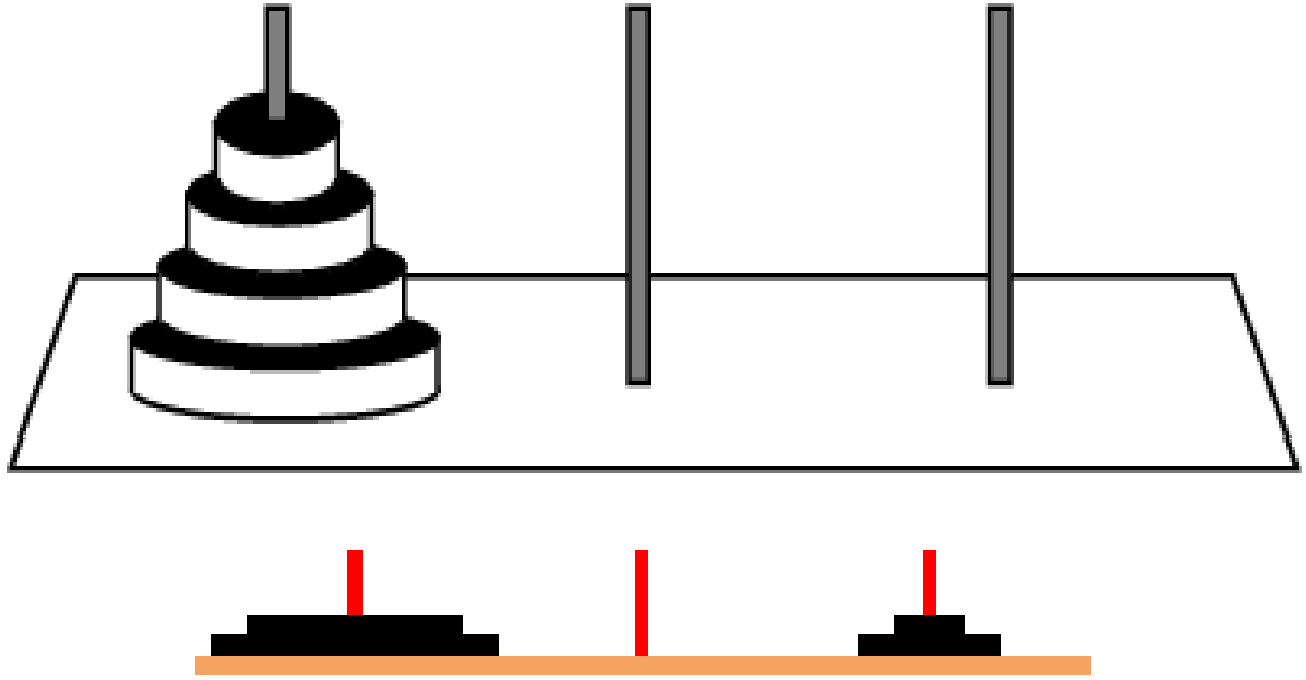
TEKRARLAMALI BAĞINTILAR

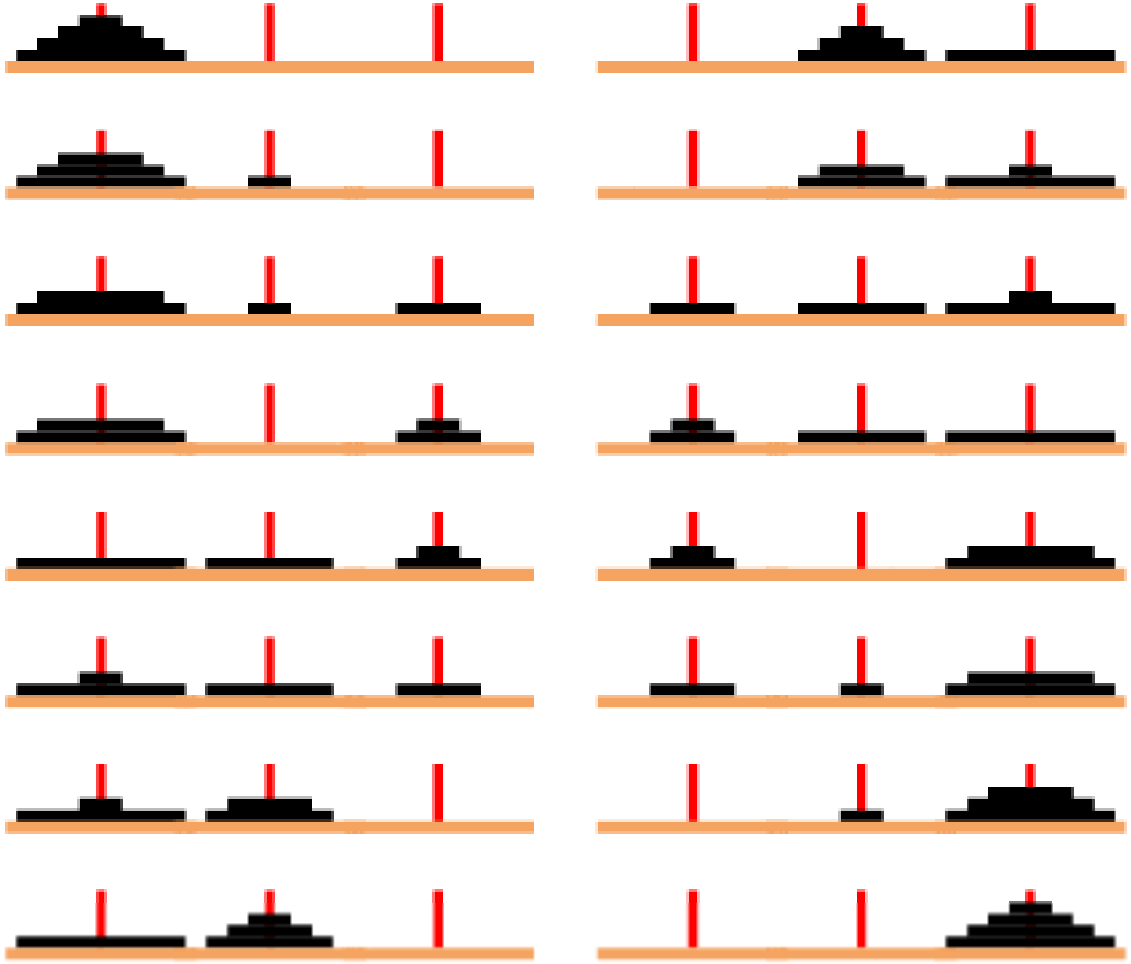
Giriş

- **Tanım 6.1.1:** Bir $a_0,a_1,\ldots$ dizisi için bir **tekrarlamalı bağıntı** a_n ile ondan önce gelen $a_0,a_1,\ldots,a_{n-1}$ arasında bir denklemdir.

- **Örnek 6.1.4:** 111 alt karakter dizisini kapsamayan n adet bitten oluşan tüm karakter dizilerinin sayısını S_n göstersin. S_1, S_2,... için bir tekrarlamalı bağıntı ve S dizisi için başlangıç koşullarını oluşturalım.
- 111 alt karakter dizisini içermeyen n-bitlik tüm karakter dizilerinin sayısını sayacağız:
  - 0 ile başlayanlar
  - 10 ile başlayanlar
  - 11 ile başlayanlar

Hanoi Kuleleri





Tekrarlamalı Bağıntıların Çözümleri

- **Tanım 6.2.4 :**

  $a_n=c_1a_{n-1}+c_2a_{n-2}+\ldots+c_ka_{n-k}$, $c_k \geq 0$

  biçiminde verilen yineleme bağıntısına **sabit katsayılı k mertebeden bir doğrusal homojen tekrarlamalı bağıntısı** denir.

- **Teorem 6.2.7 :** Sabit katsayılı ikinci-mertebeden, doğrusal tekrarlamalı bağıntısı

  $$a_n=c_1a_{n-1}+c_2a_{n-2} \qquad (6.2)$$

  ve $a_0=c_0$, $a_1=c_1$ olsun.
  Eğer S ve T bu denklemin bir çözümü iseler, bu durumda U=bS+dT değerleri de (6.2) denkleminin bir çözümdür.
  Eğer r değeri

  $$t^2-c_1t-c_2=0 \qquad (6.3)$$

  denkleminin bir kökü ise, bu durumda n=0,1,... için r^n 'de (6.2) denklemin bir çözümüdür. (6.2) ile tanımlanan dizi a

  $$a_0=C_0, \quad a_1=C_1$$

  ve r_1, r_2 , $r_1 \neq r_2$, değerleri (6.3) denkleminin kökleri iseler, bu durumda

  $$a_n=br_1^n+dr_2^n, \quad n=0,1,....$$

  olacak şekilde b ve d sabitleri vardır.

- **Teorem 6.2.9 :**

  $a_n=c_1a_{n-1}+c_2a_{n-2}$, (6.4)

  sabit katsayılı, ikinci mertebeden, doğrusal homojen tekrarlamalı bağıntısı verilsin. (6.4) denklemini sağlayan dizi a ve

  $a_0=C_0$, $a_1=C_1$

  olsunlar. Eğer

  $t^2-c_1t-c_2=0$ (6.5)

  denkleminin her iki kökü bir r değerine eşit ise, bu durumda

  $a_n=br^n+dnr^n$, $n=0,1,\ldots$

  olacak şekilde b ve d sabitleri vardır.

Algoritma Analizi

Bu algoritma

$s_1, s_2, \ldots, s_n$

dizisindeki en büyük elemanı seçer ve bu elemanı en sona yere yerleştirir. Sonra bir özyinelemeli algoritma kullanarak dizinin elemanlarını artan sırada yerleştirir.

Girdi: $s_1, s_2, \ldots, s_n$ ve dizinin boyunu gösteren n

Çıktı: Artan sırada düzenlenmiş $s_1, s_2, \ldots, s_n$ dizisi

```
1.  selection_sort (s,n) {
2.    //base case
3.    if (n ==1)
4.        return
5.    //find largest
6.    max_index :=1
7.    for i =2 to n
8.          if (s_i > s_max_index)
9.              max_index :=i
10.   //move largest to end
11.   swap(s_n, s_max_index )
12.   selection_sort (s,n-1)
13. }
```

Bu algoritma bir artan dizi içinde bir değer arar. Eğer değer bulunursa, bu değerin damgasını aksi takdirde 0 döndürür.

Girdi: Artan sırada verilmiş bir , $i \geq 1$, $s_i, s_{i+1}, \ldots, s_j$ dizisi, bir *key* değeri,i ve j

Çıktı: s_k=*key* olacak şekilde bir k damgası ya da 0

```
1.  binary_search(s,i,j,key) {
2.      if (i>j) //not found
3.          return 0
4.      k :=⌊(i +j )/2⌋
5.      if (key ==s_k )  //found
6.          return  k
7.      if  (key <s_k )  //search left half
8.          j :=k-1
9.      else  //search right half
10.         i :=k+1
11.     return  binary_search(s,i,j,key)
12. }
```