4.4 Sıralama

- → kelime listesi: alfabetik olarak, uzunluğuna göre
- → şehirler: nüfusuna, alanına, posta koduna göre
- → **bazı algoritmalar sıralı koleksiyonlarda oldukça başarılı**
  - → anagram, ikili arama

algoritmalar

- → çok sayıda sıralama algoritması
- → bilg.bil. önemli algoritmalardan – sıralama
- → eleman sayısı arttıkça sıralama sorunlu olur
- → aramaya benzer olarak, etkinlik eleman sayısıyla ilişkili
- → az elemanlılarda karmaşık algoritmalar başarısız olabilir

işlemler

analiz ederken

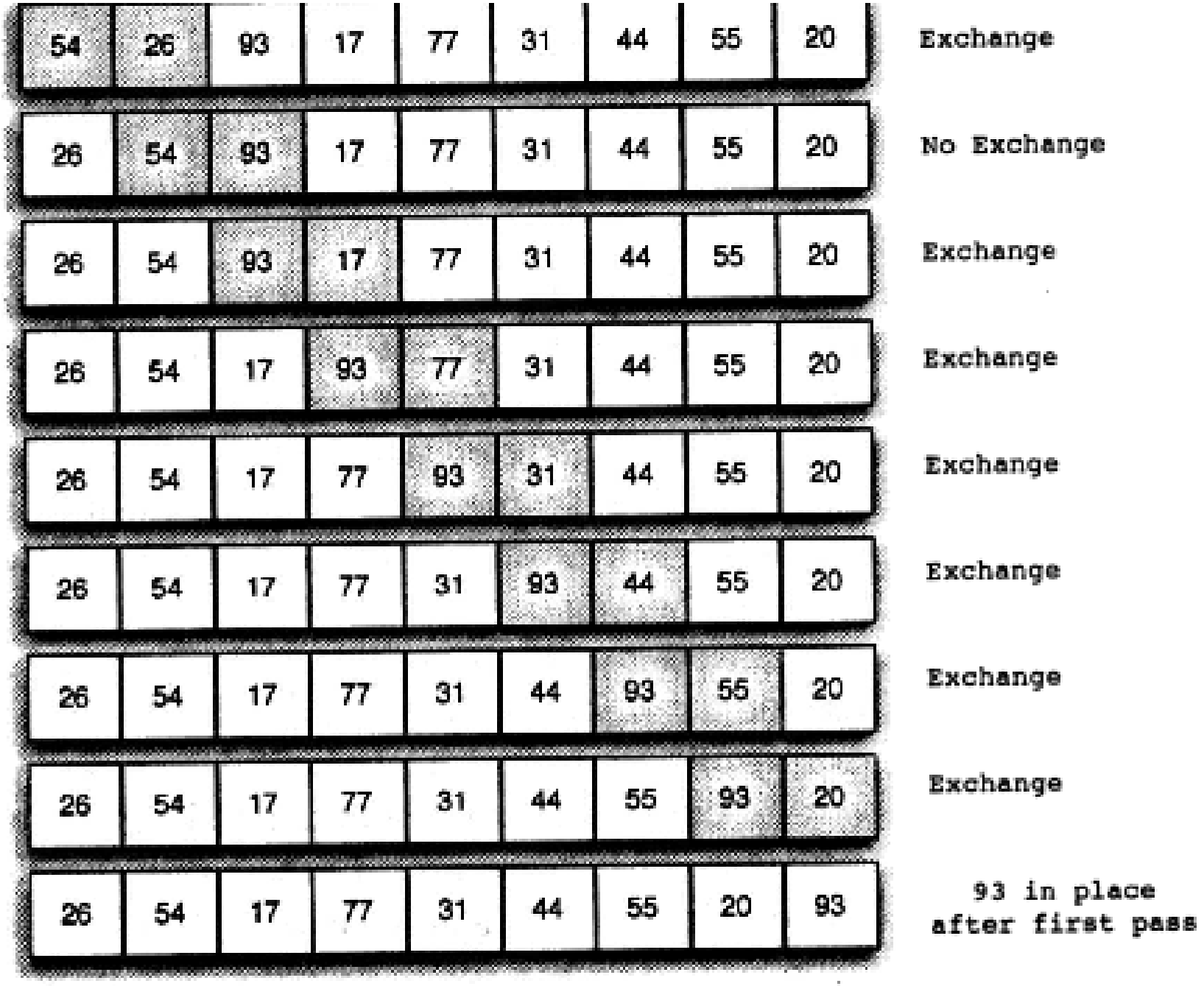
1. büyüklük sınaması: toplam karşılaştırma adedi
2. doğru sıralarına koyma işlemi

4.4.1 baloncuk (bubble) sıralama

- → listede dolaş
- → komşu elemanları karşılaştır
- → gerekirse yer değiştir
- → her geçişte sonraki enbüyük doğru yerdedir
- → her bir eleman doğru yerine gelinceye kadar devam et

ilk geçiş

ilk geçiş



açıklama

→ ilk geçişte (n-1) eleman karşılaştırılır

→ ikincisinde (n-2) eleman karşılaştırılır

→ vs

gerçekleme

gerçekleme

```
1       def bubbleSort(alist):
2           for passnum in range(len(alist)-1,0,-1):
3               for i in range(passnum):
4                   if alist[i]>alist[i+1]:
5                       alist[i],alist[i+1]=alist[i+1],alist[i]
```

takas işlemi

→ takas işlemi, diğer dillerden farklıdır

→ normalde

```
1  temp = a
2  a = b
3  b = temp
```

takas işlemi

→ takas işlemi, diğer dillerden farklıdır

→ normalde

```
1  temp = a
2  a = b
3  b = temp
```

→ pythonda

```
1  a, b = b, a
```

Pass	Comparisons
1	$n-1$
2	$n-2$
3	$n-3$
...	
$n-1$	1

Table 4.7: Comparisons for Each Pass of Bubble Sort

analiz

En kötü durum

→ karşılaştırma adetleri=

→ (n-1)+(n-2)+...+1

→ (n-1)*n/2= n^2/2 - n/2

→ dolayısıyla O(n^2)

analiz

En kötü durum

→ karşılaştırma adetleri =

→ (n-1)+(n-2)+...+1

→ (n-1)*n/2 = n^2/2 - n/2

→ dolayısıyla O(n^2)

En iyi durum

→ takas olmaz O(n)

Ortalama durum

→ ikisinin ortalaması

sonuç

- → en verimsizi
- → iyi: ilk geçişte eldeki/işlenen listenin sıralı olup-olmadığı anlaşılabiliyor

short bubble

→ sıralandığında bitir: short bubble

```
1       def shortBubbleSort(alist):
2           exchanges = True
3           passnum = len(alist)-1
4           while passnum > 0 and exchanges:
5              exchanges = False
6              for i in range(passnum):
7                  if alist[i]>alist[i+1]:
8                      exchanges = True
9                      alist[i],alist[i+1]=alist[i+1],alist[i]
10             passnum = passnum-1
```

→ passnum geçişte hiçbir değişiklik olmadıysa liste sıralı (sıralanma işlemi tamam)

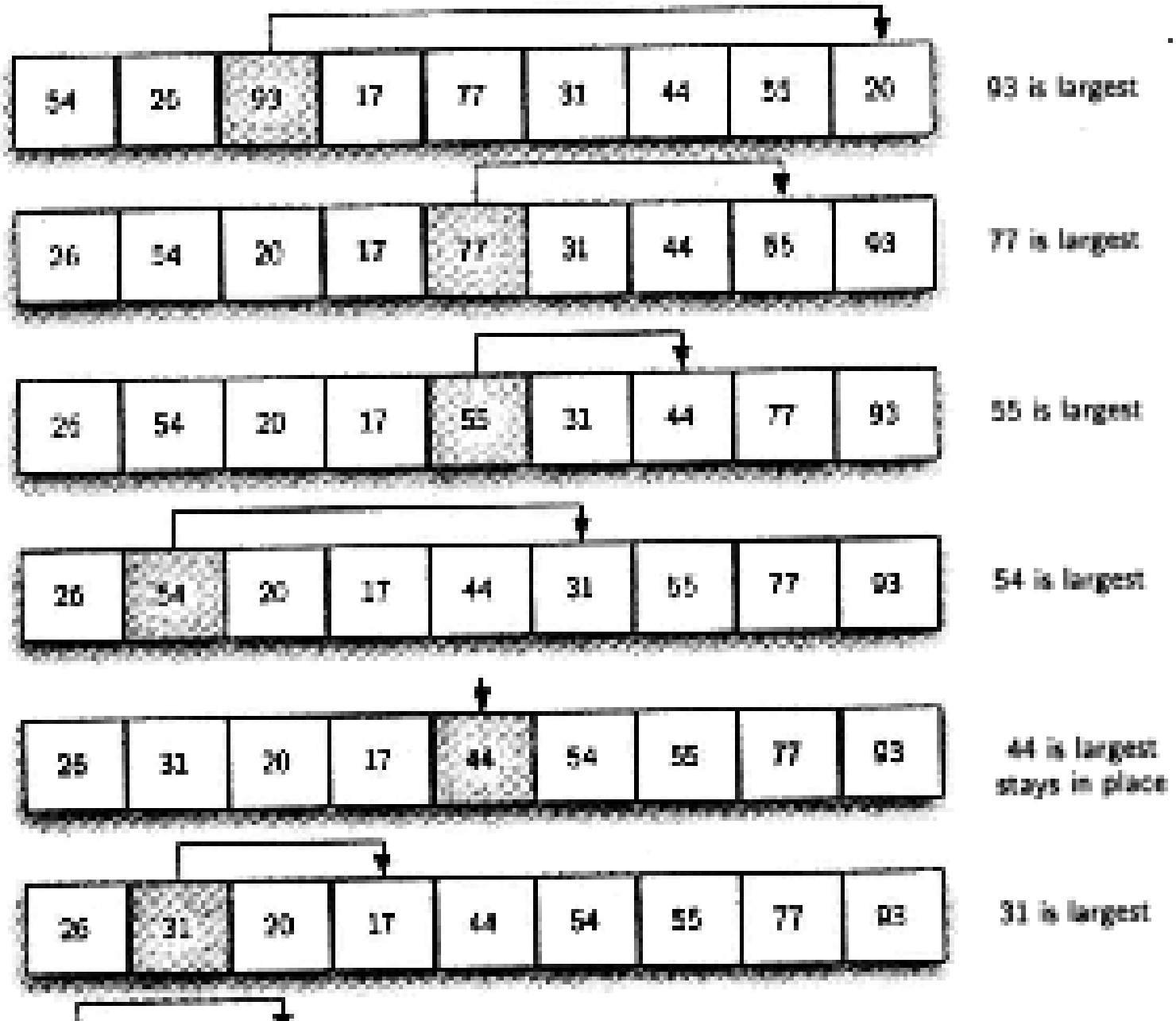
→ devam etmeye gerek yok

4.4.2 seçmeli (selection) sıralama

- → listedeki en büyük elemanı bul
- → en sona koy
- → sonraki enbüyüğü, yanına
- → vs

seçmeli sıralama

→ (n-1.) geçişte herkes doğru yerindedir



gerçekleme

gerçekleme

```
1       def selectionSort(alist):
2           for fillslot in range(len(alist)-1,0,-1):
3               positionOfMax=0
4               for location in range(1,fillslot+1):
5                   if alist[location]>alist[positionOfMax]:
6                       positionOfMax = location
7
8               alist[positionOfMax],alist[fillslot] = \
9                           alist[fillslot],alist[positionOfMax]
```

analiz

- → bubble ile aynı sayıda karşılaştırma
- → dolayısıyla O(n^2)
- → yerdeğiştirme miktarı daha az
- → yürütme zamanı daha kısa
- → örnek listemizde bubble: 20 takas, selection: 8 takas