Bölüm 3: Sıralama Algoritmaları

Bölüm 3: Sıralama Algoritmaları · Çalışma Soruları

Kararlılık ve yerinde sıralama | Kabarcık | Eklemeli | Seçimli | Shell | Birleştirme | Hızlı | Heap sort | Saymalı ve basamak | Karşılaştırma

Sercan KÜLCÜ · Giresun Üniversitesi · Bilgisayar Mühendisliği

© Sercan Külcü. Tüm hakları saklıdır. · Sunumun etkileşimli HTML sürümü:
https://sercankulcu.github.io/files/algorithms/slides/Bolum_03_Siralama_Algoritmalari_Calisma_Sorulari.html

ÇOKTAN SEÇMELİ · SIRALAMA ÖLÇÜTLERİ

Soru 1: Kararsız Algoritma

Kayıtlar yalnızca sayıya göre sıralanıyor: (3, A), (3, C), (1, B). Aşağıdaki algoritmalardan hangisi (sunumdaki kodlarıyla) A ile C'nin göreli sırasını **bozabilir**?

A) Kabarcık sıralaması

B) Eklemeli sıralama

C) Seçimli sıralama

D) Birleştirme sıralaması

Soru 2: Bayraklı Kabarcık: Turlar

Bayraklı kabarcık sıralaması [5, 2, 4, 6, 1, 3] dizisine uygulanıyor. Her tur sonunda dizi nedir, algoritma hangi turda durur?

```
1  boolean yerDegistiMi;
2  for(int i = 0; i < n - 1; i++) {
3    yerDegistiMi = false;
4    for(int j = 0; j < n - 1 - i; j++) {
5      if(dizi[j] > dizi[j+1]) {
6        int gecici = dizi[j];
7        dizi[j] = dizi[j+1];
8        dizi[j+1] = gecici;
9        yerDegistiMi = true;
10     }
11   }
12   if(yerDegistiMi == false) break;
13 }
```

Adım 1 / 7

DEĞİŞKENLER

—

ÇIKTI

Soru 2: Bayraklı Kabarcık: Turlar

Bayraklı kabarcık sıralaması [5, 2, 4, 6, 1, 3] dizisine uygulanıyor. Her tur sonunda dizi nedir, algoritma hangi turda durur?

```
▶ boolean yerDegistiMi;
for(int i = 0; i < n - 1; i++) {
  yerDegistiMi = false;
  for(int j = 0; j < n - 1 - i; j++) {
    if(dizi[j] > dizi[j+1]) {
      int gecici = dizi[j];
      dizi[j] = dizi[j+1];
      dizi[j+1] = gecici;
      yerDegistiMi = true;
    }
  }
  if(yerDegistiMi == false) break;
}
```

DEĞİŞKENLER

`dizi = [5, 2, 4, 6, 1, 3]`

ÇIKTI



n = 6. Her tur komşu çiftleri soldan sağa karşılaştırır; tur sonunda kalanların en büyüğü sona yerleşir.

Adım 2 / 7

Soru 2: Bayraklı Kabarcık: Turlar

Bayraklı kabarcık sıralaması [5, 2, 4, 6, 1, 3] dizisine uygulanıyor. Her tur sonunda dizi nedir, algoritma hangi turda durur?

```
1  boolean yerDegistiMi;
2  for(int i = 0; i < n - 1; i++) {
▶    yerDegistiMi = false;
▶    for(int j = 0; j < n - 1 - i; j++) {
▶      if(dizi[j] > dizi[j+1]) {
▶        int gecici = dizi[j];
▶        dizi[j] = dizi[j+1];
▶        dizi[j+1] = gecici;
▶        yerDegistiMi = true;
▶      }
11   }
12   if(yerDegistiMi == false) break;
13 }
```

DEĞİŞKENLER

`i = 0` `takas = 4` `yerDegistiMi = true`

`dizi = [2, 4, 5, 1, 3, 6]`

ÇIKTI

tur i = 0 sonrası

2	4	5	1	3	6
0	1	2	3	4	5

Tur i = 0: 5 karşılaştırma, **4 takas**. 6 son yerine (indis 5) oturdu; bayrak true.

Adım 3 / 7

Soru 2: Bayraklı Kabarcık: Turlar

Bayraklı kabarcık sıralaması [5, 2, 4, 6, 1, 3] dizisine uygulanıyor. Her tur sonunda dizi nedir, algoritma hangi turda durur?

```
1  boolean yerDegistiMi;
2  for(int i = 0; i < n - 1; i++) {
▶    yerDegistiMi = false;
▶    for(int j = 0; j < n - 1 - i; j++) {
▶      if(dizi[j] > dizi[j+1]) {
▶        int gecici = dizi[j];
▶        dizi[j] = dizi[j+1];
▶        dizi[j+1] = gecici;
▶        yerDegistiMi = true;
▶      }
11   }
12   if(yerDegistiMi == false) break;
13 }
```

DEĞİŞKENLER

`i = 1` `takas = 2` `yerDegistiMi = true` `dizi = [2, 4, 1, 3, 5, 6]`

ÇIKTI

tur i = 1 sonrası

2	4	1	3	5	6
0	1	2	3	4	5

Tur i = 1: 4 karşılaştırma, **2 takas**. 5 son yerine (indis 4) oturdu; bayrak true.

Adım 4 / 7

Soru 2: Bayraklı Kabarcık: Turlar

Bayraklı kabarcık sıralaması [5, 2, 4, 6, 1, 3] dizisine uygulanıyor. Her tur sonunda dizi nedir, algoritma hangi turda durur?

```
1  boolean yerDegistiMi;
2  for(int i = 0; i < n - 1; i++) {
▶    yerDegistiMi = false;
▶    for(int j = 0; j < n - 1 - i; j++) {
▶      if(dizi[j] > dizi[j+1]) {
▶        int gecici = dizi[j];
▶        dizi[j] = dizi[j+1];
▶        dizi[j+1] = gecici;
▶        yerDegistiMi = true;
▶      }
11   }
12   if(yerDegistiMi == false) break;
13 }
```

DEĞİŞKENLER

`i = 2` `takas = 2` `yerDegistiMi = true`

`dizi = [2, 1, 3, 4, 5, 6]`

ÇIKTI

tur i = 2 sonrası

2	1	3	4	5	6
0	1	2	3	4	5

Tur i = 2: 3 karşılaştırma, **2 takas**. 4 son yerine (indis 3) oturdu; bayrak true.

Soru 2: Bayraklı Kabarcık: Turlar

Bayraklı kabarcık sıralaması [5, 2, 4, 6, 1, 3] dizisine uygulanıyor. Her tur sonunda dizi nedir, algoritma hangi turda durur?

```
1  boolean yerDegistiMi;
2  for(int i = 0; i < n - 1; i++) {
▶    yerDegistiMi = false;
▶    for(int j = 0; j < n - 1 - i; j++) {
▶      if(dizi[j] > dizi[j+1]) {
▶        int gecici = dizi[j];
▶        dizi[j] = dizi[j+1];
▶        dizi[j+1] = gecici;
▶        yerDegistiMi = true;
▶      }
11   }
12   if(yerDegistiMi == false) break;
13 }
```

DEĞİŞKENLER

`i = 3` `takas = 1` `yerDegistiMi = true`

`dizi = [1, 2, 3, 4, 5, 6]`

ÇIKTI

tur i = 3 sonrası



Tur i = 3: 2 karşılaştırma, **1 takas**. 3 son yerine (indis 2) oturdu; bayrak true.

Soru 2: Bayraklı Kabarcık: Turlar

Bayraklı kabarcık sıralaması [5, 2, 4, 6, 1, 3] dizisine uygulanıyor. Her tur sonunda dizi nedir, algoritma hangi turda durur?

```
1  boolean yerDegistiMi;
2  for(int i = 0; i < n - 1; i++) {
▶    yerDegistiMi = false;
▶    for(int j = 0; j < n - 1 - i; j++) {
▶      if(dizi[j] > dizi[j+1]) {
6        int gecici = dizi[j];
7        dizi[j] = dizi[j+1];
8        dizi[j+1] = gecici;
9        yerDegistiMi = true;
10     }
11   }
▶    if(yerDegistiMi == false) break;
13 }
```

DEĞİŞKENLER

`i = 4` `takas = 0` `yerDegistiMi = false`

`dizi = [1, 2, 3, 4, 5, 6]`

ÇIKTI

tur i = 4 sonrası



Tur i = 4: 1 karşılaştırma, hiç takas yok → bayrak false → **break**. Toplam 5 + 4 + 3 + 2 + 1 = 15 karşılaştırma, 9 takas.

ÇOKTAN SEÇMELİ · KABARCIK SIRALAMASI

Soru 3: Sıralı Girdide Bayrak

1000 öğeli, zaten artan sıralı bir diziye **bayraklı** kabarcık sıralaması uygulanırsa kaç karşılaştırma yapılır?

A) 499.500

B) 1.000

C) 999

D) 0

ÇIKTI SORUSU · EKLEMELİ SIRALAMA

Soru 4: Karşılaştırma ve Kaydırma Sayısı

Eklemeli sıralama [5, 2, 4, 6, 1, 3] dizisini sıralarken `dizi[j] > gecici` karşılaştırması kaç kez yapılır, kaç kaydırma yapılır?

```
for(int i = 1; i < n; i++) {
  int gecici = dizi[i];
  int j = i - 1;
  while(j >= 0 && dizi[j] > gecici) {
    dizi[j+1] = dizi[j];
    j = j - 1;
  }
  dizi[j+1] = gecici;
}
```

▶ **Cevabı göster**

ÇOKTAN SEÇMELİ · EKLEMELİ SIRALAMA

Soru 5: İkili Arama ile Ekleme

Eklemeli sıralamada anahtarın yerleşeceği konum, sıralı kısımda **ikili arama** ile bulunursa en kötü durum zaman karmaşıklığı ne olur?

A) $\Theta(n \log n)$, çünkü her konum $O(\log n)$'de bulunur

B) $\Theta(\log n)$, çünkü arama logaritmik olur

C) $\Theta(n^2)$, çünkü kaydırmalar hâlâ $\Theta(n^2)$'dir

D) $\Theta(n)$, çünkü kaydırma gerekmez

Soru 6: Her Turdan Sonra Yazdırma

Her dış turdan sonra yazdıran eklemeli sıralama `A = {5, 7, 3, 4, 2, 8}` için ne yazdırır?

```
1 for (int i = 1; i < A.length; ++i) {
2   int anahtar = A[i];
3   int j = i - 1;
4   while (j >= 0 && A[j] > anahtar) {
5     A[j + 1] = A[j];
6     j = j - 1;
7   }
8   A[j + 1] = anahtar;
9   System.out.println(Arrays.toString(A));
10 }
```

Adım 1 / 9

DEĞİŞKENLER

—

ÇIKTI

Soru 6: Her Turdan Sonra Yazdırma

Her dış turdan sonra yazdıran eklemeli sıralama `A = {5, 7, 3, 4, 2, 8}` için ne yazdırır?

```
▶ for (int i = 1; i < A.length; ++i) {
▶   int anahtar = A[i];
▶   int j = i - 1;
▶   while (j >= 0 && A[j] > anahtar) {
5     A[j + 1] = A[j];
6     j = j - 1;
7   }
▶   A[j + 1] = anahtar;
▶   System.out.println(Arrays.toString(A));
10 }
```

DEĞİŞKENLER

`i = 1` `anahtar = 7` `j = 0`

`A = [5, 7, 3, 4, 2, 8]`

ÇIKTI

```
[5, 7, 3, 4, 2, 8]
```



i = 1: anahtar = 7, A[0] = 5 ≤ 7 → while hiç dönmez; 7 yerinde kalır. Dizi değişmeden yazdırılır.

Soru 6: Her Turdan Sonra Yazdırma

Her dış turdan sonra yazdıran eklemeli sıralama `A = {5, 7, 3, 4, 2, 8}` için ne yazdırır?

```
1  for (int i = 1; i < A.length; ++i) {
▶    int anahtar = A[i];
▶    int j = i - 1;
▶    while (j >= 0 && A[j] > anahtar) {
▶      A[j + 1] = A[j];
▶      j = j - 1;
7    }
8    A[j + 1] = anahtar;
9    System.out.println(Arrays.toString(A));
10 }
```

DEĞİŞKENLER

`i = 2` `anahtar = 3` `j = -1`

`A = [5, 5, 7, 4, 2, 8]`

ÇIKTI

```
[5, 7, 3, 4, 2, 8]
```



anahtar = 3. Kendinden büyük **2** öğe birer sağa kaydırıldı; j = -1 olunca koşul yanlış. Boşalan hücre j + 1 = 0.

Soru 6: Her Turdan Sonra Yazdırma

Her dış turdan sonra yazdıran eklemeli sıralama `A = {5, 7, 3, 4, 2, 8}` için ne yazdırır?

```
1  for (int i = 1; i < A.length; ++i) {
2    int anahtar = A[i];
3    int j = i - 1;
4    while (j >= 0 && A[j] > anahtar) {
5      A[j + 1] = A[j];
6      j = j - 1;
7    }
▶    A[j + 1] = anahtar;
▶    System.out.println(Arrays.toString(A));
10 }
```



`A[0] = 3` ; A[0..2] sıralı. Satır yazdırılır.

Adım 4 / 9

DEĞİŞKENLER

`i = 2` `anahtar = 3` `j = -1`

`A = [3, 5, 7, 4, 2, 8]`

ÇIKTI

```
[5, 7, 3, 4, 2, 8]
[3, 5, 7, 4, 2, 8]
```

Soru 6: Her Turdan Sonra Yazdırma

Her dış turdan sonra yazdıran eklemeli sıralama `A = {5, 7, 3, 4, 2, 8}` için ne yazdırır?

```
1  for (int i = 1; i < A.length; ++i) {
▶    int anahtar = A[i];
▶    int j = i - 1;
▶    while (j >= 0 && A[j] > anahtar) {
▶      A[j + 1] = A[j];
▶      j = j - 1;
7    }
8    A[j + 1] = anahtar;
9    System.out.println(Arrays.toString(A));
10 }
```

DEĞİŞKENLER

`i = 3` `anahtar = 4` `j = 0`

`A = [3, 5, 5, 7, 2, 8]`

ÇIKTI

```
[5, 7, 3, 4, 2, 8]
[3, 5, 7, 4, 2, 8]
```



anahtar = 4. Kendinden büyük **2** öğe birer sağa kaydırıldı; j = 0 olunca koşul yanlış. Boşalan hücre j + 1 = 1.

Adım 5 / 9

Soru 6: Her Turdan Sonra Yazdırma

Her dış turdan sonra yazdıran eklemeli sıralama `A = {5, 7, 3, 4, 2, 8}` için ne yazdırır?

```
1  for (int i = 1; i < A.length; ++i) {
2    int anahtar = A[i];
3    int j = i - 1;
4    while (j >= 0 && A[j] > anahtar) {
5      A[j + 1] = A[j];
6      j = j - 1;
7    }
▶    A[j + 1] = anahtar;
▶    System.out.println(Arrays.toString(A));
10 }
```



`A[1] = 4` ; A[0..3] sıralı. Satır yazdırılır.

Adım 6 / 9

DEĞİŞKENLER

`i = 3` `anahtar = 4` `j = 0`

`A = [3, 4, 5, 7, 2, 8]`

ÇIKTI

```
[5, 7, 3, 4, 2, 8]
[3, 5, 7, 4, 2, 8]
[3, 4, 5, 7, 2, 8]
```

Soru 6: Her Turdan Sonra Yazdırma

Her dış turdan sonra yazdıran eklemeli sıralama `A = {5, 7, 3, 4, 2, 8}` için ne yazdırır?

```
1  for (int i = 1; i < A.length; ++i) {
▶    int anahtar = A[i];
▶    int j = i - 1;
▶    while (j >= 0 && A[j] > anahtar) {
▶      A[j + 1] = A[j];
▶      j = j - 1;
7    }
8    A[j + 1] = anahtar;
9    System.out.println(Arrays.toString(A));
10 }
```

DEĞİŞKENLER

`i = 4` `anahtar = 2` `j = -1`

`A = [3, 3, 4, 5, 7, 8]`

ÇIKTI

```
[5, 7, 3, 4, 2, 8]
[3, 5, 7, 4, 2, 8]
[3, 4, 5, 7, 2, 8]
```



anahtar = 2. Kendinden büyük **4** öğe birer sağa kaydırıldı; j = -1 olunca koşul yanlış. Boşalan hücre j + 1 = 0.

Adım 7 / 9

Soru 6: Her Turdan Sonra Yazdırma

Her dış turdan sonra yazdıran eklemeli sıralama `A = {5, 7, 3, 4, 2, 8}` için ne yazdırır?

```
1  for (int i = 1; i < A.length; ++i) {
2    int anahtar = A[i];
3    int j = i - 1;
4    while (j >= 0 && A[j] > anahtar) {
5      A[j + 1] = A[j];
6      j = j - 1;
7    }
▶    A[j + 1] = anahtar;
▶    System.out.println(Arrays.toString(A));
10 }
```



`A[0] = 2` ; A[0..4] sıralı. Satır yazdırılır.

Adım 8 / 9

DEĞİŞKENLER

`i = 4` `anahtar = 2` `j = -1`

`A = [2, 3, 4, 5, 7, 8]`

ÇIKTI

```
[5, 7, 3, 4, 2, 8]
[3, 5, 7, 4, 2, 8]
[3, 4, 5, 7, 2, 8]
[2, 3, 4, 5, 7, 8]
```

Soru 6: Her Turdan Sonra Yazdırma

Her dış turdan sonra yazdıran eklemeli sıralama `A = {5, 7, 3, 4, 2, 8}` için ne yazdırır?

```
▶ for (int i = 1; i < A.length; ++i) {
▶   int anahtar = A[i];
▶   int j = i - 1;
▶   while (j >= 0 && A[j] > anahtar) {
5     A[j + 1] = A[j];
6     j = j - 1;
7   }
▶   A[j + 1] = anahtar;
▶   System.out.println(Arrays.toString(A));
10 }
```

DEĞİŞKENLER

`i = 5` `anahtar = 8` `j = 4`

`A = [2, 3, 4, 5, 7, 8]`

ÇIKTI

```
[5, 7, 3, 4, 2, 8]
[3, 5, 7, 4, 2, 8]
[3, 4, 5, 7, 2, 8]
[2, 3, 4, 5, 7, 8]
[2, 3, 4, 5, 7, 8]
```



i = 5: anahtar = 8, solundaki 7 ≤ 8 → kaydırma yok, 8 yerinde kalır. Dizi değişmeden yazdırılır. Döngü biter: 5 satır yazıldı, son satır sıralı dizidir.

Soru 7: Durum ve Sınır

Eklemeli sıralamanın çalışma süresi için hangisi doğrudur?

A) Tüm girdiler için $\Theta(n^2)$'dir; sıralı girdide de iç döngü i kez döner.

B) En iyi durumu $O(1)$'dir; sıralı girdide hiç kaydırma yapılmaz.

C) En kötü $O(n^2)$, en iyi $\Omega(n)$ olduğu için süresi $\Theta(n \log n)$'dir.

D) En kötü $\Theta(n^2)$, en iyi $\Theta(n)$; tüm girdiler için $O(n^2)$ ve $\Omega(n)$.

ÇIKTI SORUSU · SEÇİMLİ SIRALAMA

Soru 8: Seçimli: Ekran Çıktısı

Dizi [7, 3, 9, 4, 2] iken aşağıdaki kod ne yazdırır? Kaç gerçek takas yapılır?

```
for (int i = 0; i < dizi.length - 1; i++) {
  int gecici = i;
  for (int j = i + 1; j < dizi.length; j++) {
    if (dizi[j] < dizi[gecici]) {
      gecici = j;
    }
  }
  if (gecici != i) {
    yerDegistir(dizi, i, gecici);
  }
  System.out.println(Arrays.toString(dizi));
}
```

▶ **Cevabı göster**

Soru 9: Shell Sıralaması İzleme

`kabukSirala` [9, 8, 3, 7, 5, 6, 4, 1] dizisine uygulanıyor. Her aralık değerinden sonra dizi nedir?

```
1  void kabukSirala(int[] dizi) {
2    int n = dizi.length;
3    for (int aralik = n / 2; aralik > 0;
4                             aralik /= 2)
5      for (int i = aralik; i < n; i++) {
6        int gecici = dizi[i], j = i;
7        while (j >= aralik
8               && dizi[j - aralik] > gecici) {
9          dizi[j] = dizi[j - aralik];
10         j -= aralik;
11       }
12       dizi[j] = gecici;
13     }
14 }
```

Adım 1 / 7

DEĞİŞKENLER

—

ÇIKTI

[redacted]

Soru 9: Shell Sıralaması İzleme

`kabukSirala` [9, 8, 3, 7, 5, 6, 4, 1] dizisine uygulanıyor. Her aralık değerinden sonra dizi nedir?

```
1  void kabukSirala(int[] dizi) {
▶    int n = dizi.length;
3    for (int aralik = n / 2; aralik > 0;
4                             aralik /= 2)
5      for (int i = aralik; i < n; i++) {
6        int gecici = dizi[i], j = i;
7        while (j >= aralik
8               && dizi[j - aralik] > gecici) {
9          dizi[j] = dizi[j - aralik];
10         j -= aralik;
11       }
12       dizi[j] = gecici;
13     }
14 }
```

DEĞİŞKENLER

`dizi = [9, 8, 3, 7, 5, 6, 4, 1]`

ÇIKTI

başlangıç

9	8	3	7	5	6	4	1
0	1	2	3	4	5	6	7

n = 8; aralık dizisi n/2 = 4, 2, 1. Aynı renkteki öğeler (aralarında aralik kadar indis olanlar) birlikte eklemeli sıralanır.

Adım 2 / 7

Soru 9: Shell Sıralaması İzleme

`kabukSirala` [9, 8, 3, 7, 5, 6, 4, 1] dizisine uygulanıyor. Her aralık değerinden sonra dizi nedir?

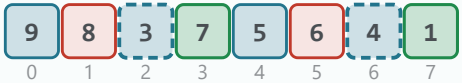
```
1  void kabukSirala(int[] dizi) {
2    int n = dizi.length;
▶    for (int aralik = n / 2; aralik > 0;
▶                             aralik /= 2)
5      for (int i = aralik; i < n; i++) {
6        int gecici = dizi[i], j = i;
7        while (j >= aralik
8               && dizi[j - aralik] > gecici) {
9          dizi[j] = dizi[j - aralik];
10         j -= aralik;
11       }
12       dizi[j] = gecici;
13     }
14 }
```

DEĞİŞKENLER

`aralik = 4` `dizi = [9, 8, 3, 7, 5, 6, 4, 1]`

ÇIKTI

aralik = 4: gruplar



aralik = 4: gruplar [9, 5], [8, 6], [3, 4], [7, 1].

Soru 9: Shell Sıralaması İzleme

`kabukSirala` [9, 8, 3, 7, 5, 6, 4, 1] dizisine uygulanıyor. Her aralık değerinden sonra dizi nedir?

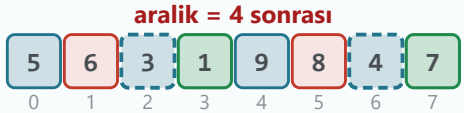
```
1  void kabukSirala(int[] dizi) {
2    int n = dizi.length;
3    for (int aralik = n / 2; aralik > 0;
4                             aralik /= 2)
▶      for (int i = aralik; i < n; i++) {
▶        int gecici = dizi[i], j = i;
▶        while (j >= aralik
▶               && dizi[j - aralik] > gecici) {
▶          dizi[j] = dizi[j - aralik];
▶          j -= aralik;
▶        }
12       dizi[j] = gecici;
13     }
14 }
```

DEĞİŞKENLER

`aralik = 4` `kaydırma = 3`

`dizi = [5, 6, 3, 1, 9, 8, 4, 7]`

ÇIKTI



Her grup kendi içinde sıralandı (**3 kaydırma**) → [5, 6, 3, 1, 9, 8, 4, 7].

Adım 4 / 7

Soru 9: Shell Sıralaması İzleme

`kabukSirala` [9, 8, 3, 7, 5, 6, 4, 1] dizisine uygulanıyor. Her aralık değerinden sonra dizi nedir?

```
1  void kabukSirala(int[] dizi) {
2    int n = dizi.length;
▶   for (int aralik = n / 2; aralik > 0;
▶                            aralik /= 2)
5      for (int i = aralik; i < n; i++) {
6        int gecici = dizi[i], j = i;
7        while (j >= aralik
8               && dizi[j - aralik] > gecici) {
9          dizi[j] = dizi[j - aralik];
10         j -= aralik;
11       }
12       dizi[j] = gecici;
13     }
14 }
```

DEĞİŞKENLER

`aralik = 2` `dizi = [5, 6, 3, 1, 9, 8, 4, 7]`

ÇIKTI

aralik = 2: gruplar



aralik = 2: gruplar [5, 3, 9, 4], [6, 1, 8, 7].

Soru 9: Shell Sıralaması İzleme

`kabukSirala` [9, 8, 3, 7, 5, 6, 4, 1] dizisine uygulanıyor. Her aralık değerinden sonra dizi nedir?

```
1  void kabukSirala(int[] dizi) {
2    int n = dizi.length;
3    for (int aralik = n / 2; aralik > 0;
4                             aralik /= 2)
▶      for (int i = aralik; i < n; i++) {
▶        int gecici = dizi[i], j = i;
▶        while (j >= aralik
▶               && dizi[j - aralik] > gecici) {
▶          dizi[j] = dizi[j - aralik];
▶          j -= aralik;
▶        }
12       dizi[j] = gecici;
13     }
14 }
```

DEĞİŞKENLER

`aralik = 2` `kaydırma = 5`

`dizi = [3, 1, 4, 6, 5, 7, 9, 8]`

ÇIKTI

aralik = 2 sonrası



Her grup kendi içinde sıralandı (**5 kaydırma**) → [3, 1, 4, 6, 5, 7, 9, 8].

Soru 9: Shell Sıralaması İzleme

`kabukSirala` [9, 8, 3, 7, 5, 6, 4, 1] dizisine uygulanıyor. Her aralık değerinden sonra dizi nedir?

```
1  void kabukSirala(int[] dizi) {
2    int n = dizi.length;
3    for (int aralik = n / 2; aralik > 0;
4                             aralik /= 2)
▶      for (int i = aralik; i < n; i++) {
▶        int gecici = dizi[i], j = i;
▶        while (j >= aralik
▶               && dizi[j - aralik] > gecici) {
▶          dizi[j] = dizi[j - aralik];
▶          j -= aralik;
▶        }
12       dizi[j] = gecici;
13     }
14 }
```

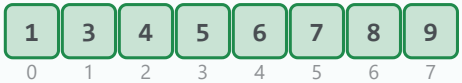
DEĞİŞKENLER

`aralik = 1` `kaydırma = 3`

`dizi = [1, 3, 4, 5, 6, 7, 8, 9]`

ÇIKTI

aralik = 1 sonrası



aralik = 1 sıradan eklemeli sıralamadır; dizi neredeyse sıralı olduğu için yalnızca **3 kaydırma** gerekti. Sonuç [1, 3, 4, 5, 6, 7, 8, 9].

Adım 7 / 7

ÇOKTAN SEÇMELİ · SHELL SIRALAMASI

Soru 10: Shell Sıralamasının Özellikleri

Shell sıralaması (n/2, n/4, ..., 1 aralıklarıyla) için hangisi doğrudur?

A) Kararlıdır, çünkü her grup eklemeli sıralamayla sıralanır.

B) Θ(n) ek bellek ister, çünkü her grup ayrı bir diziye kopyalanır.

C) Aralık 1'e inmeden önce dizi her zaman sıralanmış olur.

D) Son tur sıradan eklemeli sıralamadır ve az kaydırma ister.

Soru 11: İki Sıralı Diziyi Birleştirme

Birleştirme adımı A = [4, 6, 15, 18] ve B = [2, 9, 10, 21] dizilerine uygulanıyor. C hangi sırayla dolar, kaç karşılaştırma yapılır?

```
int i = 0, j = 0, k = 0;
while (i < A.length && j < B.length) {
  if (A[i] <= B[j])
    C[k++] = A[i++];
  else
    C[k++] = B[j++];
}
while (i < A.length)
  C[k++] = A[i++];
while (j < B.length)
  C[k++] = B[j++];
```

Adım 1 / 10

DEĞİŞKENLER

—

ÇIKTI

[redacted]

Soru 11: İki Sıralı Diziyi Birleştirme

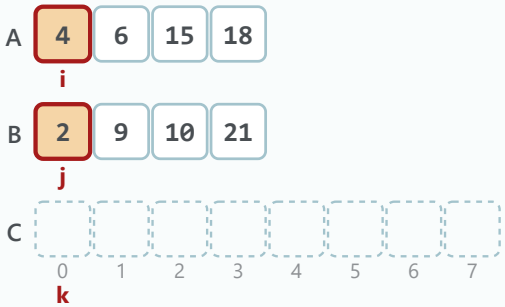
Birleştirme adımı A = [4, 6, 15, 18] ve B = [2, 9, 10, 21] dizilerine uygulanıyor. C hangi sırayla dolar, kaç karşılaştırma yapılır?

```
▶ int i = 0, j = 0, k = 0;
2 while (i < A.length && j < B.length) {
3   if (A[i] <= B[j])
4     C[k++] = A[i++];
5   else
6     C[k++] = B[j++];
7 }
8 while (i < A.length)
9   C[k++] = A[i++];
10 while (j < B.length)
11   C[k++] = B[j++];
```

DEĞİŞKENLER

`i = 0` `j = 0` `k = 0`

ÇIKTI



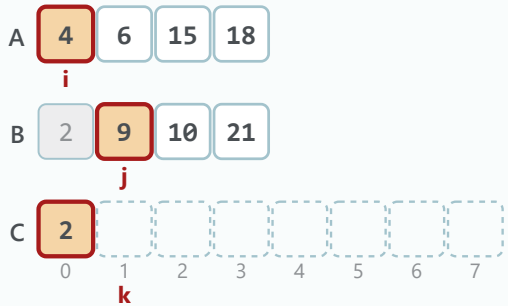
İki sıralı dizi: A = [4, 6, 15, 18], B = [2, 9, 10, 21]. Yalnızca iki dizinin önündeki (sarı) öğeler karşılaştırılır; C boş.

Adım 2 / 10

Soru 11: İki Sıralı Diziyi Birleştirme

Birleştirme adımı A = [4, 6, 15, 18] ve B = [2, 9, 10, 21] dizilerine uygulanıyor. C hangi sırayla dolar, kaç karşılaştırma yapılır?

```
1  int i = 0, j = 0, k = 0;
▶  while (i < A.length && j < B.length) {
▶    if (A[i] <= B[j])
4      C[k++] = A[i++];
▶    else
▶      C[k++] = B[j++];
7  }
8  while (i < A.length)
9    C[k++] = A[i++];
10 while (j < B.length)
11   C[k++] = B[j++];
```



A[0] = 4 > B[0] = 2 → C[0] = 2; j ve k ilerler.

Adım 3 / 10

DEĞİŞKENLER

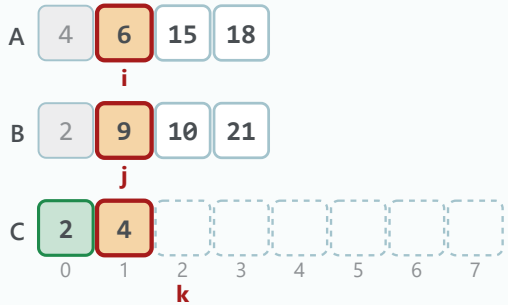
`i = 0` `j = 1` `k = 1` `C = [2]`

ÇIKTI

Soru 11: İki Sıralı Diziyi Birleştirme

Birleştirme adımı A = [4, 6, 15, 18] ve B = [2, 9, 10, 21] dizilerine uygulanıyor. C hangi sırayla dolar, kaç karşılaştırma yapılır?

```
1  int i = 0, j = 0, k = 0;
2  while (i < A.length && j < B.length) {
3    if (A[i] <= B[j])
4      C[k++] = A[i++];
5    else
6      C[k++] = B[j++];
7  }
8  while (i < A.length)
9    C[k++] = A[i++];
10 while (j < B.length)
11   C[k++] = B[j++];
```



A[0] = 4 ≤ B[1] = 9 → C[1] = 4; i ve k ilerler.

DEĞİŞKENLER

i = 1 | j = 1 | k = 2 | C = [2, 4]

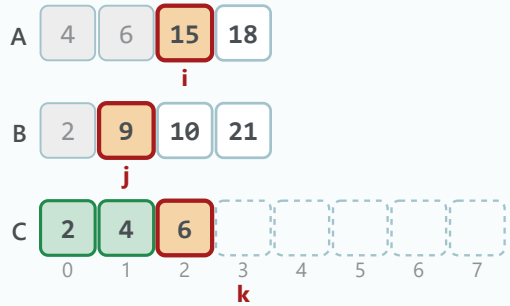
ÇIKTI

Adım 4 / 10

Soru 11: İki Sıralı Diziyi Birleştirme

Birleştirme adımı A = [4, 6, 15, 18] ve B = [2, 9, 10, 21] dizilerine uygulanıyor. C hangi sırayla dolar, kaç karşılaştırma yapılır?

```
1  int i = 0, j = 0, k = 0;
▶  while (i < A.length && j < B.length) {
▶    if (A[i] <= B[j])
▶      C[k++] = A[i++];
5    else
6      C[k++] = B[j++];
7  }
8  while (i < A.length)
9    C[k++] = A[i++];
10 while (j < B.length)
11   C[k++] = B[j++];
```



A[1] = 6 ≤ B[1] = 9 → C[2] = 6; i ve k ilerler.

DEĞİŞKENLER

`i = 2` `j = 1` `k = 3` `C = [2, 4, 6]`

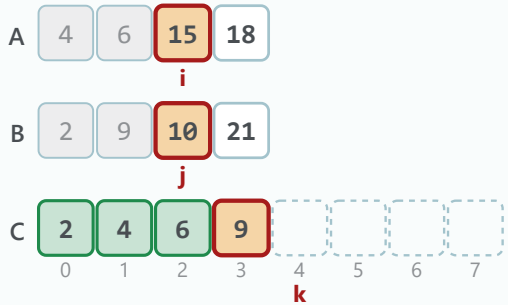
ÇIKTI

Adım 5 / 10

Soru 11: İki Sıralı Diziyi Birleştirme

Birleştirme adımı A = [4, 6, 15, 18] ve B = [2, 9, 10, 21] dizilerine uygulanıyor. C hangi sırayla dolar, kaç karşılaştırma yapılır?

```
1  int i = 0, j = 0, k = 0;
▶  while (i < A.length && j < B.length) {
▶    if (A[i] <= B[j])
4      C[k++] = A[i++];
▶    else
▶      C[k++] = B[j++];
7  }
8  while (i < A.length)
9    C[k++] = A[i++];
10 while (j < B.length)
11   C[k++] = B[j++];
```



A[2] = 15 > B[1] = 9 → C[3] = 9; j ve k ilerler.

DEĞİŞKENLER

`i = 2` `j = 2` `k = 4` `C = [2, 4, 6, 9]`

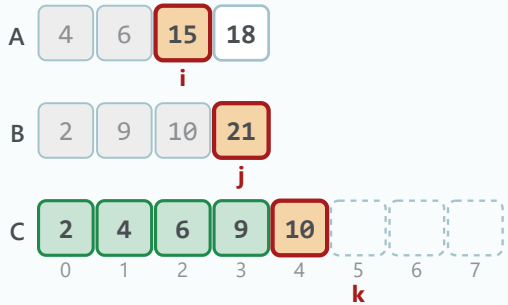
ÇIKTI

Adım 6 / 10

Soru 11: İki Sıralı Diziyi Birleştirme

Birleştirme adımı A = [4, 6, 15, 18] ve B = [2, 9, 10, 21] dizilerine uygulanıyor. C hangi sırayla dolar, kaç karşılaştırma yapılır?

```
1  int i = 0, j = 0, k = 0;
▶  while (i < A.length && j < B.length) {
▶    if (A[i] <= B[j])
4      C[k++] = A[i++];
▶    else
▶      C[k++] = B[j++];
7  }
8  while (i < A.length)
9    C[k++] = A[i++];
10 while (j < B.length)
11   C[k++] = B[j++];
```



A[2] = 15 > B[2] = 10 → C[4] = 10; j ve k ilerler.

DEĞİŞKENLER

`i = 2` `j = 3` `k = 5`

`C = [2, 4, 6, 9, 10]`

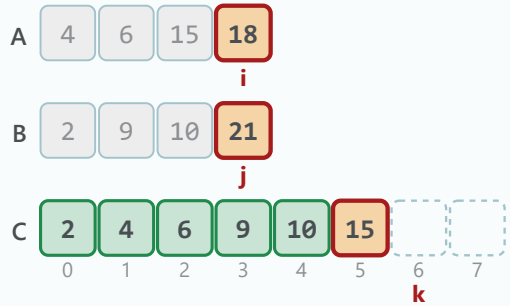
ÇIKTI

Adım 7 / 10

Soru 11: İki Sıralı Diziyi Birleştirme

Birleştirme adımı A = [4, 6, 15, 18] ve B = [2, 9, 10, 21] dizilerine uygulanıyor. C hangi sırayla dolar, kaç karşılaştırma yapılır?

```
1  int i = 0, j = 0, k = 0;
▶  while (i < A.length && j < B.length) {
▶    if (A[i] <= B[j])
▶      C[k++] = A[i++];
5    else
6      C[k++] = B[j++];
7  }
8  while (i < A.length)
9    C[k++] = A[i++];
10 while (j < B.length)
11   C[k++] = B[j++];
```



A[2] = 15 ≤ B[3] = 21 → C[5] = 15; i ve k ilerler.

DEĞİŞKENLER

`i = 3` `j = 3` `k = 6`

`C = [2, 4, 6, 9, 10, 15]`

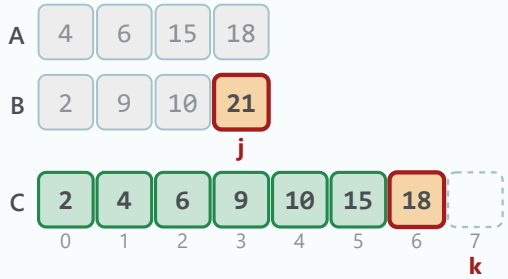
ÇIKTI

Adım 8 / 10

Soru 11: İki Sıralı Diziyi Birleştirme

Birleştirme adımı A = [4, 6, 15, 18] ve B = [2, 9, 10, 21] dizilerine uygulanıyor. C hangi sırayla dolar, kaç karşılaştırma yapılır?

```
1  int i = 0, j = 0, k = 0;
▶  while (i < A.length && j < B.length) {
▶    if (A[i] <= B[j])
▶      C[k++] = A[i++];
5    else
6      C[k++] = B[j++];
7  }
8  while (i < A.length)
9    C[k++] = A[i++];
10 while (j < B.length)
11   C[k++] = B[j++];
```



A[3] = 18 ≤ B[3] = 21 → C[6] = 18; i ve k ilerler. A bitti; ilk while sona erer.

Adım 9 / 10

DEĞİŞKENLER

`i = 4` `j = 3` `k = 7`

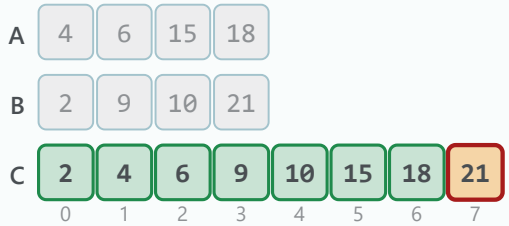
`C = [2, 4, 6, 9, 10, 15, 18]`

ÇIKTI

Soru 11: İki Sıralı Diziyi Birleştirme

Birleştirme adımı A = [4, 6, 15, 18] ve B = [2, 9, 10, 21] dizilerine uygulanıyor. C hangi sırayla dolar, kaç karşılaştırma yapılır?

```
1 int i = 0, j = 0, k = 0;
2 while (i < A.length && j < B.length) {
3   if (A[i] <= B[j])
4     C[k++] = A[i++];
5   else
6     C[k++] = B[j++];
7 }
▶ while (i < A.length)
9   C[k++] = A[i++];
▶ while (j < B.length)
▶   C[k++] = B[j++];
```



B'de kalan 21 karşılaştırmasız kopyalanır. Sonuç [2, 4, 6, 9, 10, 15, 18, 21]: 8 yazma, **7 karşılaştırma** → Θ(n).

Adım 10 / 10

DEĞİŞKENLER

`i = 4` `j = 4` `k = 8`

`C = [2, 4, 6, 9, 10, 15, 18, 21]`

ÇIKTI

```
[2, 4, 6, 9, 10, 15, 18, 21]
```

Soru 12: birlestir Çağrıları

`birlestirmeliSirala(dizi, 0, 6)` çağrısı [38, 27, 43, 3, 9, 82, 10] dizisini sıralarken `birlestir` kaç kez çağrılır? İlk dört birleştirmenin sonucu (birleştirilen aralık) nedir?

```
void birlestirmeliSirala(int[] dizi,
                         int sol, int sag) {
  if (sol < sag) {
    int orta = sol + (sag - sol) / 2;
    birlestirmeliSirala(dizi, sol, orta);
    birlestirmeliSirala(dizi, orta + 1, sag);
    birlestir(dizi, sol, orta, sag);
  }
}
```

▶ **Cevabı göster**

ÇOKTAN SEÇMELİ · BİRLEŞTİRME SIRALAMASI

Soru 13: 8 Öğede En Çok Karşılaştırma

Birleştirme sıralaması 8 öğeli bir diziyi en kötü durumda en fazla kaç öğe karşılaştırmasıyla sıralar?

A) 24

B) 17

C) 28

D) 7

Soru 14: Eşitlikte Hangisi?

`birlestir` metodunda `L[i] <= R[j]` koşulu `L[i] < R[j]` olarak değiştirilirse ne olur?

A) Dizi artık doğru sıralanmaz; eşit öğeler kaybolur

B) Dizi yine sıralanır ama algoritma kararlı olmaz

C) Dizi sıralanır ama süre $O(n^2)$'ye çıkar

D) Hiçbir şey değişmez; eşitlikte iki seçim aynıdır

Soru 15: Ters Çift Sayımı

Bir dizideki ters çift sayısını (i < j iken dizi[i] > dizi[j]) O(n log n) sürede bulan bir yöntem önerin. [3, 1, 2, 5, 4] için sonucu bulun.

▶ **Cevabı göster**

Soru 16: Lomuto Bölümleme

`parcala(dizi, 0, 7)` [7, 2, 1, 6, 8, 5, 3, 4] dizisi için çalışıyor. Her j'de i ve dizi nasıl değişir, metot ne döndürür?

```
1  int parcala(int[] dizi, int alt, int ust) {
2    int pivot = dizi[ust];
3    int i = alt - 1;
4    for (int j = alt; j < ust; j++) {
5      if (dizi[j] <= pivot) {
6        i++;
7        takas(dizi, i, j);
8      }
9    }
10   takas(dizi, i + 1, ust);
11   return i + 1;
12 }
```

Adım 1 / 10

DEĞİŞKENLER

—

ÇIKTI

[redacted]

Soru 16: Lomuto Bölümleme

`parcala(dizi, 0, 7)` [7, 2, 1, 6, 8, 5, 3, 4] dizisi için çalışıyor. Her j'de i ve dizi nasıl değişir, metot ne döndürür?

```
1  int parcala(int[] dizi, int alt, int ust) {
▶    int pivot = dizi[ust];
▶    int i = alt - 1;
4    for (int j = alt; j < ust; j++) {
5      if (dizi[j] <= pivot) {
6        i++;
7        takas(dizi, i, j);
8      }
9    }
10   takas(dizi, i + 1, ust);
11   return i + 1;
12 }
```



pivot = dizi[7] = 4 (koyu); i = −1: ≤ pivot bölgesi (mavi) boş. Kırmızı: > pivot, beyaz: henüz bakılmadı.

Adım 2 / 10

DEĞİŞKENLER

`pivot = 4` `i = -1`

`dizi = [7, 2, 1, 6, 8, 5, 3, 4]`

ÇIKTI

Soru 16: Lomuto Bölümleme

`parcala(dizi, 0, 7)` [7, 2, 1, 6, 8, 5, 3, 4] dizisi için çalışıyor. Her j'de i ve dizi nasıl değişir, metot ne döndürür?

```
1  int parcala(int[] dizi, int alt, int ust) {
2    int pivot = dizi[ust];
3    int i = alt - 1;
▶    for (int j = alt; j < ust; j++) {
▶      if (dizi[j] <= pivot) {
6        i++;
7        takas(dizi, i, j);
8      }
9    }
10   takas(dizi, i + 1, ust);
11   return i + 1;
12 }
```



j = 0: 7 > 4 → yalnızca j ilerler; kırmızı bölge büyür.

Adım 3 / 10

DEĞİŞKENLER

`pivot = 4` `i = -1` `j = 0`

`dizi = [7, 2, 1, 6, 8, 5, 3, 4]`

ÇIKTI

Soru 16: Lomuto Bölümleme

`parcala(dizi, 0, 7)` [7, 2, 1, 6, 8, 5, 3, 4] dizisi için çalışıyor. Her j'de i ve dizi nasıl değişir, metot ne döndürür?

```
1  int parcala(int[] dizi, int alt, int ust) {
2    int pivot = dizi[ust];
3    int i = alt - 1;
▶    for (int j = alt; j < ust; j++) {
▶      if (dizi[j] <= pivot) {
▶        i++;
▶        takas(dizi, i, j);
8      }
9    }
10   takas(dizi, i + 1, ust);
11   return i + 1;
12 }
```

DEĞİŞKENLER

`pivot = 4` `i = 0` `j = 1`

`dizi = [2, 7, 1, 6, 8, 5, 3, 4]`

ÇIKTI



j = 1: 2 ≤ 4 → i = 0; takas(dizi, 0, 1) → [2, 7, 1, 6, 8, 5, 3, 4].

Adım 4 / 10

Soru 16: Lomuto Bölümleme

`parcala(dizi, 0, 7)` [7, 2, 1, 6, 8, 5, 3, 4] dizisi için çalışıyor. Her j'de i ve dizi nasıl değişir, metot ne döndürür?

```
1  int parcala(int[] dizi, int alt, int ust) {
2    int pivot = dizi[ust];
3    int i = alt - 1;
▶    for (int j = alt; j < ust; j++) {
▶      if (dizi[j] <= pivot) {
▶        i++;
▶        takas(dizi, i, j);
8      }
9    }
10   takas(dizi, i + 1, ust);
11   return i + 1;
12 }
```

DEĞİŞKENLER

`pivot = 4` `i = 1` `j = 2`

`dizi = [2, 1, 7, 6, 8, 5, 3, 4]`

ÇIKTI



j = 2: 1 ≤ 4 → i = 1; takas(dizi, 1, 2) → [2, 1, 7, 6, 8, 5, 3, 4].

Adım 5 / 10

Soru 16: Lomuto Bölümleme

`parcala(dizi, 0, 7)` [7, 2, 1, 6, 8, 5, 3, 4] dizisi için çalışıyor. Her j'de i ve dizi nasıl değişir, metot ne döndürür?

```
1  int parcala(int[] dizi, int alt, int ust) {
2    int pivot = dizi[ust];
3    int i = alt - 1;
▶    for (int j = alt; j < ust; j++) {
▶      if (dizi[j] <= pivot) {
6        i++;
7        takas(dizi, i, j);
8      }
9    }
10   takas(dizi, i + 1, ust);
11   return i + 1;
12 }
```



j = 3: 6 > 4 → yalnızca j ilerler; kırmızı bölge büyür.

Adım 6 / 10

DEĞİŞKENLER

`pivot = 4` `i = 1` `j = 3`

`dizi = [2, 1, 7, 6, 8, 5, 3, 4]`

ÇIKTI

Soru 16: Lomuto Bölümleme

`parcala(dizi, 0, 7)` [7, 2, 1, 6, 8, 5, 3, 4] dizisi için çalışıyor. Her j'de i ve dizi nasıl değişir, metot ne döndürür?

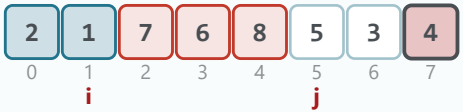
```
1  int parcala(int[] dizi, int alt, int ust) {
2    int pivot = dizi[ust];
3    int i = alt - 1;
▶    for (int j = alt; j < ust; j++) {
▶      if (dizi[j] <= pivot) {
6        i++;
7        takas(dizi, i, j);
8      }
9    }
10   takas(dizi, i + 1, ust);
11   return i + 1;
12 }
```

DEĞİŞKENLER

`pivot = 4` `i = 1` `j = 4`

`dizi = [2, 1, 7, 6, 8, 5, 3, 4]`

ÇIKTI



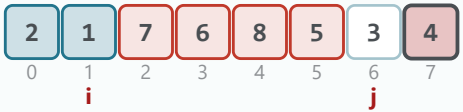
j = 4: 8 > 4 → yalnızca j ilerler; kırmızı bölge büyür.

Adım 7 / 10

Soru 16: Lomuto Bölümleme

`parcala(dizi, 0, 7)` [7, 2, 1, 6, 8, 5, 3, 4] dizisi için çalışıyor. Her j'de i ve dizi nasıl değişir, metot ne döndürür?

```
1  int parcala(int[] dizi, int alt, int ust) {
2    int pivot = dizi[ust];
3    int i = alt - 1;
▶    for (int j = alt; j < ust; j++) {
▶      if (dizi[j] <= pivot) {
6        i++;
7        takas(dizi, i, j);
8      }
9    }
10   takas(dizi, i + 1, ust);
11   return i + 1;
12 }
```



j = 5: 5 > 4 → yalnızca j ilerler; kırmızı bölge büyür.

Adım 8 / 10

DEĞİŞKENLER

`pivot = 4` `i = 1` `j = 5`

`dizi = [2, 1, 7, 6, 8, 5, 3, 4]`

ÇIKTI

Soru 16: Lomuto Bölümleme

`parcala(dizi, 0, 7)` [7, 2, 1, 6, 8, 5, 3, 4] dizisi için çalışıyor. Her j'de i ve dizi nasıl değişir, metot ne döndürür?

```
1  int parcala(int[] dizi, int alt, int ust) {
2    int pivot = dizi[ust];
3    int i = alt - 1;
▶    for (int j = alt; j < ust; j++) {
▶      if (dizi[j] <= pivot) {
▶        i++;
▶        takas(dizi, i, j);
8      }
9    }
10   takas(dizi, i + 1, ust);
11   return i + 1;
12 }
```

DEĞİŞKENLER

`pivot = 4` `i = 2` `j = 6`

`dizi = [2, 1, 3, 6, 8, 5, 7, 4]`

ÇIKTI



j = 6: 3 ≤ 4 → i = 2; takas(dizi, 2, 6) → [2, 1, 3, 6, 8, 5, 7, 4].

Soru 16: Lomuto Bölümleme

`parcala(dizi, 0, 7)` [7, 2, 1, 6, 8, 5, 3, 4] dizisi için çalışıyor. Her j'de i ve dizi nasıl değişir, metot ne döndürür?

```
1  int parcala(int[] dizi, int alt, int ust) {
2    int pivot = dizi[ust];
3    int i = alt - 1;
4    for (int j = alt; j < ust; j++) {
5      if (dizi[j] <= pivot) {
6        i++;
7        takas(dizi, i, j);
8      }
9    }
▶    takas(dizi, i + 1, ust);
▶    return i + 1;
12 }
```

DEĞİŞKENLER

`pivot = 4` `i = 2`

`dizi = [2, 1, 3, 4, 8, 5, 7, 6]`

ÇIKTI

```
3
```



Döngü bitti: takas(dizi, i + 1 = 3, 7) → [2, 1, 3, 4, 8, 5, 7, 6]. **return 3**: pivot kesin yerinde; solu ≤ 4, sağı > 4. 7 karşılaştırma → Θ(n).

Adım 10 / 10

ÇOKTAN SEÇMELİ · HIZLI SIRALAMA

Soru 17: Sıralı Girdi ve Son Öğe Pivot

Son öğeyi pivot alan hızlı sıralama, 1000 öğeli **zaten sıralı** bir diziyle çağrılırsa kaç karşılaştırma yapar ve özyineleme derinliği ne olur?

A) 9.966 karşılaştırma, derinlik 10

B) 499.500 karşılaştırma, derinlik 1000

C) 999 karşılaştırma, derinlik 1

D) 1.000.000 karşılaştırma, derinlik 500

Soru 18: Max-Heap Kurma

`yiginSirala` metodunun ilk aşaması (heap kurma) [2, 7, 4, 1, 8, 6, 5] dizisine uygulanıyor. Hangi takaslar yapılır ve oluşan max-heap nedir?

```
1 void yiginSirala(int[] dizi) {
2   int n = dizi.length;
3   for (int i = n / 2 - 1; i >= 0; i--)
4     yiginlastir(dizi, n, i);
5   // 2. aşama: kökü sona al, süz ...
6 }
```

Adım 1 / 7

DEĞİŞKENLER

—

ÇIKTI

Soru 18: Max-Heap Kurma

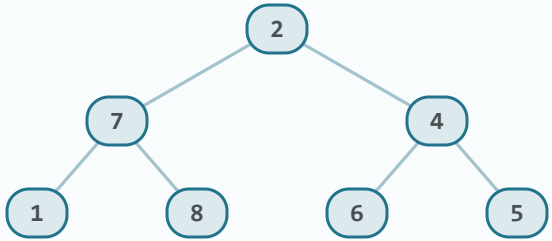
`yiginSirala` metodunun ilk aşaması (heap kurma) [2, 7, 4, 1, 8, 6, 5] dizisine uygulanıyor. Hangi takaslar yapılır ve oluşan max-heap nedir?

```
1 void yiginSirala(int[] dizi) {
▶   int n = dizi.length;
▶   for (int i = n / 2 - 1; i >= 0; i--)
4     yiginlastir(dizi, n, i);
5   // 2. aşama: kökü sona al, süz ...
6 }
```

DEĞİŞKENLER

`n = 7` `dizi = [2, 7, 4, 1, 8, 6, 5]`

ÇIKTI



Dizi tam ikili ağaç olarak düşünülür: dizi[i]'nin çocukları 2i + 1 ve 2i + 2. Yapraklar (i ≥ 3) zaten tek öğeli heap'tir; i = n/2 − 1 = 2'den köke doğru süzülür.

Soru 18: Max-Heap Kurma

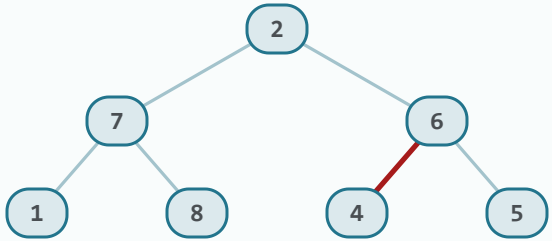
`yiginSirala` metodunun ilk aşaması (heap kurma) [2, 7, 4, 1, 8, 6, 5] dizisine uygulanıyor. Hangi takaslar yapılır ve oluşan max-heap nedir?

```
1 void yiginSirala(int[] dizi) {
2   int n = dizi.length;
▶   for (int i = n / 2 - 1; i >= 0; i--)
▶     yiginlastir(dizi, n, i);
5   // 2. aşama: kökü sona al, süz ...
6 }
```

DEĞİŞKENLER

`i = 2` `dizi = [2, 7, 6, 1, 8, 4, 5]`

ÇIKTI



yiginlastir(i = 2): 4 en büyük çocuğundan küçük → 4 ↔ 6 (indis 2 ↔ 5). Dizi [2, 7, 6, 1, 8, 4, 5].

Soru 18: Max-Heap Kurma

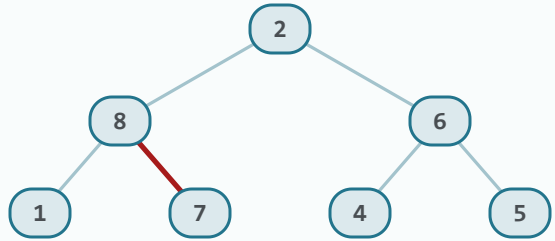
`yiginSirala` metodunun ilk aşaması (heap kurma) [2, 7, 4, 1, 8, 6, 5] dizisine uygulanıyor. Hangi takaslar yapılır ve oluşan max-heap nedir?

```
1 void yiginSirala(int[] dizi) {
2   int n = dizi.length;
▶   for (int i = n / 2 - 1; i >= 0; i--)
▶     yiginlastir(dizi, n, i);
5   // 2. aşama: kökü sona al, süz ...
6 }
```

DEĞİŞKENLER

`i = 1` `dizi = [2, 8, 6, 1, 7, 4, 5]`

ÇIKTI



yiginlastir(i = 1): 7 en büyük çocuğundan küçük → 7 ↔ 8 (indis 1 ↔ 4). Dizi [2, 8, 6, 1, 7, 4, 5].

Soru 18: Max-Heap Kurma

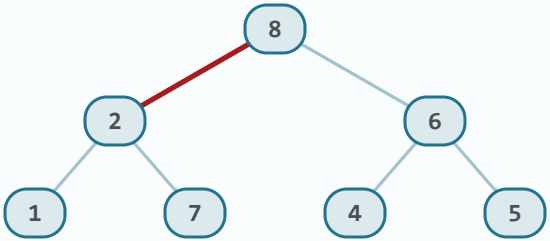
`yiginSirala` metodunun ilk aşaması (heap kurma) [2, 7, 4, 1, 8, 6, 5] dizisine uygulanıyor. Hangi takaslar yapılır ve oluşan max-heap nedir?

```
1 void yiginSirala(int[] dizi) {
2   int n = dizi.length;
▶   for (int i = n / 2 - 1; i >= 0; i--)
▶     yiginlastir(dizi, n, i);
5   // 2. aşama: kökü sona al, süz ...
6 }
```

DEĞİŞKENLER

`i = 0` `dizi = [8, 2, 6, 1, 7, 4, 5]`

ÇIKTI



yiginlastir(i = 0): 2 en büyük çocuğundan küçük → 2 ↔ 8 (indis 0 ↔ 1). Dizi [8, 2, 6, 1, 7, 4, 5].

Soru 18: Max-Heap Kurma

`yiginSirala` metodunun ilk aşaması (heap kurma) [2, 7, 4, 1, 8, 6, 5] dizisine uygulanıyor. Hangi takaslar yapılır ve oluşan max-heap nedir?

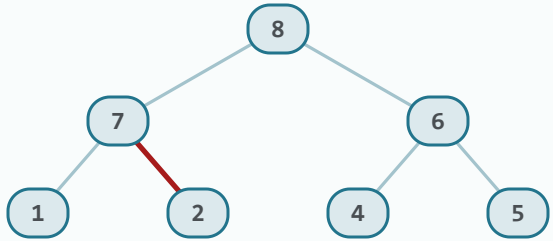
```
1 void yiginSirala(int[] dizi) {
2   int n = dizi.length;
3   for (int i = n / 2 - 1; i >= 0; i--)
▶     yiginlastir(dizi, n, i);
5   // 2. aşama: kökü sona al, süz ...
6 }
```

DEĞİŞKENLER

`i = 0` `dizi = [8, 7, 6, 1, 2, 4, 5]`

ÇIKTI

```
[8, 7, 6, 1, 2, 4, 5]
```



Süzme sürer: 2 en büyük çocuğundan küçük → 2 ↔ 7 (indis 1 ↔ 4). Dizi [8, 7, 6, 1, 2, 4, 5]. Max-heap hazır: kök 8 en büyük öğe.

Soru 18: Max-Heap Kurma

`yiginSirala` metodunun ilk aşaması (heap kurma) [2, 7, 4, 1, 8, 6, 5] dizisine uygulanıyor. Hangi takaslar yapılır ve oluşan max-heap nedir?

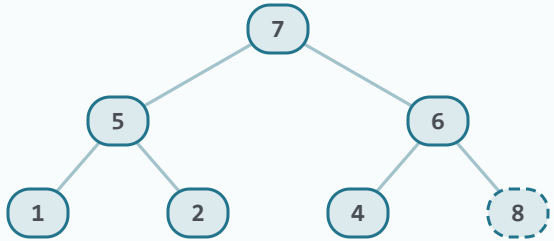
```
1 void yiginSirala(int[] dizi) {
2   int n = dizi.length;
3   for (int i = n / 2 - 1; i >= 0; i--)
4     yiginlastir(dizi, n, i);
▶   // 2. aşama: kökü sona al, süz ...
6 }
```

DEĞİŞKENLER

`dizi = [7, 5, 6, 1, 2, 4, 8]`

ÇIKTI

```
[8, 7, 6, 1, 2, 4, 5]
```



2. aşamanın ilk turu: kök 8 ↔ dizi[6] = 5 → 8 sıralı bölgeye geçer (kesikli); heap boyutu 6. yiginlastir(0): 5 ↔ 7 → [7, 5, 6, 1, 2, 4, 8].

ÇOKTAN SEÇMELİ · HEAP SORT

Soru 19: Heap Sort Özellikleri

Heap sort için aşağıdakilerden hangisi doğrudur?

A) Aşağıdan yukarı heap kurma O(n log n), sıralama aşaması O(n) sürer.

B) Kök–son takası eşit anahtarların sırasını koruduğu için kararlıdır.

C) Sıralı girdide Θ(n) sürede biter.

D) En kötü durumda bile O(n log n) sürer ve O(1) ek bellek kullanır.

ÇOKTAN SEÇMELİ · DOĞRUSAL ZAMANLI SIRALAMA

Soru 20: Karar Ağacı Sınırı

3 farklı öğeyi yalnızca karşılaştırmayla sıralayan **her** algoritma en kötü durumda en az kaç karşılaştırma yapmak zorundadır?

A) 2

B) 3

C) 6

D) 9

Soru 21: Saymalı Sıralama İzleme

`saymaliSirala` [3, 1, 4, 1, 3, 0] dizisi için çalışıyor. Sayaç dizisi ve çıktı dizisi nasıl dolar?

```
1  int[] saymaliSirala(int[] dizi) { // öğeler ≥ 0
2    if (dizi.length == 0) return dizi;
3    int maks = Arrays.stream(dizi).max().getAsInt();
4    int[] sayac = new int[maks + 1];
5    for (int x : dizi) sayac[x]++;
6    for (int i = 1; i <= maks; i++)
7      sayac[i] += sayac[i - 1];
8    int[] cikti = new int[dizi.length];
9    for (int i = dizi.length - 1; i >= 0; i--) {
10     cikti[sayac[dizi[i]] - 1] = dizi[i];
11     sayac[dizi[i]]--;
12   }
13   return cikti;
14 }
```

Adım 1 / 9

DEĞİŞKENLER

—

ÇIKTI

Soru 21: Saymalı Sıralama İzleme

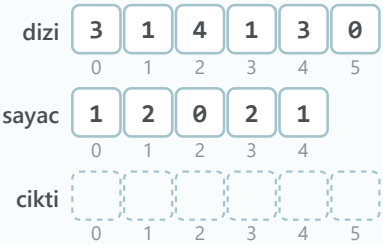
`saymaliSirala` [3, 1, 4, 1, 3, 0] dizisi için çalışıyor. Sayaç dizisi ve çıktı dizisi nasıl dolar?

```
1  int[] saymaliSirala(int[] dizi) { // öğeler ≥ 0
2    if (dizi.length == 0) return dizi;
3    int maks = Arrays.stream(dizi).max().getAsInt();
4    int[] sayac = new int[maks + 1];
▶    for (int x : dizi) sayac[x]++;
6    for (int i = 1; i <= maks; i++)
7      sayac[i] += sayac[i - 1];
8    int[] cikti = new int[dizi.length];
9    for (int i = dizi.length - 1; i >= 0; i--) {
10     cikti[sayac[dizi[i]] - 1] = dizi[i];
11     sayac[dizi[i]]--;
12   }
13   return cikti;
14 }
```

DEĞİŞKENLER

`sayac = [1, 2, 0, 2, 1]`

ÇIKTI



Sayma döngüsü bitti: sayac[x] = x anahtarının adedi → [1, 2, 0, 2, 1]. maks = 4 olduğu için sayaç 5 hücrelidir.

Soru 21: Saymalı Sıralama İzleme

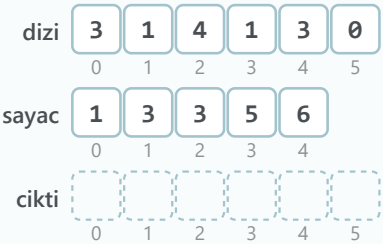
`saymaliSirala` [3, 1, 4, 1, 3, 0] dizisi için çalışıyor. Sayaç dizisi ve çıktı dizisi nasıl dolar?

```
1  int[] saymaliSirala(int[] dizi) { // öğeler ≥ 0
2    if (dizi.length == 0) return dizi;
3    int maks = Arrays.stream(dizi).max().getAsInt();
4    int[] sayac = new int[maks + 1];
5    for (int x : dizi) sayac[x]++;
▶    for (int i = 1; i <= maks; i++)
▶      sayac[i] += sayac[i - 1];
8    int[] cikti = new int[dizi.length];
9    for (int i = dizi.length - 1; i >= 0; i--) {
10     cikti[sayac[dizi[i]] - 1] = dizi[i];
11     sayac[dizi[i]]--;
12   }
13   return cikti;
14 }
```

DEĞİŞKENLER

`sayac = [1, 3, 3, 5, 6]`

ÇIKTI

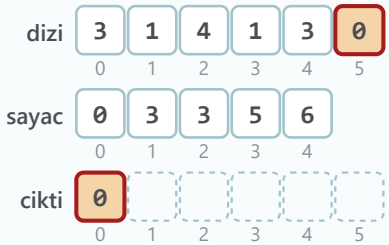


Önek toplamı: sayac[x] = anahtarı ≤ x olan öğe sayısı → [1, 3, 3, 5, 6]. sayac[x] − 1, x'in çıktıdaki *son* konumudur.

Soru 21: Saymalı Sıralama İzleme

`saymaliSirala` [3, 1, 4, 1, 3, 0] dizisi için çalışıyor. Sayaç dizisi ve çıktı dizisi nasıl dolar?

```
1  int[] saymaliSirala(int[] dizi) { // öğeler ≥ 0
2    if (dizi.length == 0) return dizi;
3    int maks = Arrays.stream(dizi).max().getAsInt();
4    int[] sayac = new int[maks + 1];
5    for (int x : dizi) sayac[x]++;
6    for (int i = 1; i <= maks; i++)
7      sayac[i] += sayac[i - 1];
8    int[] cikti = new int[dizi.length];
▶    for (int i = dizi.length - 1; i >= 0; i--) {
▶      cikti[sayac[dizi[i]] - 1] = dizi[i];
▶      sayac[dizi[i]]--;
12   }
13   return cikti;
14 }
```



i = 5: dizi[5] = 0 → cikti[sayac[0] − 1] = cikti[0] = 0; sayac[0] = 0.

Adım 4 / 9

DEĞİŞKENLER

`i = 5` `dizi[i] = 0` `sayac = [0, 3, 3, 5, 6]`

ÇIKTI

Soru 21: Saymalı Sıralama İzleme

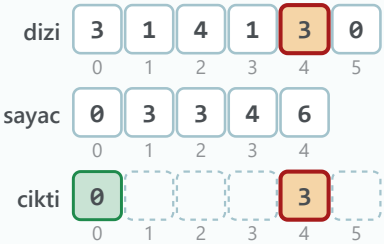
`saymaliSirala` [3, 1, 4, 1, 3, 0] dizisi için çalışıyor. Sayaç dizisi ve çıktı dizisi nasıl dolar?

```
1  int[] saymaliSirala(int[] dizi) { // öğeler ≥ 0
2    if (dizi.length == 0) return dizi;
3    int maks = Arrays.stream(dizi).max().getAsInt();
4    int[] sayac = new int[maks + 1];
5    for (int x : dizi) sayac[x]++;
6    for (int i = 1; i <= maks; i++)
7      sayac[i] += sayac[i - 1];
8    int[] cikti = new int[dizi.length];
▶    for (int i = dizi.length - 1; i >= 0; i--) {
▶      cikti[sayac[dizi[i]] - 1] = dizi[i];
▶      sayac[dizi[i]]--;
12   }
13   return cikti;
14 }
```

DEĞİŞKENLER

`i = 4` `dizi[i] = 3` `sayac = [0, 3, 3, 4, 6]`

ÇIKTI



i = 4: dizi[4] = 3 → cikti[sayac[3] − 1] = cikti[4] = 3; sayac[3] = 4.

Adım 5 / 9

Soru 21: Saymalı Sıralama İzleme

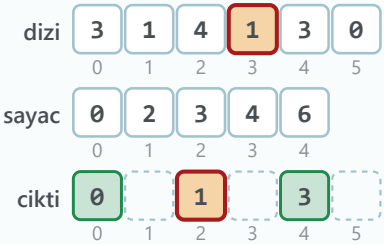
`saymaliSirala` [3, 1, 4, 1, 3, 0] dizisi için çalışıyor. Sayaç dizisi ve çıktı dizisi nasıl dolar?

```
1  int[] saymaliSirala(int[] dizi) { // öğeler ≥ 0
2    if (dizi.length == 0) return dizi;
3    int maks = Arrays.stream(dizi).max().getAsInt();
4    int[] sayac = new int[maks + 1];
5    for (int x : dizi) sayac[x]++;
6    for (int i = 1; i <= maks; i++)
7      sayac[i] += sayac[i - 1];
8    int[] cikti = new int[dizi.length];
▶    for (int i = dizi.length - 1; i >= 0; i--) {
▶      cikti[sayac[dizi[i]] - 1] = dizi[i];
▶      sayac[dizi[i]]--;
12   }
13   return cikti;
14 }
```

DEĞİŞKENLER

`i = 3` `dizi[i] = 1` `sayac = [0, 2, 3, 4, 6]`

ÇIKTI



i = 3: dizi[3] = 1 → cikti[sayac[1] − 1] = cikti[2] = 1; sayac[1] = 2.

Adım 6 / 9

Soru 21: Saymalı Sıralama İzleme

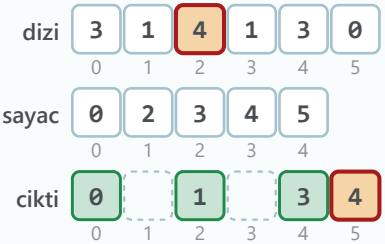
`saymaliSirala` [3, 1, 4, 1, 3, 0] dizisi için çalışıyor. Sayaç dizisi ve çıktı dizisi nasıl dolar?

```
1  int[] saymaliSirala(int[] dizi) { // öğeler ≥ 0
2    if (dizi.length == 0) return dizi;
3    int maks = Arrays.stream(dizi).max().getAsInt();
4    int[] sayac = new int[maks + 1];
5    for (int x : dizi) sayac[x]++;
6    for (int i = 1; i <= maks; i++)
7      sayac[i] += sayac[i - 1];
8    int[] cikti = new int[dizi.length];
▶    for (int i = dizi.length - 1; i >= 0; i--) {
▶      cikti[sayac[dizi[i]] - 1] = dizi[i];
▶      sayac[dizi[i]]--;
12   }
13   return cikti;
14 }
```

DEĞİŞKENLER

`i = 2` `dizi[i] = 4` `sayac = [0, 2, 3, 4, 5]`

ÇIKTI



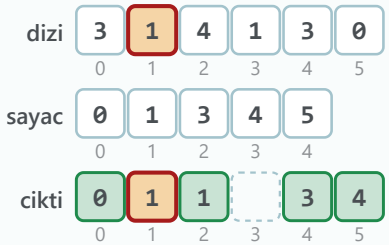
i = 2: dizi[2] = 4 → cikti[sayac[4] − 1] = cikti[5] = 4; sayac[4] = 5.

Adım 7 / 9

Soru 21: Saymalı Sıralama İzleme

`saymaliSirala` [3, 1, 4, 1, 3, 0] dizisi için çalışıyor. Sayaç dizisi ve çıktı dizisi nasıl dolar?

```
1  int[] saymaliSirala(int[] dizi) { // öğeler ≥ 0
2    if (dizi.length == 0) return dizi;
3    int maks = Arrays.stream(dizi).max().getAsInt();
4    int[] sayac = new int[maks + 1];
5    for (int x : dizi) sayac[x]++;
6    for (int i = 1; i <= maks; i++)
7      sayac[i] += sayac[i - 1];
8    int[] cikti = new int[dizi.length];
▶    for (int i = dizi.length - 1; i >= 0; i--) {
▶      cikti[sayac[dizi[i]] - 1] = dizi[i];
▶      sayac[dizi[i]]--;
12   }
13   return cikti;
14 }
```



i = 1: dizi[1] = 1 → cikti[sayac[1] − 1] = cikti[1] = 1; sayac[1] = 1.

Adım 8 / 9

DEĞİŞKENLER

`i = 1` `dizi[i] = 1` `sayac = [0, 1, 3, 4, 5]`

ÇIKTI

Soru 21: Saymalı Sıralama İzleme

`saymaliSirala` [3, 1, 4, 1, 3, 0] dizisi için çalışıyor. Sayaç dizisi ve çıktı dizisi nasıl dolar?

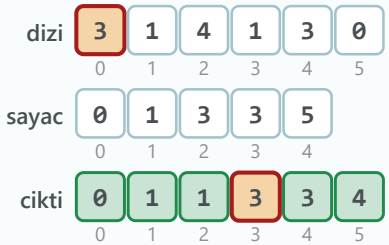
```
1  int[] saymaliSirala(int[] dizi) { // öğeler ≥ 0
2    if (dizi.length == 0) return dizi;
3    int maks = Arrays.stream(dizi).max().getAsInt();
4    int[] sayac = new int[maks + 1];
5    for (int x : dizi) sayac[x]++;
6    for (int i = 1; i <= maks; i++)
7      sayac[i] += sayac[i - 1];
8    int[] cikti = new int[dizi.length];
▶    for (int i = dizi.length - 1; i >= 0; i--) {
▶      cikti[sayac[dizi[i]] - 1] = dizi[i];
▶      sayac[dizi[i]]--;
12   }
13   return cikti;
14 }
```

DEĞİŞKENLER

`i = 0` `dizi[i] = 3` `sayac = [0, 1, 3, 3, 5]`

ÇIKTI

```
[0, 1, 1, 3, 3, 4]
```



i = 0: dizi[0] = 3 → cikti[sayac[3] − 1] = cikti[3] = 3; sayac[3] = 3. Çıktı: [0, 1, 1, 3, 3, 4]. Karşılaştırma yok: Θ(n + k).

ÇIKTI SORUSU · DOĞRUSAL ZAMANLI SIRALAMA

Soru 22: Basamak Sıralaması Geçişleri

[329, 457, 657, 839, 436, 720, 355] dizisi LSD basamak sıralamasıyla (her geçişte kararlı saymalı sıralama) sıralanıyor. Her geçişten sonraki dizi nedir?

▶ **Cevabı göster**

ÇOKTAN SEÇMELİ · KARŞILAŞTIRMA

Soru 23: Hangi Algoritma?

Disk üzerinde, belleğe sığmayan büyük bir dosyayı sıralamak ya da bir bağlı listeyi sıralamak için en uygun algoritma hangisidir?

A) Birleştirme sıralaması

B) Heap sort (yerinde)

C) Hızlı sıralama (rastgele pivot)

D) Seçimli sıralama (az takas)

Son

Bölüm 3: Sıralama Algoritmaları: Çalışma Soruları

Konu anlatımı: Bölüm 3: Sıralama Algoritmaları · Terim sözlüğü