Bölüm 4: Geri İzleme Algoritmaları

Bölüm 4: Geri İzleme Algoritmaları · Çalışma Soruları

Seç · keşfet · geri al | Durum uzayı ve budama | Alt küme ve permütasyon | n-vezir

En az çakışma | Sudoku | Çizge boyama ve Hamilton yolu | Kelime arama

Sercan KÜLCÜ · Giresun Üniversitesi · Bilgisayar Mühendisliği

© Sercan Külcü. Tüm hakları saklıdır. · Sunumun etkileşimli HTML sürümü:
https://sercankulcu.github.io/files/advanced_algorithms/slides/Bolum_04_Geri_Izleme_Algoritmalari_Calisma_Sorulari.html

ÇOKTAN SEÇMELİ · TEMEL FİKİR

Soru 1: Kaba Kuvvetten Farkı

Geri izlemeyi (backtracking) kaba kuvvetten ayıran temel özellik hangisidir?

A) Kısıtı bozan bir kısmi çözümün devamını hiç üretmez

B) Tüm tam çözümleri üretir ve sonra hepsini test eder

C) Her adımda yerel en iyi seçimi yapar ve geri dönmez

D) Alt problemlerin sonuçlarını bir tabloda saklar

ÇOKTAN SEÇMELİ · TEMEL FİKİR

Soru 2: Geri Almayı Unutmak

Genel şablondaki `cikar(kismi, a)` (geri al) satırı silinirse ne olur?

A) Özyineleme hiç bitmez ve yığın taşması oluşur

B) Önceki seçim kalır ve sonraki adayları denerken durum bozulur

C) Yalnızca ilk çözüm bulunur, diğer çözümler kaybolur

D) Hiçbir şey değişmez; özyineleme durumu kendiliğinden geri yükler

ÇIKTI SORUSU · ALT KÜME VE PERMÜTASYON

Soru 3: Permütasyon Üretimi

Aşağıdaki kod `uret(new ArrayList<>(), 3)` ile çağrılıyor (`kul` 4 hücreli boolean dizi). Ne yazar? Durum uzayı ağacında kök dahil kaç düğüm vardır?

```
static boolean[] kul = new boolean[4];
static void uret(List<Integer> yol, int n) {
  if (yol.size() == n) {
    for (int x : yol) System.out.print(x);
    System.out.print(" ");
    return;
  }
  for (int v = 1; v <= n; v++) {
    if (!kul[v]) {
      kul[v] = true; yol.add(v);        // seç
      uret(yol, n);                     // keşfet
      yol.remove(yol.size() - 1); // geri al
      kul[v] = false;
    }
  }
}
```

▶ **Cevabı göster**

Soru 4: Alt Küme Toplamı İzleme

a = {3, 5, 6, 7} ve HEDEF = 15 için `ara(0, 0)` durum uzayı ağacını nasıl gezer? Bulunan ilk alt küme hangisidir?

```
1 boolean ara(int i, int toplam) {
2   if (toplam == HEDEF) return true;
3   if (i == n || toplam > HEDEF) return false;
4   secildi[i] = true;              // al
5   if (ara(i + 1, toplam + a[i])) return true;
6   secildi[i] = false;             // alma
7   return ara(i + 1, toplam);
8 }
```



a = {3, 5, 6, 7}, HEDEF = 15. Her düzeyde bir öğe için iki dal: önce "al" (+a[i]), sonra "alma" (−). Kökte toplam 0.

Adım 1 / 8

DEĞİŞKENLER

`i = 0` `toplam = 0` `seçilen = []`

ÇIKTI

Soru 4: Alt Küme Toplamı İzleme

a = {3, 5, 6, 7} ve HEDEF = 15 için `ara(0, 0)` durum uzayı ağacını nasıl gezer? Bulunan ilk alt küme hangisidir?

```
1 boolean ara(int i, int toplam) {
2   if (toplam == HEDEF) return true;
3   if (i == n || toplam > HEDEF) return false;
▶   secildi[i] = true;              // al
▶   if (ara(i + 1, toplam + a[i])) return true;
6   secildi[i] = false;             // alma
7   return ara(i + 1, toplam);
8 }
```



ara(1, 3): a[0] = 3 alındı → toplam 3. Önce a[1] = 5 alınmayı deneyecek.

Adım 2 / 8

DEĞİŞKENLER

`i = 1` `toplam = 3` `seçilen = [3]`

ÇIKTI

Soru 4: Alt Küme Toplamı İzleme

a = {3, 5, 6, 7} ve HEDEF = 15 için `ara(0, 0)` durum uzayı ağacını nasıl gezer? Bulunan ilk alt küme hangisidir?

```
1 boolean ara(int i, int toplam) {
2   if (toplam == HEDEF) return true;
3   if (i == n || toplam > HEDEF) return false;
▶   secildi[i] = true;              // al
▶   if (ara(i + 1, toplam + a[i])) return true;
6   secildi[i] = false;             // alma
7   return ara(i + 1, toplam);
8 }
```



ara(2, 8): a[1] = 5 alındı → toplam 8. Önce a[2] = 6 alınmayı deneyecek.

Adım 3 / 8

DEĞİŞKENLER

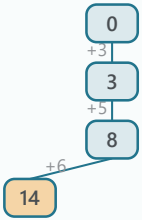
`i = 2` `toplam = 8` `seçilen = [3, 5]`

ÇIKTI

Soru 4: Alt Küme Toplamı İzleme

a = {3, 5, 6, 7} ve HEDEF = 15 için `ara(0, 0)` durum uzayı ağacını nasıl gezer? Bulunan ilk alt küme hangisidir?

```
1 boolean ara(int i, int toplam) {
2   if (toplam == HEDEF) return true;
3   if (i == n || toplam > HEDEF) return false;
▶   secildi[i] = true;              // al
▶   if (ara(i + 1, toplam + a[i])) return true;
6   secildi[i] = false;             // alma
7   return ara(i + 1, toplam);
8 }
```



ara(3, 14): a[2] = 6 alındı → toplam 14. Önce a[3] = 7 alınmayı deneyecek.

Adım 4 / 8

DEĞİŞKENLER

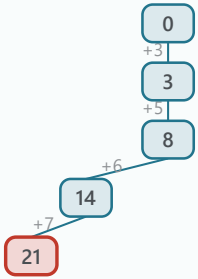
`i = 3` `toplam = 14` `seçilen = [3, 5, 6]`

ÇIKTI

Soru 4: Alt Küme Toplamı İzleme

a = {3, 5, 6, 7} ve HEDEF = 15 için `ara(0, 0)` durum uzayı ağacını nasıl gezer? Bulunan ilk alt küme hangisidir?

```
1 boolean ara(int i, int toplam) {
2   if (toplam == HEDEF) return true;
▶  if (i == n || toplam > HEDEF) return false;
4   secildi[i] = true;              // al
5   if (ara(i + 1, toplam + a[i])) return true;
6   secildi[i] = false;             // alma
7   return ara(i + 1, toplam);
8 }
```



ara(4, 21): 21 > 15 → **budandı**, false. Bu düğümün altı hiç üretilmez.

Adım 5 / 8

DEĞİŞKENLER

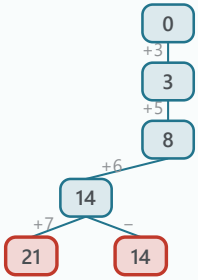
`i = 4` `toplam = 21` `seçilen = [3, 5, 6, 7]`

ÇIKTI

Soru 4: Alt Küme Toplamı İzleme

a = {3, 5, 6, 7} ve HEDEF = 15 için `ara(0, 0)` durum uzayı ağacını nasıl gezer? Bulunan ilk alt küme hangisidir?

```
1 boolean ara(int i, int toplam) {
2   if (toplam == HEDEF) return true;
▶  if (i == n || toplam > HEDEF) return false;
4   secildi[i] = true;              // al
5   if (ara(i + 1, toplam + a[i])) return true;
6   secildi[i] = false;             // alma
7   return ara(i + 1, toplam);
8 }
```



ara(4, 14): öğe kalmadı (i == n) ve toplam 15 değil → false; geri dön.

Adım 6 / 8

DEĞİŞKENLER

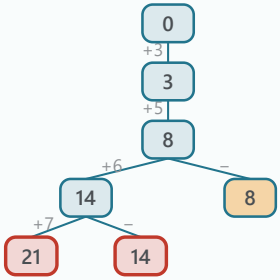
`i = 4` `toplam = 14` `seçilen = [3, 5, 6]`

ÇIKTI

Soru 4: Alt Küme Toplamı İzleme

a = {3, 5, 6, 7} ve HEDEF = 15 için `ara(0, 0)` durum uzayı ağacını nasıl gezer? Bulunan ilk alt küme hangisidir?

```
1 boolean ara(int i, int toplam) {
2   if (toplam == HEDEF) return true;
3   if (i == n || toplam > HEDEF) return false;
4   secildi[i] = true;              // al
5   if (ara(i + 1, toplam + a[i])) return true;
▶   secildi[i] = false;             // alma
▶   return ara(i + 1, toplam);
8 }
```



ara(3, 8): a[2] = 6 alınmadı → toplam 8. Önce a[3] = 7 alınmayı deneyecek.

Adım 7 / 8

DEĞİŞKENLER

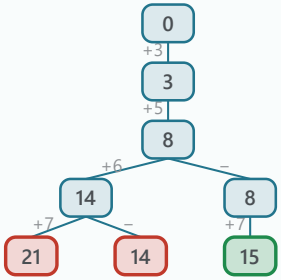
`i = 3` `toplam = 8` `seçilen = [3, 5]`

ÇIKTI

Soru 4: Alt Küme Toplamı İzleme

a = {3, 5, 6, 7} ve HEDEF = 15 için `ara(0, 0)` durum uzayı ağacını nasıl gezer? Bulunan ilk alt küme hangisidir?

```
1 boolean ara(int i, int toplam) {
▶   if (toplam == HEDEF) return true;
3   if (i == n || toplam > HEDEF) return false;
4   secildi[i] = true;              // al
5   if (ara(i + 1, toplam + a[i])) return true;
6   secildi[i] = false;             // alma
7   return ara(i + 1, toplam);
8 }
```



ara(4, 15): toplam == HEDEF → **true**; true tüm çağrılar boyunca yukarı taşınır. Çözüm {3, 5, 7}.

Adım 8 / 8

DEĞİŞKENLER

`i = 4` `toplam = 15` `seçilen = [3, 5, 7]`

ÇIKTI

```
true
```

ÇOKTAN SEÇMELİ · ALT KÜME VE PERMÜTASYON

Soru 5: Permütasyon Ağacı

{1, 2, 3, 4} kümesinin tüm permütasyonlarını geri dönüşle üreten aramanın durum uzayı ağacında kök dahil kaç düğüm vardır?

A) 24

B) 41

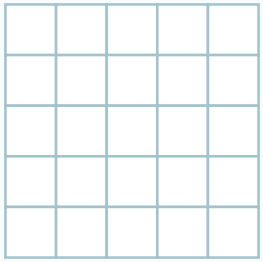
C) 65

D) 256

Soru 6: 5 Vezir İzleme

n = 5 için `coz(t, 0)` ilk çözümü nasıl bulur? Kırmızımsı hücreler, o satırda önceki vezirlerin tehdit ettiği sütunlardır.

```
▶ boolean coz(int[][] t, int satir) {
▶   if (satir == n) return true;
3   for (int sutun = 0; sutun < n; sutun++) {
4     if (guvenliMi(t, satir, sutun)) {
5       t[satir][sutun] = 1;
6       if (coz(t, satir + 1)) return true;
7       t[satir][sutun] = 0;   // geri al
8     }
9   }
10  return false;
11 }
```



n = 5, boş tahta. coz(t, 0): satırlar yukarıdan aşağı doldurulur; her satırda sütun 0, 1, 2, … sırayla denenir.

Adım 1 / 7

DEĞİŞKENLER

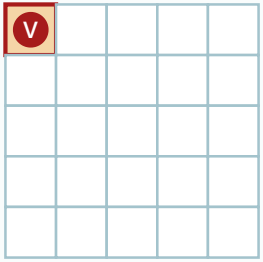
`satir = 0`

ÇIKTI

Soru 6: 5 Vezir İzleme

n = 5 için `coz(t, 0)` ilk çözümü nasıl bulur? Kırmızımsı hücreler, o satırda önceki vezirlerin tehdit ettiği sütunlardır.

```
1  boolean coz(int[][] t, int satir) {
2    if (satir == n) return true;
3    for (int sutun = 0; sutun < n; sutun++) {
▶      if (guvenliMi(t, satir, sutun)) {
▶        t[satir][sutun] = 1;
▶        if (coz(t, satir + 1)) return true;
▶        t[satir][sutun] = 0;   // geri al
8      }
9    }
10   return false;
11 }
```



Satır 0: sütun 0 güvenli → t[0][0] = 1, coz(t, 1).

Adım 2 / 7

DEĞİŞKENLER

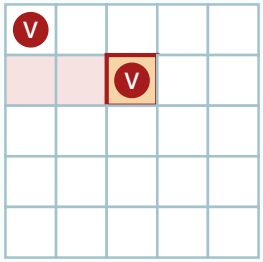
`satir = 0` `sutun = 0`

ÇIKTI

Soru 6: 5 Vezir İzleme

n = 5 için `coz(t, 0)` ilk çözümü nasıl bulur? Kırmızımsı hücreler, o satırda önceki vezirlerin tehdit ettiği sütunlardır.

```
1  boolean coz(int[][] t, int satir) {
2    if (satir == n) return true;
3    for (int sutun = 0; sutun < n; sutun++) {
▶      if (guvenliMi(t, satir, sutun)) {
▶        t[satir][sutun] = 1;
▶        if (coz(t, satir + 1)) return true;
▶        t[satir][sutun] = 0;   // geri al
8      }
9    }
10   return false;
11 }
```



Satır 1: sütun 0, 1 tehdit altında; sütun 2 güvenli → t[1][2] = 1, coz(t, 2).

Adım 3 / 7

DEĞİŞKENLER

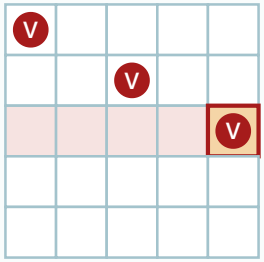
`satir = 1` `sutun = 2`

ÇIKTI

Soru 6: 5 Vezir İzleme

n = 5 için `coz(t, 0)` ilk çözümü nasıl bulur? Kırmızımsı hücreler, o satırda önceki vezirlerin tehdit ettiği sütunlardır.

```
1  boolean coz(int[][] t, int satir) {
2    if (satir == n) return true;
3    for (int sutun = 0; sutun < n; sutun++) {
▶      if (guvenliMi(t, satir, sutun)) {
▶        t[satir][sutun] = 1;
▶        if (coz(t, satir + 1)) return true;
▶        t[satir][sutun] = 0;   // geri al
8      }
9    }
10   return false;
11 }
```



Satır 2: sütun 0, 1, 2, 3 tehdit altında; sütun 4 güvenli → t[2][4] = 1, coz(t, 3).

Adım 4 / 7

DEĞİŞKENLER

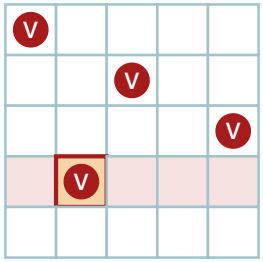
`satir = 2` `sutun = 4`

ÇIKTI

Soru 6: 5 Vezir İzleme

n = 5 için `coz(t, 0)` ilk çözümü nasıl bulur? Kırmızımsı hücreler, o satırda önceki vezirlerin tehdit ettiği sütunlardır.

```
1  boolean coz(int[][] t, int satir) {
2    if (satir == n) return true;
3    for (int sutun = 0; sutun < n; sutun++) {
▶      if (guvenliMi(t, satir, sutun)) {
▶        t[satir][sutun] = 1;
▶        if (coz(t, satir + 1)) return true;
▶        t[satir][sutun] = 0;   // geri al
8      }
9    }
10   return false;
11 }
```



Satır 3: sütun 0 tehdit altında; sütun 1 güvenli → t[3][1] = 1, coz(t, 4).

Adım 5 / 7

DEĞİŞKENLER

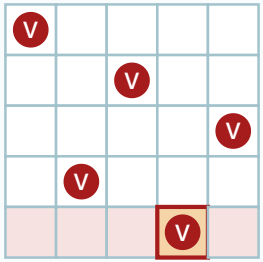
`satir = 3` `sutun = 1`

ÇIKTI

Soru 6: 5 Vezir İzleme

n = 5 için `coz(t, 0)` ilk çözümü nasıl bulur? Kırmızımsı hücreler, o satırda önceki vezirlerin tehdit ettiği sütunlardır.

```
1  boolean coz(int[][] t, int satir) {
2    if (satir == n) return true;
3    for (int sutun = 0; sutun < n; sutun++) {
▶      if (guvenliMi(t, satir, sutun)) {
▶        t[satir][sutun] = 1;
▶        if (coz(t, satir + 1)) return true;
▶        t[satir][sutun] = 0;   // geri al
8      }
9    }
10   return false;
11 }
```



Satır 4: sütun 0, 1, 2 tehdit altında; sütun 3 güvenli → t[4][3] = 1, coz(t, 5).

Adım 6 / 7

DEĞİŞKENLER

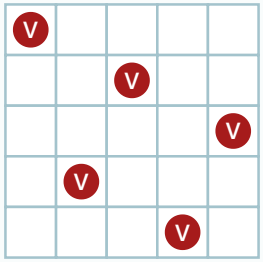
`satir = 4` `sutun = 3`

ÇIKTI

Soru 6: 5 Vezir İzleme

n = 5 için `coz(t, 0)` ilk çözümü nasıl bulur? Kırmızımsı hücreler, o satırda önceki vezirlerin tehdit ettiği sütunlardır.

```
1  boolean coz(int[][] t, int satir) {
▶    if (satir == n) return true;
3    for (int sutun = 0; sutun < n; sutun++) {
4      if (guvenliMi(t, satir, sutun)) {
5        t[satir][sutun] = 1;
6        if (coz(t, satir + 1)) return true;
7        t[satir][sutun] = 0;   // geri al
8      }
9    }
10   return false;
11 }
```



satir == n → true; true yığın boyunca yukarı taşınır. Çözüm [0, 2, 4, 1, 3]: 5 yerleştirme, **hiç geri dönüş yok** (4 vezirde 8 yerleştirme, 4 geri dönüş gerekmişti).

Adım 7 / 7

DEĞİŞKENLER

`satir = 5`

ÇIKTI

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

ÇIKTI SORUSU · N-VEZİR

Soru 7: 6 Vezir: İlk Çözüm

Dersteki `coz` metoduna her yerleştirmede artan bir `yer` sayacı eklendi. n = 6 için program ne yazar (sütunlar satır sırasıyla)?

```
static int yer = 0;
static boolean coz(int[][] t, int satir) {
  if (satir == n) return true;
  for (int s = 0; s < n; s++) {
    if (guvenliMi(t, satir, s)) {
      t[satir][s] = 1; yer++;
      if (coz(t, satir + 1)) return true;
      t[satir][s] = 0;
    }
  }
  return false;
}
// main (n = 6): coz(t, 0); sonra her satırdaki
// vezirin sütunu bir diziye yazılır:
System.out.println(
    Arrays.toString(sut) + " " + yer);
```

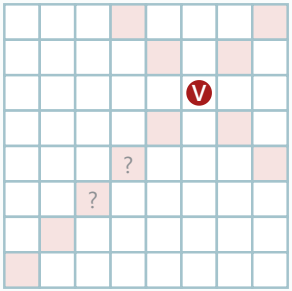
▶ **Cevabı göster**

ÇIKTI SORUSU · N-VEZİR

Soru 8: O(1) Güvenlik Testi

n = 8 için bayrak dizileriyle çalışan sürümde (2, 5) karesine bir vezir konuyor. Hangi `sut`, `k1` ve `k2` hücreleri true olur? Ardından (4, 3) ve (5, 2) kareleri güvenli midir?

```
boolean[] sut = new boolean[n];
boolean[] k1 = new boolean[2*n-1]; // r+c
boolean[] k2 = new boolean[2*n-1]; // r-c+n-1
// (r, c) güvenli mi?
int a = r + c, b = r - c + n - 1;
if (sut[c] || k1[a] || k2[b]) continue;
sut[c] = k1[a] = k2[b] = true;     // yerleştir
```



▶ **Cevabı göster**

ÇOKTAN SEÇMELİ · N-VEZİR

Soru 9: Tüm Çözümleri Saymak

İlk çözümde duran `coz` metodunu tüm çözümleri sayacak biçimde değiştirmenin doğru yolu hangisidir?

A) satir == n iken sayac++ yapıp false döndürmek

B) satir == n iken sayac++ yapıp true döndürmek

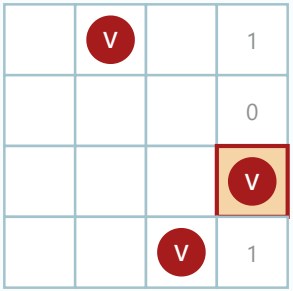
C) Geri alma satırını silip döngüyü n kez çalıştırmak

D) Her yerleştirmede sayac++ yapıp aramayı bitirmek

ÇIKTI SORUSU · EN AZ ÇAKIŞMA

Soru 10: Çakışma Sayıları

4 × 4 tahtada her sütunda bir vezir var; sütun 0, 1, 2, 3'teki vezirlerin satırları [2, 0, 3, 2]. Onarım için sütun 3'teki vezir seçildi. Bu vezir sütunu içinde her satıra taşınsa diğer vezirlerden kaçıyla çakışır? En az çakışma onu nereye taşır?



▶ **Cevabı göster**

ÇOKTAN SEÇMELİ · EN AZ ÇAKIŞMA

Soru 11: En Az Çakışmanın Sınırı

En az çakışma sezgiseli ile geri izleme karşılaştırıldığında aşağıdakilerden hangisi doğrudur?

A) En az çakışma tamdır; çözüm yoksa bunu kanıtlar

B) Geri izleme büyük n'de en az çakışmadan hızlıdır

C) En az çakışma tüm çözümleri sayabilir, geri dönüş yalnızca birini bulur

D) En az çakışma adım sınırına ulaşırsa çözüm olmadığı söylenemez

ÇIKTI SORUSU · SUDOKU

Soru 12: Kutunun Köşesi

9 × 9 Sudoku'nun `gecerliMi` metodundaki kutu hesabı aşağıda. Kod ne yazar? (7, 4) hücresinin kutusu hangi satır ve sütunları kapsar?

```
int[][] h = {{7, 4}, {2, 8}, {5, 5}};
for (int[] x : h) {
  int bs = x[0] / 3 * 3;
  int bk = x[1] / 3 * 3;
  System.out.print(bs + " " + bk + " ");
}
```

▶ **Cevabı göster**

ÇOKTAN SEÇMELİ · SUDOKU

Soru 13: MRV Sezgiseli

Sudoku çözücüsünde MRV (minimum remaining values) sezgiseli neyi değiştirir?

A) Her hücrede rakamları 9'dan 1'e doğru dener

B) Geçerlilik testini satır, sütun ve kutu için O(1) yapar

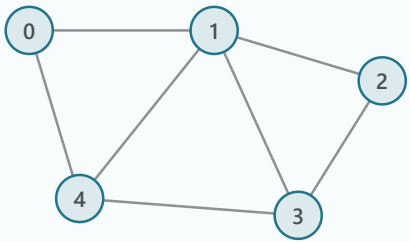
C) Çıkmaza girince en eski kararı geri alır

D) Sıradaki hücre olarak en az adayı kalanı seçer

Soru 14: Çizge Boyama İzleme

Kenarlar 0–1, 0–4, 1–2, 1–3, 1–4, 2–3, 3–4; m = 3. `boya(0)` renkleri nasıl atar, nerede geri döner?

```
1  boolean boya(int v) {
2    if (v == n) return true;
3    for (int c = 1; c <= m; c++) {
4      if (uygunMu(v, c)) { // komşularda c yok
5        renk[v] = c;
6        if (boya(v + 1)) return true;
7        renk[v] = 0;              // geri al
8      }
9    }
10   return false;
11 }
```



5 düğüm, m = 3 renk (r1, r2, r3). Düğümler 0, 1, 2, … sırayla boyanır; her düğümde renk 1, 2, 3 sırayla denenir.

Adım 1 / 12

DEĞİŞKENLER

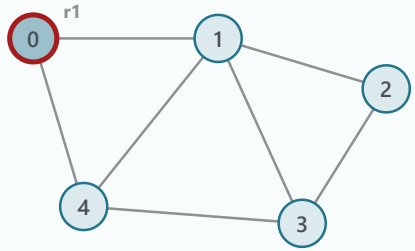
`renk = [0, 0, 0, 0, 0]`

ÇIKTI

Soru 14: Çizge Boyama İzleme

Kenarlar 0–1, 0–4, 1–2, 1–3, 1–4, 2–3, 3–4; m = 3. `boya(0)` renkleri nasıl atar, nerede geri döner?

```
1  boolean boya(int v) {
2    if (v == n) return true;
3    for (int c = 1; c <= m; c++) {
▶      if (uygunMu(v, c)) { // komşularda c yok
▶        renk[v] = c;
▶        if (boya(v + 1)) return true;
7        renk[v] = 0;              // geri al
8      }
9    }
10   return false;
11 }
```



Düğüm 0: renk 1 uygun → atanır, boya(1).

Adım 2 / 12

DEĞİŞKENLER

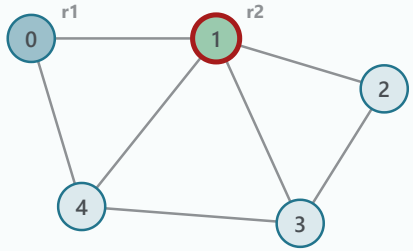
`renk = [1, 0, 0, 0, 0]`

ÇIKTI

Soru 14: Çizge Boyama İzleme

Kenarlar 0–1, 0–4, 1–2, 1–3, 1–4, 2–3, 3–4; m = 3. `boya(0)` renkleri nasıl atar, nerede geri döner?

```
1  boolean boya(int v) {
2    if (v == n) return true;
3    for (int c = 1; c <= m; c++) {
▶      if (uygunMu(v, c)) { // komşularda c yok
▶        renk[v] = c;
▶        if (boya(v + 1)) return true;
7        renk[v] = 0;              // geri al
8      }
9    }
10   return false;
11 }
```



Düğüm 1: renk 1 komşuda var; renk 2 uygun → atanır, boya(2).

Adım 3 / 12

DEĞİŞKENLER

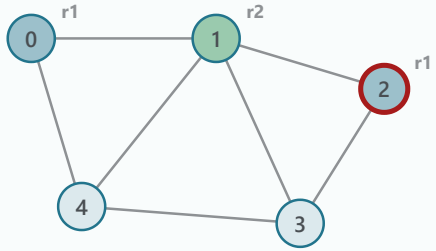
`renk = [1, 2, 0, 0, 0]`

ÇIKTI

Soru 14: Çizge Boyama İzleme

Kenarlar 0–1, 0–4, 1–2, 1–3, 1–4, 2–3, 3–4; m = 3. `boya(0)` renkleri nasıl atar, nerede geri döner?

```
1  boolean boya(int v) {
2    if (v == n) return true;
3    for (int c = 1; c <= m; c++) {
▶      if (uygunMu(v, c)) { // komşularda c yok
▶        renk[v] = c;
▶        if (boya(v + 1)) return true;
7        renk[v] = 0;              // geri al
8      }
9    }
10   return false;
11 }
```



Düğüm 2: renk 1 uygun → atanır, boya(3).

Adım 4 / 12

DEĞİŞKENLER

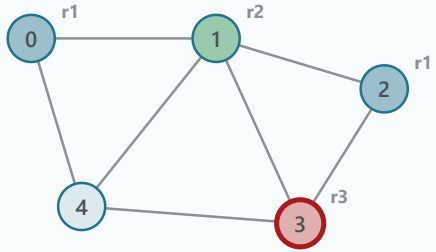
`renk = [1, 2, 1, 0, 0]`

ÇIKTI

Soru 14: Çizge Boyama İzleme

Kenarlar 0–1, 0–4, 1–2, 1–3, 1–4, 2–3, 3–4; m = 3. `boya(0)` renkleri nasıl atar, nerede geri döner?

```
1  boolean boya(int v) {
2    if (v == n) return true;
3    for (int c = 1; c <= m; c++) {
▶      if (uygunMu(v, c)) { // komşularda c yok
▶        renk[v] = c;
▶        if (boya(v + 1)) return true;
7        renk[v] = 0;              // geri al
8      }
9    }
10   return false;
11 }
```



Düğüm 3: renk 1, 2 komşuda var; renk 3 uygun → atanır, boya(4).

DEĞİŞKENLER

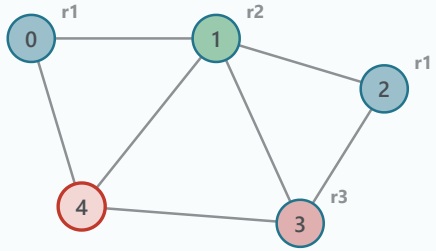
`renk = [1, 2, 1, 3, 0]`

ÇIKTI

Soru 14: Çizge Boyama İzleme

Kenarlar 0–1, 0–4, 1–2, 1–3, 1–4, 2–3, 3–4; m = 3. `boya(0)` renkleri nasıl atar, nerede geri döner?

```
1  boolean boya(int v) {
2    if (v == n) return true;
3    for (int c = 1; c <= m; c++) {
4      if (uygunMu(v, c)) { // komşularda c yok
5        renk[v] = c;
6        if (boya(v + 1)) return true;
7        renk[v] = 0;              // geri al
8      }
9    }
▶    return false;
11 }
```



Düğüm 4: denenecek renk kalmadı → **return false** (çıkmaz).

Adım 6 / 12

DEĞİŞKENLER

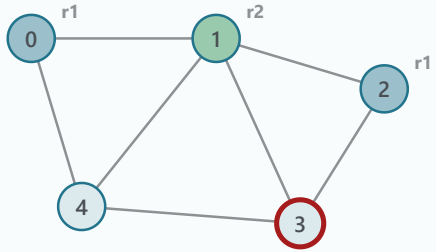
`renk = [1, 2, 1, 3, 0]`

ÇIKTI

Soru 14: Çizge Boyama İzleme

Kenarlar 0–1, 0–4, 1–2, 1–3, 1–4, 2–3, 3–4; m = 3. `boya(0)` renkleri nasıl atar, nerede geri döner?

```
1  boolean boya(int v) {
2    if (v == n) return true;
3    for (int c = 1; c <= m; c++) {
4      if (uygunMu(v, c)) { // komşularda c yok
5        renk[v] = c;
6        if (boya(v + 1)) return true;
▶        renk[v] = 0;                // geri al
8      }
9    }
10   return false;
11 }
```



Geri al: düğüm 3'ün rengi (3) silinir; aynı düğümde sıradaki renk denenecek.

Adım 7 / 12

DEĞİŞKENLER

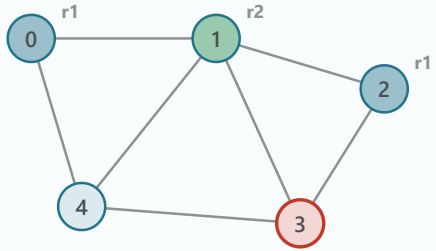
`renk = [1, 2, 1, 0, 0]`

ÇIKTI

Soru 14: Çizge Boyama İzleme

Kenarlar 0–1, 0–4, 1–2, 1–3, 1–4, 2–3, 3–4; m = 3. `boya(0)` renkleri nasıl atar, nerede geri döner?

```
1  boolean boya(int v) {
2    if (v == n) return true;
3    for (int c = 1; c <= m; c++) {
4      if (uygunMu(v, c)) { // komşularda c yok
5        renk[v] = c;
6        if (boya(v + 1)) return true;
7        renk[v] = 0;              // geri al
8      }
9    }
▶    return false;
11 }
```



Düğüm 3: denenecek renk kalmadı → **return false** (çıkmaz).

Adım 8 / 12

DEĞİŞKENLER

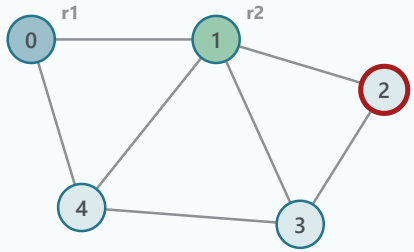
`renk = [1, 2, 1, 0, 0]`

ÇIKTI

Soru 14: Çizge Boyama İzleme

Kenarlar 0–1, 0–4, 1–2, 1–3, 1–4, 2–3, 3–4; m = 3. `boya(0)` renkleri nasıl atar, nerede geri döner?

```
1  boolean boya(int v) {
2    if (v == n) return true;
3    for (int c = 1; c <= m; c++) {
4      if (uygunMu(v, c)) { // komşularda c yok
5        renk[v] = c;
6        if (boya(v + 1)) return true;
▶        renk[v] = 0;                // geri al
8      }
9    }
10   return false;
11 }
```



Geri al: düğüm 2'nin rengi (1) silinir; aynı düğümde sıradaki renk denenecek.

Adım 9 / 12

DEĞİŞKENLER

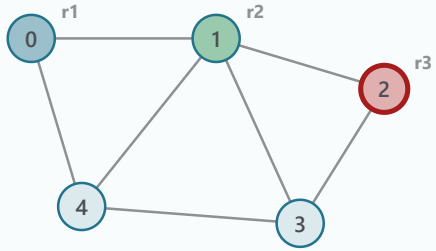
`renk = [1, 2, 0, 0, 0]`

ÇIKTI

Soru 14: Çizge Boyama İzleme

Kenarlar 0–1, 0–4, 1–2, 1–3, 1–4, 2–3, 3–4; m = 3. `boya(0)` renkleri nasıl atar, nerede geri döner?

```
1  boolean boya(int v) {
2    if (v == n) return true;
3    for (int c = 1; c <= m; c++) {
▶      if (uygunMu(v, c)) { // komşularda c yok
▶        renk[v] = c;
▶        if (boya(v + 1)) return true;
7        renk[v] = 0;              // geri al
8      }
9    }
10   return false;
11 }
```



Düğüm 2: renk 2 komşuda var; renk 3 uygun → atanır, boya(3).

Adım 10 / 12

DEĞİŞKENLER

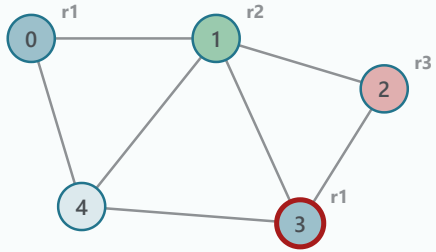
`renk = [1, 2, 3, 0, 0]`

ÇIKTI

Soru 14: Çizge Boyama İzleme

Kenarlar 0–1, 0–4, 1–2, 1–3, 1–4, 2–3, 3–4; m = 3. `boya(0)` renkleri nasıl atar, nerede geri döner?

```
1  boolean boya(int v) {
2    if (v == n) return true;
3    for (int c = 1; c <= m; c++) {
▶      if (uygunMu(v, c)) { // komşularda c yok
▶        renk[v] = c;
▶        if (boya(v + 1)) return true;
7        renk[v] = 0;              // geri al
8      }
9    }
10   return false;
11 }
```



Düğüm 3: renk 1 uygun → atanır, boya(4).

Adım 11 / 12

DEĞİŞKENLER

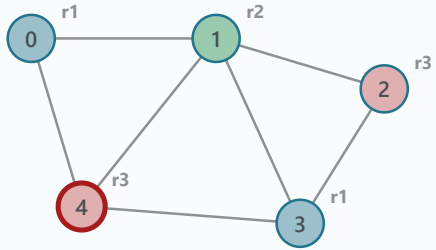
`renk = [1, 2, 3, 1, 0]`

ÇIKTI

Soru 14: Çizge Boyama İzleme

Kenarlar 0–1, 0–4, 1–2, 1–3, 1–4, 2–3, 3–4; m = 3. `boya(0)` renkleri nasıl atar, nerede geri döner?

```
1  boolean boya(int v) {
2    if (v == n) return true;
3    for (int c = 1; c <= m; c++) {
▶      if (uygunMu(v, c)) { // komşularda c yok
▶        renk[v] = c;
▶        if (boya(v + 1)) return true;
7        renk[v] = 0;              // geri al
8      }
9    }
10   return false;
11 }
```



Düğüm 4: renk 1, 2 komşuda var; renk 3 uygun → atanır, boya(5). Beş düğüm de boyandı: geçerli 3-boyama [1, 2, 3, 1, 3].

Adım 12 / 12

DEĞİŞKENLER

`renk = [1, 2, 3, 1, 3]`

ÇIKTI

```
true
```

KLASİK SORU · ÇİZGE BOYAMA VE HAMILTON YOLU

Soru 15: Tek Uzunluklu Döngü

5 düğümlü döngü çizgesi C_5 (0–1–2–3–4–0), `boya(0)` ile m = 2 renkle boyanmaya çalışılıyor. Arama ne döndürür ve neden? En az kaç renk gerekir?

▶ **Cevabı göster**

Soru 16: Kelime Arama İzleme

Izgara ABCE / SFCS / ADEE. (1, 0)'dan başlayan `ara` çağrısı "SADE" kelimesini nasıl bulur? (ds = {−1, 1, 0, 0}, dk = {0, 0, −1, 1})

```
1  boolean ara(char[][] iz, String kelime,
2              int i, int s, int k,
3              boolean[][] ziyaret) {
4    if (i == kelime.length()) return true;
5    if (s < 0 || s >= iz.length || k < 0
6        || k >= iz[0].length || ziyaret[s][k]
7        || iz[s][k] != kelime.charAt(i))
8      return false;
9    ziyaret[s][k] = true;
10   for (int d = 0; d < 4; d++)
11     if (ara(iz, kelime, i + 1, s + ds[d],
12             k + dk[d], ziyaret))
13       return true;
14   ziyaret[s][k] = false;       // geri al
15   return false;
16 }
```

A	B	C	E
S	F	C	S
A	D	E	E

Izgara ABCE / SFCS / ADEE, kelime "SADE". ara(iz, "SADE", 0, 1, 0, ziyaret): (1, 0)'daki S'den başlanır. Yön sırası: yukarı, aşağı, sol, sağ.

Adım 1 / 8

DEĞİŞKENLER

`i = 0` `(s, k) = (1, 0)`

ÇIKTI

Soru 16: Kelime Arama İzleme

Izgara ABCE / SFCS / ADEE. (1, 0)'dan başlayan `ara` çağrısı "SADE" kelimesini nasıl bulur? (ds = {−1, 1, 0, 0}, dk = {0, 0, −1, 1})

```
1  boolean ara(char[][] iz, String kelime,
2              int i, int s, int k,
3              boolean[][] ziyaret) {
4    if (i == kelime.length()) return true;
5    if (s < 0 || s >= iz.length || k < 0
6        || k >= iz[0].length || ziyaret[s][k]
7        || iz[s][k] != kelime.charAt(i))
8      return false;
▶    ziyaret[s][k] = true;
▶    for (int d = 0; d < 4; d++)
11     if (ara(iz, kelime, i + 1, s + ds[d],
12             k + dk[d], ziyaret))
13       return true;
14   ziyaret[s][k] = false;      // geri al
15   return false;
16 }
```

A	B	C	E
S	F	C	S
A	D	E	E

(1, 0) = 'S' kelimenin 0. harfiyle eşleşti → ziyaret edildi işaretlenir, 4 komşu denenir.

Adım 2 / 8

DEĞİŞKENLER

`i = 0` `(s, k) = (1, 0)`

ÇIKTI

Soru 16: Kelime Arama İzleme

Izgara ABCE / SFCS / ADEE. (1, 0)'dan başlayan `ara` çağrısı "SADE" kelimesini nasıl bulur? (ds = {−1, 1, 0, 0}, dk = {0, 0, −1, 1})

```
1  boolean ara(char[][] iz, String kelime,
2              int i, int s, int k,
3              boolean[][] ziyaret) {
4    if (i == kelime.length()) return true;
5    if (s < 0 || s >= iz.length || k < 0
6        || k >= iz[0].length || ziyaret[s][k]
7        || iz[s][k] != kelime.charAt(i))
8      return false;
▶    ziyaret[s][k] = true;
▶    for (int d = 0; d < 4; d++)
11     if (ara(iz, kelime, i + 1, s + ds[d],
12             k + dk[d], ziyaret))
13       return true;
14   ziyaret[s][k] = false;      // geri al
15   return false;
16 }
```

A	B	C	E
S	F	C	S
A	D	E	E

(0, 0) = 'A' kelimenin 1. harfiyle eşleşti → ziyaret edildi işaretlenir, 4 komşu denenir.

Adım 3 / 8

DEĞİŞKENLER

`i = 1` `(s, k) = (0, 0)`

ÇIKTI

Soru 16: Kelime Arama İzleme

Izgara ABCE / SFCS / ADEE. (1, 0)'dan başlayan `ara` çağrısı "SADE" kelimesini nasıl bulur? (ds = {−1, 1, 0, 0}, dk = {0, 0, −1, 1})

```
1  boolean ara(char[][] iz, String kelime,
2              int i, int s, int k,
3              boolean[][] ziyaret) {
4    if (i == kelime.length()) return true;
5    if (s < 0 || s >= iz.length || k < 0
6        || k >= iz[0].length || ziyaret[s][k]
7        || iz[s][k] != kelime.charAt(i))
8      return false;
9    ziyaret[s][k] = true;
10   for (int d = 0; d < 4; d++)
11     if (ara(iz, kelime, i + 1, s + ds[d],
12             k + dk[d], ziyaret))
13       return true;
▶    ziyaret[s][k] = false;      // geri al
▶    return false;
16 }
```

A	B	C	E
S	F	C	S
A	D	E	E

(0, 0)'dan hiçbir komşu 'D' ile sürmüyor (B, sınır ya da ziyaretli) → **işaret kaldırılır**, false.

Adım 4 / 8

DEĞİŞKENLER

`i = 1` `(s, k) = (0, 0)`

ÇIKTI

Soru 16: Kelime Arama İzleme

Izgara ABCE / SFCS / ADEE. (1, 0)'dan başlayan `ara` çağrısı "SADE" kelimesini nasıl bulur? (ds = {−1, 1, 0, 0}, dk = {0, 0, −1, 1})

```
1  boolean ara(char[][] iz, String kelime,
2              int i, int s, int k,
3              boolean[][] ziyaret) {
4    if (i == kelime.length()) return true;
5    if (s < 0 || s >= iz.length || k < 0
6        || k >= iz[0].length || ziyaret[s][k]
7        || iz[s][k] != kelime.charAt(i))
8      return false;
▶    ziyaret[s][k] = true;
▶    for (int d = 0; d < 4; d++)
11     if (ara(iz, kelime, i + 1, s + ds[d],
12             k + dk[d], ziyaret))
13       return true;
14   ziyaret[s][k] = false;      // geri al
15   return false;
16 }
```

A	B	C	E
S	F	C	S
A	D	E	E

(2, 0) = 'A' kelimenin 1. harfiyle eşleşti → ziyaret edildi işaretlenir, 4 komşu denenir.

Adım 5 / 8

DEĞİŞKENLER

`i = 1` `(s, k) = (2, 0)`

ÇIKTI

Soru 16: Kelime Arama İzleme

Izgara ABCE / SFCS / ADEE. (1, 0)'dan başlayan `ara` çağrısı "SADE" kelimesini nasıl bulur? (ds = {−1, 1, 0, 0}, dk = {0, 0, −1, 1})

```
1  boolean ara(char[][] iz, String kelime,
2              int i, int s, int k,
3              boolean[][] ziyaret) {
4    if (i == kelime.length()) return true;
5    if (s < 0 || s >= iz.length || k < 0
6        || k >= iz[0].length || ziyaret[s][k]
7        || iz[s][k] != kelime.charAt(i))
8      return false;
▶    ziyaret[s][k] = true;
▶    for (int d = 0; d < 4; d++)
11     if (ara(iz, kelime, i + 1, s + ds[d],
12             k + dk[d], ziyaret))
13       return true;
14   ziyaret[s][k] = false;      // geri al
15   return false;
16 }
```

A	B	C	E
S	F	C	S
A	D	E	E

(2, 1) = 'D' kelimenin 2. harfiyle eşleşti → ziyaret edildi işaretlenir, 4 komşu denenir.

Adım 6 / 8

DEĞİŞKENLER

`i = 2` `(s, k) = (2, 1)`

ÇIKTI

Soru 16: Kelime Arama İzleme

Izgara ABCE / SFCS / ADEE. (1, 0)'dan başlayan `ara` çağrısı "SADE" kelimesini nasıl bulur? (ds = {−1, 1, 0, 0}, dk = {0, 0, −1, 1})

```
1  boolean ara(char[][] iz, String kelime,
2              int i, int s, int k,
3              boolean[][] ziyaret) {
4    if (i == kelime.length()) return true;
5    if (s < 0 || s >= iz.length || k < 0
6        || k >= iz[0].length || ziyaret[s][k]
7        || iz[s][k] != kelime.charAt(i))
8      return false;
▶    ziyaret[s][k] = true;
▶    for (int d = 0; d < 4; d++)
11     if (ara(iz, kelime, i + 1, s + ds[d],
12             k + dk[d], ziyaret))
13       return true;
14   ziyaret[s][k] = false;      // geri al
15   return false;
16 }
```

A	B	C	E
S	F	C	S
A	D	E	E

(2, 2) = 'E' kelimenin 3. harfiyle eşleşti → ziyaret edildi işaretlenir, 4 komşu denenir.

Adım 7 / 8

DEĞİŞKENLER

`i = 3` `(s, k) = (2, 2)`

ÇIKTI

Soru 16: Kelime Arama İzleme

Izgara ABCE / SFCS / ADEE. (1, 0)'dan başlayan `ara` çağrısı "SADE" kelimesini nasıl bulur? (ds = {−1, 1, 0, 0}, dk = {0, 0, −1, 1})

```
1  boolean ara(char[][] iz, String kelime,
2              int i, int s, int k,
3              boolean[][] ziyaret) {
▶  if (i == kelime.length()) return true;
5  if (s < 0 || s >= iz.length || k < 0
6      || k >= iz[0].length || ziyaret[s][k]
7      || iz[s][k] != kelime.charAt(i))
8    return false;
9  ziyaret[s][k] = true;
10 for (int d = 0; d < 4; d++)
11   if (ara(iz, kelime, i + 1, s + ds[d],
12           k + dk[d], ziyaret))
13     return true;
14 ziyaret[s][k] = false;      // geri al
15 return false;
16 }
```

A	B	C	E
S	F	C	S
A	D	E	E

i == 4 == kelime uzunluğu → true. Yol: (1,0) S → (2,0) A → (2,1) D → (2,2) E.

Adım 8 / 8

DEĞİŞKENLER

`i = 4`

ÇIKTI

```
true
```

ÇOKTAN SEÇMELİ · KELİME ARAMA

Soru 17: Ziyaret İşaretini Kaldırmamak

Kelime aramada `ziyaret[s][k] = false` satırı unutulursa ne olur?

A) Aynı hücre bir yolda iki kez kullanılabilir hâle gelir ve yol uzar

B) Çıkmaz bir yolda kullanılan hücre başka yollarda kullanılamaz

C) Arama sonsuz döngüye girer ve yığın taşar

D) Hiçbir şey değişmez; işaret yalnızca hız içindir

ÇOKTAN SEÇMELİ · ANALİZ

Soru 18: Karmaşıklık

Geri izlemede en kötü durum karmaşıklıkları için hangi eşleştirme doğrudur?

A) Hamilton yolu $O(2^n)$, kelime arama $O(R \cdot C)$

B) n-vezir $O(n^2)$, Sudoku $O(9m)$

C) Sudoku $O(9^m)$, çizge m-boyama $O(n \cdot m^n)$

D) Alt kümeler $O(n!)$, permütasyonlar $O(2^n)$

Son

Bölüm 4: Geri İzleme Algoritmaları: Çalışma Soruları

Konu anlatımı: Geri izleme algoritmaları sunumu · Terim sözlüğü