



# SET MATRİX ZEROES

HAZIRLAYANLAR:

Azra ATEŞOĞLU

Nisa Naz KORKMAZ

# Giriş



**1-Matris Nedir?**

**2-»Set Matrix Zeroes « Problemi ,Neden Önemli Olduğu,Çözümü**

**3- Ele aldığımız Problemin Pseudo kodu**

**5-Sınırlamalar ve Kullanılan Veri Yapıları**

**5-Kısıtlar ve Beklentiler**

**6-Algoritma Ayrıntıları**

**7-Problemin Çözümüne Yönelik Kodlama**

**8-Kodun Başarısını Test Etme**

**9-Günlük Hayattan Örnekler**

**10-Sıkça Sorulan Sorular**

**11-Sonuçlar ve Öneriler**

**12-Kaynaklar ve Github Karekodu**

# MATRİSLERE BAKALIM ...



## 1. Excel Tabloları

Satırlar genellikle farklı kayıtları (kişiler, ürünler, tarihler vs.), sütunlar ise bu kayıtlara ait özellikleri (ad, fiyat, tarih gibi) temsil eder.

## 2. Sensör Verileri

Birçok sensör ağı (örneğin hava kalitesi izleme, sıcaklık ölçümleri) belirli bölgelerde düzenli aralıklarla veri toplar.

Bu veriler coğrafi olarak ya da zaman bazlı olarak matris şeklinde saklanabilir.

Saat → 10:00 | 10:05 | 10:10

Sensör1 → 25°C | 26°C | 27°C

Sensör2 → 24°C | 25°C | 25°C

## 3. Görsel Veriler (Piksel Düzenleri)

- Her dijital görüntü aslında bir matristir.
- Her pikselin bir (renk) değeri vardır ve bu değerler satır-satır yerleştirilmiş matrisler şeklinde tutulur.

**Tablo (matris) tabanlı veri yapıları**, bir hücredeki veri hatalıysa veya özel bir durumu (örneğin 0) temsil ediyorsa, bu bilgi tüm satır ya da sütunu etkileyebilir.

$$M = \begin{bmatrix} 4 & 6 & 5 \\ 9 & 8 & 0 \\ 2 & 4 & 1 \end{bmatrix}$$

İşte bu durum, “**Set Matrix Zeroes**” problemi gibi senaryoları ortaya çıkarır.

**? Peki ya şöyle bir durumla karşılaşırsak:**


**Bir tablodaki tek bir hücrenin durumu, gerçekten tüm satırı ve sütunu değiştirmemizi gerektiriyorsa?**

Bu durumda yapılacak işlemler hem **veri bütünlüğü**, hem de **sistem güvenliği** açısından önem taşır. Ve çözüm için akıllı bir algoritma gerekir...

# “Set Matrix Zeroes” Problemi Nedir?



- Elimizde  $M \times N$  boyutunda bir matris (2D liste) olduğunu düşünelim.
- Eğer herhangi bir hücre 0 ise, **o hücrenin bulunduğu satırın ve sütunun tamamı 0 olmalı.**
- Bu işlem, **tüm sıfırlar aynı anda etkili olacak şekilde** yapılmalıdır.

 Yani bir sıfırın değiştirdiği değer, başka bir satır/sütunu etkilememelidir.

**Amaç:** Tüm bu işlemi orijinal matrisi değiştirerek ve mümkünse ekstra alan kullanmadan yapabilmek.

## Günlük Hayattaki Karşılığı:

- ☐ Elektronik devrelerde kısa devre oluşması gibi, bir arıza tüm bağlantıyı etkileyebilir.
- ☐ Görsel bir sistemde bir bozuk piksel tüm bir çizgiyi etkileyebilir.

# NEDEN ÖNEMLİ?



Diyelim ki bir Excel tablosu düşünüyorsun ve bu tabloya hata içeren bir “0” değeri girilmiş. Eğer bu 0 değeri bir “hata”yı simgeliyorsa, bu durumda bu hatanın **görsel veya sistematik olarak tüm satıra ve sütuna yayılması** gerekebilir.

Çünkü:

- Aynı satırdaki diğer veriler bu hatalı veriyle ilişkili olabilir.
- Aynı sütundaki veriler de o alanla ilişkili olabilir.

# Çözüm

- Matrisin ilk satırı ve ilk sütunu işaretleme alanı olarak kullanılır.
- Sıfır olan hücrelerin satır ve sütun bilgisi bu bölgelerde tutulur.
- İlk satır ve ilk sütunun sıfır içerip içermediği, iki ayrı boolean değişken - *firstRowZero* ve *firstColZero*- ile kontrol edilir.
- Böylece tüm işlemler *sabit* ek alanla yapılır ve alan karmaşıklığı  $O(1)$  olur.

$$M = \begin{bmatrix} 4 & 6 & 5 \\ 9 & 8 & 0 \\ 2 & 4 & 1 \end{bmatrix}$$

$$M = \begin{bmatrix} 4 & 6 & 0 \\ 0 & 8 & 0 \\ 2 & 4 & 1 \end{bmatrix}$$



{ }

$$M = \begin{bmatrix} 4 & 6 & 5 \\ 9 & 8 & 0 \\ 2 & 4 & 1 \end{bmatrix}$$

$$M = \begin{bmatrix} 4 & 6 & 0 \\ 0 & 0 & 0 \\ 2 & 4 & 0 \end{bmatrix}$$



# Pseudo Code

```
FUNCTION SetZeroes(matrix):
    m = number of rows in matrix
    n = number of columns in matrix

    // Check if first row contains any zero
    firstRowZero = FALSE
    FOR j = 0 TO n-1:
        IF matrix[0][j] == 0:
            firstRowZero = TRUE
            BREAK

    // Check if first column contains any zero
    firstColZero = FALSE
    FOR i = 0 TO m-1:
        IF matrix[i][0] == 0:
            firstColZero = TRUE
            BREAK

    // Use first row and first column as markers
    FOR i = 1 TO m-1:
        FOR j = 1 TO n-1:
            IF matrix[i][j] == 0:
                matrix[i][0] = 0 // Mark the corresponding row
                matrix[0][j] = 0 // Mark the corresponding column
```

```
// Set zeros based on markers (excluding first row and column)
FOR i = 1 TO m-1:
    FOR j = 1 TO n-1:
        IF matrix[i][0] == 0 OR matrix[0][j] == 0:
            matrix[i][j] = 0

// Handle first row if needed
IF firstRowZero:
    FOR j = 0 TO n-1:
        matrix[0][j] = 0

// Handle first column if needed
IF firstColZero:
    FOR i = 0 TO m-1:
        matrix[i][0] = 0
```

## 🎯 Amaç:

Matris içinde herhangi bir hücre 0 ise, o hücrenin bulunduğu satır ve sütunun tamamını 0 yapmak.

## Neden Bu Yöntem ?

- ✓ Ekstra listeye ihtiyaç duymaz.
- ✓ Bellek kullanımını azaltır ( $O(1)$  alan karmaşıklığı).
- ✓ Büyük matrislerde verimlidir.

 Not:

Bu yöntem yalnızca  $O(1)$  alan gerektirir ancak **kod yapısı biraz daha karmaşık** olabilir.

Avantajı, **bellek verimliliği**dir.

# Sınırlamalar

- İlk satır veya ilk sütun sıfır içeriyorsa dikkatli kontrol gerekir.
- Kod yapısı diğer yöntemlere göre biraz daha karmaşık olabilir.
- Okunabilirlik azalabilir, ancak alan açısından daha verimlidir.

# Kullanılan Veri Yapıları

- Matrisin **kendisi** veri yapısı olarak kullanılır.
- Ekstra HashSet, liste veya dizi gibi bir *veri yapısı* kullanılmaz.
- İlk satır ve sütun, işaretleme için yeniden amaçlandırılır.

# Kısıtlar ve Beklentiler

Zaman Karmaşıklığı	$O(m \times n)$
Alan Karmaşıklığı	$O(1)$ – In Place Method (seçtiğimiz yöntem) $O(m+n)$ – HashSet Method
Girdi Büyüklüğü	$1 \leq m, n \leq 200$ $-2^{31} \leq \text{matrix}[i][j] \leq 2^{31} - 1$
Edge Case'ler	i. Hiç sıfır olmayan matris ii. Tüm matris sıfır iii. Tek satır veya tek sütun 0 ilk satır veya sütundaysa (in place için dikkat edilmeli.

# Algoritma Ayrıntıları

- **Başlangıç kontrolü:**

İlk satır ve sütun taranarak sıfır içerip içermediği *firstRowZero* ve *firstColZero* ile belirlenir.

- **İşaretleme:**

Kalan matris taranır. *Sıfır* görüldüğünde, ilgili satır ve sütunun ilk hücresi sıfır yapılır.

- **Sıfırlama:**

İşaretlere göre, kalan hücreler sıfırlanır.

- **Son kontrol:**

Değişkenler ***true*** ise ilk satır veya ilk sütun tamamen sıfırlanır.



# Gerçek Hayat Kullanım Senaryosu



- **Görüntü işleme:** Hatalı bir piksel tespit edildiğinde, tüm satır/sütun maskelenerek görüntüdeki etki alanı sıfırlanabilir.
- **Veri gizliliği:** Kimlik bilgisi eksik bir kayıt tespit edildiğinde, tüm satır sistemden çıkarılabilir veya sıfırlanabilir.
- **Veri temizliği:** Eksik değer içeren veriler, model eğitime zarar vermemesi için tüm satır/sütun olarak işaretlenir veya kaldırılır.

# Sıkça Sorulan Sorular

- In-place çözüm neden önemlidir?
- İlk satır ve sütun neden işaretleme (flag) için kullanılıyor?
- İlk satır ve sütunu nasıl koruruz?
- Naive çözüm ile in-place çözüm arasındaki temel fark nedir?

# Sonuçlar ve Öneriler🔍

- In-place yaklaşımı, alan kısıtının olduğu ortamlarda tercih edilmelidir.
- Kodun karmaşıklığı artar, ancak performans ve verimlilik kazanılır.
- Matrisin kendisini veri yapısı olarak kullanmak, donanım dostu çözümler sunar.

# Kaynaklar

- ChatGPT (OpenAI, 2024) – Support for conceptual explanations and algorithm design.
- Claude AI (Anthropic) – Assistance with pseudocode structuring.
- [LeetCode – Set Matrix Zeroes](#) – Reference for problem description and constraints
- W3Schools – Data Structures Basics
- GeeksforGeeks – Set Matrix Zeroes Explanation

Kaynak kodlar için yandaki QR okutabilirsiniz.





# Teşekkürler

Azra ATEŞOĞLU  
Nisa Naz KORKMAZ