Hamming Kodlama Simülatörü

# 🔍 Proje Amacı

Bu proje, Hamming kodlama algoritması ile hata tespiti ve düzeltme mekanizmasını görselleştirerek öğretmek amacıyla geliştirilmiştir. Kullanıcıdan alınan ikili veri üzerinde Hamming kodu hesaplanır, belleğe kaydedilir, istenirse rastgele hata eklenebilir ve bu hata tespit edilip düzeltilebilir.

# ⚙️ Kullanılan Teknolojiler

- Python 3  
- Tkinter (Grafik Arayüz)  
- random (Hatalı bit üretmek için)

# 🧮 Projenin Özellikleri

- Veri Girişi: 8, 16 veya 32 bitlik ikili veri girişini destekler.  
- Hamming Kod Hesaplama: Girilen veriye uygun Hamming kodu hesaplanır ve belleğe kaydedilir.  
- Hatalı Bit Ekleme: Rastgele bir bit pozisyonu seçilerek veri bozulur.  
- Hata Tespiti ve Düzeltme: Hamming algoritmasıyla bozulmuş bitin pozisyonu bulunur ve düzeltme yapılır.  
- Arayüz: Basit, kullanıcı dostu Tkinter arayüzü ile işlemler kolayca yapılır.

# 🖥️ Ekran Görüntüsü

metin, ekran görüntüsü, yazı tipi içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

# metin, ekran görüntüsü, yazı tipi içeren bir resim Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

# 

# 🧠 Çalışma Mantığı

1. Veri girişi: Kullanıcı 8, 16 veya 32 bit uzunluğunda 0 ve 1'lerden oluşan veri girer.  
2. Hamming kodu hesaplanır: Veriye uygun sayıda parite biti hesaplanarak veri içerisine yerleştirilir.  
3. Belleğe yazılır: Oluşan Hamming kodu arayüzde gösterilir.  
4. Hata eklenebilir: Rastgele bir bit pozisyonu değiştirilerek hata simüle edilir.  
5. Hata kontrolü: Hamming kodu üzerinden hangi pozisyonda hata olduğu bulunur.  
6. Düzeltme: Hatalı bit eski haline döndürülerek veri onarılır.

# Kodun Ana Bölümleri ve Açıklamaları

**a) calculate\_hamming()**  
Bu fonksiyon:

* Kullanıcıdan alınan ikili veriyi işler.
* Hamming algoritmasına göre parity (parite) bitlerini hesaplar.
* Hamming kodunu oluşturur ve belleğe kaydeder.

**b) add\_error()**

* Bellekte rastgele bir biti bozar.
* Ekrana bozulan bit pozisyonunu yazar.

**c) check\_and\_correct()**

* Sendrom (parity kontrolü) ile hatayı tespit eder.
* Bozulan biti düzeltir.
* Kullanıcıya bilgi verir.

# Karşılaşılan Zorluklar ve Çözümleri

* **Hata pozisyonunun doğru tespit edilememesi:** Parite bitlerinin hesaplanmasında yapılan küçük mantıksal hatalar nedeniyle başlangıçta pozisyon yanlış tespit ediliyordu. Kodun mantığı adım adım kontrol edilerek düzeltildi.
* **Tkinter GUI’nin güncellenmemesi:** Bellek ve sonuç kutularının her işlemde temizlenip güncellenmesi gerektiği unutulduğunda karışık sonuçlar görüntüleniyordu. delete() ve insert() fonksiyonlarıyla bu sağlandı.

# Gelecekte Geliştirme Fikirleri

* Hatalı bit sayısı arttırılarak 2-bit, 3-bit hata testleri yapılabilir (Hamming bu durumda yalnızca tespit edebilir, düzeltemez).
* GUI tasarımı daha kullanıcı dostu hale getirilebilir (örneğin renkli hata uyarıları, görsel simgeler vs.).
* Hamming dışında farklı hata düzeltme algoritmaları (Reed-Solomon gibi) eklenebilir.

📂 Kurulum ve Kullanım

1. Python 3 yüklü olduğundan emin olun.  
2. Aşağıdaki komutla gerekli dosyaları indirin:  
 git clone https://github.com/kullaniciadi/hamming-simulator.git  
 cd hamming-simulator  
3. Uygulamayı çalıştırın:  
 python hamming\_simulator.py

# 👤 Geliştirici

Ad Soyad: Merve Gök  
Üniversite/Bölüm: Btü/Bilgisayar mühendisliği

Youtube: <https://youtu.be/xPBZib_Shk8>

GitHub: <https://github.com/merveegokk/hamming-code>