PROJE UYGULAMA RAPORU: AŞAMA 2 - KODLAMA VE TEST Tarih: 19.12.2025

2.1 Geliştirme Ortamı ve Dil Seçimi

MOD-64 algoritmasının kodlanması aşamasında **Python** programlama dili tercih edilmiştir. Python'un seçilme nedenleri şunlardır:

- **Bit Seviyesi İşlemler:** Python, büyük tamsayılar üzerinde bit kaydırma (<<, >>) ve mantıksal operatörleri (XOR, AND) doğrudan ve hızlı bir şekilde uygulayabilmektedir.
- **Okunabilirlik:** Feistel Ağı mimarisinin karmaşık mantığını sade ve anlaşılır bir sözdizimi ile ifade etmeye olanak tanır.
- **Geliştirme Ortamı (IDE):** Kodlama işlemi **Visual Studio Code (VS Code)** ortamında gerçekleştirilmiştir.

Geliştirilen Temel Fonksiyonlar:

Algoritma, nesne tabanlı bir yapı içinde aşağıdaki 3 ana fonksiyon üzerine inşa edilmiştir:

1. **Anahtar_Uret(parola)**: Kullanıcıdan alınan string tipindeki parolayı ASCII dönüşümü ile 64-bitlik bir tamsayıya çevirir. Ardından her tur (8 tur) için bit kaydırma yöntemiyle özgün alt anahtarlar (Round Keys) üretir.
2. **Sifrele(duz_metin, anahtar)**: 64-bitlik düz metin bloğunu alır. 8 turluk Feistel döngüsüne sokar. Her turda sağ bloğu F-Fonksiyonu (XOR + S-Kutusu + Permütasyon) ile işleyip sol blokla XOR işlemine tabi tutar. Sonuçta şifreli sayısal veriyi (ciphertext) döndürür.
3. **Desifrele(sifreli_metin, anahtar)**: Şifreli veriyi alır. Şifreleme işleminin aynısını uygular, ancak bu sefer Anahtar_Uret fonksiyonundan gelen anahtarları **ters sırada** kullanır. Bu sayede işlem matematiksel olarak geri alınır.

2.2 Test ve Doğrulama

Algoritmanın güvenilirliğini ve Feistel mimarisinin doğru çalıştığını kanıtlamak amacıyla iki kritik test senaryosu uygulanmıştır.

Test 1: Basit Doğrulama (Correctness)

- **Amaç:** Verilen düz metnin şifrelendikten sonra doğru anahtarla tekrar eski haline döndürülebildiğini kanıtlamak.
- **Girdi:** Metin: "KALE", Anahtar: "AN"
- **Beklenen Sonuç:** Deşifreleme çıktısının "KALE" olması.

- **Gerçekleşen Sonuç:** Algoritma metni başarıyla şifrelemiş ve şifreli veriyi doğru anahtarla çözerek "KALE"çıktısını vermiştir.
- **Durum:** ✅ BAŞARILI

Test 2: Anahtar Hassasiyeti (Avalanche Effect)

- **Amaç:** Anahtardaki çok küçük bir değişikliğin (1 bit veya 1 harf), şifre çözme işlemini tamamen başarısız kıldığını göstermek (Kriptografik Çığ Etkisi).
- **Senaryo:**
  1. Metin "AN" anahtarı ile şifrelendi.
  2. Şifreli veri, "AM" anahtarı ile (Sadece 1 harf farklı) çözülmeye çalışıldı.
- **Beklenen Sonuç:** Anlamsız veya bozuk bir metin çıkması.
- **Gerçekleşen Sonuç:** Sistem, yanlış anahtar verildiğinde orijinal metin yerine [Anlamsız Veri] veya bozuk karakterler üretmiştir.
- **Durum:** ✅ BAŞARILI

Sonuç: Gerçekleştirilen kodlama ve testler sonucunda, MOD-64 algoritmasının Feistel mimarisine uygun olarak çalıştığı, şifreleme-deşifreleme döngüsünü tamamladığı ve anahtar değişikliklerine karşı duyarlı olduğu doğrulanmıştır.