



T.C
KOCAEVİ SAĞLIK VE TEKNOLOJİ ÜNİVERSİTESİ
MÜHENDİSLİK VE DOĞA BİLİMLERİ FAKÜLTESİ
BİLGİSAYAR/YAZILIM MÜHENDİSLİĞİ

PROJE KONUSU: HANGMAN GAME

**ÖĞRENCİ ADI: SUDE ÖZCELİK
ÖĞRENCİ NUMARASI: 240502053**

**ÖĞRENCİ ADI: BEREN TOPALOĞLU
ÖĞRENCİ NUMARASI: 240502025**

DERS SORUMLUSU:

PROF. DR./DR. ÖĞR. ÜYESİ FULYA AKDENİZ

TARİH: 13.11.2025

1. GİRİŞ

1.1 Projenin amacı:

- **Amaç:** Gelişmiş bir Adam Asmaca (Hangman) oyunu uygulamasını Python dilinde oluşturmak.
- **Temel Özellikler:** Puanlama sistemi, kategori seçimi, özel hamleler (İpucu ve İşlem Çözme) ve skor kaydetme.
- **Hedef Kitle:** Kelime oyunlarını seven ve basit matematik becerilerini test etmek isteyen kullanıcılar.

2. GEREKSİNİM ANALİZİ

2.1 Arayüz gereksinimleri:

- Oyunun her adımda mevcut hata hakkı, toplam puan, bonus puanı ve kelime kategorisi bilgileri gösterilmelidir (oyunu_goster).
- Kullanıcıya tahmin, ipucu al (I) ve çözüm yap (C) olmak üzere 3 temel aksiyon seçeneği sunulmalıdır.
- Adam Asmaca figürünün, kalan hata sayısına göre güncel halinin **ASCII Art** formatında çizilmesi gereklidir (adam_asmaca_ciz).
- Hata, başarı, uyarı ve sonuç mesajları (kazanma/kayıbetme) kullanıcıya açıkça iletilmelidir.
- Skor tablosu, oyun sonunda JSON dosyasından okunarak sıralı bir şekilde gösterilmelidir.

2.2 Fonksiyonel gereksinimler:

- Rastgele kelime seçme ve kategori atama.
- Kullanıcıdan tek harf veya tam kelime tahmini alma.
- Harf tahmini doğru ise puan artışı ve bonus puanı verme.
- Harf tahmini yanlış ise puan düşüşü ve hata hakkı azaltma.
- Kullanıcının C komutu ile rastgele bir matematik işlemi sorulması ve sonucuna göre gizli harf açma mekanizmasını çalıştırma.
- Kullanıcının I komutu ile bonus puanı karşılığında kategori ipucu verme.
- Kullanıcı skorlarını scores.json dosyasına kaydetme ve sıralama.

2.3 Diyagramlar

- **Use-Case Diagram (Kullanım Senaryosu Diyagramı):** Oyuncunun oyunu başlatması, harf tahmin etmesi, ipucu alması, işlem çözmesi, kazanması ve skor kaydetmesi gibi temel işlevleri göstermelidir.
- **Activity Diagram (Aktivite Diyagramı):** oyunu_baslat fonksiyonu içerisindeki ana döngüyü (Giriş Al - Seçimi Değerlendir - Durumu Güncelle - Çıkış Kontrolü) görselleştirmelidir.

3 TASARIM

3.1 Mimari tasarım:

- **Modüler Yaklaşım:** Uygulama, her biri belirli bir görevde odaklanmış ayrı Python fonksiyonlarına bölünmüştür (örneğin: rastgele_kelime_sec, oyunu_goster, skoru_kaydet).
- **Veri Yönetimi:** Oyun verileri (kelimeler, puanlar, figürler) kodun başında global sabitler olarak tanımlanmıştır, bu da kolay yapılandırma sağlar.
- **Durum Yönetimi:** Oyun durumu, kalan_hata, total_puan, bonus_puan ve tahmin_edilen_harfler değişkenleri ile merkezi olarak oyunu_baslat döngüsünde yönetilir.
- **Girdi/Çıktı Arayüzü:** Konsol tabanlı (CLI) bir arayüz kullanılmıştır.
- **Veri Akış Diyagramı (DAD) Eklenmesi:** (Ana oyunu başlatma, tahmini işleme, puanı güncelleme ve sonuçlandırma akışını gösteren bir DAD buraya eklenmelidir.)

3.2 Kullanılacak teknolojiler

- **Yazılım Dili:** Python (Yüksek seviye, okunabilir ve hızlı prototipleme için idealdir).
- **Harici Kütüphaneler:**
- random: Kelime seçimi ve hesap_cozme fonksiyonunda matematik işlemlerini rastgele oluşturmak için kullanılır.
- os: Konsol ekranını temizlemek (os.system('cls'/'clear')) için kullanılır.
- json: Skorların kalıcı olarak diske kaydedilmesi ve okunması için kullanılır (skoru_kaydet).
- **Diğer Teknolojiler:** Veri kalıcılığı için JSON formatı.

3.3 Veri tabanı tasarımı

- **Veri Tabanı Tasarımı Açıklaması:** Uygulama, geleneksel bir veri tabanı (SQL/NoSQL) kullanmak yerine, skor verilerini basit ve hafif bir yapı olan **JSON dosyası** (scores.json) içinde saklar.
- **ER Diyagramı Eklenmesi:** (Bu projede bir ER diyagramı gerekmeyez. Basitçe bir Skor tablosunun alanları (kullanıcı, skor, kelime) açıklanabilir.)

3.4 Kullanıcı arayüzü tasarımı

- **Kullanıcı Arayüzü Açıklaması:** Arayüz, kullanıcının komutlarla (H/I/C) etkileşimde bulunduğu bir konsol uygulamasıdır. Kullanıcıya sürekli güncel ve temiz bir ekran sunulur.
- **Yazılımdan Ekran Çıktıları:** (Buraya oyunun farklı aşamalarından 3-4 adet konsol çıktısı ekran görüntüsü eklenmelidir.)
 - Oyunun Başlangıç Ekranı (Kategori, Puanlar ve İlk Adam Asmaca Figürü).
 - İşlem Çözme Ekranı ve Girdi İstemi.
 - Oyun Sonu ve Skor Tablosu.
- **Uygulamanın Nasıl Çalıştırılacağı:**
 1. Python yorumlayıcısının yüklü olduğundan emin olun.
 2. Kod dosyasını (hangman_oyunu.py gibi) bir terminalde açın.
 3. python hangman_oyunu.py komutu ile oyunu başlatın.
 4. İstenildiğinde kullanıcı adınızı girerek oyuna başlayın.

4 UYGULAMA

4.1 Kodlanan bileşenlerin açıklamaları

- tr_to_en: Türkçe karakter içeren girdileri büyük harf ve İngilizce karşılığına dönüştürerek standartlaştırır.
- rastgele_kelime_sec: KELİMELER sözlüğünden kategori ve kelimeyi seçer.
- adam_asmaca_ciz: MAKS_HATA üzerinden kalan hata sayısına göre figürün doğru aşamasını ekrana basar.
- oyunu_goster: Ekranı temizler, oyun durumunu (puan, hata, kelime) ve figürü çizer.
- hesap_cozme: Rastgele matematik işlemi sorar; doğru bilinirse harf açar, yanlış bilinirse hata hakkı düşürür.
- ipucu_alma: 1 bonus puanı karşılığında kelimenin kategorisini kullanıcıya bildirir.
- skoru_kaydet: Oyun sonunda elde edilen skoru JSON dosyasına kaydeder ve ilk 5 skoru gösterir.
- oyunu_baslat: Oyunun ana döngüsünü, kullanıcı etkileşimini ve kurallarını yönetir.

4.2 Görev dağılımı:

- Kodun yazım aşamasında ortak çalışıldı. Geliştirme, ekleme ve çıkarma aşamasını Sude Özçelik üstlenmiştir. Raporlama süresince aynı şekilde birlikte çalışılmıştır.

4.3 Karşılaşılan zorluklar ve çözüm yöntemleri

- **Zorluk:** Türkçe karakterlerin (İ, Ş, Ğ vb.) tahmin ve gizli kelime karşılaşmasında sorun yaratması.

- **Çözüm:** tr_to_en fonksiyonu yazarak tüm girdiler ve gizli kelime standartlaştırılmış
- **Zorluk:** skoru_kaydet fonksiyonunda JSON dosyasının ilk kez oluşturulması veya boş olması durumunda hata alınması.
- **Çözüm:** os.path.exists ve os.path.getsize ile dosyanın varlığı ve boş olup olmadığı kontrol edilerek json.load işlemi güvenli hale getirilmiştir.
- **Zorluk:** hesap_cozme fonksiyonunda tam bölme gerektiren durumların yönetilmesi (Çıkarma ve Bölme işlemleri).
- **Çözüm:** Bölme işlemi seçildiğinde, tam sayı sonuç elde etmek için birinci sayının, ikinci sayının katı olarak rastgele atanması sağlanmıştır (sayi1 = sayi2 * random.randint(1, 10)).

4.4 Proje isterlerine göre eksik yönler

- Proje isterleri dahilinde kodlanamayan bir bileşen bulunmamaktadır.

5 TEST VE DOĞRULAMA

5.1 Yazılımın test süreci

- **tr_to_en Testi:** "Şeftali" girdisinin "SEFTALI" çıktısını verdiği kontrolü.
- **hesap_cozme Testi (Doğru Yanıt):** İşlemın doğru bilinmesi durumunda puan artışı ve harf açılmasının kontrolü.
- **skoru_kaydet Testi:** Skoru kaydettikten sonra JSON dosyasının içeriğinin doğru olup olmadığını kontrolü.

5.2 Yazılımın doğrulanması

- **Tam ve Doğru Çalışan Bileşenler:**
- tr_to_en, rastgele_kelime_sec, adam_asmaca_ciz, ipucu_alma ve skoru_kaydet fonksiyonları beklenen işlevlerini doğru şekilde yerine getirmiştir.
- **Eksik ya da Hatalı Çalışan Bileşenler:**
- Testler sonucunda önemli bir hata tespit edilmemiştir. Tüm kritik bileşenler doğru ve beklenen şekilde çalışmaktadır.

6 GitHub Bağlantıları:

[240502053-cpu/AdamasmaProje: Programlama Lab Proje](https://github.com/240502053-cpu/AdamasmaProje)

<https://github.com/240502053-cpu/AdamasmaProje/blob/main/ProgramlamLab.py>