**SAKARYA ÜNİVERSİTESİ**

**BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

**PROGRAMLAMA DİLLERİNİN PRENSİPLERİ DERSİ**

1. **ÖDEVİ RAPOR DOSYASI**

****

**HAZIRLAYAN**

**ÖMER AKKIYAL**

**G231210058**

**27.05.2025**

**1. Giriş**

Bu proje kapsamında bir **uzay aracı simülasyonu** gerçekleştirilmiştir.  
Projenin amacı, farklı gezegenlerde zamanın ilerlemesini, uzay araçlarının çıkış ve varış süreçlerini, yolcuların yaşam sürelerini ve nüfus değişimlerini doğru bir şekilde modelleyerek, bir simülasyon ortamı oluşturmaktır.

**2. Yapılanlar**

Projede aşağıdaki işlemler gerçekleştirilmiştir:

* **Gezegenler** için ayrı zaman birimleri ve nüfus değerleri tanımlandı.
* **Uzay araçları** için çıkış ve varış gezegenleri, çıkış tarihi, yolculuk mesafesi ve yolcular listelendi.
* **Kişiler** (yolcular) için kalan ömür hesaplamaları yapıldı.
* Zaman ilerlemesi her saat için sağlanarak:
  + Gezegenlerin tarihi güncellendi,
  + Yolcuların yaşam süresi azaltıldı,
  + Uzay araçlarının hareketleri ve varış kontrolleri gerçekleştirildi,
  + Nüfus değişimleri otomatik olarak yapıldı.
* Program süresince simülasyon konsola yazdırıldı.

**3. Karşılaşılan Zorluklar**

Proje sırasında aşağıdaki zorluklar yaşandı:

* **Zaman hesaplamalarında** (özellikle tarihe saat ekleme, iki tarih arasındaki saat farkı) taşmaları doğru yönetmek zorlayıcı oldu.
* **Ekran temizleme** işlemlerinde (cmd /c cls) Java'nın doğal desteği olmadığı için her iterasyonda yeni bir process açılması programı yavaşlattı.
* Çok fazla gezegen olduğunda konsol çıktısının **yatay taşması** ve okunamaz hale gelmesi problemi ortaya çıktı.
* **NullPointerException** hatası alındı; doğru veri eşlemesi ve sıralaması yapılarak sorun çözüldü.

Bu problemler üzerinde yoğun çalışılarak çözümler üretildi.

**4. Optimizasyonlar ve Geliştirmeler**

Programın performansını artırmak amacıyla aşağıdaki optimizasyonlar yapılmıştır:

* Gezegenlerin çıktısı **satır başına 5 gezegen** olacak şekilde düzenlenerek okunabilirlik artırıldı.
* Gereksiz new nesne oluşturmalardan kaçınıldı, Iterator kullanılarak güvenli koleksiyon işlemleri sağlandı.
* Süre ölçümü (System.currentTimeMillis()) ile programın çalışma süresi tespit edildi.

Bu optimizasyonlar sonucunda programın hem hızı hem de kullanıcı deneyimi büyük ölçüde iyileştirildi.

**5. Sonuç**

Bu proje sayesinde Java'da:

* Nesne yönelimli programlama (OOP) prensipleri,
* Koleksiyon yönetimi (ArrayList, Map kullanımı),
* Tarih ve zaman hesaplamaları,
* Performans optimizasyon teknikleri gibi konularda önemli bir deneyim kazanılmıştır.

Karşılaşılan problemler çözüldükçe hem kod kalitesi hem de kişisel tecrübe artmıştır.  
Projenin sonunda hem doğru çalışan hem de performanslı bir simülasyon ortaya çıkarılmıştır.