T.C.
SAKARYA ÜNİVERSİTESİ

BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

PROGRAMLAMA DİLLERİNİN PRENSİPLERİ ÖDEV RAPORU

Gezegenler Arası Seyahat Simülasyonu

B231210081 - Umut Arda VURAL

SAKARYA

Nisan, 2025

Programlama Dillerinin Prensipleri Dersi

Gezegenler Arası Seyahat Simülasyonu

Umut Arda Vural

[a] *B231210081 1/A*

Özet

Problemimiz, gezegenler arası seyahat etmesi planlanan uzay araçlarının belirlenen rotalar doğrultusunda güvenli şekilde varış noktalarına ulaşıp ulaşamadığının kontrol edilmesidir. Bu süreçte uzay araçlarının o anki durumu, konumu ve kalan seyahat süresi dikkate alınarak her bir gezegenin nüfus verilerinin doğru ve tutarlı biçimde güncellenmesi gerekmektedir. Uzay aracı imha olmuşsa ya da tüm yolcular hayatını kaybetmişse, bu durum kullanıcıya saat başı güncellenen kontrol ekranında anlık olarak gösterilmelidir. Her saat sonunda; araçların konumları, durumları, yaşam süresi olan yolcular ve gezegenler arasındaki mesafeler kontrol edilerek sistematik biçimde güncellenmelidir. Elde edilen tüm veriler kaydedilmeli ve sistemin son durumunu gösteren bir rapor oluşturulmalıdır.Bu ödevimizin , böyle bir simülasyon senaryosunu anlayıp modelleyebilmemiz için verilmiş olması muhtemeldir.

Anahtar Kelimeler: Tutarlılık , Güncellik , Doğruluk , Yönetilebilirlik

1.GELİŞTİRİLEN YAZILIM AŞAMALARI

1.1 Ödev nasıl gerçekleştirildi

Gerekli sınıf yapıları oluşturduğum projede, Gezegen, Kişi ve Uzay Aracı sınıflarında birer ArrayList tanımladım ve bu sınıflarda nesneleri sıralı bir şekilde tuttum. Dosya okuma sınıfında ise önce gezegenleri okuyarak gezegen listesine ekledim. Ardından, uzay araçlarını okuyarak her bir uzay aracının varış ve kalkış gezegenlerine referans ekledim. Son olarak, her bir uzay aracı nesnesini uzay aracı listesine ekledim. Kişileri okurken, her bir kişinin bulunduğu uzay aracı için referansları işleyerek nesneleri oluşturma işlemini de tamamladım. Dosyadan okuma işlemiyle birlikte, tüm verilerin doğru bir şekilde işlenmesini sağladım.

Zaman sınıfı ise her gezegen nesnesi için oluşturuldu simülasyon sınıfı içinde zaman sınıflarını fonksiyonları kullanıldı. Zaman sınıfı içinde gezegenin saat arttırma işlemleri gerçekleştirildi. Gerekli zamanlarda tarih arttırma işlemleride yapıldı. Console sınıfı ise jarı çalıştırırken gerekli ekran temizleme işlemini gerçekleştirdi.Main sınıfı ile program çalıştırıldı .

1.2 Hangi yöntem neden kullanıldı

1.2.1 ArrayList kullanımı

Bu projede verileri sıralı ve düzenli şekilde tutabilmek için ArrayList veri yapıları kullandım. Gezegen, Uzay Aracı ve Kişi sınıflarının her birinde ayrı bir ArrayList tanımlayarak, okunan verileri sıralı olarak listeledim ve gerektiğinde kolay erişim sağladım.

1.2.2 Nesne tabanlı programlama

Nesne tabanlı programlama yapısına uygun olarak, sınıflarımda get ve set metotları tanımladım. Bu sayede hem kapsüllemeyi sağladım hem de verilere kontrollü bir şekilde ulaştım. Bu yapı, uygulamanın okunabilirliğini ve yönetilebilirliğini artırdı.

1.2.3 Zaman değişkeni

Tarih işlemleri için LocalDate sınıfını tercih ettim. Bu sınıf sayesinde tarih karşılaştırmaları, formatlamaları ya da tarihsel sıralamaları daha kolay ve hatasız bir şekilde gerçekleştirebildim. Ayrıca, tarih üzerinde manuel işlem yapma ihtiyacını ortadan kaldırmış oldum.

1.2.4 Dosyadan okuma sınıfı

Dosya okuma işlemlerini yaparken, verileri doğru sırayla işleyebilmek adına önce gezegenleri okuyup listeye ekledim. Ardından uzay araçlarını okudum ve kalkış/varış gezegenleri daha önceden oluşturulan gezegen listesinden referans olarak uzay aracı nesnelerine bağladım. Bu, veri bütünlüğünü koruma mı sağladı.

Kişi verilerini işlerken de benzer bir yöntem izledim. Her kişinin bağlı olduğu uzay aracını, daha önce oluşturduğum uzay aracı listesi üzerinden bulup referans olarak bağladım. Böylece kişilerin konumlandığı uzay araçları ile bağlantıları doğru şekilde kuruldu.

1.2.5 Hata kontrolleri

Uygulamanın hatasız çalışmasını sağlamak adına, özellikle referans atamaları sırasında null kontrolü yaptım. Örneğin, bir uzay aracının kalkış ya da varış gezegeni ya da bir kişinin ait olduğu uzay aracı bulunamazsa, sistemin çökmesini engellemek amacıyla gerekli ön kontrolleri ekledim. Bu sayede hatalı ya da eksik veri girişlerinden kaynaklanabilecek olası NullPointerException gibi hataların önüne geçmiş oldum.

Tüm bu yöntemler, projenin sürdürülebilir, esnek ve güvenli bir yapıda ilerlemesini sağlamak amacıyla bilinçli olarak tercih edilmiştir.

1.2.6 Zaman sınıfı

Zaman sınıfını her gezegen nesnesi için gezegen nesnesi içinde yapıcı metod ile oluşturarak her gezegen için bağımsız bir zaman elde etmiş oldum . Zaman kontrolleri zaman sınıfı içinde yapılarak gerekli tarih işlemleri tamamlanmıştır.

1.2.7 Gezegen sınıfı

Gezegen sınıfı içerisinde gezegen ile ilgili verileri tuttum . Uzay araçları için kalkış ve varış gezegen nesnesini geri döndüren bir metod sayesinde aralarında bağlantı kurmuş oldum

1.2.8 Uzay araci sınıfı

İçerisinde gerekli alanları (değişken) tanımlamalarını yanında varış ve kalkış gezegeninin nesne referanslarını tuttum . Bu şekilde varış gezegenine kolayca erişmiş oldum . Yapıcı metod içinde varış tarihini hesaplayan metodum ile varış tarihini değişkene atama gerçekleştirildi.

1.2.9 Simülasyon ve main sınıfı

Tüm olayların gerçekleştiği sınıf yapısı .Sırası ile önce gezegenlerin zamanı artacak , kişileirn yaşları azalacak, araç kontrolü , konum işlemleri , gezegen nufusu ayarlama işlemleri yapıldı .Main sınıfı içinde bu işlemleri yapan eventSimülation fonksiyonundan sonra programın bitip bitmediği kontrol edildi bu işlemlerden önce her iterasyonda console ekranı temizlendi .

1.2.10 Varış tarihinin hesaplanması

Bu işlemleri yapabilmek için önce aracın kalkışa kadar bulunduğu gezegendeki geçen saati buldum.İkinci adımda yoldaki saati ile topladım. Son adımda varış gezegenine bu saatleri ekleyerek o gezegenin saatine göre varış tarihlerini belirledim.

1.3 Ödevin geliştirme amacı

Ödevimin amacı, gelecekte gelişen teknolojilerle insanların yeni gezegenlere seyahat etmesi ya da kolonileşme (yerleşme amaçlı konaklama) süreçlerini yönetecek bir kontrol uygulaması geliştirmektir. Bu uygulama, uzay teknolojileri ve biyoteknoloji alanlarındaki yenilikleri göz önünde bulundurarak, insanların gezegenler arası yolculuklarını organize etmek, klanolişme süreçlerini izlemek ve her iki sürecin güvenli bir şekilde yönetilmesini sağlamak amacıyla kullanılabilir. Bu sayede, uzaya seyahat eden kişilerin yeni gezegenlerde yerleşmeye çalışan bireylerin süreçlerinin doğru bir şekilde kontrol edilmesi ve yönetilmesi sağlanarak, gelecekteki uzay kolonizasyonu için sağlam bir altyapı oluşturulmuş olur.

2.ÇIKTILAR

Test amaçlı verilen verilerde, başlangıçta toplam 5 araç bulunmaktaydı. Bu araçlardan 4'ünün varıştan sonra imha edilmesi, son kalan 5. aracın da hedef gezegene ulaştığında programın sona ermesi bekleniyordu. Eldeki verilere uygun şekilde, tarihlerin tutarlılığını sağlayarak bu süreci başarıyla yönettim ve programı istenildiği gibi tamamladım.

3.SONUÇ

Bu proje, veri ilişkilerinin tutarlı, tarih bilgilerinin güncel ve sistemin genel yapısının doğru şekilde kurulması sayesinde başarıyla tamamlanmıştır. Kullanılan nesne yönelimli yapı, projenin yönetilebilirliğini artırmış ve sürdürülebilir bir mimari sunmuştur. Gerçek hayatta bu yapıların lojistik, ulaşım ve uzay teknolojilerinde karşılığı bulunmakta olup; bu projeyle bu sistemleri küçük ölçekte simüle etme fırsatı buldum. Kişisel olarak da algoritmik düşünme, hata ayıklama ve sınıflar arası ilişki kurma konularında kendimi geliştirdim.

* Ödev Sorumlusu. Umut Arda Vural B231210081,

Mail Adresi: umut.vural@ogr.sakarya.edu.tr