**T.C.**

**SAKARYA ÜNİVERSİTESİ**

**BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

PROGRAMLAMA DİLLERİNİN PRENSİPLERİ 2.ÖDEV RAPORU

**Gezegenler Arası Yolculuk Projesi (C)**

Nargiza Zhyrgalbek kyzy-G201210575

Ders Grubu : 2.Öğretim C Grubu

**ÖZET**

Bu proje, insanların uzay araçlarıyla farklı gezegenler arasında seyahat ettiği, dinamik bir uzay simülasyonu uygulamasıdır. Uygulama, Kisiler.txt, Araclar.txt ve Gezegenler.txt adındaki metin dosyalarından kişi, uzay aracı ve gezegen bilgilerini okuyarak simülasyonu başlatır. Simülasyon, her bir saatlik adımı bir döngü iterasyonu olarak kabul eder ve gezegenlerin türüne (Kayaç, Gaz Devi, Buz Devi, Cüce) göre kişilerin yaşlanma hızlarının farklılık göstermesi gibi gerçekçi unsurları içerir. Kalan ömrü sıfıra düşen kişiler ölür, tüm yolcuları ölen uzay araçları imha edilir. Program, tüm uzay araçları hedeflerine varana kadar çalışır ve bu süreçte ekranı sürekli temizleyerek anlık durumu günceller. Projenin odak noktası, C dilinde struct'lar ve fonksiyon işaretçileri kullanarak nesne yönelimli kalıtım ve polimorfizm benzetimlerinin nasıl yapılabileceğini göstermektir.

# ÖĞRENDİKLERİM

Projenin en kritik ve öğretici kısmı, Gezegen yapısını bir "temel sınıf" gibi tasarlayıp, KayacGezegen, GazDeviGezegen, BuzDeviGezegen ve CuceGezegen yapılarını bu temel sınıftan türetilmiş gibi modellemektir. Bunu başarmak için, türetilmiş yapıların ilk üyesi olarak temel Gezegen yapısını dahil ettim. Böylece türetilmiş yapıların işaretçileri, temel yapı işaretçisi gibi davranabilir. Polimorfik davranış için Gezegen yapısı içine bir fonksiyon işaretçisi ekledim. Her türetilmiş gezegen türü, kendi özel yaşlandırma fonksiyonunu bu işaretçiye atadı. Bu sayede, bir gezegen işaretçisi üzerinden genel bir yaslandir fonksiyonu çağrıldığında, C çalışma zamanında doğru türe ait özel fonksiyonun çalışmasını sağlayarak polimorfizmi simüle ettim. Bu yaklaşım, C'nin düşük seviye kontrol yeteneğiyle OOP'nin modülerliğini birleştirmeyi öğretti. Simülasyon sırasında kişi ve uzay aracı listelerinin dinamik olarak değişmesi (yeni kişilerin eklenmesi, ölen kişilerin çıkarılması, imha edilen araçların silinmesi) nedeniyle malloc, calloc ve free gibi fonksiyonları dikkatli bir şekilde kullanmayı öğrendim. Bellek sızıntılarını ve geçersiz bellek erişimlerini önlemek için her tahsis edilen belleği doğru zamanda ve doğru şekilde serbest bırakmanın kritik olduğunu pekiştirdim. Özellikle dinamik listelerden eleman silme işlemlerinde pointer yönetiminin inceliklerini anladım.

# YAPTIKLARIM VE ZORLANDIĞIM KISIMLAR

İstenilenler dahilinde tüm komutları yerine getirebildim ve ekrana gerekli işlemleri doğru şekilde yazdırdım. Gerçek takvim kurallarına uygun doğru ve tutarlı tarih/saat ilerletme ve gezegenlerin kendi iç saatleriyle global simülasyon saati arasındaki senkronizasyonu sağlamak detaylı ve hatasız bir mantık inşası gerektirdi. Simülasyonun dinamik doğası (kişilerin ölmesi, uzay araçlarının imha edilmesi) nedeniyle malloc ve free işlemlerini doğru ve hatasız bir şekilde yaparak bellek sızıntılarını ve hatalı erişimleri önlemek kritik bir noktaydı. Projeyi işlevsel ve bağımsız modüllere ayırmak, her modülün sorumluluğunu net bir şekilde belirlemek ve modüller arası bağımlılıkları minimize etmek, büyük bir kod tabanının organizasyonu açısından önemliydi.