**ROBOT SİSTEMİ PROJESİ**

Erdem Özer - Muhammed Kurfeyiz

Bilgisayar Mühendisliği Bölümü  
Kocaeli Üniversitesi

[erdemcemozer@hotmail.com](mailto:erdemcemozer@hotmail.com), [m.kurfeyiz98@gmail.com](mailto:m.kurfeyiz98@gmail.com)

**Özet**

Robot sistemi sayesinde, kullanıcının belirlediği robotlar ve özellikleri kullanarak farklı problemler çözmeye yarayan bir uygulamadır.

**1. Giriş**

Robot sistemi aralarında hiyerarşik bir bağ bulunan ve birbirleri ile ilişkilendirilen nesnelerin Java dilinin özellikleri olan Polymorphism ve Inheritance kullanarak oluşturulan robotlarla problem çözmeye yarar.

Grafik kısmı sayesinde robotların konumlarını çok rahat görebileceğimiz bir uygulamadır.

**2. Temel Bilgiler**

Bu proje geliştirilirken “NetBeans” Tümleşik Geliştirme Ortamı (IDE) kullanılmıştır.

**3. Tasarım**

Robot sistemi için hazırlanan bu projenin programlanma aşamaları aşağıda belirtilmiştir.

**3.1 Hiyerarşik Yapı**

Robot sisteminde oluşturulabilir 6 farklı temel sınıf bulunmaktadır. Bu temel sınıfların birbirleri ile ortak özelliklerini belirleyen, bir tanesi interface bir tanesi class olmak üzere iki adet yapı vardır. Tüm sınıfların ortak özelliklerini içeren temel ata “Robotlar” sınıfı bulunmaktadır.

Oluşturulup, özellikleri girilebilen “Spider, Tekerlekli, Paletli, Seri, Paralel ve Hibrit” sınıfları, bu sınıfların ortak özelliklerini oluşturan “Gezgin Robotlar ve Manipülatör Robotlar(Interface)” ve tüm sınıfların atası olan “Robotlar” yapısı fotoğraflar(Resim 6.1) kısmındaki hiyerarşik yapıyı kullanmaktadır.

**3.2 Yöntemler**

Projede kullanılan fonksiyonlar aşağıda açıklanmıştır.

ileri(int y) : Bu fonksiyon robotun konumunu birimler şeklinde ileri taşır.

geri(int y) : Bu fonksiyon robotun konumunu birimler şeklinde geri taşır.

saga(int x) : Bu fonksiyon robotun konumunu birimler şeklinde sağa taşır.

sola(int x) : Bu fonksiyon robotun konumunu birimler şeklinde sola taşır.

sonKonum() : Robotun hareketi bittikten sonraki konumunu kullanıcıya döndürür.

sureyiGoster(int mesafe, String robot) : “mesafe” değerini alarak robotun bütün hareketinin süresini bulur.

EngelGecmeSuresiBul() : Gezgin robotların engel üzerlerinden geçme zamanını hesaplar, spider robot ise engelden geçemez.

**3.3 Sınıfların Özellikleri**

Robotlar (Interface) : Tüm sınıfların atasıdır; ileri, geri, sağa, sola, sonKonum ve sureyiGoster methodları bodysiz tanımlanmıştır.

Gezgin\_Robotlar(Class) : Spider, tekerlekli, paletli ve hibrit robotların atasıdır. Bu sınıfların ortak özellik ve fonksiyonların bodyleri burada tanımlanmıştır.

Manipulator\_Robotlar(Interface): Seri, paralel ve hibrit robotların atasıdır. Bu sınıfta fonksiyonlar bodysiz olarak tanımlanmıştır.

Spider(Class) : Gezgin Robotlardan miras aldığı fonksiyon ve değişkenler bu sınıfın constructorında kullanılır.

Tekerlekli(Class) : Gezgin Robotlardan miras aldığı fonksiyon ve değişkenler bu sınıfın constructorında kullanılır.

Paletli(Class) : Gezgin Robotlardan miras aldığı fonksiyon ve değişkenler bu sınıfın constructorında kullanılır.

Seri(Class) : Manipülatör Robotlardan miras aldığı fonksiyon ve değişkenler bu sınıfın constructorında kullanılır.

Paralel(Class) : Manipülatör Robotlardan miras aldığı fonksiyon ve değişkenler bu sınıfın constructorında kullanılır.

Hibrit(Class) : Manipülatör ve Gezgin Robotlardan miras aldığı fonksiyon ve değişkenler bu sınıfın constructorında kullanılır.

**4. Karşılaşılan Sorunlar ve Çözümleri**

1. Çoklu kalıtım sağlanması için Gezgin robotları class olarak tanımlarken, Manipülatör robotları interface olarak tanımladık.

2. Abstraction kullanmadığımız için methodlar alt sınıflarda sorun çıkardı bunun için hepsini tek tek override ettik.

**5. Kazanımlar**

Robot sisteminin bize kazandırdıkları;

* Interface kullanımı
* Çoklu kalıtım
* Override kullanımı

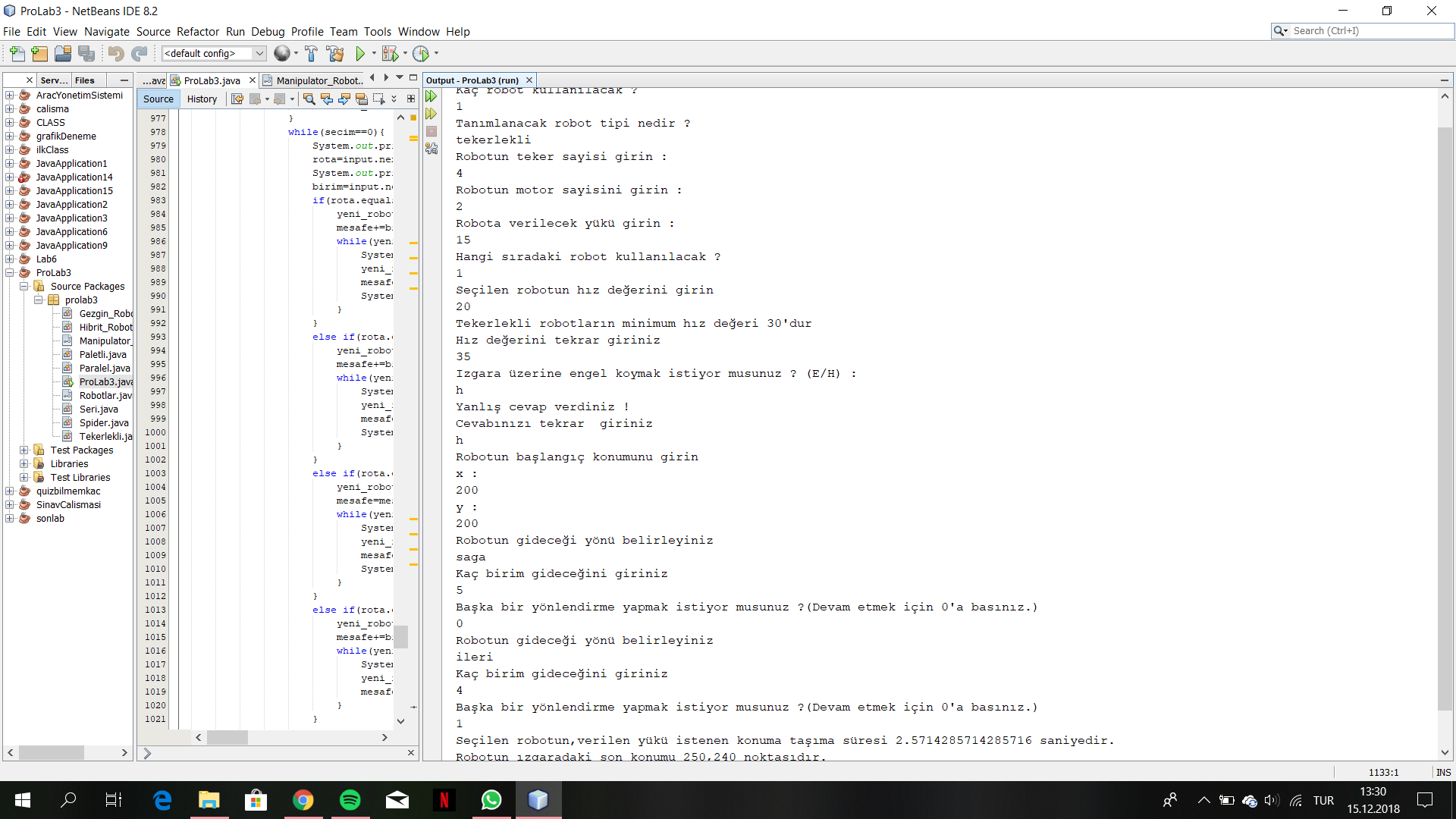
**6. Çalışma Mantığı**

Kullanıcının seçtiği sayıda ve yine kullanıcı tarafından belirlenen özelliklerle robotlar oluşturan uygulama, daha sonra yine kullanıcının kendi oluşturduğu robotlardan birini seçtirerek belirli 3 problemden birini çözer. Kullanıcı seçtiği robot türüne göre farklı özellikler girer ve bu robotların arasından kullanmak istediğini seçerek programa devam eder.

**7. Fotoğraflar**

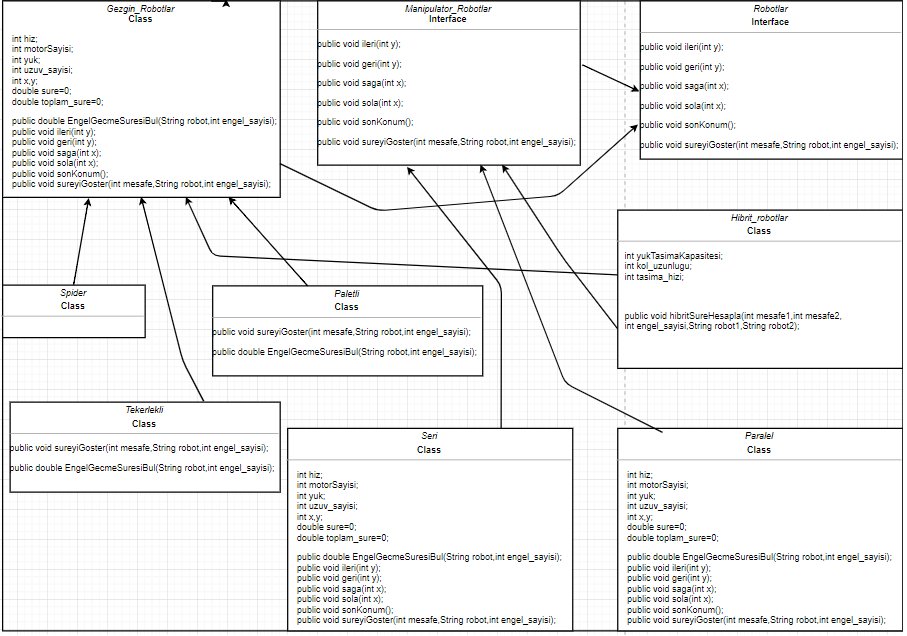
****

7.1 Robotların hiyerarşik yapısı



7.2 Ekran çıktısı

**8. UML Diyagramı**

****

**9. Kaba kod**

1- BAŞLA

2- ROBOT SAYISI SEÇ

3- ROBOT TÜRLERİNİ SEÇ

4- SEÇİLEN ROBOTUN ÖZELLİKLERİNİ VE YÜK MİKTARLARINI GİR

5- EĞER ENGEL KOYULACAKSA SAYISINI VE YERİNİ SEÇ

6- ROBOTUN VEYA KOLLARIN HANGİ YÖNE KAÇ BİRİM GİDECEĞİNİ GİR

7- EĞER ROBOT TÜRÜ YADA ÖZELLİKLERİ SEÇİLEN YÖNLER İÇİN ELVERİŞLİ İSE ROBOTUN SON KONUMU VE SON KONUMA GİDERKEN HARCADIĞI SÜRE DÖDÜRÜLÜR

8- EĞER DEĞİLSE ROBOT ENGEL ÖNÜNDE DURUR YADA KOL UZUNLUĞU YETMEZ VE HATA VERİR

9- BİTİR

**10. Kaynakça**

**-** [**https://www.tutorialspoint.com/java/java\_object\_classes.htm**](https://www.tutorialspoint.com/java/java_object_classes.htm)

**-** [**https://www.yusufaytas.com/inheritance-ve-polymorphism/**](https://www.yusufaytas.com/inheritance-ve-polymorphism/)

**-** [**http://www.onurcanalp.com/2014/08/21/encapsulation-inheritance-ve-polymorphism/**](http://www.onurcanalp.com/2014/08/21/encapsulation-inheritance-ve-polymorphism/)

**-** [**https://www.mobilhanem.com/java-abstraction/**](https://www.mobilhanem.com/java-abstraction/)

**-** [**https://www.tutorialspoint.com/java/java\_interfaces.htm**](https://www.tutorialspoint.com/java/java_interfaces.htm)