# KOCAELİ ÜNİVERSİTESİ

# BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

# BLM209 PROGRAMLAMA LAB. I

# PROJE 3

180202116 Mert BİLGİÇ [bilgic.mert44@gmail.com](mailto:bilgic.mert44@gmail.com)

ÖZET

Programımız kullanıcıdan öncelikle kaç tane robot girmek istediği sormaktadır devamında tanımlanacak robot sayısı tipini istemektedir.Bunun devamı kullanıcının seçtiği robot tipine göre şekillenmektedir.Kullanıcı hangi robot tipini seçerse robot için gerekli olan bilgiler kullanıcıdan alınmaktadır.Devamında ise hangi robot ile işlem yapmak istediği istenmektedir.İşlem yapmak istediği robotun türüne göre gerekli işlemleri yapılır.Eğer kullanıcı hibrit robotu seçtiyse aynı zamanda daha öncede Gezgin robotlardan girdiği bir robot türünü girerse kullanıcıdan tekrar ilgili gezgin robot hakkında bilgi girmesi istenmemiştir.İlerleyen kısımda kullanıcıda kısımlarda kullanıcıdan engel sayısı engel koordinatları istenmektedir.Engel sayısının girişi bittiğinde robotun konumu alınır ve robotun hareket işlemleri yapıldıktan sonra sonuç harita üzerinde gösterilir.

***Anahtar Kelimeler: Robot,Engel,Konum***

GİRİŞ

Bu şablon MS-WORD 2007 ile hazırlanmış ve MS-Word 2007 dosyayı olarak kaydedilmiştir.Kenar boşlukları sütün genişlikleri satır aralıkları ve stilleri belgenin içine gömülüdür.

Bu projenin genel amacı nesneye dayalı programlama yapısını öğretmektir.Projede gezgin manipülatör ve hibrit olmak üzere üç tip robot vardır.Bu robotlarda kendi içinde dallanmaktadır.Gezgin robotlar tekerlekli paletli ve spider olmak üzere üçe ayrılır.Manipülatör robotlar kendi içinde seri ve paralel olmak üzere ikiye ayrılır.Hibrit robotlarda manipülatör ve gezgin robotların özelliklerini taşımaktadır.Bu projedeki amacımız bu robotların özelliklerini kullanarak nasıl davrandıkları hangi durumlarda nasıl tepki verdiğini ölçmektir.Robot adı altında oluşturduğumuz sınıfın içinde tüm robotların ortak özellikleri bulunmaktadır.Bu Robot sınıfın içindeki özellikler ve fonksiyonlar GezginRobotlar ve Manipülatör robotlara kalıtım yöntemiyle aktarılmıştır.Daha sonra bu sınıfları da kendi alt sınıflarına bu özellikleri kalıtım yoluyla aktarmıştır.Genel yapı bu şekilde oluşturulduktan sonra robotları oluşturmak için RobotOlustur adlı bir sınıf oluşturulmuştur.Robotlarla alakalı bilgiler bu sınıfın içinde verilen cevaplara uygun şekilde ilerleyerek kullanıcıya aktarılmıştır. Kullanıcıya işlem yapmak istediği robot sorulduktan sonra kullanıcıdan engel sayısı istenmiş ve istenen engel sayısına uygun şekilde kullanıcıdan engel koordinatları alınmıştır.Robotun konumu alındıktan sonra kullanıcı robotu yönlendirmiştir.Girilen bilgilere uygun görsel ekran kullanıcıya aktarılarak işlem sonlandırılmaktadır.

YÖNTEM

Bu kısımda izlediğimiz yolu ve çalışmalarımızı anlatmaktayız.

Projede izlediğimiz yolu özetlemek gerekirse öncelikli amacımız projeyi nesneyi dayalı programlama yapısına uygun şekle getirerek en verimli şekilde kod yazmaktır.

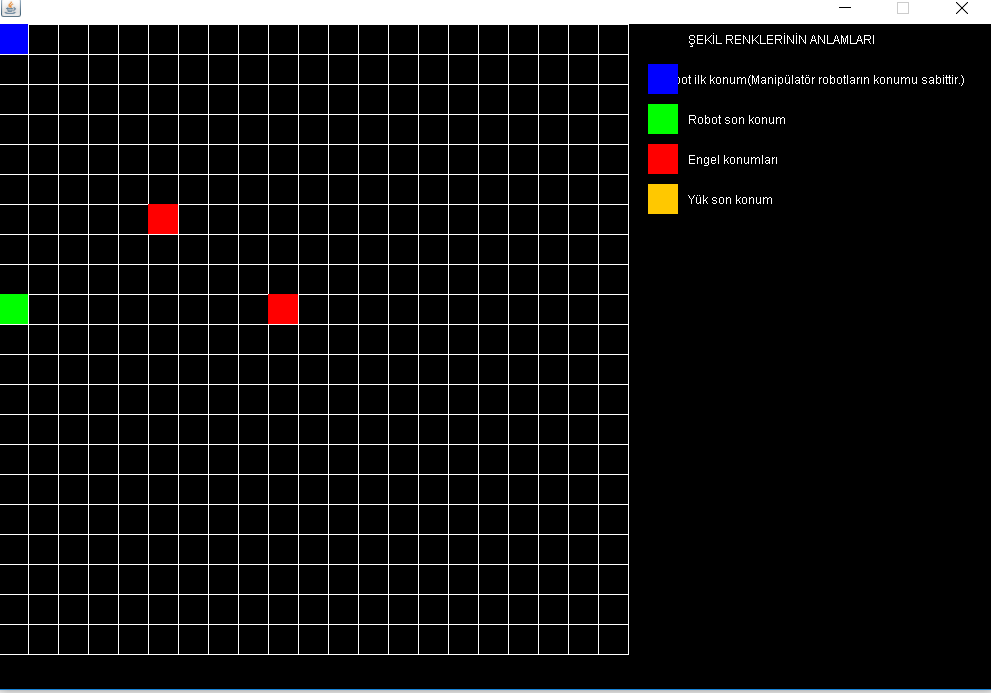
İzlediğimiz yolu sırasıyla şu şekilde aktarabiliriz:

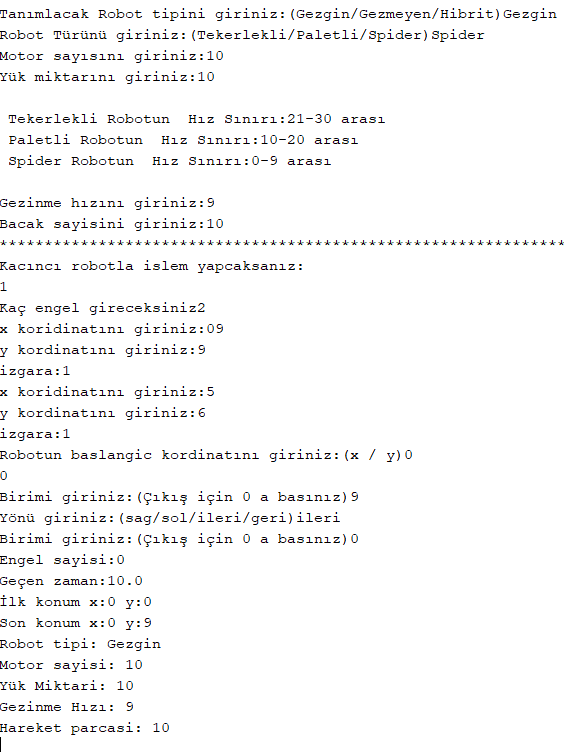
Projeyi inceledikten sonra genel yapıyı oluşturduk.Genel yapıyı oluşturduktan sonra verileri main içinden almak yerine bir sınıfın içinde almaya karar verdik bunu yapma amacımız nesneye dayalı programlamayı daha iyi bir şekilde öğrenmekti.Bu yapıyı kullanmak bize nesneye dayalı program yapısı hakkında bir çok bilgi edinmemizi sağladı.Daha sonra uygulayacağımız fonksiyonları verilim bir şekilde sınıfların içine yazmaya çalıştık.Gezgin robotlar için yazdığımız fonksiyonları tekerlekli spider paletli ve hibrit robotun kullanabileceği şekilde yazdık.Manipülatör robotlar robotlar içinde aynı işlemi uygulamaya çalıştık.Hibrit robotlar içinde iç extends yapamadığımız için Gezmeyen robotlar için interface oluşturup bunu implements ederken Gezgin robotları içine extends ettik en sonunda gerekli değerler main için aktarıldı ve seçilen robotun işlemleri başlıyacak şekilde ayarlandı.

SONUÇ

Sonuç olarak projede nesneye dayalı programlamanın temeli işlemleri öğrenildi aynı zamanda verilerin görsel ekranı hakkında kullanıcıya nasıl aktarılması gerektiği öğrenildi

DENEYSEL SONUÇLAR





Deneysel sonuçta görüldüğü gibi robotların hız sınırı olduğu alınan engel koordinatlarının harita üzerinde yerleştirilebildiği aynı zamanda robotun robottun konumundaki değişiklik gözlemlenebilir.Robotun herhangi bir engele takılıp takılmadığını değerlerinin aktarıldığını gözlemleyebiliriz.

KABA KOD

Bu kısımda programın algoritması anlatılmaya çalışılmıştır

Kullanıcıdan veri alınmaya başladığı kısımda başlarsak kullanıcıdan robot sayısına göre bir for döngüsü olusturulur daha sonra istenilen tip sorulup gerekli if(koşul) koşuluna girilir.İf koşulu içinde her robotta ortak olan motor sayisi robot tipi yük miktarı kullanıcıdan alınır.Eğer seçilen robot Gezgin ise gezin koşulunun içinde atamalar yapılır atamaların sonucunda hareket ettirilmek istenilen robot kullanıcıdan alınır.Bu değerler main in içine döner oluşturulmak istenilen if(koşul) objenin içine girer ver.Bu veriler RobotOlustur sınıfın içinde yapılır.

Üst kısımda kullanıcıdan alınan veri ve bu verinin işlemesi anlatılmıştır.

public double EngeldenGecmeSüresiniBul() fonksiyonunda devam edersek paletli ve tekerlekli içinde bu fonksiyonu kullanırız bunu gönderilen nesnenin robot tipine bakarak yaparız. if(getRobot\_tipi().equals("Tekerlekli")) veya else if(getRobot\_tipi().equals("Paletli")) şeklinde belirlenilen koşullara girer eğer dönen robot spider olursa else return 0 şeklinde Maine dönüş sağlanır.

Gezgin robotun bir diğer önemli fonksiyonu kullanıcının harita üzerindeki engel ve robotları belirlediği kısımdır.Bu kısım şu şekilde yapılmıştır.

Kullanıcıdan engel sayisi alındıktan sonra for(int i=0;i<engel\_s;i++) şeklinde döngü oluşturulup x ve y koordinatları kullanıcıdan alınmıştır alınan bu koordinatlar izgara[x1[i]][y1[i]]=1; şeklinde bir değişkenin içinde 1 olarak belirlenmiştir.Burda bir atanmasının sebebi izgara değişkenin içindeki tüm değerler 0 iken engel olan indislerdeki değer 1 olucak bu şekilde haritanın içinde bilgiler tutulmuş oluyor.Robotun ilk konumu ilkx=x2 ilky=y2; şeklinde değişkenlere atılıyor bunu yapmamızın sebebi devamında robotun başlangıç konumunu kaybetmemek. System.out.print("Yönügiriniz:(sag/sol/ileri/geri)"); System.out.print("Birimi giriniz:(Çıkış için 0 a basınız)"); ile değerler alınıp daha sonra if(yön.equals("sag")){for(;temp>0;temp--){if(izgara[x2][y2]==1){ if(getRobot\_tipi().equals("Spider")){System.out.println("Spider robot engel geçemez engele takıldınız");c++;x2-=1;break;}engel++;} şeklindeki bir kod parçası ile robot hareket ettirilmiş eğer spider ise döngü sonlandrılarak görsel ekrana geçilir.

Manipülatör robotlarda yine gezgin robotlarda işlemlerin bir benzeri yapılır burada yükü bırakırken Pisagor uygulayıp uzaklığı öyle kontrol ederiz if(yön.equals("sag")){ if(x1<=getKol\_uzunluğu()){ x1+=temp;}else{System.out.println("Kol uzunlugunun dışına çıktınız"); }}

HibritRobot ta ise gezgin robotlar extends gezmeyen robotlar implements edilmişt durumdadır.HibritRobotlar işlemin bir kısmını gezgin robotların içindeki iki fonksiyonu kullnarak yapmaktadır

public static void IzgaraHaraketleri()

public double EngeldenGecmeSüresiniBul()

buradaki işlemi bittikten sonra Izgaraharaketleri fonksiyonunun kaldığı yerden manipülatör robot olarak devam etmesi sağlanır.Bunun için

public static void uzaklıkKontrol()

fonksiyonunu kullanır.Son durum ekranda görsel ile gösterilir.

KAYNAKÇA

Sıfırdan İleri Seviyeye Komple Java Geliştirici Kursu (Mustafa Murat Çoşkun)

Sıfırdan İleri Seviyeye Uygulamalı Java Kursu(Emre Altunbilek)

<https://www.tutorialspoint.com/java/java_inheritance.htm>

http://www.baskent.edu.tr/~tkaracay/etudio/ders/prg/java/ch15/inheritance.htm

