**ARAÇ YÖNETİM SİSTEMİ**

Ali Recep KARACA, Mehmet Fırat KÖMÜRCÜ

Bilgisayar Mühendisliği Bölümü

[recepkaraca@yandex.com](mailto:recepkaraca@yandex.com) , [mehmetfiratkomurcu@hotmail.com](mailto:mehmetfiratkomurcu@hotmail.com)

*Bu proje; hava, kara ve su taşıtları ile taşımacılık sektöründeki firmalar için kolay bir yönetim sistemi sunmaktadır. Proje içerisinde yeni araç ekleme, var olan araçlar üzerinde özellik güncellemesi yapabilme, var olan tüm araçların mevcut durumlarını listeleme gibi özellikler bulunmaktadır. Ayrıca, program içerisinde var olan tüm araçların bilgileri bir dosyaya yazılarak, bu dosya üzerinden de var olan tüm araçların mevcut durumları kontrol edilebilmektedir.*

**1. Problemin Tanımlanması**

Bu projede çözümlenmesi gereken sorun; araçlar üzerine çalışan firmalar için kolay bir yönetim sistemi sunacak program geliştirilmesidir. Bu program içerisinde mevcut araçların güncellenebilir, tüm araçların listelenebilir ve yeni bir araç girişi yapılabilmektedir. Tüm bu işlemlerin yanında, ayrıca tüm bu araç verilerini bir dosyada toplayan ve programı kullanan kişinin, mevcut tüm araçların son durumlarını inceleyebileceği bir dosya kayıt sistemi bulunmaktadır. Bu sistem sayesinde var olan tüm araçların bilgileri toplu bir şekilde kullanıcıya sunulmaktadır.

**2. Giriş**

Verilen projenin konusu; bir araç yönetim sistemi geliştirilmesidir. Bu sistem içerisinde tüm araçları listeleme, mevcut araçların özelliklerini değiştirme ve yeni araç ekleme gibi özellikler de programa dahil edilmelidir. Bunların yanında tüm bu araçların verileri bir dosyada tutulmalıdır.

Verilen projenin amacı; araç yönetim sistemi uygulaması geliştirilerek, bu proje içerisinde olan araç, kara taşıtları, hava taşıtları, deniz taşıtları, otomobil, bisiklet, gemi, uçak, uçan gemi isimli tüm sınıfların yazılarak, proje içerisinde encapsulation, inheritence, polymorphism, abstraction yapılarının kullanılmasını sağlayarak, öğrencilerin nesneye yönelik programlama konusunda tecrübe kazanmasını ve bu konuyu bir uygulama üzerinde deneyimleyerek konuyu pekiştirmelerini ve anlamalarını sağlamaktır.

**3. Temel Bilgiler**

Projeyi gerçekleştirme aşamasında Java programlama dili kullanılmıştır. Ayrıca; yapmış olduğumuz proje; Java programlama dili kullanılarak nesneye yönelik programlama mantığında oluşturulmuştur.

Yazılan bu program Windows 10 üzerindeki NetBeans 8.2 derleyicisi üzerinde derlenmiş olup, derleme aşamasında herhangi bir hatayla karşılaşılmamıştır. Ayrıca derlenen bu program yine Windows 10 üzerinde çalıştırılmış ve programın çalışması sırasında yine herhangi bir hatayla karşılaşılmamıştır.

**4. Genel Yapı**

Projede ilk olarak “Arac” isimli bir interface oluşturulmuştur. Bu interface içerisine “getMarka”, “setMarka”, “getHiz”, “getRenk”, “setRenk”, “getYolcuSayisi”, “setYolcuSayisi”, “getTekerlek sayisi”, “setTekerlekSayisi”, “getUretimYili”, “setUretimYili”, “getFiyat”, “setFiyat”, “hizlan”, “yavaşla”, “dur” isimli fonksiyonlar tanımlanmıştır. Burada; Arac class’ı bir interface olduğundan dolayı, içerisindeki tüm fonksiyonların bodyleri boş bırakılmış, kendisini kalıtan tüm sınıfların bu boş body içeren fonksiyonları doldurmaları sağlanmıştır. Ayrıca bu “Arac” isimli sınıfın interface yapılmasını gerektiren sebep; proje içerisinde kullanılacak olan çoklu kalıtım probleminde, Java programlama dili kullanılarak birden fazla sınıf kalıtılamayacağından dolayı bazı sınıfların interface yapılması gerektiğindendir.

Ardından; “KaraTasitlari” isimli bir “abstract” biçiminde sınıf daha oluşturulmuştur ve bu sınıf “Arac” isimli sınıfı implement etmektedir. Burada, “KaraTasitlari” sınıfının abstract yapılmasındaki amaç; “KaraTasitlari” sınıfının kullanılarak herhangi bir nesne üretilmemesini sağlamaktır. Bu sınıf içerisinde, Java programlama dilindeki interface kullanım şeklinin bir parçası olarak, “Arac” isimli sınıf içerisindeki tüm body içermeyen fonksiyonlar override edilmiştir. Ayrıca projede “KaraTasitlari” sınıfı için istenen motorlu/motorsuz taşıt sınıflandırması da bir değişken yardımıyla tamamlanmıştır. Bu değişkenin get, set metotları yazılmıştır. Ayrıca bu sınıf içerisinde, birisi varsayılan yapılandırıcı olmak üzere iki farklı yapılandırıcı eklenmiştir.

“DenizTasitlari” sınıfı da, “KaraTasitlari” sınıfında olduğu gibi abstract bir sınıf olarak tanımlanmış ve “Arac” isimli sınıfın kalıtılması adına “Arac” sınıfı implement edilmiştir. Yine bu sınıfta; “KaraTasitlari” isimli sınıfta da olduğu gibi, Java programlama dilinde interface kullanımı gereği, implement ettiğimiz sınıf olan “Arac” isimli sınıfın tüm body içermeyen fonksiyonları bu sınıf içerisinde de override edilmiştir. Ayrıca bu sınıf içerisinde, birisi varsayılan yapılandırıcı olmak üzere iki farklı yapılandırıcı eklenmiştir.

“HavaTasitlari” sınıfı ise, “UcanGemi” isimli sınıfın hem “HavaTasitlari” hem de “DenizTasitlari” isimli sınıfları kalıtıyor olması gerektiğinden dolayı, Java programlama dilinde de çoklu kalıtım desteklenmediğinden, ayrıca “DenizTasitlari” isimli sınıfın da normal bir sınıf olarak tanımlanmış olmasından dolayı, bir interface sınıf olarak tanımlanmıştır. Bu sınıf, projede bizlerden istenen kıstaslara uygun olarak “Arac” isimli sınıfı extend etmektedir. Ayrıca bu sınıf içerisinde, yine projede bizlerden istenen, uçaklara özgü bir özellik olan, uçağın inip inmediğini kontrol etmesi adına “inis”, “indiMi” isimli fonksiyonlar tanımlanmıştır. Java programlama dilindeki interface yapısı gereği, tanımlanan bu fonksiyonlar body içermemektedir. Bu fonksiyonları bodyleri, yine bu interface sınıfını kalıtan nesneler tarafından tanımlanacaktır.

“Otomobil” isimli sınıf, projede bizlerden istenen kıstaslara uygun olarak, “KaraTasitlari” isimli sınıfı kalıtmaktadır. Kalıtımı yapılan “KaraTasitlari” sınıfı bir abstract sınıf olduğundan dolayı, “KaraTasitlari” sınıfı içerisinde abstract biçiminde tanımlanan tüm fonksiyonlar “Otomobil” sınıfı içerisinde de tanımlanmıştır. Bu sınıf içerisinde, “Bisiklet” isimli sınıftan farklı olarak “getYakitTuru” ve “setYakitTuru” adında iki fonksiyon bulunmaktadır. Bu fonksiyonlar, otomobillerin yakıt içermesinden dolayı “Otomobil” sınıfı içerisinde tanımlanmışlardır. Ayrıca bu sınıf içerisinde, birisi varsayılan yapılandırıcı olmak üzere iki farklı yapılandırıcı eklenmiştir.

“Bisiklet” isimli sınıf, yine “Otomobil” isimli sınıfta olduğu gibi “KaraTasitlari” isimli sınıfı extend etmektedir. Kalıtımı yapılan “KaraTasitlari” sınıfı bir abstract sınıf olduğundan dolayı, “KaraTasitlari” sınıfı içerisinde abstract biçiminde tanımlanan tüm fonksiyonlar “Bisiklet” sınıfı içerisinde de tanımlanmıştır. Bu sınıf içerisinde, “Otomobil” sınıfından farklı olarak, “getYakitTuru” ve “setYakitTuru” fonksiyonları bulunmamaktadır. Bunun sebebi; bisiklet isimli aracın motorsuz bir araç olmasından kaynaklanmaktadır. Ayrıca bu sınıf içerisinde, birisi varsayılan yapılandırıcı olmak üzere iki farklı yapılandırıcı eklenmiştir.

“Gemi” isimli sınıf, projede bizlerden istenen kıstaslara uygun olarak, “DenizTasitlari” isimli sınıfı extend etmektedir. Kalıtımı yapılan “DenizTasitlari” sınıfı bir abstract sınıf olduğundan dolayı, “DenizTasitlari” sınıfı içerisinde abstract biçiminde tanımlanan tüm fonksiyonlar “Gemi” sınıfı içerisinde de tanımlanmıştır. Ayrıca bu sınıf içerisinde, birisi varsayılan yapılandırıcı olmak üzere iki farklı yapılandırıcı eklenmiştir.

“Ucak” isimli sınıf, “HavaTasitlari” isimli sınıfı implement etmektedir. Java programlama dili yapısı gereği, “HavaTasitlari” isimli interface içerisinde tanımlanan tüm bodysiz fonksiyonların bodyleri “Ucak” sınıfı içerisinde doldurulmuştur. Ayrıca yine, “HavaTasitlari” interface sınıfı içerisinde bulunan abstract fonksiyonlar da “Ucak” isimli sınıf içerisinde tanımlanmıştır.

“UcanGemi” isimli sınıf, hem “HavaTasitlari” isimli sınıfı, hem de “DenizTasitlari” isimli sınıfı içermesi gerekmektedir. Bunun sebebi; projede bizlerden “UcanGemi” isimli sınıfın çoklu kalıtım gerçekleştirmesi gerektiğinin belirtilmenden kaynaklanmaktadır. Bunun için; “UcanGemi” isimli sınıf, “DenizTasitlari” isimli sınıfı extend etmekte, ayrıca “HavaTasitlari” isimli interface sınıfı da implement ederek çoklu kalıtım gerçekleştirilmiştir. Bunun bu şekilde yapılmasının sebebi; Java programa dilinin çoklu kalıtımı bu şekilde desteklemesinden kaynaklanmaktadır. Ayrıca bu sınıf içerisinde, birisi varsayılan yapılandırıcı olmak üzere iki farklı yapılandırıcı tanımlanmıştır.

Son olarak; “VehicleManagementSystem” isimli sınıf içerisinde, “karaTasitlar”, “denizTasitlar” ve “havaTasitlar” isimlerindea arraylistler oluşturulmuştur. Ardından, bilgisayar üzerinde “AracKayit.txt” isimli bir dosya olup olmadığı kontrol edilmiştir. Eğer bilgisayar üzerinde böyle bir dosya yoksa ilk olarak oluşturulması sağlanmıştır. Ardından, kullanıcıdan yapmasını istediği işlem girişi alınmaktadır. Burada kullanıcı yeni veri girişi yapabilmekte, tüm mevcut araçları listeleyebilmekte ve mevcut olan araçlar üzerinde özellik güncellemesi yapabilmektedir. Bu kullanıcı işlem seçimi sırasında, eğer “AracKayit.txt” isimli dosya içerisine daha önceden herhangi bir araç kayıt edilmişse, bu dosya içerisinde bulunan tüm araçlar önce program içerisine aktarılmakta olup, aktarılan tüm bu araçların birer nesneleri oluşturulmaktadır.

Eğer kullanıcı işlem olarak veri girişini seçerse, yine bu sınıf içerisinde bulunan “VeriGirisiListesi” isimli fonksiyon çalışmaktadır. Bu “VeriGirisiListesi” isimli fonksiyon içerisinde ilk olarak, kullanıcıya veri girişi yapacağı araç türünü seçmesi istenmektedir. Bunlar; otomobil, bisiklet, gemi, uçak ve uçan gemi araçlarından herhangi biri olmak zorundadır. Kullanıcı araç türü seçimini yaptıktan sonra, ilgili aracın nesnesinin yaratılabilmesi için gerekli olan tüm veriler kullanıcıdan istenmektedir. Kullanıcının girmiş olduğu veriler ışığında, oluşturulmuş olunan nesneler, “karaTasitlar”, “denizTasitlar”, “havaTasitlar” isimli arraylistlerden ilgili olanına aktarılmakta olup burada, polymorphism özelliğinden faydalanılmaktadır.

Eğer kullanıcı işlem olarak veri listelemeyi seçerse, yine bu sınıf içerisinde bulunan “VeriListele” isimli fonksiyon çalıştırılmaktadır. Bu “VeriListele” isimli fonksiyon içerisinde ilk olarak, kullanıcıya listelenmek istenilen veri tipi sorulmaktadır. Bunlar; “Kara Taşıtları”, “Deniz Taşıtları” veya “Hava Taşıtları” seçeneklerinden birisi olmalıdır. Kullanıcı listelenmesini istediği veri tipi seçimini yaptıktan sonra, ilgili veri tipine uygun olarak, içerisinde oluşturulan nesneler bulunan arraylistler içerisinden veriler çekilerek ekranda basılmakta olup, kullanıcının ilgili araçlar üzerindeki tüm detayları görmesi sağlanmaktadır.

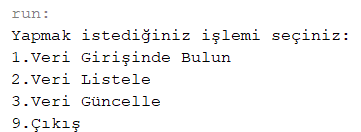
Eğer kullanıcı işlem olarak veri güncellemeyi seçerse, yine bu sınıf içerisinde bulunan “VeriGuncelle” isimli fonksiyon çalıştırılmaktadır. Bu “VeriGuncelle” isimli fonksiyon içerisinde ilk olarak, kullanıcıdan güncellemek istediği aracın “ID” numarasını girmesi istenmektedir. Kullanıcı araç “ID” numarasını girdikten sonra bu numara kontrol edilmekte olup, eğer bu “ID” numarasına sahip herhangi bir araç bulunamazsa kullanıcıya bir uyarı mesajı verilmektedir. Eğer kullanıcının girmiş olduğu bu “ID” numarası gerçek bir numaraysa ve bir araca aitse, ilk olarak bu “ID” numarasının hangi araç tipinde olduğu tespit edilmektedir. Ardından, bulunan bu araç tipiyle alakalı, güncellenebilecek veriler ekranda listelenmektedir ve kullanıcı bu verilerden istediği kadarını değiştirebilir. Burada, her araç tipine göre ekrana farklı menüler basılmaktadır.

Programın en son aşamasında ise, oluşturulan, güncellenen tüm bu araç nesnelerinin verileri “AracKayit.txt” isimli dosyaya yazılmakta olup, programın daha sonraki çalıştırılmaları sırasında bu verilerin silinmesi engellenmiş olmaktadır. Oluşturulan tüm araçlar bu dosya içerisine kayıt edilmektedir. Ayrıca, programın daha sonraki çalıştırılmaları sırasında, “AracKayit.txt” dosyasındaki bu veriler program içerisine aktarılmakta olup, bu veriler ışığında yeni nesneler üretilerek veri kaybının önüne geçilmesi sağlanmıştır.

**5. Proje Hakkında Diğer Veriler**

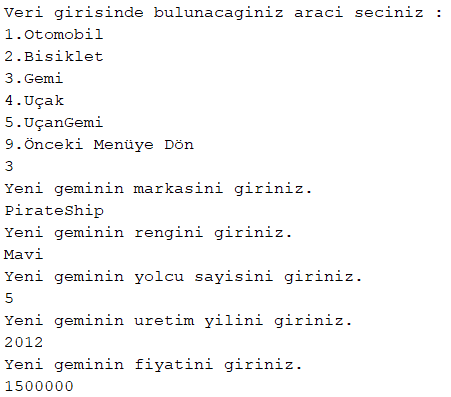
**5.1 Programın Çalışması Sırasından Bazı Görseller**

İşlem seçim ekranı:



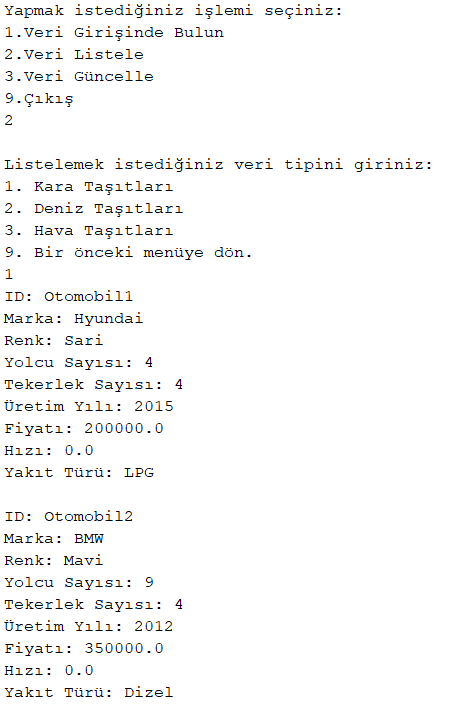
*Şekil 1:* İşlem Seçimi

Yeni araç girişi:



*Şekil 2:* Araç Girişi

Araçların listelenmesi:



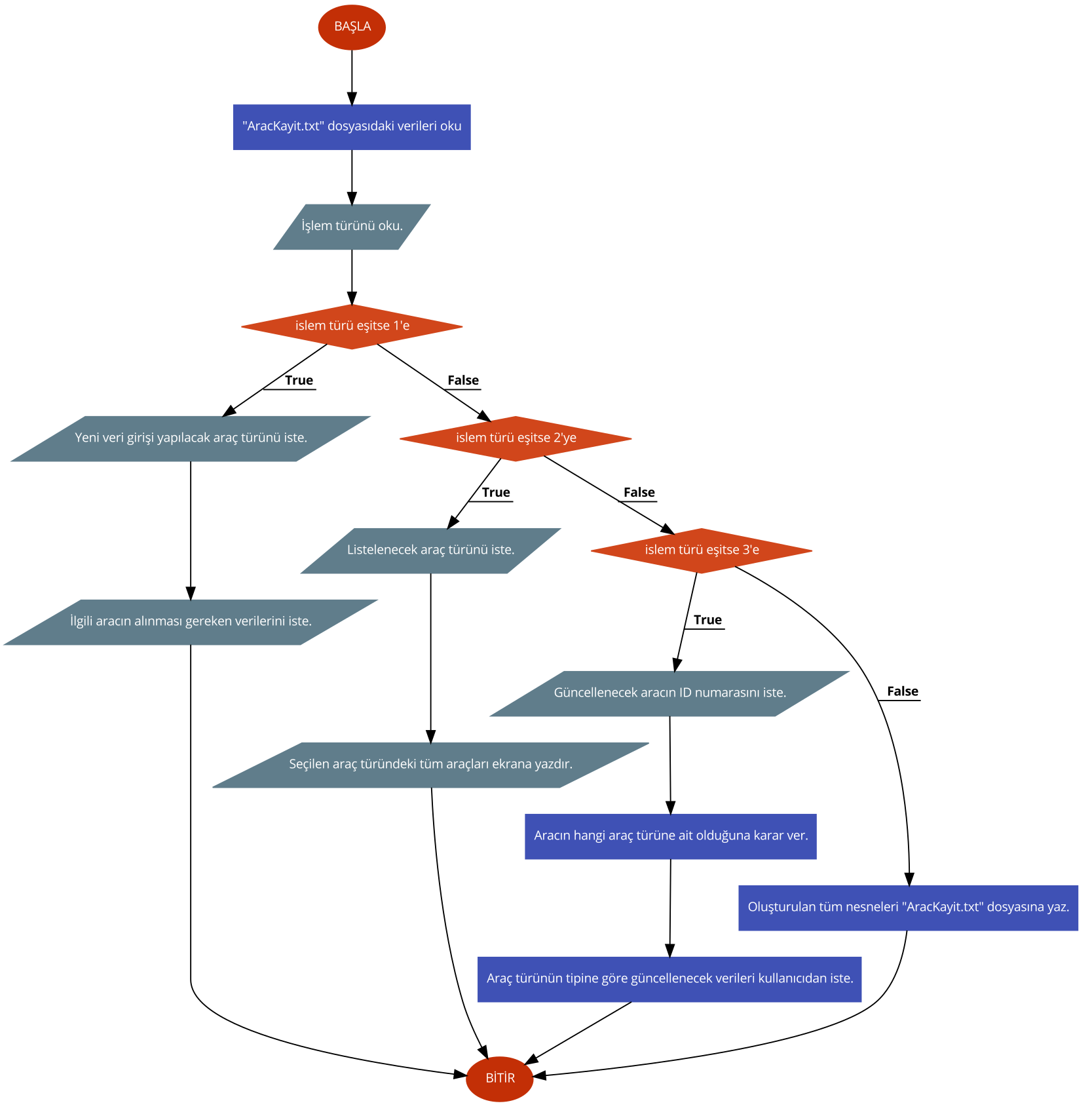
*Şekil 3*: Araçların listelenmesi

**6. Sonuçlar**

* Taşıtların, markası, yakıt türü (motorlu), hızı, yolcu sayısı, tekerlek sayısı, fiyatı, üretim yılı ve rengi isimlerinde özellikleri bulunmaktadır.
* Taşıtlar hızlanabilmekte, yavaşlayabilmekte ve durabilmektedir.
* Proje içerisinde Encapsulation, Inheritance, Polymorphism, Abstraction yapılarının hepsi kullanılmıştır.
* Yapıcı (constructor) fonksiyonlar (parametreli ve parrametresiz olmak üzere en az iki adet) yazılmıştır.
* Sınıflar içerisinde, tüm özellikler için get ve set fonksiyonları tanımlanmıştır.
* Projede bizlerden istenen kıstaslardan olan, yeni veri girişinde bulunulması, tüm verilerin listelenmesi ve tüm verilerin güncellenmesi özellikleri program içerisinde bulunmaktadır. Bu özellikler kullanıcıya, bir arayüz yardımı ile sunulmakta ve kullanıcı bu arayüz üzerinden ilgili işlemi kendisi seçebilmektedir.

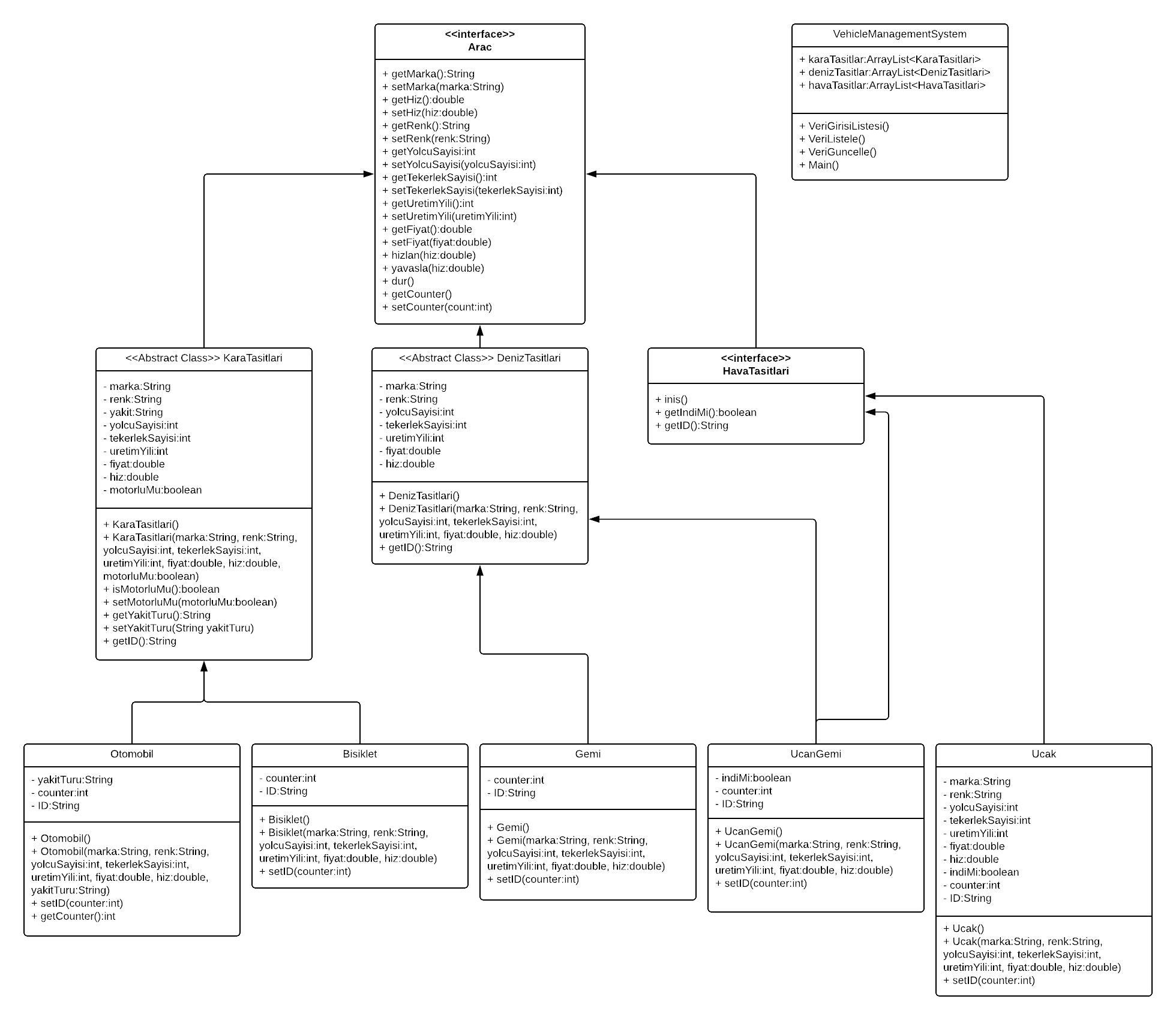
Yukarıdaki değerlendirmeler ışığında, yapmış olduğumuz program, proje sırasında bizlerden istenen tüm adımları yerine getirmekte olup, hiçbir eksik fonksiyonu bulunmamaktadır. Program, belirlenen hedefe tam anlamıyla ulaşmaktadır.

**7. Akış Şeması**



*Şekil 5*: Akış Şeması

**8. UML Diyagramı**



**9. Referanslar**

<https://www.tutorialspoint.com/java/java_files_io.htm> (Access Date: 17.03.2018)

<https://gelecegiyazanlar.turkcell.com.tr/konu/android/egitim/android-101/javada-dosya-islemleri> (Access Date: 17.03.2018)

<https://www.tutorialspoint.com/java/java_polymorphism.htm> (Access Date: 17.03.2018)

<https://www.tutorialspoint.com/java/java_interfaces.htm> (Access Date: 17.03.2018)

<https://javatutorial.net/java-abstraction-example> (Access Date: 17.03.2018)

<https://www.journaldev.com/1775/multiple-inheritance-in-java> (Access Date: 18.03.2018)

<https://www.dotnetperls.com/arraylist-java> (Access Date: 18.03.2018)