# GEZGİN KARGO PROBLEMİ PROJESİ

Osman Şimşek
Bilgisayar Mühendisliği
Kocaeli Üniversitesi
180202048
simsekosman15@hotmail.com

Yener Emin Elibol
Bilgisayar Mühendisliği
Kocaeli Üniversitesi
180202054
yenereminelibol@gmail.com

***Gezgin Satıcı Probleminde amaç, bir satıcının bulunduğu şehirden başlayarak her şehre sadece bir kez uğradıktan sonra başladığı şehre dönülmesi için en kısa yolun bulunmasıdır. Bu projede problem Gezgin Satıcı Probleminde uyarlanarak yapılmıştır.***

## I. GİRİŞ

Gezgin Kargo Probleminde satıcı tarafından girilen teslimat noktalarını birer kez geçerek başladığı yere ulaşması için gereken en kısa yolu hesaplamaktadır. Kullanıcı sistemi çalıştırdığı zaman kontrol ekranı gelmektedir. Şehirlerin adlarını girerek "Kayıt Yap" butonuna tıklayarak sisteme kayıt yapabilmektedirler. Sisteme en az 3, en fazla 10 şehir girilebilmektedir. Şehirler sisteme kaydedildikten sonra arama işlemini yaptırmaları için "Arama" butonuna tıklamaları gerekmektedirler. Program 1 saniye içinde aramayı tamamlamaktadır. Arama sonucunu programın ekranında bulunan "Sonuçlar" butonuna tıklayarak en iyi 5 sonucu görebilmektedirler. Kullanıcı gidilen yolun çizdirilmesini istediği vakit programın altında yer alan haritayı çizdir başlığı altındaki butonlara tıklayarak en iyi yolu ve diğer bulunan yolları çizdirebilmektedir.

## II. TEMEL BİLGİLER

.Program C# programlama dilinde gerçekleştirilmiş olup geliştirme ortamı olarak Visual Studio 2019 kullanılmıştır.

## III. TASARIM

### *A. Algoritma*

Program çalıştırıldığında kullanıcıyı bir ara yüz karşılar. Bu ara yüzde toplamda 9 farklı buton ve şehir isimlerinin girilmesi için bir textbox bulunmaktadır.

- 1. buton olan "Kayıt Yap" butonu textbox ile sisteme giriş yapılan şehirleri sisteme kaydedilmesini sağlar. Sisteme en az 3, en fazla 10 şehir eklenmesi gerekmektedir. Bunlar dışında fazla ya da eksik şehir eklenirse sistem uyarı verecektedir ve girilen şehirler sıfırlanacaktır. Kullanıcının tekrar giriş yapılması istenmektedir.
- 2. buton olan "Arama Yap" butonu ile sisteme girilen ( en fazla 10, en az 3 şehir) şehirler arasında en kısa yolu bulmayı sağlar. Arama yap butonu arkada çalışacak olan Floyd ve BruteForce algoritmalarını tetikler ve ilk başta Floyd algoritması çalışmaktadır. Floyd algoritması ile 81 ilin birbirine olan en kısa mesafesini ve bu illerin nasıl bir yol izlediklerini sisteme kaydeder. Kullanıcıdan alınan şehir bilgilerini random işlemine tabi tutularak 1000 tane yol ortaya çıkartılır ve Floyd algoritması ile bulunan 1000 yolu BruteForce algortimasına yollayarak bulunan varsa en iyi 5 çözüm yoksa da 5 ten az yol bulmaktadır.
- 3. buton olan "Sıfırla" butonu ile sisteme tekrar giriş yapılıp aranılması için sistemi hazırlar.
- 4. buton olan "Sonuçlar" butonu ile arama işlemi yapıldıktan sonra bulunan çözümlerin sonuçlar sayfasına listelenip gösterilmesini sağlamaktadır.
- 5. buton olan "En İyi Sonuç" butonu ile arama yapıldıktan sonra bulunan en iyi yolun harita üstüne çizdirilmesini sağlamaktadır.
- 6. buton olan "2. Sonuç" butonu ile arama işlemi yapıldıktan sonra bulunan en iyi 2. yolun harita üstünde çizdirilmesini sağlar. Eğer ki sistem tarafından 1 sonuç bulunduysa çizdirilme işlemi gerçekleşmeyecektir.
- 7. buton olan "3. Sonuç" butonu ile arama işlemi yapıldıktan sonra bulunan en iyi 3. yolun harita üstünde çizdirilmesini sağlar. Eğer ki sistem tarafından 2 sonuç bulunduysa çizdirilme işlemi gerçekleşmeyecektir.
- 8. buton olan "4. Sonuç" butonu ile arama işlemi yapıldıktan sonra bulunan en iyi 4. yolun harita üstünde çizdirilmesini sağlar. Eğer ki sistem tarafından 3 sonuç bulunduysa çizdirilme işlemi gerçekleşmeyecektir.
- 9. buton olan "5. Sonuç" butonu ile arama işlemi yapıldıktan sonra bulunan en iyi 5. yolun harita üstünde çizdirilmesini sağlar. Eğer ki sistem tarafından 1 sonuç bulunduysa çizdirilme işlemi gerçekleşmeyecektir.

### *B. Kullanılan Bazı Fonksiyonlar ve classlar*

- Public class FloydWarshall:
  Bu class ta kendisine gönderilen komşuluk matrisi ile 81 ilin birbirlerine olan en kısa mesafesini ve nasıl bir yol izlediği bulunmaktadır. Bulunan bu bilgiler 2-Opt algortimasında kullanılmak üzere saklanmaktadır.

- Public class BruteForce:
  Bu class ta arama yap butonu çalıştırıldığı zaman kullanıcının girdiği şehirlerden random çekilerek 100 farklı yol elde edildikten sonra bu bilgileri ve FloydWarshall algoritmasından gelen bilgileri gönderilerek varsa en iyi 5 çözüm bulunur, 5 çözüm bulunamassa bile en az 1 çözüm bulabilir.

- Private void btnaramayap_Click:
  Bu fonksiyonda Kullanıcının sisteme giriş yapmış olduğu şehir bilgileri alınarak 1000 farklı random sonuç elde edilmeye çalışılır. Bulunan sonuçlar 2-Opt algoritmasına gönderilir. 2-Opt algoritması çalıştırılıp en iyi 5 veya 5 ten az sonuç elde edilmesi işlemini gerçekleştirir.

- Private void btnsistemisıfırla_Click:
  Bu fonksiyon kullanıcı sisteme tekrar giriş yapıp arama yaptırmak istediği zaman sistemin sıfırlayıp kullanıcıya tekrar giriş yapmasını hazırlar.

- Private void btnsehirismi_Click:
  Bu fonksiyon kullanıcı sisteme şehir bilgilerinin girmesini sağlamaktadır. Eğerki kullanıcı 3 ten az şehir veya 10 dan fazla şehir girmesi durumunda arama yap butonuna tıklarsa uyarı verdirmesini sağlamaktadır. Ayrıca kullanıcı sisteme hatalı bir şehir ismi girmeye çalışırsa kullanıcıyı uyarmak için ekrana hata mesajı verdirecektir.

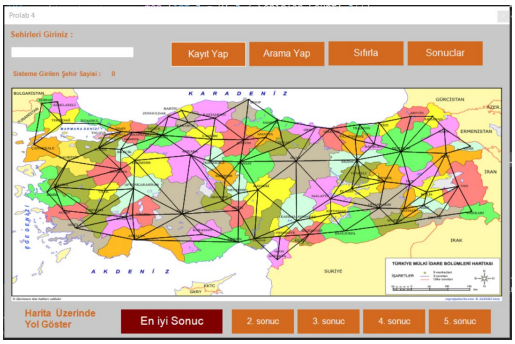
*C. Karşılaşılan Bazı Sorunlar*

- Bu projede karşılaşılan sorunlardan biri random çektirilen şehirlerden aynı olanları sildirmede sorun yaşandı.
- Bu projede karşılaşılan diğer sorunlardan birisi ekrana çizdirilen en kısa yollardan bir diğeri çizdiriliceği vakit bir öncekinin ekrandan sildirilememsi.
- Bu projede karşılışılan diğer sorunlardan birisi de Floyd algoritmasında iller arasında bulunan en kısa yolların hangi şekilde gidildiği bulunamadı.

*D. Kazanımlar*

- C# ta form yapısını ve form yapısında grafiksel işlemlerin nasıl kullanıldığı öğrenildi.
- C# ta dosya okuma işlemlerinin nasıl yapıldığı öğrenildi.
- Floyd Warshall ve BruteForce algoritmalarının nasıl izlediği hakkında bilgi sahibi olundu ve bu algoritmaları projeye eklemeyi başardık.
- C# ta List yapılarının nasıl kullanıldığı öğrenildi.

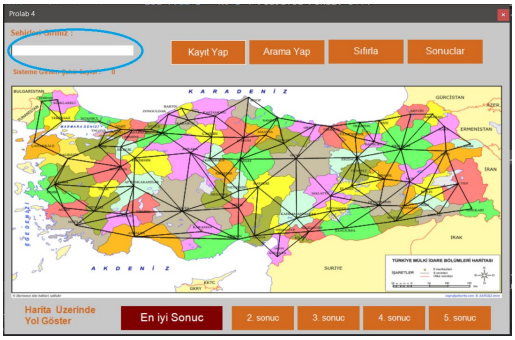
*E. Çalışma Adımları*




**Adım 1:**

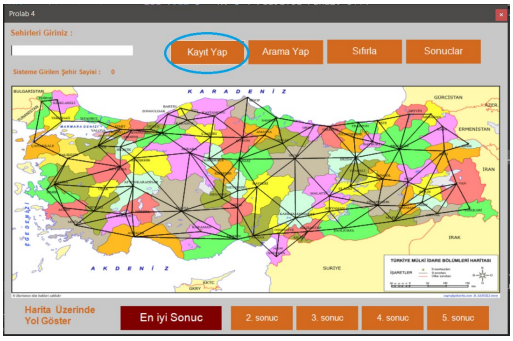
Sistem İlk Çalıştırıldığı zaman kullanıcıya bir bilgilendirme mesajı vermektedir. Bu mesajı okuduktan sonra tamam butonuna basıp devam edilmelidir

Tamam butonuna basıp geçildikten sonra kullanıcının karşısına program ekranı gelmektedir. 9 buton ve 1 textboxtan oluşmaktadır.

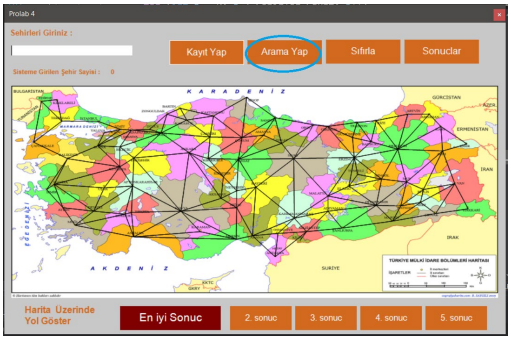



**Adım 2:**

Programda ilk olarak sisteme şehirlerin girilmesi gerekmektedir. Yandaki mavi ile çizilmiş olan textboxtan şehir isimleri girilmelidir. Şehir isimleri büyük harfle başlamalı ve türkçe karakter içermemelidir.



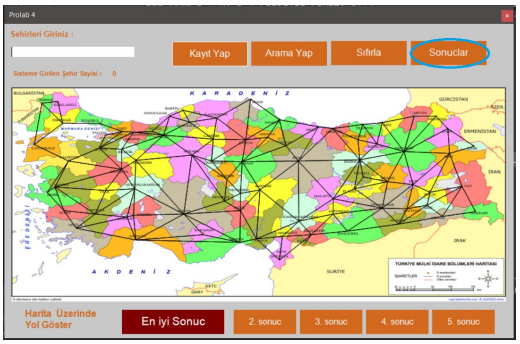

Sisteme kayıt yapılması için textboxa şehir ismi yazıldıktan sonra 'Kayıt Yap' butonuna tıklanmalıdır. Sisteme hatalı şehir ismi girildiği zaman uyarı vermektedir. Ayrıca sisteme en az 3 en fazla 10 şehir girilmesi gerekiyo bu sınırlar dışında arama yaptırılırsa sistem otomatik olarak kendini sıfırlamaktadır.




**Adım 3:**

Sisteme 3 ile 10 arasında şehir girildikten sonra arama yap butonuna tıklanmalıdır. Arama yap butonu ile sistem en kısa yolu hesaplamaktadır. Yandaki resimde sisteme 'Konya, Kayseri, Bartin, Sinop, Canakkale, Edirne, Samsun, Antalya, Mus, Bitlis' sehirleri girilmiştir.

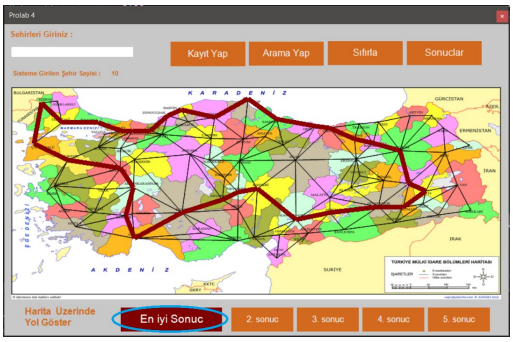
.




**Adım 4:**

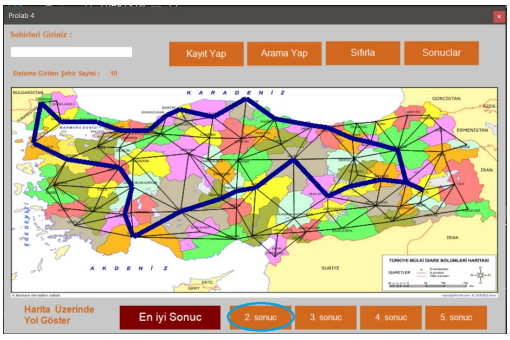
Arama butonuna tıklandıktan 1 2 saniye sonrasında arama tamamlanmaktadır. Bulunan sonuçları görmek için 'Sonuçlar' butonuna tıklanmalıdır.


'Sonuçlar' butonuna tıklandıktan sonra ekrana ufak bir pencere gelmektedir. Bu pencerede bulunan en kısa yollar iyiden kötüye sıralı bir şekilde gösterilmektedir. Bulunabilen sonuç sayısı en fazla 5 olarak kısıtlanmıştır. Eğer ki 5 sonuç bulamadıysa bile en az 1 sonuç bulunabilmektedir.



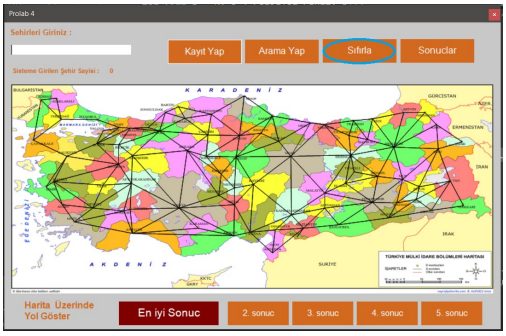

**Adım 5:**

Arama yapıldıktan sonra bulunan sonuçları ekrana bastırmak için programın altında bulunan 5 butona tıklanması gerekmektedir. Bulunan en iyi sonucu çizdirmek için 'En iyi Sonuç' butonuna tıklanması gerekmektedir.




Bulunan sonuçların sayısı 2 ye eşitse eğer ekrana 2 tane yol çizdirilebilir bunlardan biri 'En iyi Sonuç' butonuna basılarak çizdirilir diğer '2. sonuç' butonu ile çizdirilmektedir. Yandaki resimde de bulunan 2. sonuç çizdirilmiştir.

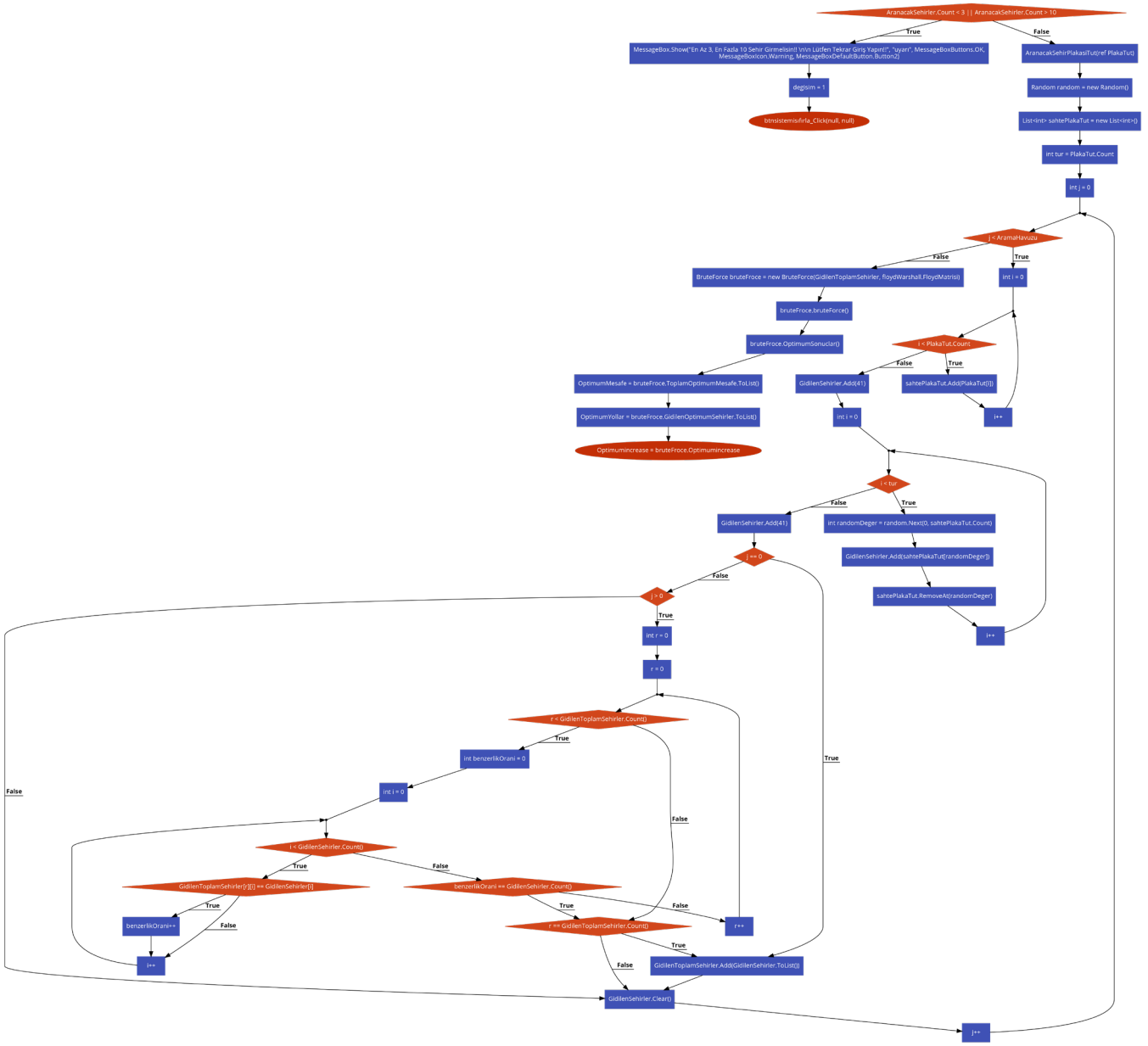
Eğerki bulunan sonuçların sayısı kadar haritaya çizim yapılabilmektedir. Yandaki resimde 10 farklı şehirden en kısa yol için arama yapıldığından 5 farklı sonuç bulunabilmiştir ve haritaya çizim yaptırabilicek diğer 3 butonda çalışır.




**Adım 6:**

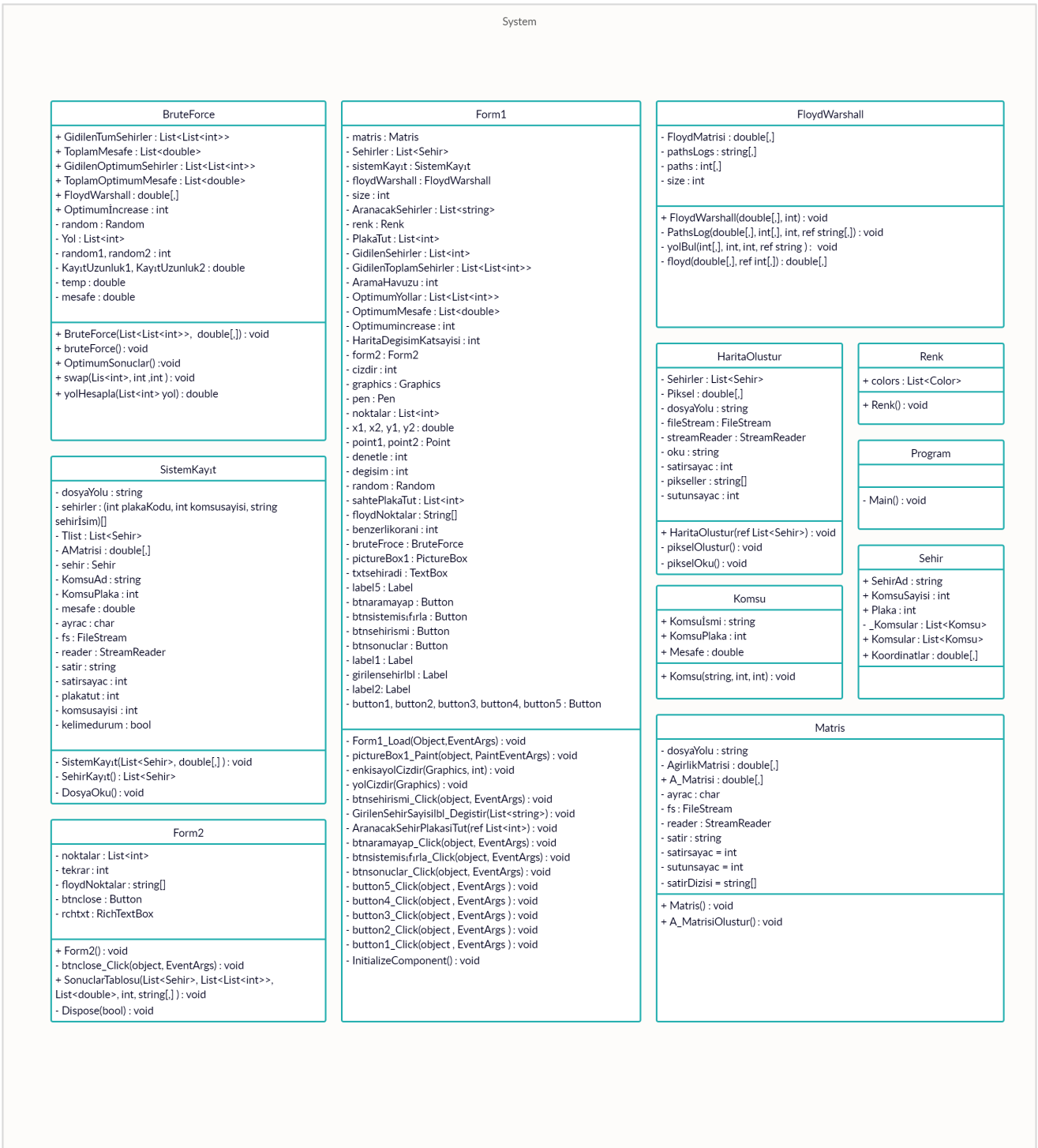
Kullanıcı sisteme tekrar giriş yapıp arama yapmak isterse yanda belirtilen 'Sıfırla' butonuna tıklanması gerekmektedir. Sıfırla butonuna tıklanınca sistem kendini tekrar arama yapmak için hazırlanmaktadır.

## F. Akış Diyagramı




*btnaramayap_Click fonksiyonunun çalışma sistemi*

## G. Uml Diagram



## KAYNAKÇA


[1] https://stackoverflow.com/

[2] http://bilgisayarkavramlari.sadievrenseker.com/category/bilgisayar-kavramlari/

[3] https://www.youtube.com/watch?v=BAejnwN4Ccw

[4] https://rosettacode.org/wiki/Floyd-Warshall_algorithm

[5] https://www.techiedelight.com/pairs-shortest-paths-floyd-warshall-algorithm/

[6] https://www.geeksforgeeks.org/

[7] https://www.youtube.com/watch?v=q6fPk0--eHY