

**Gebze Technical University  
Computer Engineering**

**CSE 443 - 2018 Fall**

**HOMEWORK 01 REPORT**

**Elif Şeyma ARMAĞAN  
151044042**

Course Assistant:

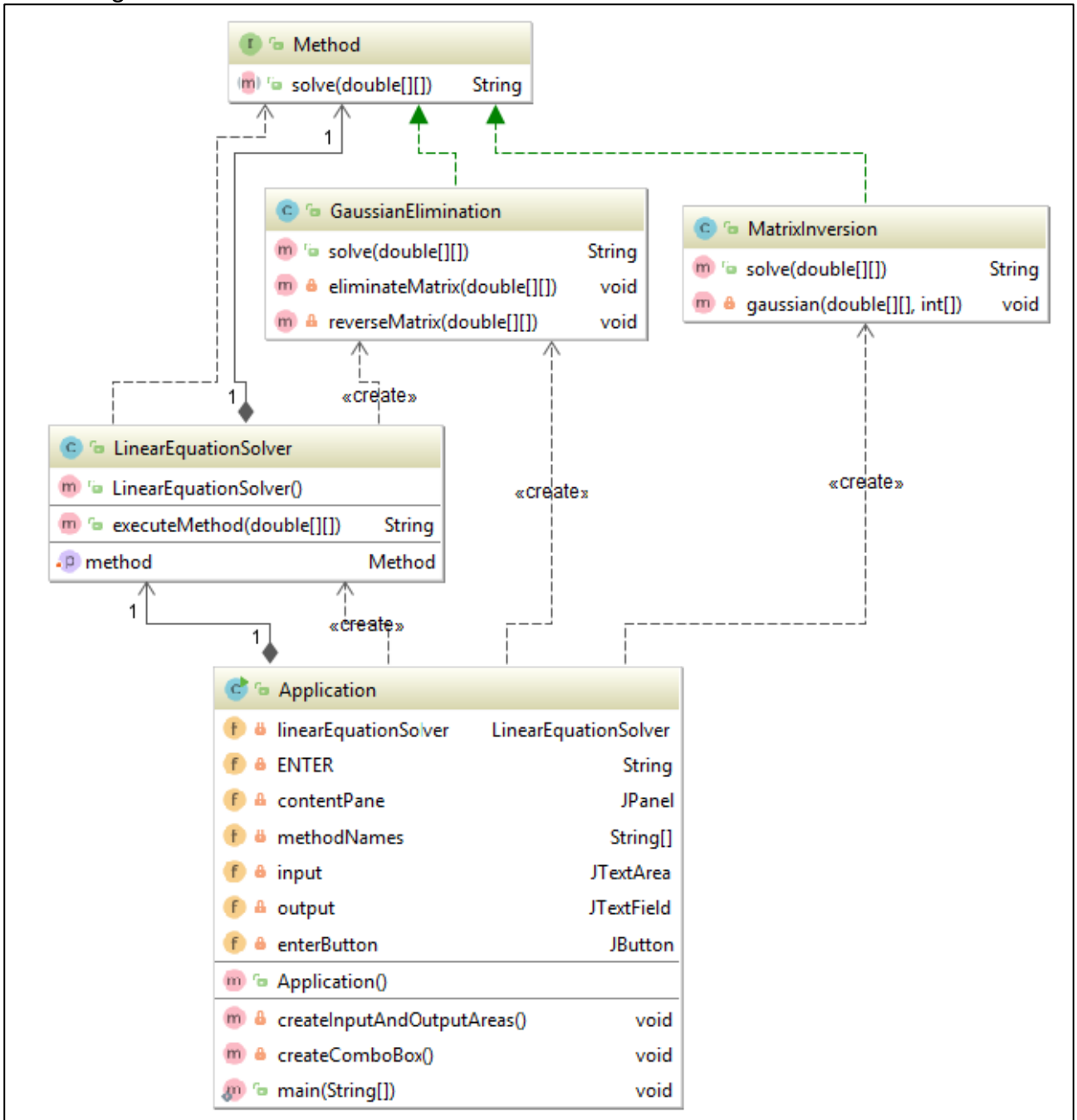
**Muhammet Ali DEDE**

## 1 PART 1

Bu soru için Strategy Pattern uyumludur. Çünkü farklı metotlar ile denklemler çözümlenmek isteniyor ve iki farklı şekilde bu sağlanıyor. Her yöntem kendi başına, kendi içinde algoritmalarını barındırıp, çözüm yapılmak istenen class'ta ortak interface vasıtasıyla tutulursa Strategy Pattern'ı uygulamış oluyoruz.

Bu pattern ile, ileride farklı bir yöntem geliştiresek entegre etmek çok rahat olacaktır. Yeni yöntemin algoritmasını barındıran, ortak interface'den gelen bir sınıf oluşturup, bu yöntemi ön yüzde bir seçenek olarak sunarız. Ve eğer kullanıcı bu yeni yöntemi seçerse çözümü çağırdığımız sınıfta ortak interface tipindeki değişkenimize yeni sınıfın objesini atarız.

Class diagram:

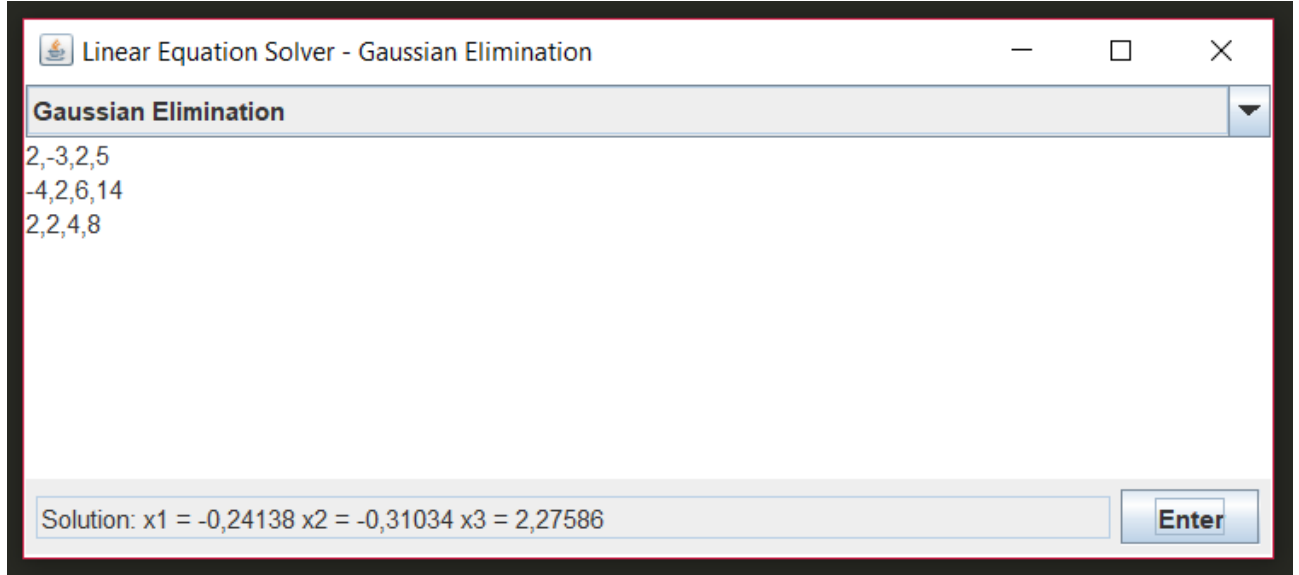


Burada “Method”, farklı algoritmalarımız için ortak interface ve her algoritma (Gaussian ve MatrixInversion) bunu implement eden bir sınıftan oluşuyor.

“LinearEquationSolver” ise, Method tipinde bir field tutarak, o an seçili olan algoritmanın solve methodunu çağırıyor yani Strategy Pattern’ın “Context” objesi.

“Application” ise ön yüzün oluşturulduğu ve kullanıcıdan algoritma seçiminin alındığı sınıf.

Örnek çalışma senaryosu:



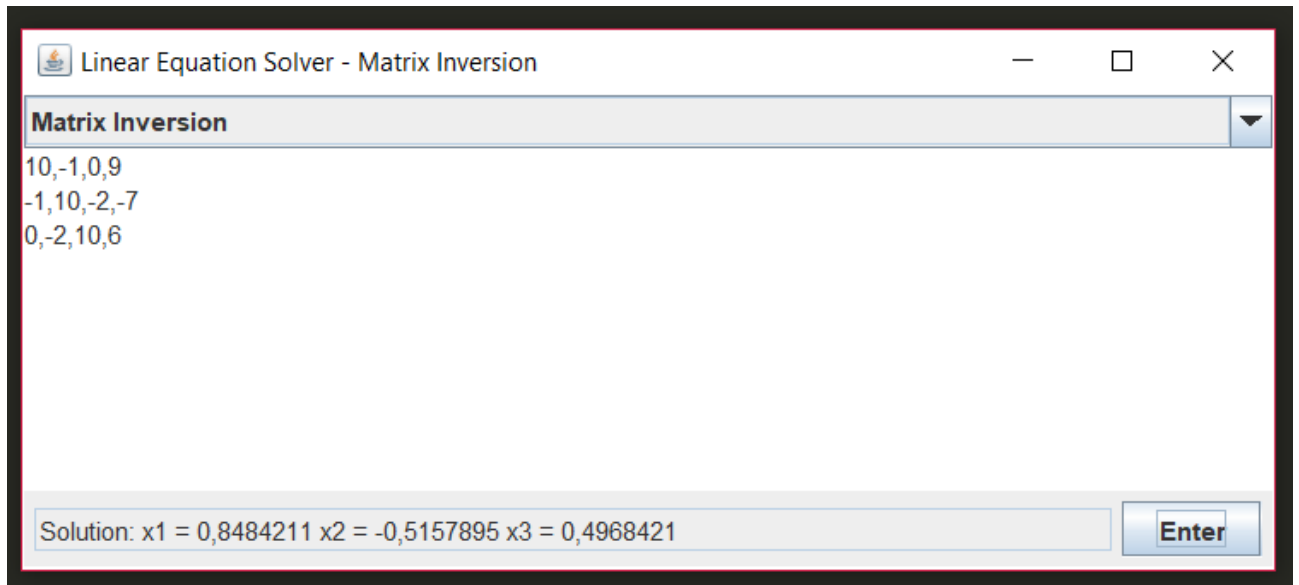
Linear Equation Solver - Gaussian Elimination

Gaussian Elimination

2,-3,2,5  
-4,2,6,14  
2,2,4,8

Solution: x1 = -0,24138 x2 = -0,31034 x3 = 2,27586

Enter



Linear Equation Solver - Matrix Inversion

Matrix Inversion

10,-1,0,9  
-1,10,-2,-7  
0,-2,10,6

Solution: x1 = 0,8484211 x2 = -0,5157895 x3 = 0,4968421

Enter

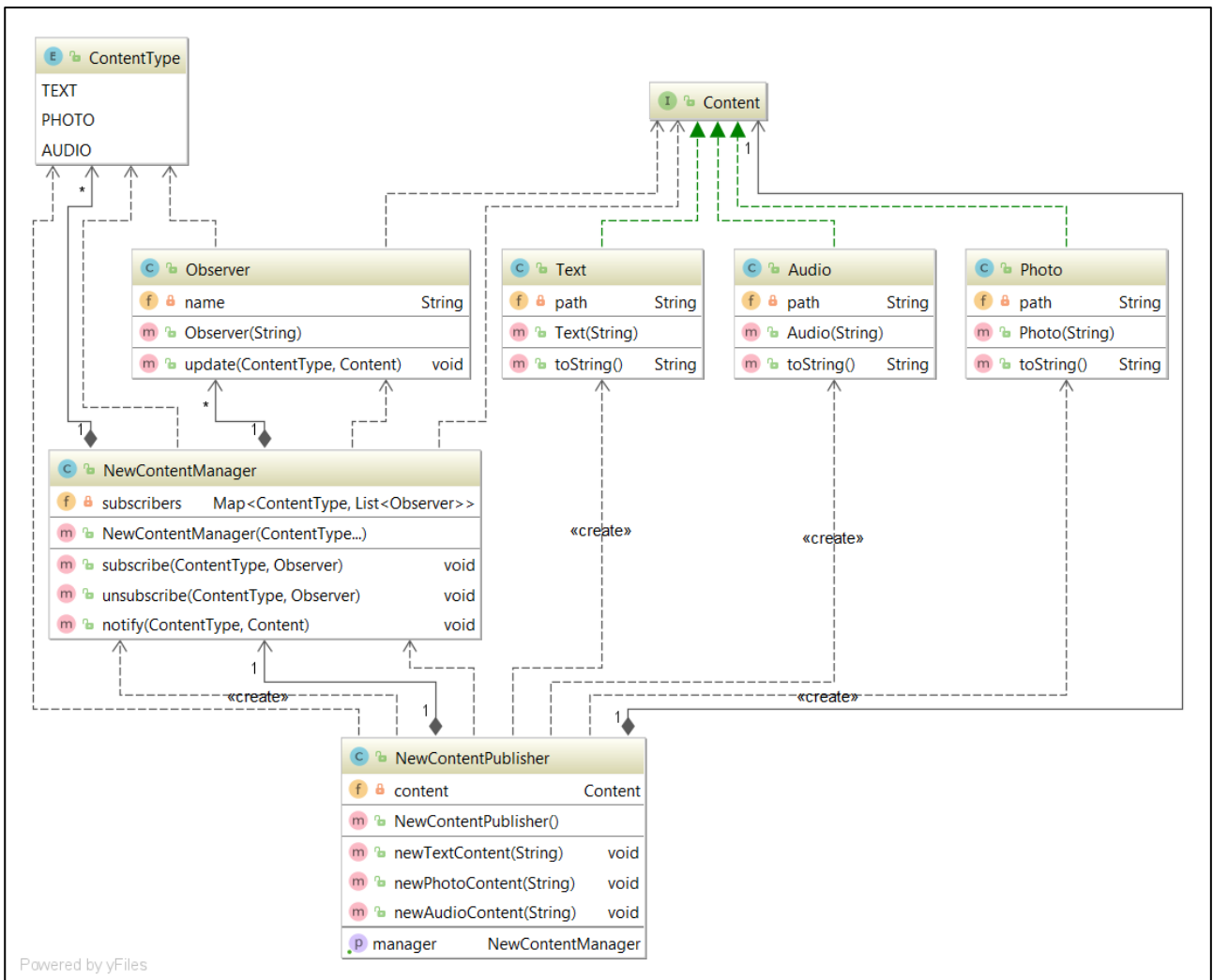
Burada input formatı, denklemlerin bilinmez ifadeleri (x1, x2 ...) olmadan yazılmış halleridir ve denklemler alt alta yazılmalıdır.

Örnek verilen denkler 3 bilinmeyenlidir ve son eleman denklemin sonucudur.

$2x_1 - 3x_2 + 2x_3 = 5$  şeklindeki denklem 2, -3, 2, 5 şeklinde yazılır.

## 2 PART 2

Bu soruda istenen yapı Observer Pattern ile sağlanır.



Öncelikle farklı veri tipleri ortak bir interface'den üretilir.

Observer'lar yani bu sorudaki subscriber'lar kendi update metoduna sahip olmalıdır.

Her bir subscriber, farklı veri tiplerine ve birden fazla veri tipine abone olabileceğinden, “NewContentManager” içerisindeki subscribe metodu ile, abone olmak isteyen Observer’ı, abone olmak istediği veri tipiyle birlikte, her bir veri tipine abone olan subscriber’ların tutulduğu map’e kaydederiz.

Bir veri tipine abone olmak isteyen kullanıcılar, “NewContentPublisher” sınıfındaki “manager” field’ına ulaşp, kendini kaydettirmelidir.

Yeni içerik geldiğinde ise, örneğin yeni bir fotoğraf içeriği geldiğini iletmek isteyen bir web sitesi, “NewContentPublisher” sınıfındaki newPhotoContent metodunu, yeni gelen içeriğin bağlantısı ile çağırmalıdır. Bu metodun içerisinde, fotoğraf tipi için kendini kaydetmiş olan tüm abonelerin update methodu çağrılır.