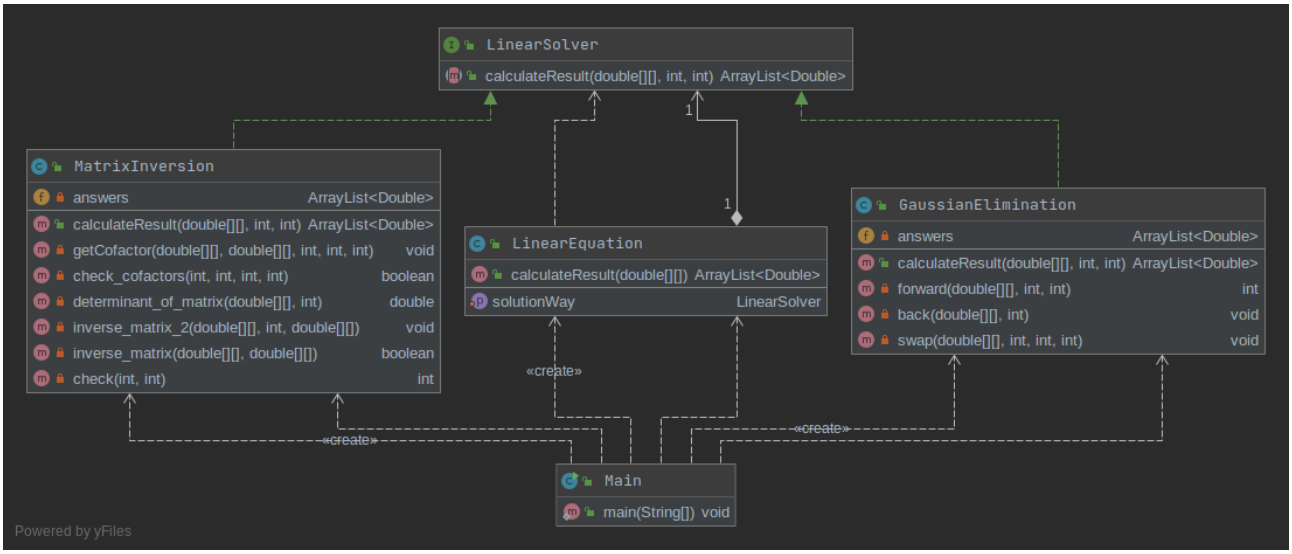
CSE 443 – OBJECT ORIENTED ANALYSIS AND DESIGN
HOMEWORK #01 – REPORT
GÖKHAN HAS – 161044067

QUESTION 1 :

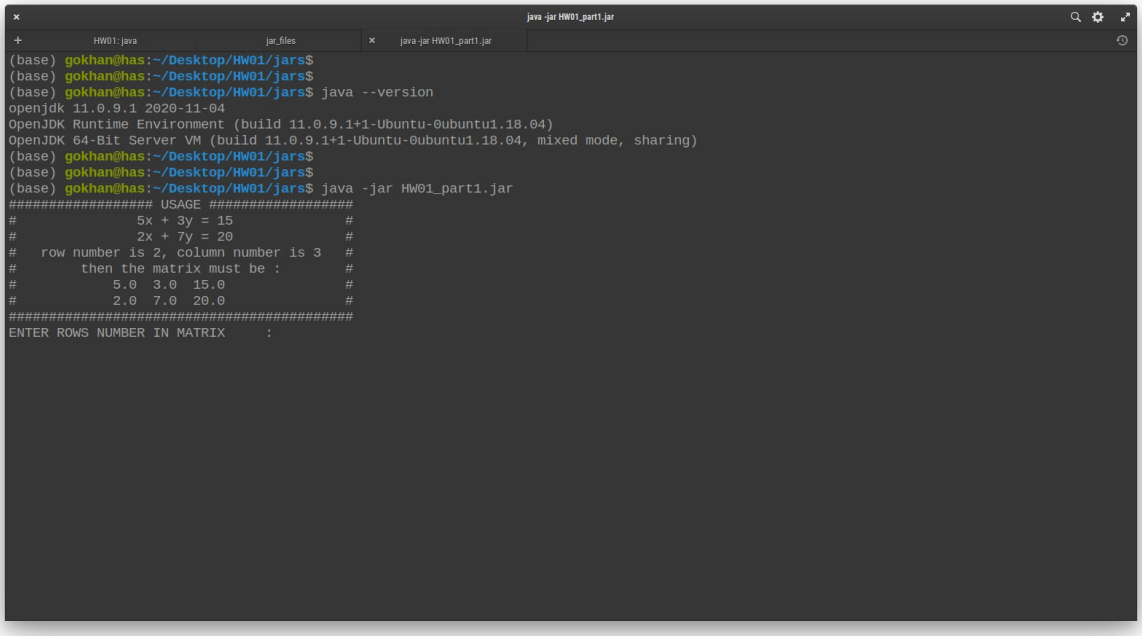
There may be more than one method for performing a transaction in our software. The strategy design template is used to select and apply a method according to the situation. Each algorithm is implemented for a class. So to summarize, the behavior or algorithm of a class at runtime can be changed according to a strategy.

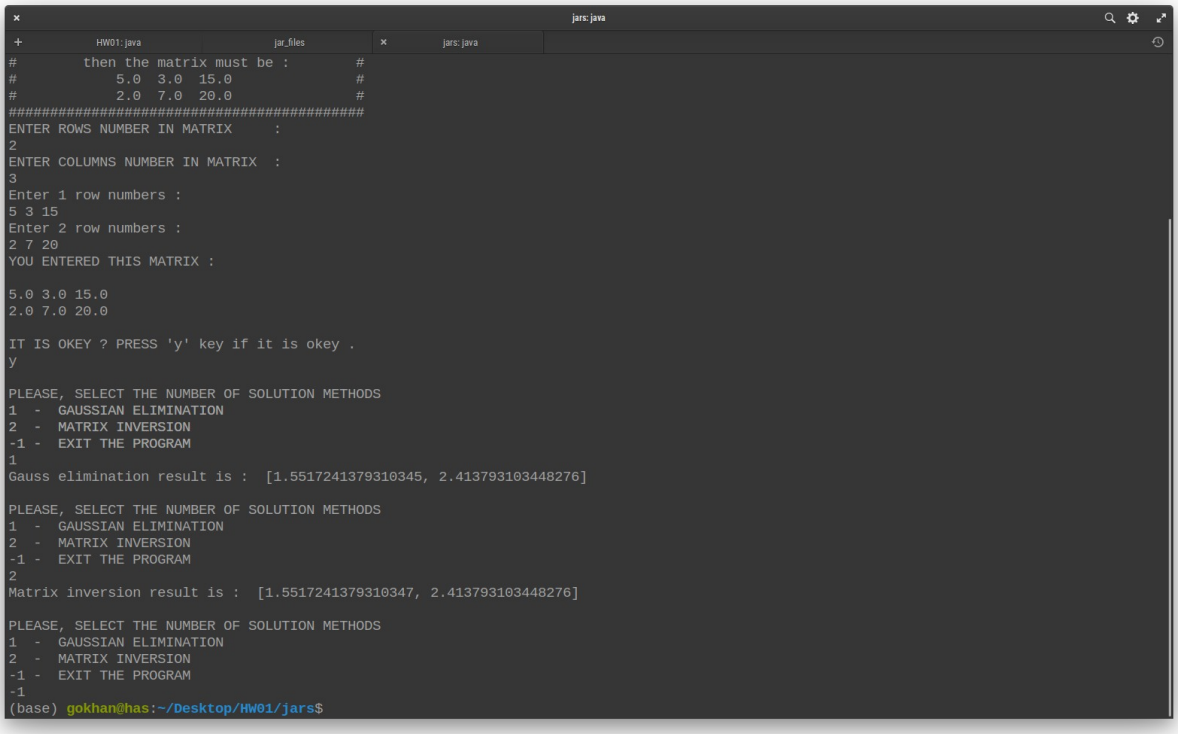
In order to apply the strategy design template, we need to define an interface named Strategy. This interface contains methods or methods that will be implemented by subclasses.



In accordance with this design, an interface named LinearSolver was created. There are two classes that implement this interface. These are MatrixInversion and GaussianElimination classes. The calculateResult function has been defined to be used in classes in this interface. Likewise, a class named LinearEquation has been defined. This class holds the LinearSolver object. The purpose here is to show that it can change dynamically. The type of that object can be of the type of MatrixInversion or GaussianElimination classes. These are considered as two separate behaviors. These classes use more than one auxiliary function to solve the problem.

As seen in the picture below, a sample usage of the program is shown to the user by running the jar file with the specified command. The user has the number of rows and columns (must be at least with row number + 1) There is no such thing as separation of coefficients matrix and result matrix. Therefore, (n) x (n + 1) should be entered as the matrix number.

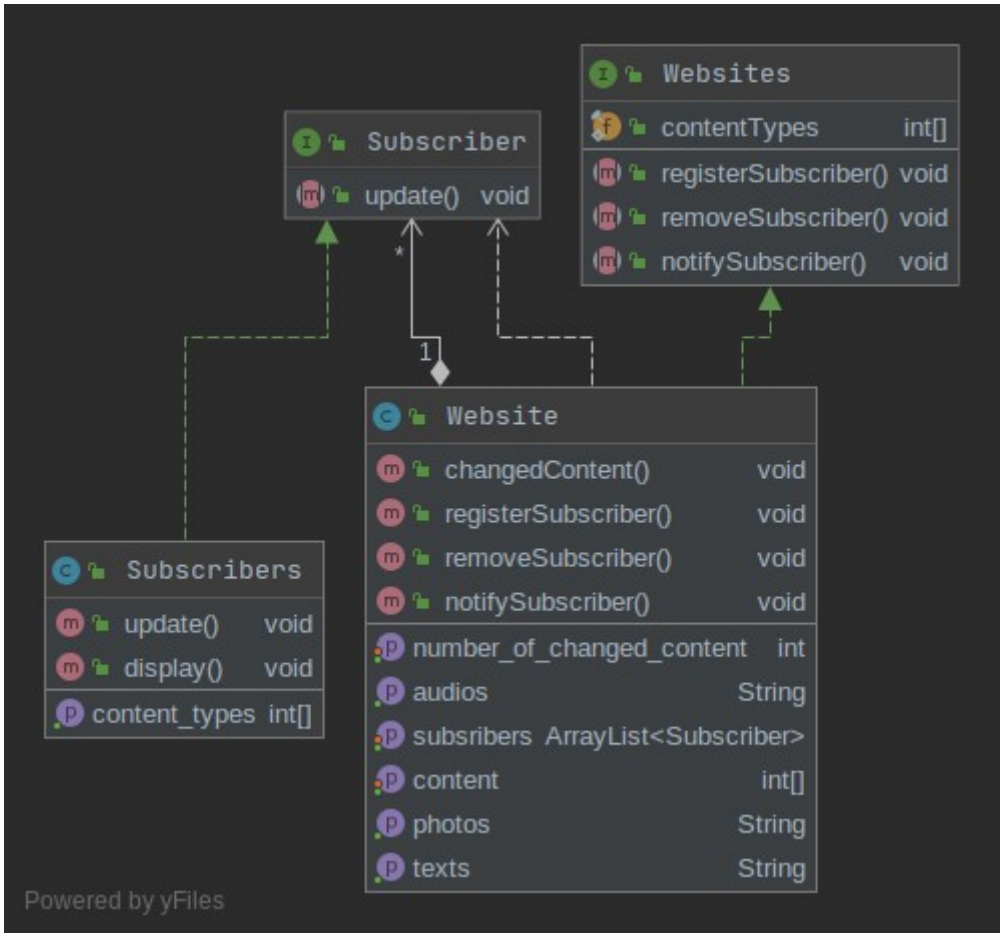




In the program, the user can dynamically change between solution methods.

QUESTION 2 :

I used the Observer design template for this question. This design pattern is preferred when there is a one-to-many relationship between objects. That is, if any change in an object affects other objects dependent on that object, this template is used.



Observer pattern uses 3 actor classes. These classes are; Subject, Observer and Client classes.

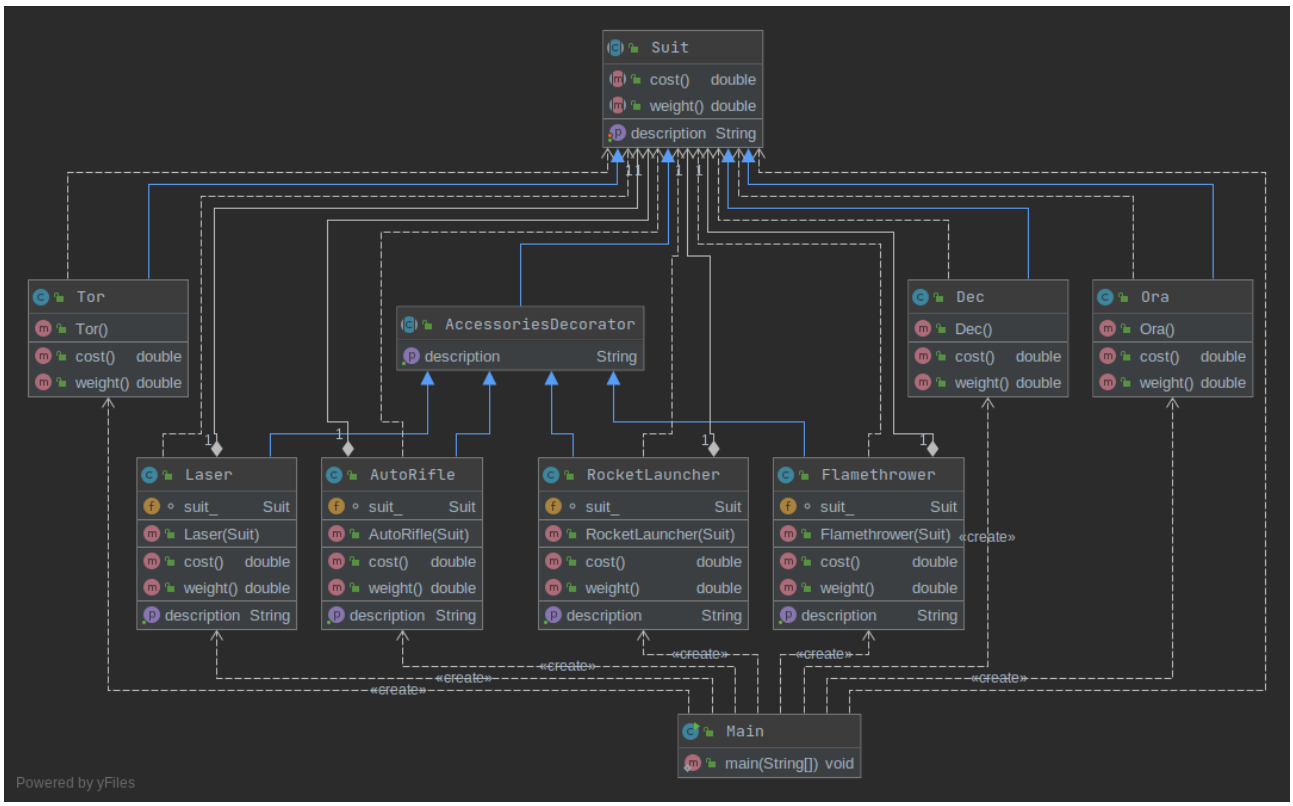
The Website keeps the observer list and includes the changedContent () method that allows adding to this list. It also provides a notification to the observer about the notifySubscriber () method. Optionally, the removeSubscriber () method has been added to the system to delete any observer. These methods are available in the Websites interface. Likewise, 3 variables are kept in String types in order to send news only to the subscriber using the relevant Website. There are subscriber types in the same interface as int array. This array is used as a kind of enum. There is a variable in the Wbsite class that keeps how many extra content has changed.

Subscriber is an interface. It contains the update () method. Observers that will inherit from this class also necessarily include the update () method. It is aimed that the subclasses benefit from a common interface.

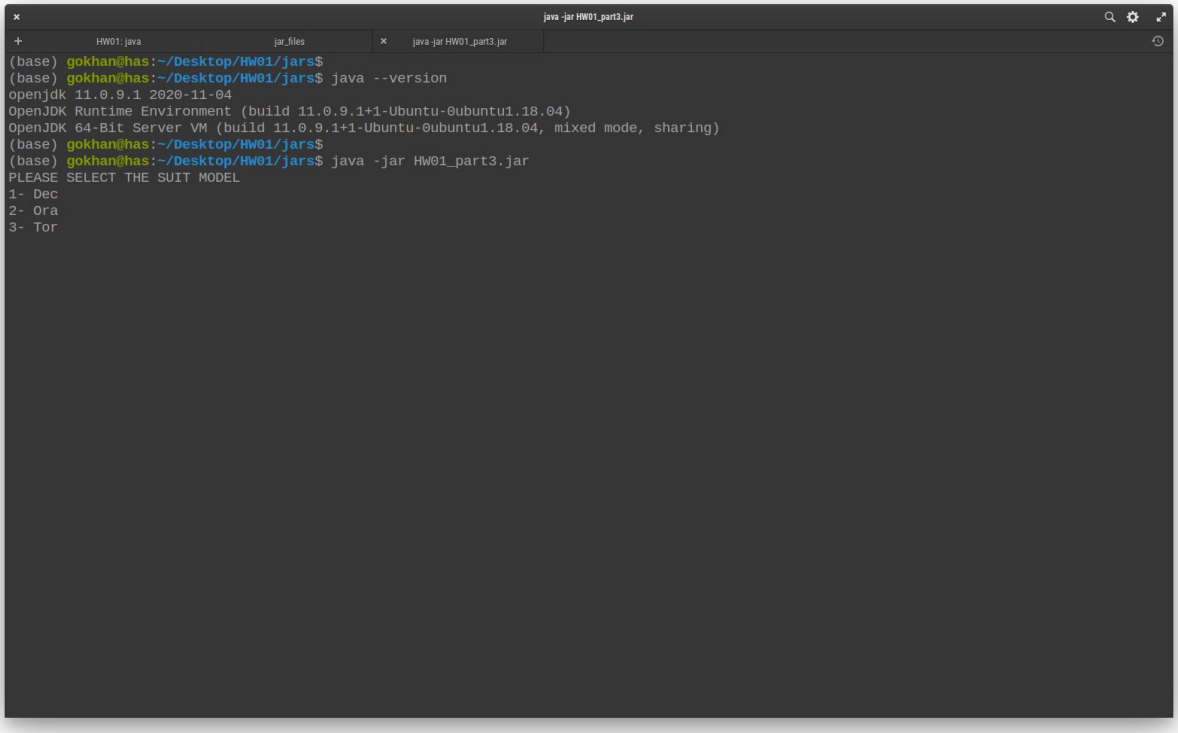
QUESTION 3:

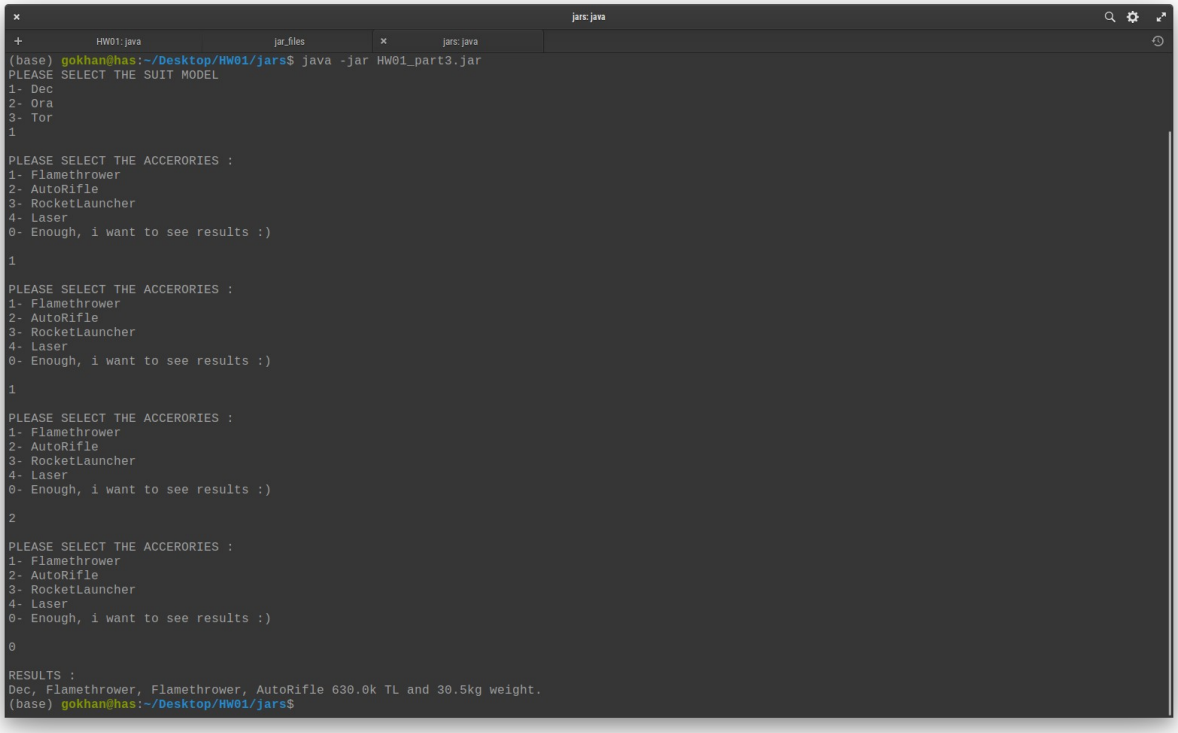
The Decorator design pattern is one of the structural design patterns. Used to dynamically add new properties to an object. It shows that we can increase the functions of an object without using inheritance. Properties added to an object of a class at runtime do not affect other objects derived from that class. Used when trying to add new properties to an object at runtime.

We can divide the Decorator design pattern into two parts as Decorator classes and Component classes. Component classes include the Decorator super class.



The AccessoriesDecorator class is derived from the Suit class. At the same time, there is a HAS-A relationship between the Decorator class and the Component class. This means that there is an instance variable of type Suit in the Decorator class. The AccessoriesDecorator class can be abstract or interface. It is necessary not to use concrete classes. The object to which the properties are dynamically added is derived from the ConcreteComponent class. The ConcreteDecorator object does the process of adding properties to the ConcreteComponent object.
In the Decorator design pattern, inheritance is used only so that classes have the same type. The functions of the object are provided by composition. If the inheritance was adhered to, the behavior of the object at compile time would be determined. In such a case, new features would not be dynamically added.





GÖKHAN HAS - 161044067