AgroMarket Comprehensive Report

Introduction

The AgroMarket project simulates an agricultural market organization, consisting of a list of farmers. This organization is designed to manage farmer information and provide essential functionalities to maintain and manipulate the farmer database. The AgroMarket class also adheres to certain legal requirements by implementing the `LegalEntity` interface. The primary goal of this project is to demonstrate object-oriented programming concepts such as encapsulation, inheritance, and polymorphism in Java.

Classes and Their Functionality

1. LegalEntity Interface

- **Purpose**: To define a contract for legal entities, ensuring that any implementing class provides methods to get the entity's address and VAT number.
- **Methods**:
  - `String getAddress()`: Returns the address of the legal entity.
  - `String getVatNumber()`: Returns the VAT number of the legal entity.

2. Farmer Class

- **Purpose**: To represent individual farmers in the agricultural market. This class holds basic information about the farmer.
- **Fields**:
  - `String name`: The name of the farmer.
  - `String farmName`: The name of the farmer's farm.
- **Constructor**:
  - `Farmer(String name, String farmName)`: Initializes the farmer with a name and farm name.
- **Methods**:
  - `getName()`: Returns the name of the farmer.
  - `setName(String name)`: Sets the name of the farmer.
  - `getFarmName()`: Returns the name of the farmer's farm.
  - `setFarmName(String farmName)`: Sets the name of the farmer's farm.
  - `toString()`: Returns a string representation of the farmer.

3. AgroMarket Class

- **Purpose**: To manage a collection of farmers and provide functionalities to add, remove, and retrieve farmers. This class also implements the `LegalEntity` interface, ensuring it provides methods to get its address and VAT number.
- **Fields**:
  - `String address`: The address of the AgroMarket.
  - `String vatNumber`: The VAT number of the AgroMarket.
  - `List<Farmer> farmers`: A list to store the farmers associated with the AgroMarket.
- **Constructor**:

    - `AgroMarket(String address, String vatNumber)`: Initializes the AgroMarket with an address and VAT number and an empty list of farmers.
- **Methods**:
    - `String getAddress()`: Returns the address of the AgroMarket.
    - `String getVatNumber()`: Returns the VAT number of the AgroMarket.
    - `void addFarmer(Farmer farmer)`: Adds a farmer to the AgroMarket.
    - `boolean removeFarmer(Farmer farmer)`: Removes a farmer from the AgroMarket and returns true if the removal was successful.
    - `List<Farmer> getFarmers()`: Returns a list of farmers associated with the AgroMarket.
    - `String toString()`: Returns a string representation of the AgroMarket, including its address, VAT number, and the list of farmers.

Example Usage

To demonstrate the functionality of the AgroMarket project, an example usage scenario is provided through a tester class.

AgroMarketTester Class

- **Purpose**: To test the functionalities of the AgroMarket class by creating instances of AgroMarket and Farmer, and performing operations like adding and removing farmers.
- **Main Method**:
    - Creates an instance of AgroMarket.
    - Creates instances of Farmer.
    - Adds farmers to the AgroMarket.
    - Removes a farmer from the AgroMarket.
    - Prints the details of the AgroMarket before and after removing a farmer.

```
java
Copy code
package finalexam.task4;

public class AgroMarketTester {
    public static void main(String[] args) {
        // Create an instance of AgroMarket
        AgroMarket market = new AgroMarket("123 Main St, Farmville",
"VAT123456");

        // Create some Farmer objects
        Farmer farmer1 = new Farmer("John Doe", "Doe Farms");
        Farmer farmer2 = new Farmer("Jane Smith", "Smith Orchards");

        // Add farmers to the market
        market.addFarmer(farmer1);
        market.addFarmer(farmer2);

        // Print the AgroMarket details
        System.out.println("AgroMarket details:");
        System.out.println(market);

        // Remove a farmer
        market.removeFarmer(farmer1);
```

```
        // Print the AgroMarket details after removal
        System.out.println("AgroMarket details after removing a farmer:");
        System.out.println(market);
    }
}
```

Conclusion

The AgroMarket project effectively demonstrates the principles of object-oriented programming in Java. By implementing the `LegalEntity` interface and managing a collection of `Farmer` objects, the project provides a clear example of how interfaces, classes, and collections can be used to build a functional and maintainable software system.