Problem Statement

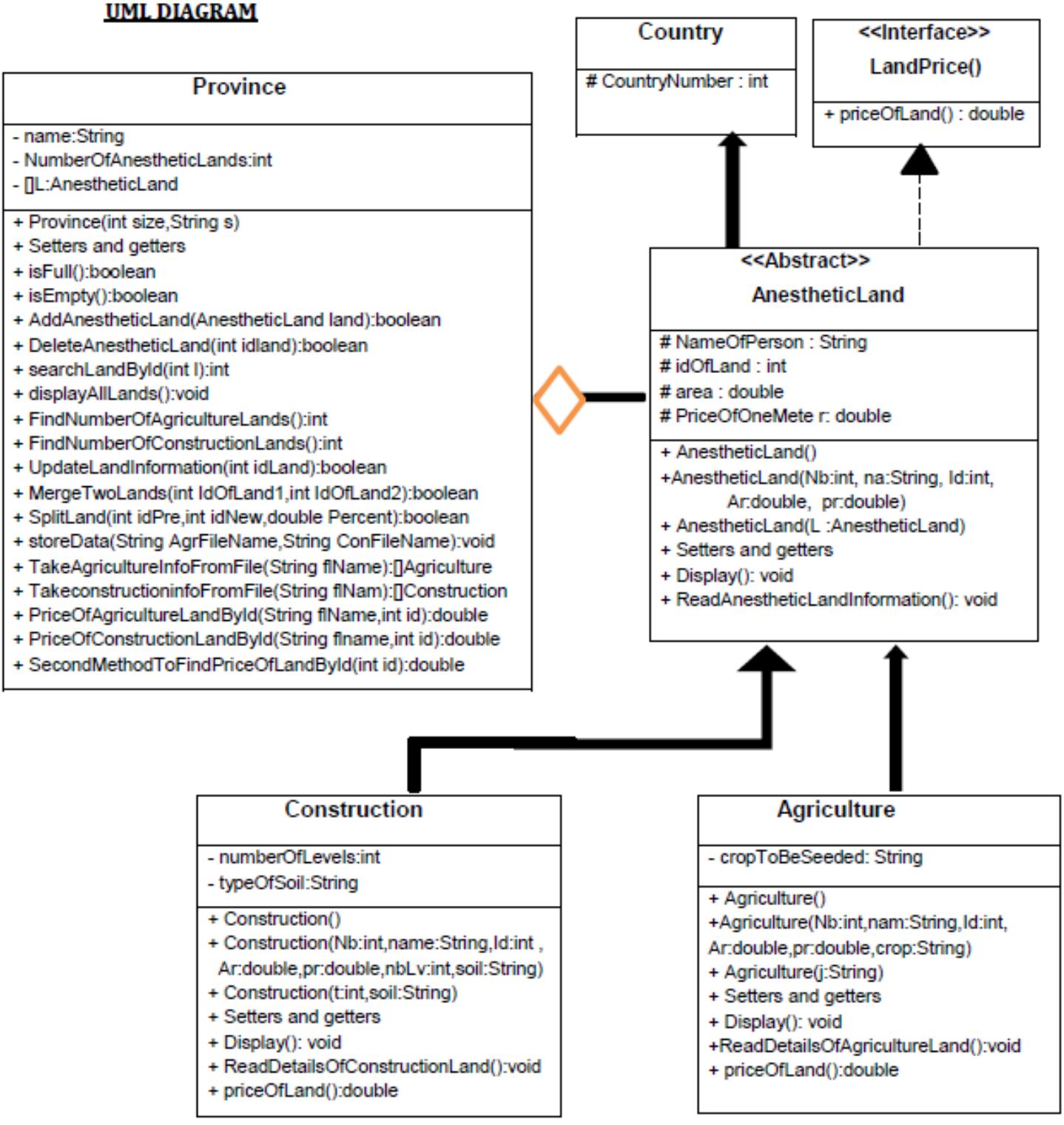
Technology has been developing for many years so people can use it in their daily life and make their life style much easier. When we say technology, we are talking about artificial intelligence that depends, in one way or another, on programming. We all use this language in everything around us without even knowing it.
The biggest problem Lebanese people are facing every day, is waiting in turn in most of the governmental departments, which is still depending on the old filing system.

Time is money. To save time and money, all the governmental departments and social services must forget what does paper mean and be always up-to-date.

In my project, I try to give a new beginning for the filing system that can be used in land sectors. Some of benefits are merging and splitting lands.
Using this system, people can know all the information they need of any land using search. Also, they will be able to sell / buy any land and register it in the government. As for engineers and contractors, it will help them take any license to their work without spending any time waiting in turn.

For your final project in Object Oriented Programming (OOP), it is required to implement a program based on the following UML diagram:

UML DIAGRAM


Country
CountryNumber : int
<<Interface>>
LandPrice()
+ priceOfLand() : double
Province
- name:String
- NumberOfAnestheticLands:int
- []L:AnestheticLand
+ Province(int size,String s)
+ Setters and getters
+ isFull():boolean
+ isEmpty():boolean
+ AddAnestheticLand(AnestheticLand land):boolean
+ DeleteAnestheticLand(int idland):boolean
+ searchLandById(int l):int
+ displayAllLands():void
+ FindNumberOfAgricultureLands():int
+ FindNumberOfConstructionLands():int
+ UpdateLandInformation(int idLand):boolean
+ MergeTwoLands(int IdOfLand1,int IdOfLand2):boolean
+ SplitLand(int idPre,int idNew,double Percent):boolean
+ storeData(String AgrFileName,String ConFileName):void
+ TakeAgricultureInfoFromFile(String flName):[]Agriculture
+ TakeconstructioninfoFromFile(String flNam):[]Construction
+ PriceOfAgricultureLandById(String flName,int id):double
+ PriceOfConstructionLandById(String flname,int id):double
+ SecondMethodToFindPriceOfLandById(int id):double
<<Abstract>>
AnestheticLand
NameOfPerson : String
idOfLand : int
area : double
PriceOfOneMete r: double
+ AnestheticLand()
+AnestheticLand(Nb:int, na:String, Id:int,
Ar:double, pr:double)
+ AnestheticLand(L :AnestheticLand)
+ Setters and getters
+ Display(): void
+ ReadAnestheticLandInformation(): void
Construction
- numberOfLevels:int
- typeOfSoil:String
+ Construction()
+ Construction(Nb:int,name:String,Id:int ,
Ar:double,pr:double,nbLv:int,soil:String)
+ Construction(t:int,soil:String)
+ Setters and getters
+ Display(): void
+ ReadDetailsOfConstructionLand():void
+ priceOfLand():double
Agriculture
- cropToBeSeeded: String
+ Agriculture()
+Agriculture(Nb:int,nam:String,Id:int,
Ar:double,pr:double,crop:String)
+ Agriculture(j:String)
+ Setters and getters
+ Display(): void
+ReadDetailsOfAgricultureLand():void
+ priceOfLand():double

Your task is to implement all the necessary classes, taking into consideration the relationships between classes as shown in the diagram.

The interface contains a method to contain the price of a specific land. The abstract class contains a constructor, a method to display various details about a land and another method to read details about a specific land.

The class **Agriculture** contains multiple constructors, a method to display details about the agriculture land, a method that reads details from user about a specific agriculture land and a method for returning the price of a specific agriculture land.

The class **construction** contains constructors, a method to display details about a specific construction land, a method that reads details from user about a specific construction land and a method that returns the price of a specific construction land

The class **Province** contains the following methods:

- A method that returns a boolean value indicating whether the province (array of lands) is full or not
- A method that returns a boolean value indicating whether the province (array of lands) is empty or not
- A method that receives a land object and adds it to the Province whenever the number of lands is less than the size of array
- A method that checks if the land exists in the province using a search method and returns a boolean value. If yes, the land is deleted from the province and the total number of lands is decreased
- A search method that receives a land identifier and verifies if it is available in the province, and then it returns the index where the land is found
- A method displaying the details of all lands in the province

- A method that returns the number of lands of type agriculture in the province
- A method that returns the number of lands of type construction in the province
- A method for updating the land that takes land identifier from the user and apply modifications according to the entered identifier. This method returns a boolean value to indicate whether the update has succeeded or failed
- A method for merging two lands. This method reads two identifiers representing the lands to be merged, put them in the first land having the first identifier and delete the second land. It should return a boolean value to indicate whether the merge occurs or not
- A method for splitting two lands where it reads from the users the old identifier of a land and the identifier of a new land after split. This method then splits the land in two, the first part has the old land's identifier and the second part has the new land's identifier. This method also takes a percentage that is used for determining the area of the new land
- A method to save all agriculture lands in file and all construction lands in another file
- A method returning an array containing all agriculture lands after reading them from file
- A method returning an array containing all construction lands after reading them from file
- A method calculating the price of an agriculture land which reads from the user the identifier of an agricultural land and the name of the file containing it and then returns the price of this land

- A method calculating the price of a construction land that reads from the user the identifier of a construction land and the name of the file containing it and then returns the price of this land
- A method returning the price of a land whose identifier is entered by the user

After implementing all the classes, you need to implement the main method that takes the necessary input from the user, tests all the methods in all the previous classes and generates the required output.

Make sure that your code follows the principles of OOP, such as encapsulation, inheritance, and polymorphism.

Make sure to include comments or any documentation/explanation for any element in your code when necessary so that your code is clear. Please note that you will be asked to present your work during the last lab of the semester.

The deadline of submission of your work is the week from May 1 to 5, 2023. You will receive an announcement from your TA's including submission links, the specified deadline for each section as well as additional guidelines for your project submission and presentation.

If you have any questions, please don't hesitate to reach out your TA Lab.

Best of luck in your work