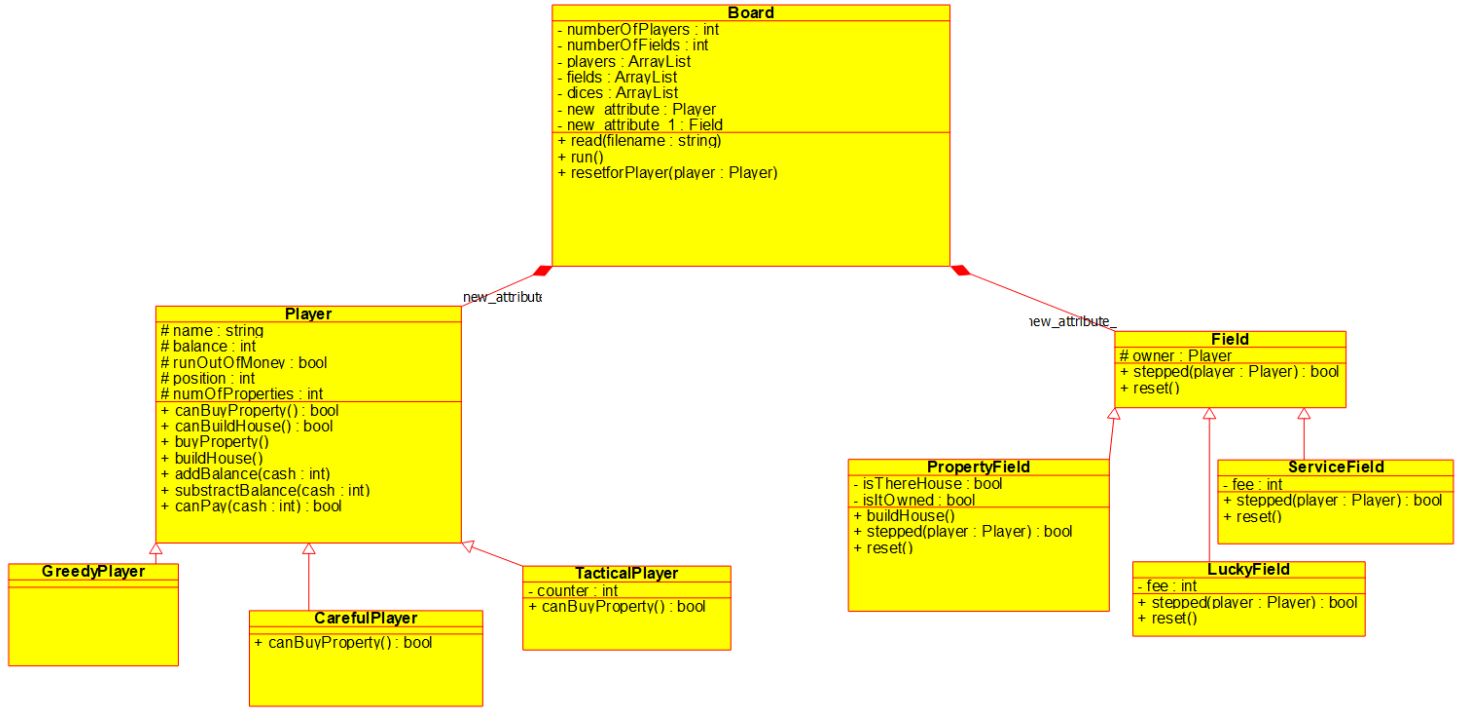
Rasul Khanbayov GGUSA3 -- Assignment – Task 3

Description of the task.

So basically we have Board. There are players and fields on the board. Players has initially 10000 money. Each player has different strategy. These are: Greedy, Careful and Tactical. We also have different fields. Fields are also divided into 3 parts: Property, Lucky, Service. Lucky field gives money to the player. However Service field gets money from player. Players can buy property for 1000, can build house for 4000. But we can not buy and build house in 1 turn. If player comes to the property owned by somebody else then he pays 500 If there is no house, otherwise pays 2000. We have dice numbers. If we divide dice numbers to number of players we get number of rounds.

UML diagram



Description of the methods:

Player class:

1. canBuyPropety() – returns true if the balance of the player is above 1000, false otherwise.
2. canBuildHouse() – returns true checks if the balance of the player is above 4000, false otherwise.
3. buyProperty() – it is void function. I subtract 1000 from player's balance.
4. buildHouse() – it is void function. I substact 4000 from player's balance.
5. addBalance(cash : int) – it is void function. It adds cash to the balance.
6. Substract(cash : int) – it is void function. It subtracts cash from the balance.
7. canPay(cash : int) – returns true if balance is above the given cash.

CarefulPlayer class:

1. canBuyProperty() – it overrides the method in its super class. It returns true if he his balance is more than 2000 basically.

TacticalPlayer class:

1. canBuyProperty() - it overrides the method in its super class. It returns true if he his balance to more than 1000 and counter is odd number.

Field class:

1. stepped(player : Player) – an abstract function.
2. reset() – an abstract function.

ServiceField class:

1. stepped(player : Player) – It sees if player can pay fee, if yes it returns true and subtracts money, or false otherwise.
2. reset() – Because Service Field is not owned by somebody it is empty function

LuckyField class:

1. stepped(player : Player) – It returns true and adds money
2. reset() – Because Lucky Field is not owned by somebody it is empty function

PropertyField class:

1. buildHouse() – It is void function. It sets isThereHouse attribute to true.
2. stepped(player : Player) – This method checks if property is owned or not. If yes player pays penalty and returns true, if player cannot pay the method returns false. If property is not owned, player checks if he can buy it. But if he already owns player checks if can build house.
3. reset() – it is void function. It sets owner to none. Other two Boolean attributes to false.

Board class:

1. Read(filename:string) – It gets a filename. It reads elements one by one. Then assign them for the corresponding places.
2. Run() – based on the dices players positions change and they buy property and so on.
3. resetForPlayer(player : Player) – This method basically resets fields that owned by the player.

<u>Testing</u>

1. Input – People and fields plays. There is a winner at the end after 15^{th} round.
2. Test1 – There is one property first player buys. Second one pays. Then first one builds house and second player pays again
3. Test2 – This time there is 2 properties. And each of them own one. They pay and build house.
4. Test3 – I am checking Service Field. Fee is 5000. After 2 rounds all of them has 0 in balance
5. Test4 – I am checking Lucky Field. Fee is 1000. After 6 rounds all has 16000 in their balance.
6. Test5 – Testing Tactical Player. He buys first time but doesn’t second time.
7. Test6 – Testing Careful Player. When he has 2000 in balance he doesn’t buy house.
8. Test7 – Invalid Input.