DOCUMENTATION

Description of project:

Simulate a simplified Capital game. There are some players with different strategies, and a cyclical board with several fields. Players can move around the board, by moving forward with the amount they rolled with a dice. A field can be a property, service, or lucky field. A property can be bought for 1000, and stepping on it the next time the player can build a house on it for 4000. If a player steps on a property field which is owned by somebody else, the player should pay to the owner 500, if there is no house on the field, or 2000, if there is a house on it. Stepping on a service field, the player should pay to the bank (the amount of money is a parameter of the field). Stepping on a lucky field, the player gets some money (the amount is defined as a parameter of the field). There are three different kind of strategies exist. Initially, every player has 10000.

Greedy player: If he steps on an unowned property, or his own property without a house, he starts buying it, if he has enough money for it.

Careful player: he buys in a round only for at most half the amount of his money.

Tactical player: he skips each second chance when he could buy. If a player has to pay, but he runs out of money because of this, he loses. In this case, his properties are lost, and become free to buy.

Read the parameters of the game from a text file. This file defines the number of fields, and then defines them. We know about all fields: the type. If a field is a service or lucky field, the cost of it is also defined. After these parameters, the file tells the number of the players, and then enumerates the players with their names and strategies. In order to prepare the program for testing, make it possible to the program to read the roll dices from the file.

Print out which player loses as a second loser.

To user:

Data in the input text file should be like this.

- First line – Number of fields (N)

The next N lines should contain the description of the fields, in the following way

First Character – Type of the field

- L for Lucky field
- S for Service
- P for Property

It cannot be anything else.

The second Character after a space will be the cost of field, If a field is a service or lucky field. It should be integer.

- The line after above - Number of players (N)

The next N lines should contain the description of the players, in the following way

First Character – Strategy of the player

- G for Greedy player
- C for Careful player
- T for Tactical player

It cannot be anything else.

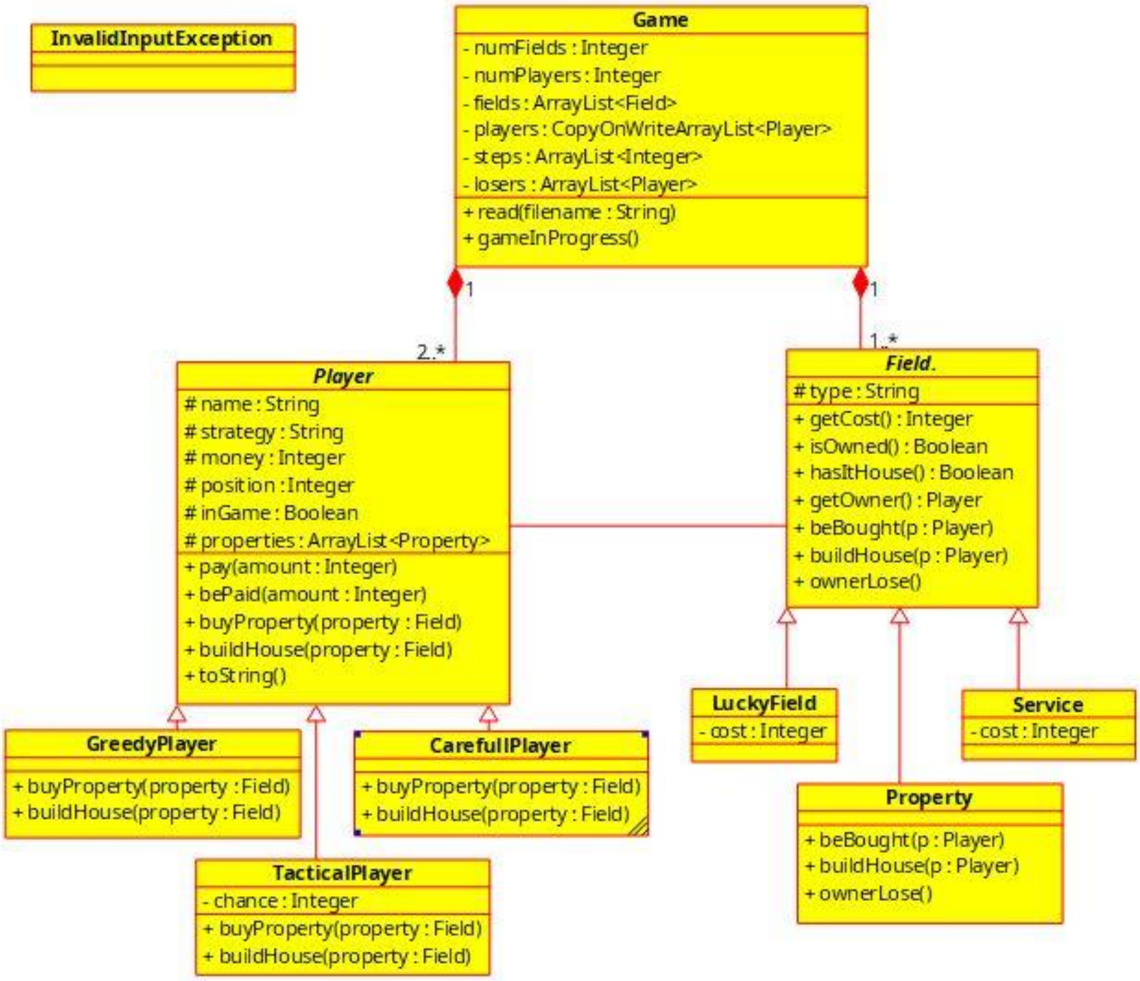
The second Character after a space will be the name of player. It should be String.

- The line after above - the roll dices.

An example of a sample input:

```
5
S 1000
L 0
P
P
S 2000
4
G Alex
T Sara
C Mary
T Jim
3 2 5 1 2 2 1 1 1 2 4 4 1 6 6 4 1 5 3 4 1 2 3 3 5 1 2 1 1 4 1 4 1 4 3
```

UML Diagram:



Test Cases:

Sample Input 1:

5

S 1000

L 0

P

P

S 2000

4

G Alex

T Sara

C Mary

T Jim

3 2 5 1 2 2 1 1 1 2 4 4 1 6 6 4 1 5 3 4 1 2 3 3 5 1 2 1 1 4 1 4 1 4 3

Sample output 1:

```
Player{name=Mary, strategy=Careful_Player} loses as a second loser.
```

Sample Input 2:

```
6
S 1000
S 2000
L 100
P
P
S 2000
3
G Alex
C Mary
T Jim
5 3 5 1 2 2 1 1 1 6 4 4 1 6 6 4 1 5 3 4 1 2 3 3 5 1 2 1 1 4 1 4 1 4 3
```

Sample output 2:

```
Player{name=Mary, strategy=Careful_Player} loses as a second loser.
```

Sample Input 3:

```
5
S 3000
L 300
P
P
S 2000
3
G Alex
C Mary
T Jim
1 3 5 1 1 2 1 1 1 6 4 4 5 6 6 4 1 5 3 4 1 2 4 3 5 1 2 1 1 4 1 4 1 4 3 3 4 5 6
```

Sample output 3:

```
Player{name=Jim, strategy=Tactical_Player} loses as a second loser.
```

Sample Input 4:

```
5
S 1000
L 300
P
S 2000
S 2000
4
G Alex
T Sara
C Mary
T Jim
1 3 5 1 1 2 1 1 1 6 4 4 5 6 6 4 1 5 3 4 1 2 4 3 5 1 2 1 1 4 1 4 1 4 3 4 3 2 1
```

Sample output 4:

```
Player{name=Alex, strategy=Greedy_Player} loses as a second loser.
```

Sample Input 5:

```
6
S 1000
S 2000
L 0
P
S 2000
P
4
G Alex
T Sara
C Mary
T Jim
1 1 1 1 1 3 5 1 1 2 1 1 1 6 4 4 5 6 6 4 1 5 3 4 1 2 4 3 5 1 2 1 1 4 1 4 1 4 3 5 4 3 2
```

Sample output 5:

```
Player{name=Sara, strategy=Tactical_Player} loses as a second loser.
```

Sample Input 6:

```
3
S 5000
L 100
P
4
C Lisa
G Mary
T Alex
T Jim
3 3 3 6 1 3 5 1 1 2 1 1 1 6 4 4 5 6 6 4 1 5 3 4 1 2 4 3 5 1 2 1 1 4 1 4 1 4 3
```

Sample output 6:

```
Player{name=Alex, strategy=Tactical_Player} loses as a second loser.
```

Sample Input 7:

```
4
S 4000
P
L 10
P
4
G Sara
T Jim
C Mary
G Alex
4 4 4 4 1 3 5 1 2 1 1 1 6 4 4 5 6 6 4 1 5 3 4 1 2 4 3 5 1 2 1 1 4 1 4 1 4 3 3 4 5 6
```

Sample output 7:

```
Player{name=Sara, strategy=Greedy_Player} loses as a second loser.
```

Sample Input 8:

```
5
S 6000
L 0
P
P
S 2000
5
G Alex
T Lily
G Kate
C Mary
T Jim
3 2 3 1 2 1 2 1 1 2 4 4 1 6 6 4 1 5 3 4  1 2 3 3 5 1 2 1 1 4 1 4 1 4 3
```

Sample output 8:

```
Player{name=Jim, strategy=Tactical_Player} loses as a second loser.
```

Sample Input 9:

```
4
S 2000
L 20
P
S 3000
5
G Alex
T Lily
G Kate
T Mary
T Jim
1 1 4 1 4 1 4 3 1 2 4 4 1 6 6 4 1 5 3 4  1 2 3 3 5 1 2 1 1 4 1 4 1 4 3
```

Sample output 9:

```
Player{name=Kate, strategy=Greedy_Player} loses as a second loser.
```

Sample Input 10:

```
7
P
L 20
S 2000
L 10
S 3000
P
S 2000
4
C Lily
G Alex
G Kate
T Jim
1 6 6 4 1 5 4 3 1 2 4 4 1 6 6 4 1 5 3 4 1 2 3 3 5 1 2 1 1 4 1 4 1 4 3
```

Sample output 10:

```
Player{name=Kate, strategy=Greedy_Player} loses as a second loser.
```