A project made by:

BENYEMNA Hamza

BITAR Aref

DIA2

Project Design Pattern and Software Development Process : Monopoly

I – Introduction:

The project is about recreating a simplified Monopoly game using many design pattern notions seen during the class this semester.

We have between 2 and 6 players. Every player has to roll 2 dices and move through the board according to his roll score.

Indeed, it is demanded to code a game that has:

- A continuous board of 40 cells (rounds are possible because the cells 39 and 0 are connected)
- A jail where to go when a triple double dice is made or when we arrive at the Go To Jail cell (position 30).
- We have to make a double dice in order to go out of the Jail.

This simplified Monopoly is using the rudimentary rules and concepts of the Monopoly, but we had still the possibility to go a little bit further if we can use more **Design Patterns** and some of general concepts of **Oriented Object Programming** (**encapsulation** for example).

But we wanted to go further.

We barely added different kind of Cells; Some for **taxe**s paid by the Players (Community Chest, Income Tax, Luxury Tax), some are with **random issues** (Chance with what we can lose or earn a random amount of money), some are Properties (Train Station, Services or Houses) that can be bought by Players in order to **earn a rent** when other Players are in it. The Free Parking is a safe Cell for everyone and every completed round by passing the Start Cell, the Players could earn money. For more fun, it is playable between **2 and 6** Players and the game ends when everybody his **bankrupted** (have no money) except one survival who is the **winner**!

II – Design Hypothesis:

Our goal is to employ methods in order to:

- Reduce the quantity of code (**optimization**)
- Simplify the creation of instances and their securitization (**verification**)
- Develop a solution that works in a dynamic context of changes, which is at the core of Design Pattern purpose (**reusability**).

We have to cope with a game to develop that is based on the reusability and dynamic changes during the game. We had 2 mains axes of thinking:

- How to simplify the creation and definition in every t time of **Cell** and **Properties** ?
- How to create a unique **Board** with a unique access point to this ?

The first point led us to think about **Factory** and **Decorator**, and the second one about **Singleton** Design Pattern.

We developed 3 Design Pattern methods through this project.

Singleton:

This is a **Creation Pattern** concept. As said before, this project needs a unique Board during the whole game. Otherwise, this is the best Pattern to use in this case. Indeed, we design a Board which is unique but composed of different kind of Cells.

Factory:

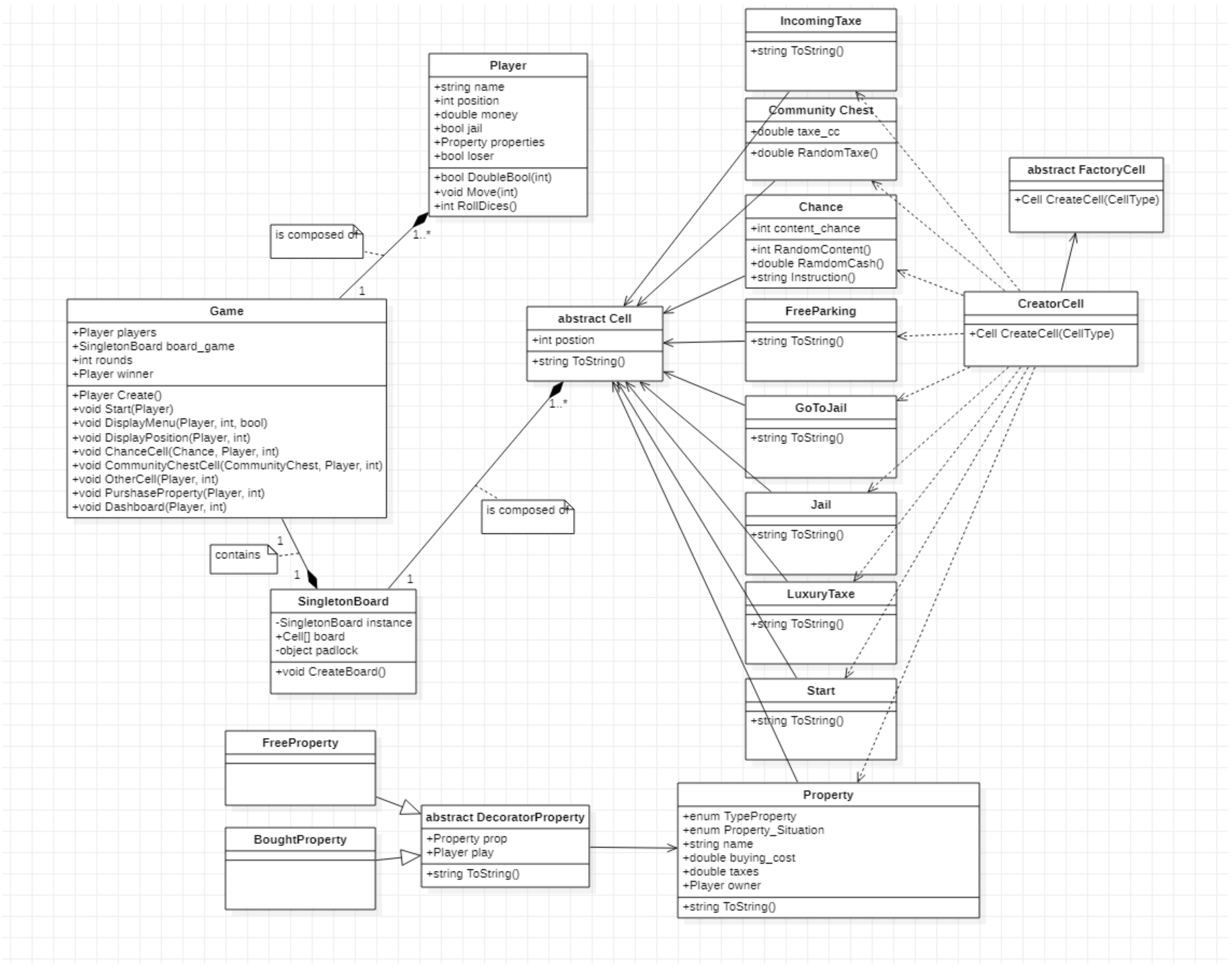
This is also a **Creation Pattern** concept. This method is really efficient when we have many objects that has the same core but are specified. In this case, we used it to create Cells. We don't instantiate Cells because it is nonsense but the specific ones that we can find on the Board. This way, Start, Jail , etc. are produced by the same pattern but for specific uses and this reusability of the **Factory** is essential regarding the 40 that it produced.

Decorator:

Lastly, we have this **Structural Pattern** concept. The aim of this project is to produce something dynamic that can be adapted whenever we need during the game without knowing especially what it is about to happen. Indeed, we use it to settle the types of Properties with different situations that can change during the game.

III – UML Diagrams:

A – Diagram of the final solution:



Explanation of the UML Diagram:

We have the Game composed of some Players and needs a unique Board. So, without Players and a Board, there is no Game of Monopoly possible (**composition**).

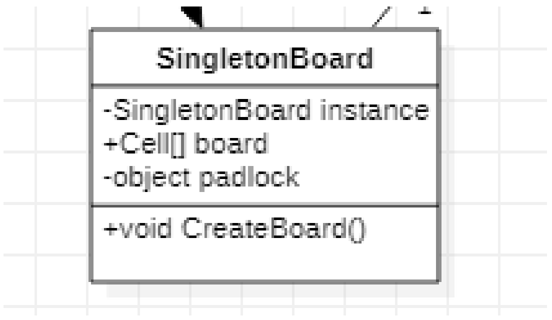
A Board need several Cells without what it would not exist (**composition**).

The Cells are not instantiated, but some classes are inheriting from it. This is why Cell class is **abstract** and others as FreeParking are **sealed**.

The UML part concerning the patterns are more explained bellow.

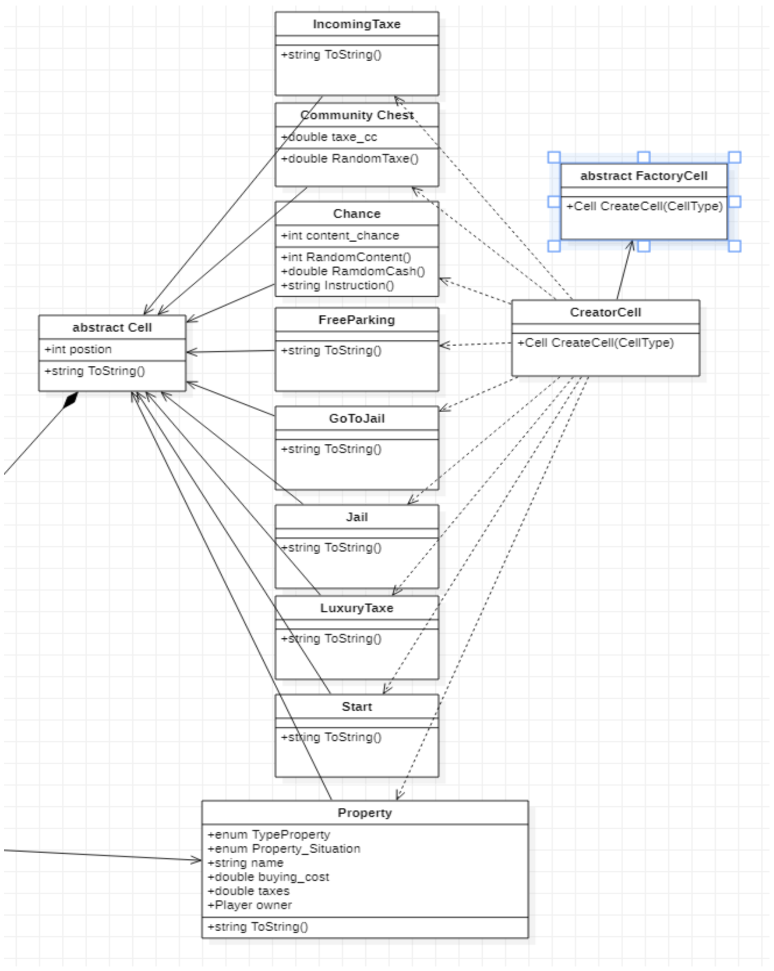
B – Sequence diagram:

a) Singleton pattern:



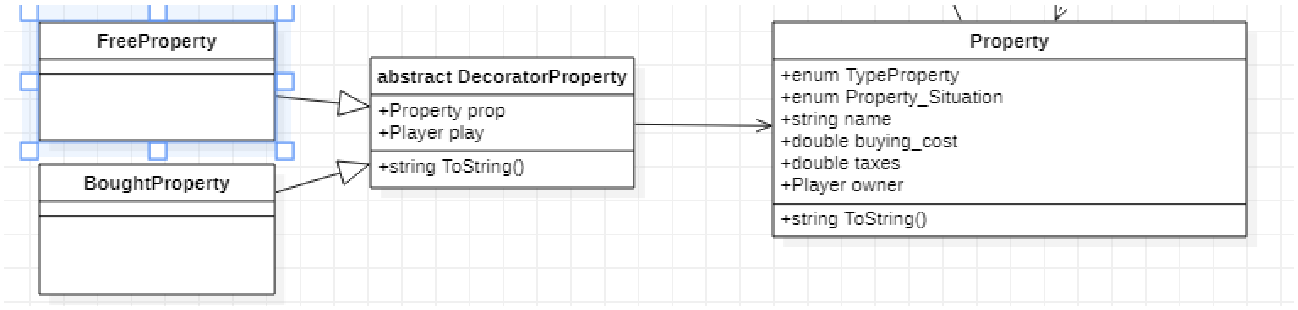
The use of a **Singleton** pattern allows to have only one instance of the Board alive at whatever moment of the Game. Indeed, this is essential to keep the same Board during the entire Game and ensure that no other is created. This is possible thanks to this pattern added to a function **padlock** that allow an Instance to keep sure that it is unique.

b) Factory pattern:



This pattern helps creating Cells more efficiently for the game because Cells are a **dynamic** object whereas there aren't two similar properties on the board. **Factory** pattern is really effective in such case where you need a large **flexibility** as in our case. Here, we have a FactoryCell which is at the core of this pattern. The "builder" that create concrete Cells is the CreatorCell that inherits from the Factory one. Indeed, it directly creates Properties, Jails, Start, etc. which are all Cells.

c) Decorator pattern:

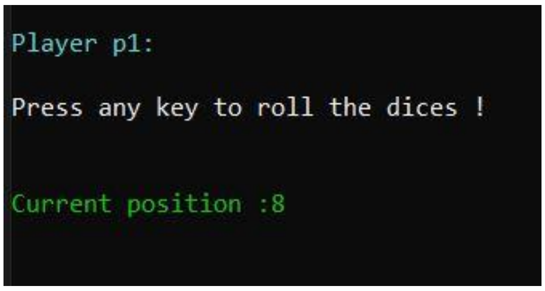


This pattern allows to the Properties to **evolve** during the game. In terms of their situation, which is sold or not(BoughProperty and FreeProperty). As in the **Factory** pattern, we don't implement Properties but "**Decorated Properties**" as Bough or Free with different specificities (a Bough Property has a tax that a Player has to pay when he arrives on this Cell).

IV) - Test Case:

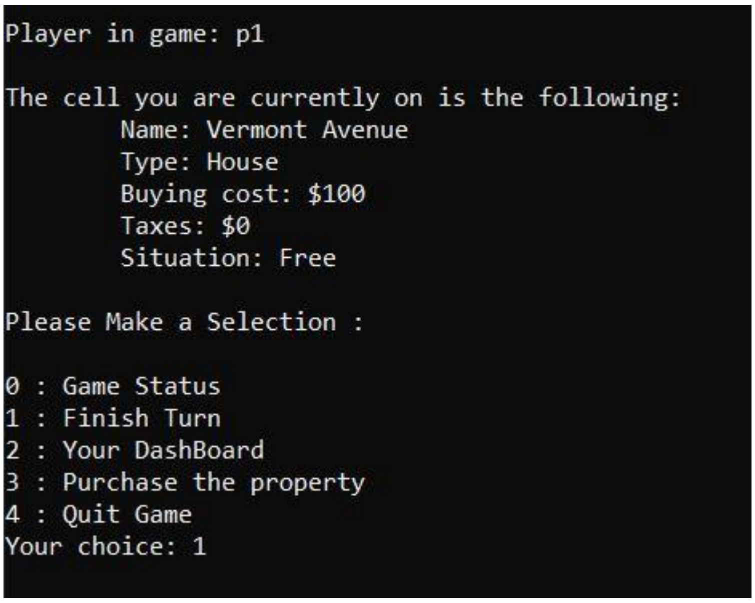
We implemented **4 test cases**. To settle it, we implemented Players and the Game in an advance state of the Game if needed. So, thanks to this **inputs** , we have **outputs** that illustrates different functionalities of our project.

Double Dice:

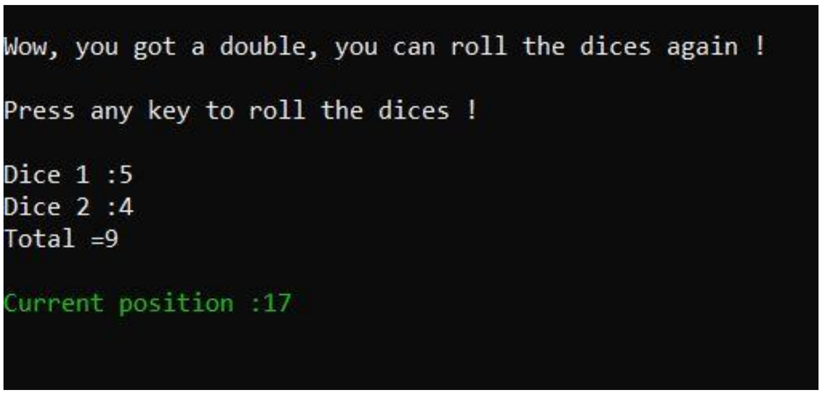


```
Player p1:

Press any key to roll the dices !


Current position :8
```

The Player p1 is defined to obtain a double (4,4) after the dice roll. Indeed, he moves to the Cell 8.

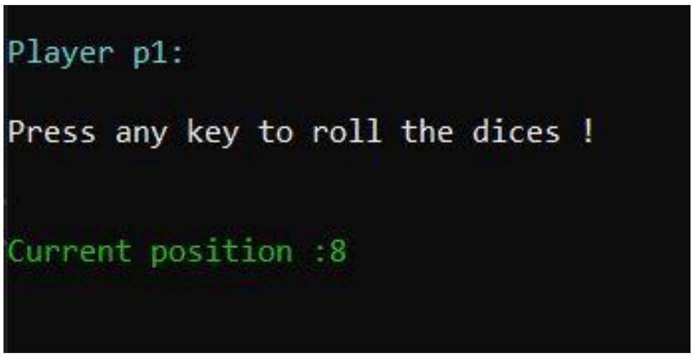

```
Player in game: p1

The cell you are currently on is the following:
        Name: Vermont Avenue
        Type: House
        Buying cost: $100
        Taxes: $0
        Situation: Free

Please Make a Selection :

0 : Game Status
1 : Finish Turn
2 : Your DashBoard
3 : Purchase the property
4 : Quit Game
Your choice: 1
```

The Player p1 then chose to finish his turn without doing nothing.

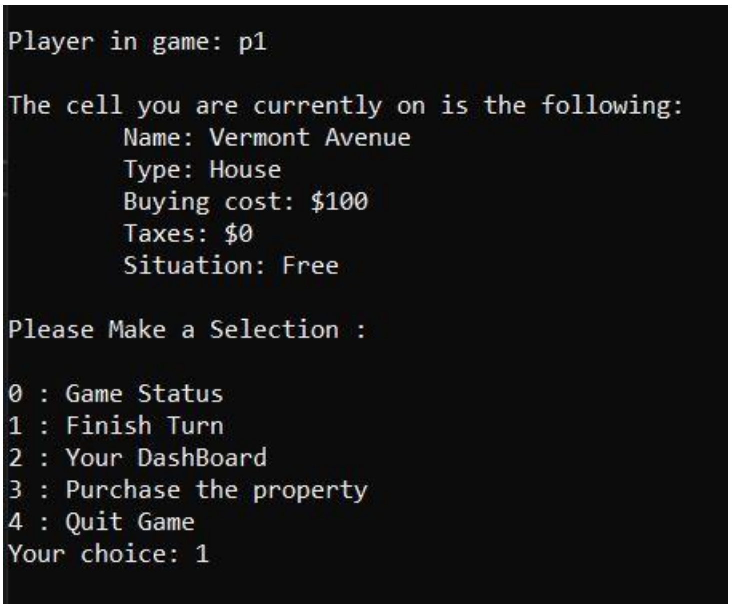

```
Wow, you got a double, you can roll the dices again !

Press any key to roll the dices !

Dice 1 :5
Dice 2 :4
Total =9

Current position :17
```

After that, as he made a double, he rolls the dice twice ! It ends here because this roll ended on a (5,4).

Triple Dice:

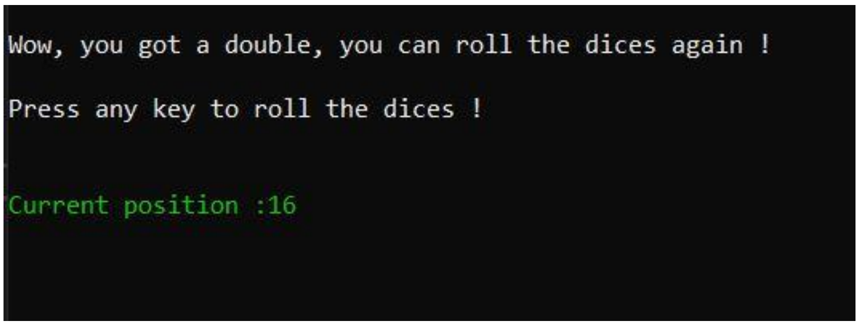

```
Player p1:

Press any key to roll the dices !


Current position :8
```

We implemented the same double (4,4) for Player p1.



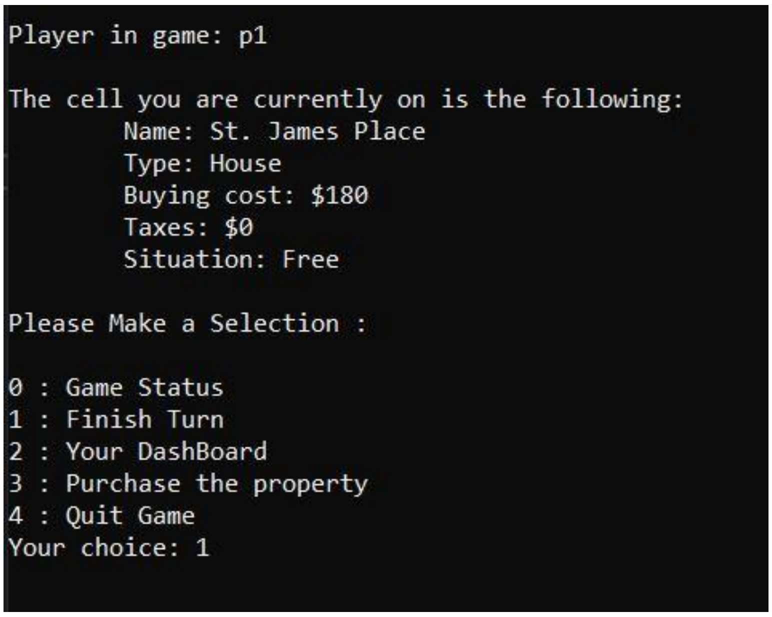
```
Player in game: p1

The cell you are currently on is the following:
        Name: Vermont Avenue
        Type: House
        Buying cost: $100
        Taxes: $0
        Situation: Free

Please Make a Selection :

0 : Game Status
1 : Finish Turn
2 : Your DashBoard
3 : Purchase the property
4 : Quit Game
Your choice: 1
```

P1 is in the Cell 8. He decides to finish his turn.



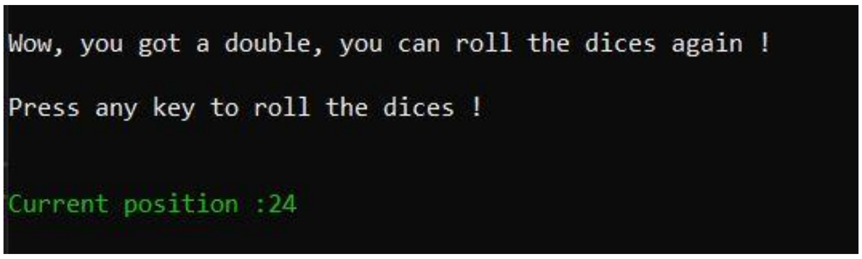
```
Wow, you got a double, you can roll the dices again !

Press any key to roll the dices !


Current position :16
```

He redoes the same double (4,4). So, he is at the Cell 16.



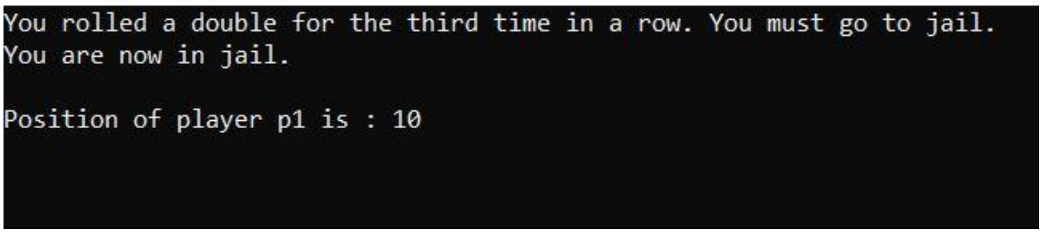
```
Player in game: p1

The cell you are currently on is the following:
        Name: St. James Place
        Type: House
        Buying cost: $180
        Taxes: $0
        Situation: Free

Please Make a Selection :

0 : Game Status
1 : Finish Turn
2 : Your DashBoard
3 : Purchase the property
4 : Quit Game
Your choice: 1
```

He finishes his turn.



```
Wow, you got a double, you can roll the dices again !

Press any key to roll the dices !


Current position :24
```

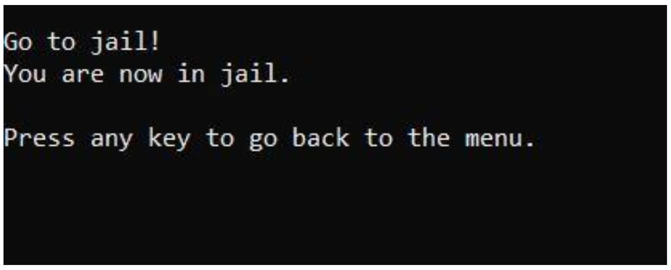
The 3rd double is done. P1 is at the Cell 24.



```
You rolled a double for the third time in a row. You must go to jail.
You are now in jail.

Position of player p1 is : 10
```

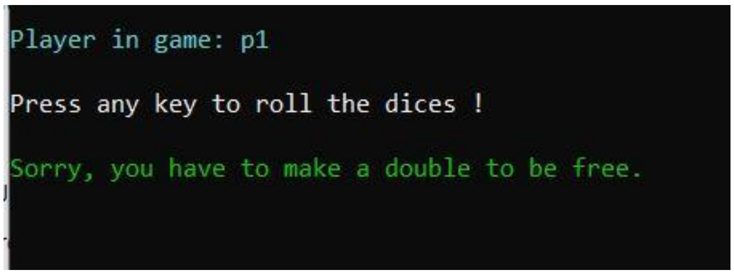
As he made a 3rd double, p1 goes to Jail at Cell 10.

Jail:



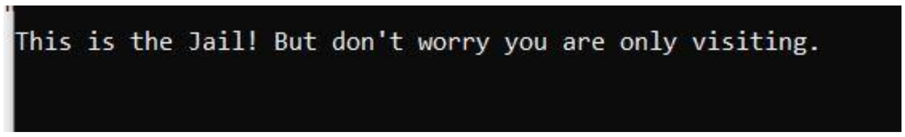
```
Go to jail!
You are now in jail.

Press any key to go back to the menu.
```

P1 has unfortunately visited Cell Go To Jail.



```
Player in game: p1

Press any key to roll the dices !

Sorry, you have to make a double to be free.
```

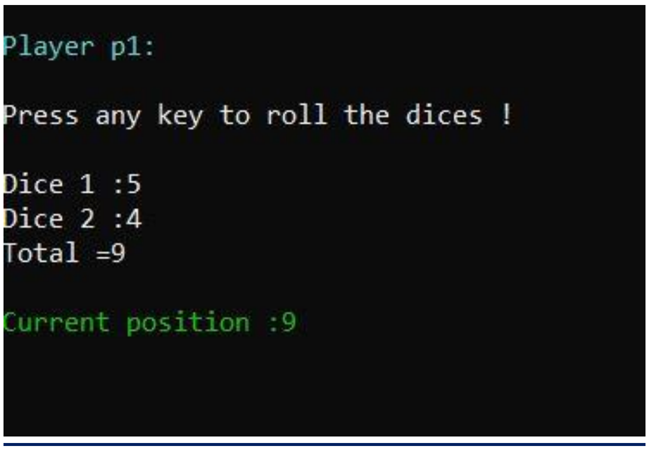
After that he has to do a double in order to be free.



```
This is the Jail! But don't worry you are only visiting.
```

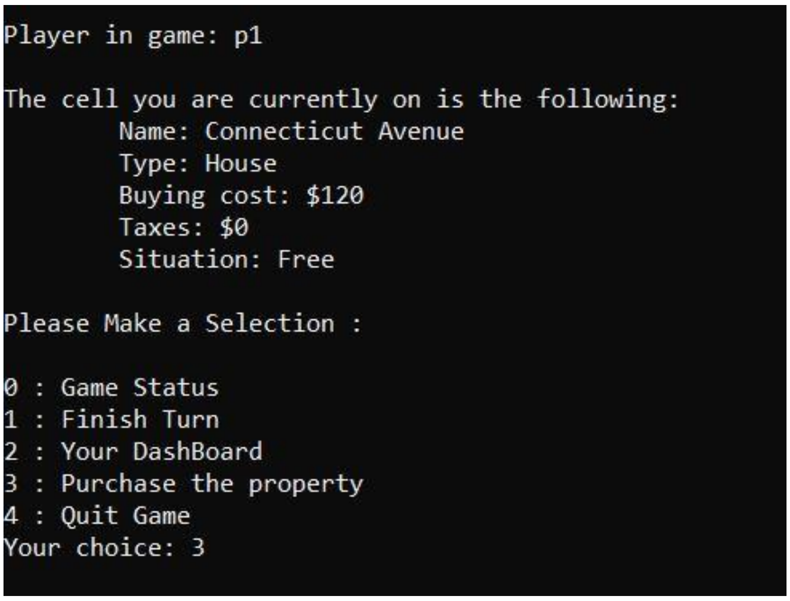
Visiting Cell 10 don't put us in prison.

Buy a Property:



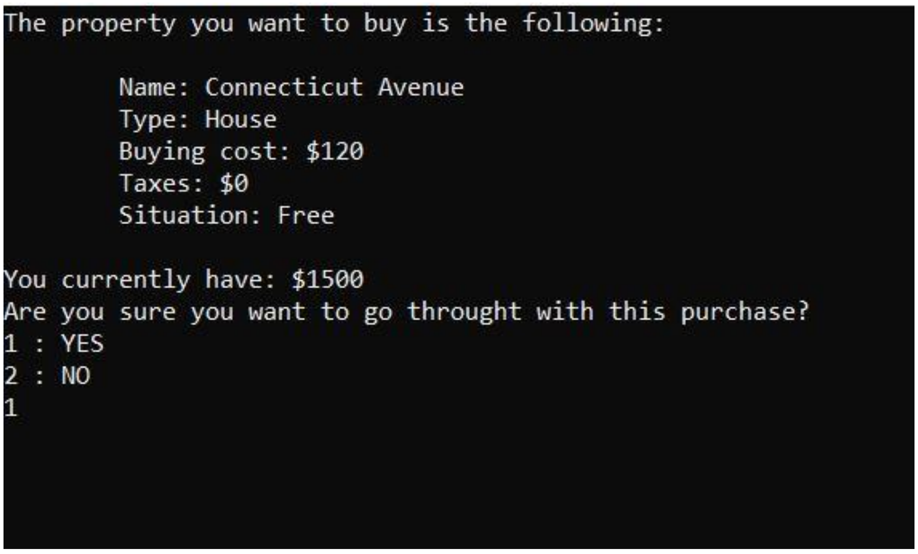
```
Player p1:

Press any key to roll the dices !

Dice 1 :5
Dice 2 :4
Total =9

Current position :9
```

P1 is settled to be at the FreeProperty in the Cell 9 with a (5,4) dice roll.



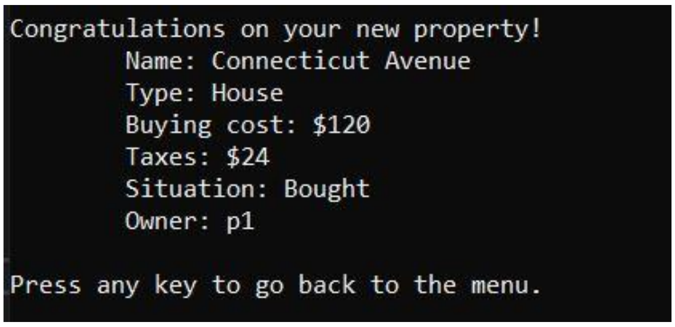
```
Player in game: p1

The cell you are currently on is the following:
        Name: Connecticut Avenue
        Type: House
        Buying cost: $120
        Taxes: $0
        Situation: Free

Please Make a Selection :

0 : Game Status
1 : Finish Turn
2 : Your DashBoard
3 : Purchase the property
4 : Quit Game
Your choice: 3
```

P1 chose the option "*purchase the property*" by taping "*3*". This Property is a house and free, so we can buy it.



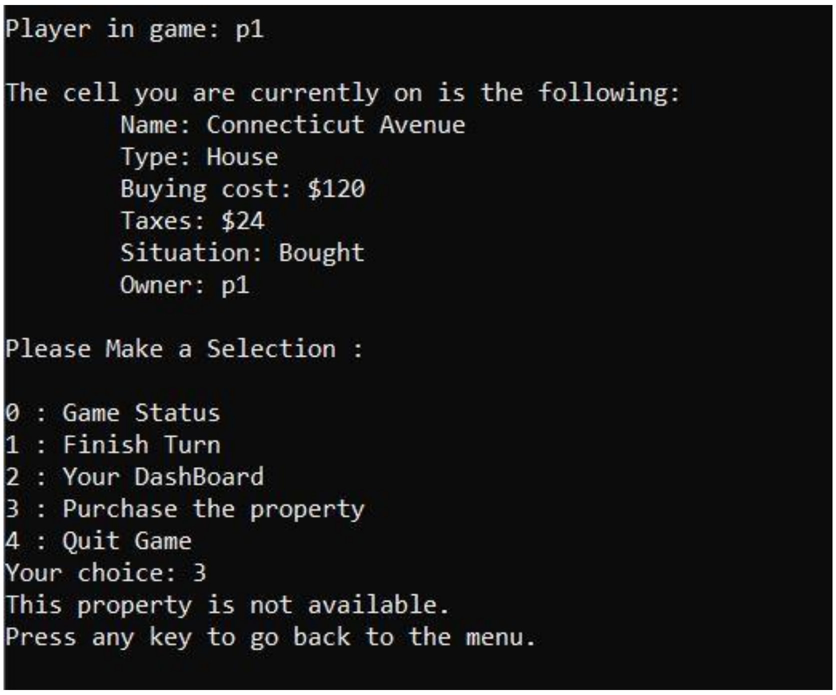
```
The property you want to buy is the following:

        Name: Connecticut Avenue
        Type: House
        Buying cost: $120
        Taxes: $0
        Situation: Free

You currently have: $1500
Are you sure you want to go throught with this purchase?
1 : YES
2 : NO
1
```

A validation message is needed to do the purchase. P1 validate the purchase by typing “1”.



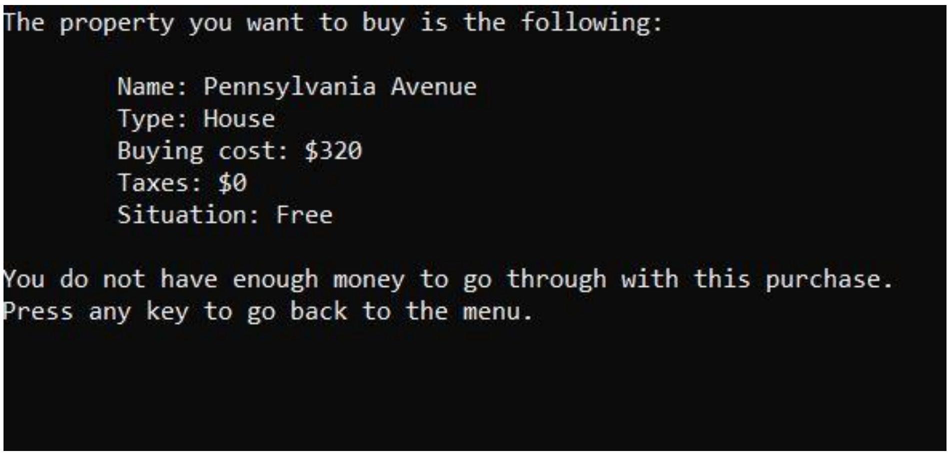
```
Congratulations on your new property!
        Name: Connecticut Avenue
        Type: House
        Buying cost: $120
        Taxes: $24
        Situation: Bought
        Owner: p1

Press any key to go back to the menu.
```

The House at Cell 8 belongs now to p1 !



```
Player in game: p1

The cell you are currently on is the following:
        Name: Connecticut Avenue
        Type: House
        Buying cost: $120
        Taxes: $24
        Situation: Bought
        Owner: p1

Please Make a Selection :

0 : Game Status
1 : Finish Turn
2 : Your DashBoard
3 : Purchase the property
4 : Quit Game
Your choice: 3
This property is not available.
Press any key to go back to the menu.
```

Furthermore, it is impossible now to purchase a Bough Property.



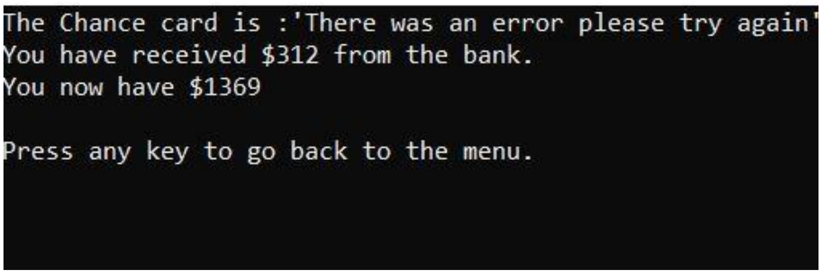
```
The property you want to buy is the following:

        Name: Pennsylvania Avenue
        Type: House
        Buying cost: $320
        Taxes: $0
        Situation: Free

You do not have enough money to go through with this purchase.
Press any key to go back to the menu.
```

Besides this, it is impossible to purchase a Property when a Player has not enough money.

V – Extra:

As explained, we wanted to go further in this project by coding a more complete and complex Game. We already presented the rules implemented. Let's see some screens of a party made until the end with the presentation of some added functionalities.

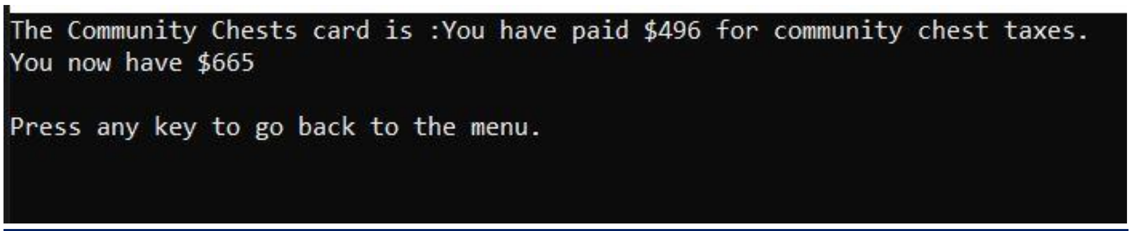
Chance:



```
The Chance card is :'There was an error please try again'
You have received $312 from the bank.
You now have $1369

Press any key to go back to the menu.
```

This Player was on a Chance Cell at the position 2, 22, or 33. He unfortunately paid a tax to the bank.

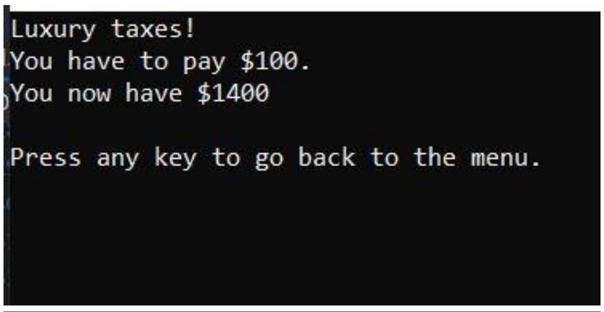
Community Chest:



```
The Community Chests card is :You have paid $496 for community chest taxes.
You now have $665

Press any key to go back to the menu.
```

This Player was in a Community Chest Cell at the position 7,17 or 36. He paid a tax to the bank.

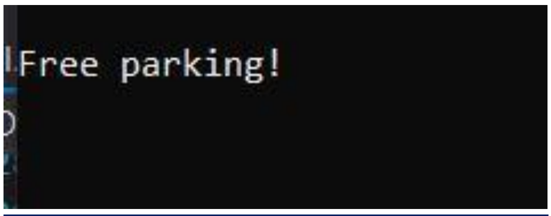
Luxury Tax:



```
Luxury taxes!
You have to pay $100.
You now have $1400

Press any key to go back to the menu.
```

This Player has paid a Luxury Tax of 100$ because he is in Cell 4.

Free Parking:

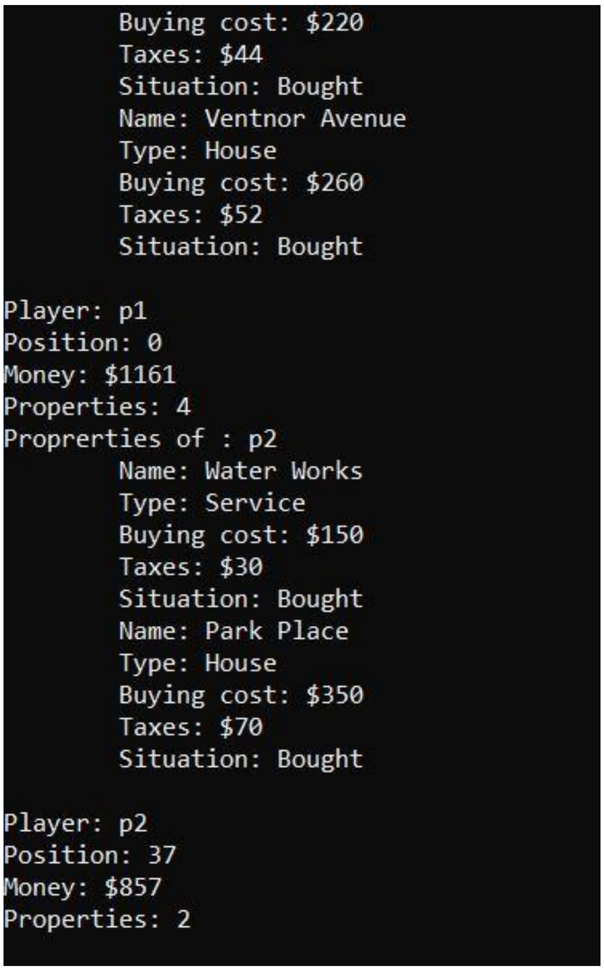


```
Free parking!
```

This Player has reached the Cell Free Parking at the position 20. He is safe of any tax in this case, he just has to pass his turn !

There are also visual interfaces in order to know how goes the Game or the Players.

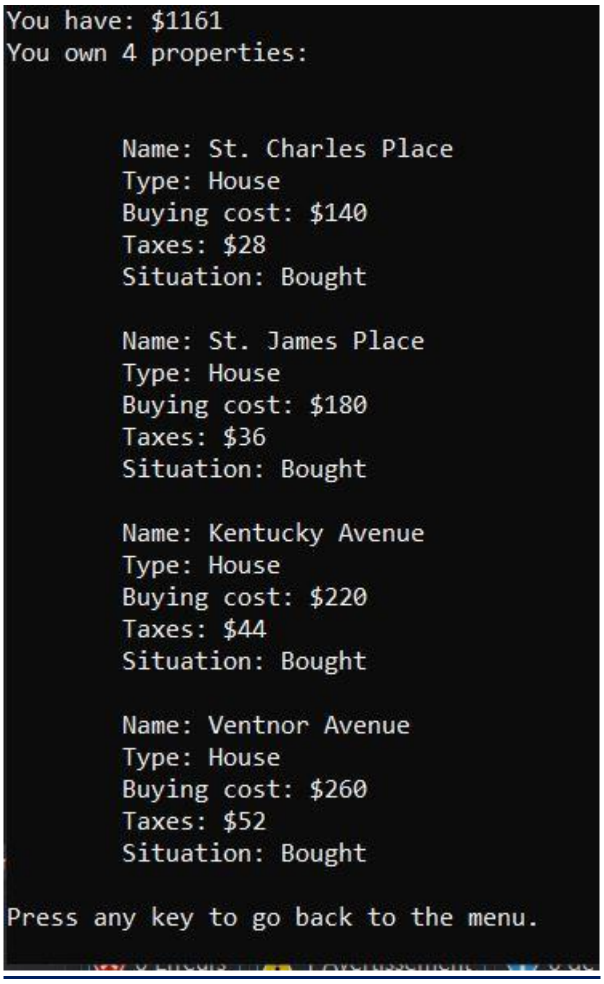
Game Status:



```
        Buying cost: $220
        Taxes: $44
        Situation: Bought
        Name: Ventnor Avenue
        Type: House
        Buying cost: $260
        Taxes: $52
        Situation: Bought

Player: p1
Position: 0
Money: $1161
Properties: 4
Proprerties of : p2
        Name: Water Works
        Type: Service
        Buying cost: $150
        Taxes: $30
        Situation: Bought
        Name: Park Place
        Type: House
        Buying cost: $350
        Taxes: $70
        Situation: Bought

Player: p2
Position: 37
Money: $857
Properties: 2
```

We have the state of a simulated Game that has gone until the end. You have all information about p1 and p2.

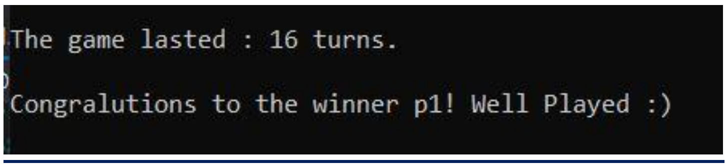
Dashboard:



```
You have: $1161
You own 4 properties:


        Name: St. Charles Place
        Type: House
        Buying cost: $140
        Taxes: $28
        Situation: Bought

        Name: St. James Place
        Type: House
        Buying cost: $180
        Taxes: $36
        Situation: Bought

        Name: Kentucky Avenue
        Type: House
        Buying cost: $220
        Taxes: $44
        Situation: Bought

        Name: Ventnor Avenue
        Type: House
        Buying cost: $260
        Taxes: $52
        Situation: Bought

Press any key to go back to the menu.
```

If the Player wants to know his money and all the Properties, he has.

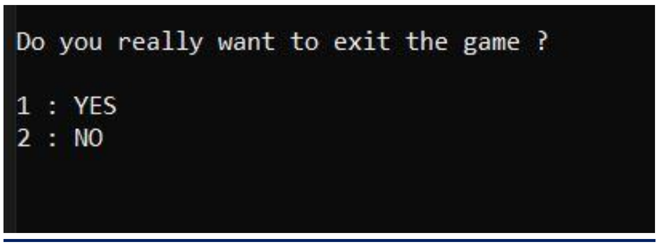
Winner:



```
The game lasted : 16 turns.

Congralutions to the winner p1! Well Played :)
```

The Player p1 has won this Game against p2 but with difficulty !

Exit the Game:



```
Do you really want to exit the game ?

1 : YES
2 : NO
```

It is possible to exit the Game. We have a validation shell. If no, we return to our menu of Player. If yes, the game is over, and the console shell is closed.