SDU

# Assignment 2 in Object-oriented Programming

<Iñigo Susilla Uanga>
<iusus23@student.sdu.dk>

**Table of contents**

# 1 Status of the Implementation

I have fulfilled all the sub tasks of the requirement. The most difficult part is to make the uno game work with all the other classes created.

## 1.1 Example output

- Provide a code snippet that demonstrates the output of the program (only a couple of rounds of gameplay).

```
How many Uno.players are we playing (2-9)
2
Name of the bot
BOT1
Name of the bot
BOT2
Your name:
IÑIGO
Adding player BOT1
Adding player BOT2
Adding player IÑIGO
```

```
-------------------------------------------------------
| Player: IÑIGO | Current top card RED_8 |
|-----------------------------------------------------|
| Player current hand and number:8   |
|  Reversecard_RED,  BLUE_7,  DRAW2_YELLOW,  DRAW2_GREEN,
BLUE_5, SKIP_RED, SKIP_YELLOW, YELLOW_7,  |
-------------------------------------------------------
Select one from your hand
0) Reversecard_RED
1) SKIP_RED
1
Playing:                            SKIP_RED
Enter to continue...
```

```
-------------------------------------------------------
| Player: IÑIGO | Current top card GREEN_9 |
|-----------------------------------------------------|
| Player current hand and number:1   |
| GREEN_8,  |
-------------------------------------------------------
Select one from your hand
```

```
0) GREEN_8
0
Playing:                                GREEN_8
the winner is: IÑIGO
```

# 2 Known Bugs

- As I tested I couldn't´t find any kind of bug

# 3 Design changes

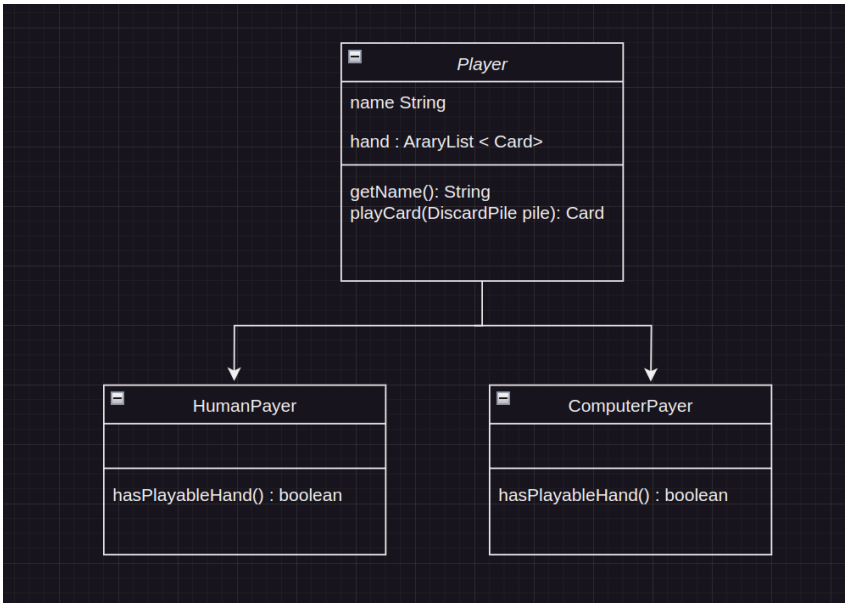
I created dome functions to make my life easier as getTopCard() or skipTurn().

# 4 Application of OOP Principles

## 4.1 Encapsulation

- I think that the whole proyect is an example of this because we cannot access the attributes from other classes to mantein the attributes “safe”.

## 4.2 Inheritance




In this case we clearly see the advantajes because the classes HumanPlayer and ComputerPlayer share some funcitions and attributes so we make another clasee that “share them”

### 4.3 Polymorphism

- When in the classes DrawPile and DiscardPiles we have some arraylist of the Class Card but we never create a Card object we are always creating instancies of their children that fit perfectly.

### 4.4 Abstraction

- With the function matches we have this because it as extension of the one before. I used a lot in the function matches. Because it is crated in the father class and later we must implement it in all his childs but in the father class is empty the body.

## 5 Improvements and Future Work

- Make the player say Uno when is its last card and the bots have a random number between 20-30 to say it first. If the player doesn’t says it draws 3 cards. (This can be implemented thought a thread)

# 6 Conclusion

- With the OOP makes the programming easier because we have some patterns that makes the abstraction of the problem easier. Also it makes the programs more safe as we use the encapsulation to protect some attributes that may be important.