**Introduction:**

This document describes the design and implementation of the Monopoly game of group 15. The project is part of the course COMP2021 Object-Oriented Programming at PolyU. The following sections describe the requirements that were implemented and the design decisions taken. The last section describes the available commands in the game.

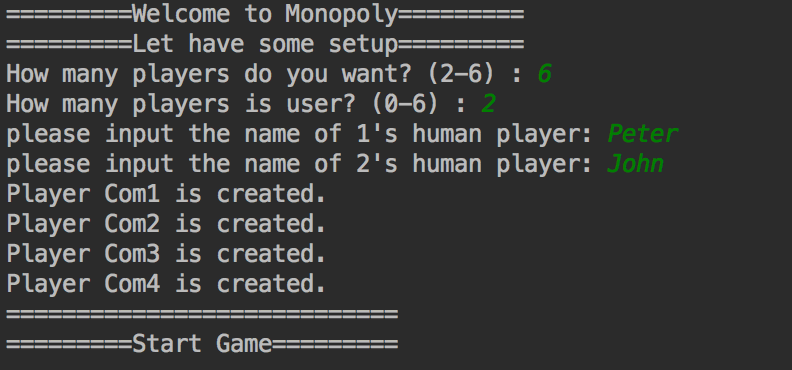
**What is Monopoly?**

***Monopoly*** is a [board game](https://en.wikipedia.org/wiki/Board_game) that originated in the United States in 1903 as a way to demonstrate that an economy which rewards wealth creation is better than one in which monopolists work under few constraints[[1]](https://en.wikipedia.org/wiki/Monopoly_(game)#cite_note-NYT-20150213-1) and to promote the economic theories of [Henry George](https://en.wikipedia.org/wiki/Henry_George) and in particular his ideas about taxation.[[3]](https://en.wikipedia.org/wiki/Monopoly_(game)#cite_note-3) The current version was first published by [Parker Brothers](https://en.wikipedia.org/wiki/Parker_Brothers) in 1935. Subtitled "The Fast-Dealing Property Trading Game", the game is named after the economic concept of [monopoly](https://en.wikipedia.org/wiki/Monopoly)—the domination of a market by a single entity. It is now owned and produced by the American game and toy company [Hasbro](https://en.wikipedia.org/wiki/Hasbro). Players move around the game-board buying or trading properties, developing their properties with houses and hotels, and collecting rent from their opponents, with the goal being to drive them all into [bankruptcy](https://en.wikipedia.org/wiki/Bankruptcy) leaving one monopolist in control of the entire economy. Since the board game was first commercially sold in the 1930s, it has become a part of popular world culture, having been locally licensed in more than 103 countries and printed in more than thirty-seven languages.

Our product is a simple version of monopoly. For instances, our game does not allow to build houses and hotels on owner’s properties. There are some more traditional rules being simplified in our product. However, the basic function and rule are no different from that of the traditional one.

**Requirements:**

Req-1. The game must support a command line user interface.



We have support to use command line for the user interface.

Req-2. The game must support both human players and computer players.

In the class ‘Player’, there is a Boolean attribute called “isComp” to check whether the player is human player or computer player. If it return true, it will be a computer player and it will automatically make decisions. If it return false, it will be a human player.

Req-3. At each step of a human player, the game must ask for a command from the player. Supported commands must include: continue, report, auto, and retire.

In the monopoly class, player can choose option each turn.

Option 0 = Draw dice

Option 1 = print current information

Option 2 = Let the computer take over

Option 3 = surrender

Req-4. Upon receiving the command continue, the game must let the player take his/her turn.

In monopoly class, after receiving the option 0, it will draw the dice and the player will move, then the player can make decision. After that, it will print out “End of Turn” and the next player will take turn.

Req-5. Upon receiving the command report, the game must print out the information of each square on the board and each player’s location on the board.

If the player choose option 1, it will print the information of each of the array ‘pos’. Then, there is a ‘getPosition’ operation in attribute ‘Player’ and it can return their position.

Req-6. Upon receiving the command auto, the game must use a computer player to takes over the player’s place. A computer player always continues until the end of the game and makes decisions in Req-8 randomly.

If the player choose option 2, it will ask the player again to ensure the option. If the player say yes, the player will be set to a computer player for a turn and make decision automatically.

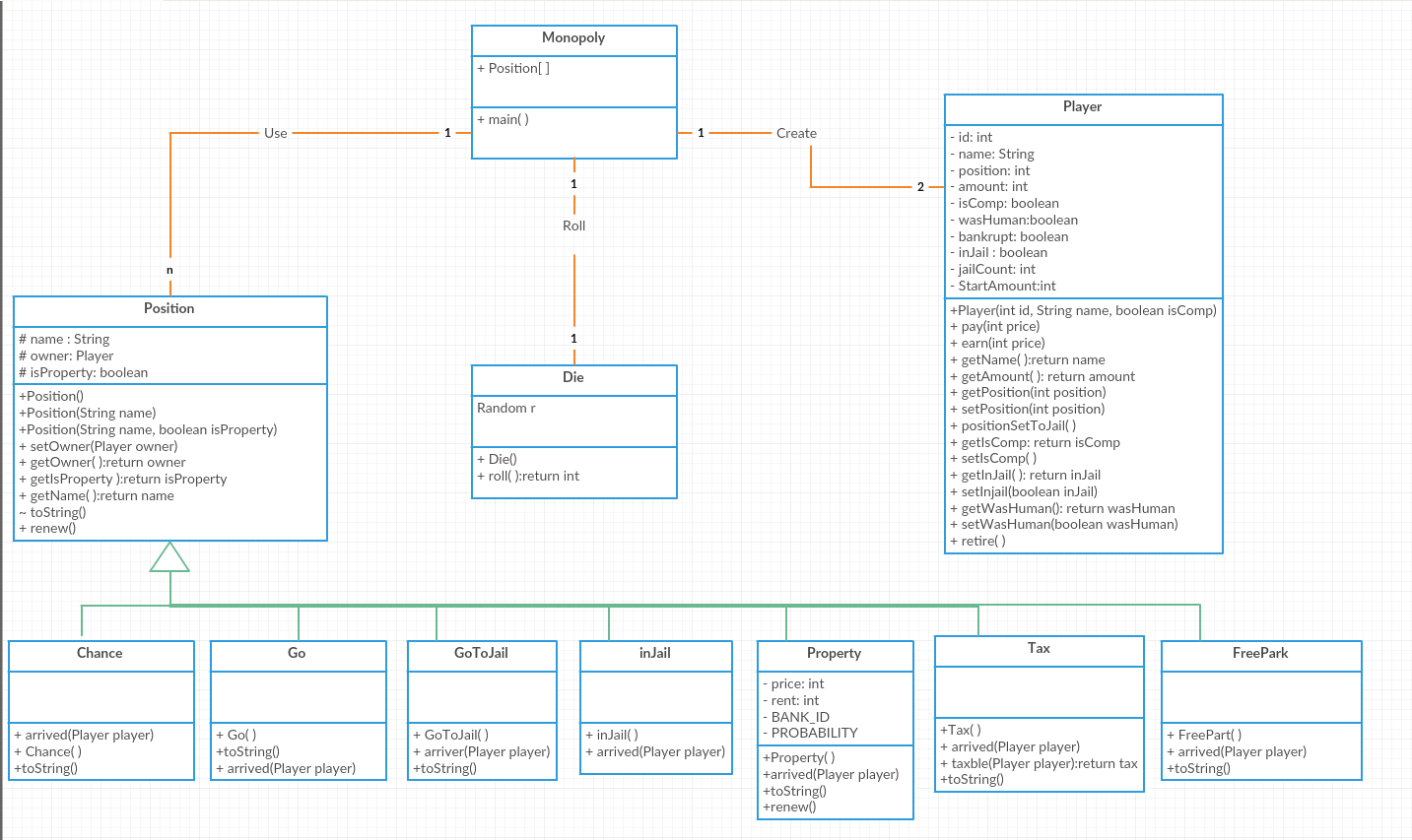
Req-7. Upon receiving the command retire, the game must make the player retire (i.e., the player leaves the game with all his/her property becoming unowned).

If the player choose option 3, it will ask the player again to ensure the option. If the player say yes, it will print out ‘bye bye’ and will set the state of the player to be bankrupt. Then, the player will no longer be exit in the game and all his property will become unowned.

Req-8. When a human player can decide whether to buy a property or to pay the fine to get out of Jail, the game must ask for the player’s input.

Whenever the human player go to a property which is unowned and he has enough money, it will print out ’Do you want to buy this property? (Y/N)’ and ask for the player input. If the player say yes, he will pay for it and own the property.

Whenever the human player is in the jail, the system will ask whether he want to pay $50 and get out of the jail. If the player chooses this option, he will pay $50 and be released.

**Design:**

There are mainly four class in this program. Monopoly includes the main operation and we process the game and call the other function in this class. The ‘Player’ class has attribute of player to store the state, the position and the wealth of each player. And for those operations on this class, we can check and return the information of each player. The next class is die and we use it to get the random number for the player to move in every round. The most important class of this program is Position. This program use an array to store every square on the board and their information are stored in the attribute in this class such as name, owner, type. If we want to get the information of a square, we can call the operation and it will return what we want. For different type of square, we use Position to be the superclass and create other subclass, such as property, jail, chance, etc. We use different class to describe different types of square.

**Quick Start Guide:**

First, input how many player (2-6).

Then, input how many human player (the other player will automatically become computer player).

Next, input the name of players.

And the game will start.

Each turn the user will be asked for input.

Input 0 if you want to throw the dice.

Input 1 if you want to know the current information of each play and each square on the broad.

Input 2 if you want the computer to take the turn and automatically make decisions.

Input 3 if you want to surrender.

If the player land on a property which is unowned and the player have enough money, he will be asked whether he want to purchase the property.

And whenever the player is in jail, he will be asked whether he want to pay $50 and get out of the jail.

The game will end when there is only one player does not bankrupt.