1 Program Scope

The program should be able to receive as input a chess move in UCI(Universal Chess Interface) format i.e e2e4, and if the movement is valid, output the board state to the user or inform the user the input isn't valid. For this matter, the standard python library is enough address the problem. For debugging purposes, a graphical interface was also required and implemented in pygame, a graphical framework for games. Also for debugging and testing pourposes, it was used the program *pgn-extract* to convert PGN game notation to UCI notation.

2 Program project

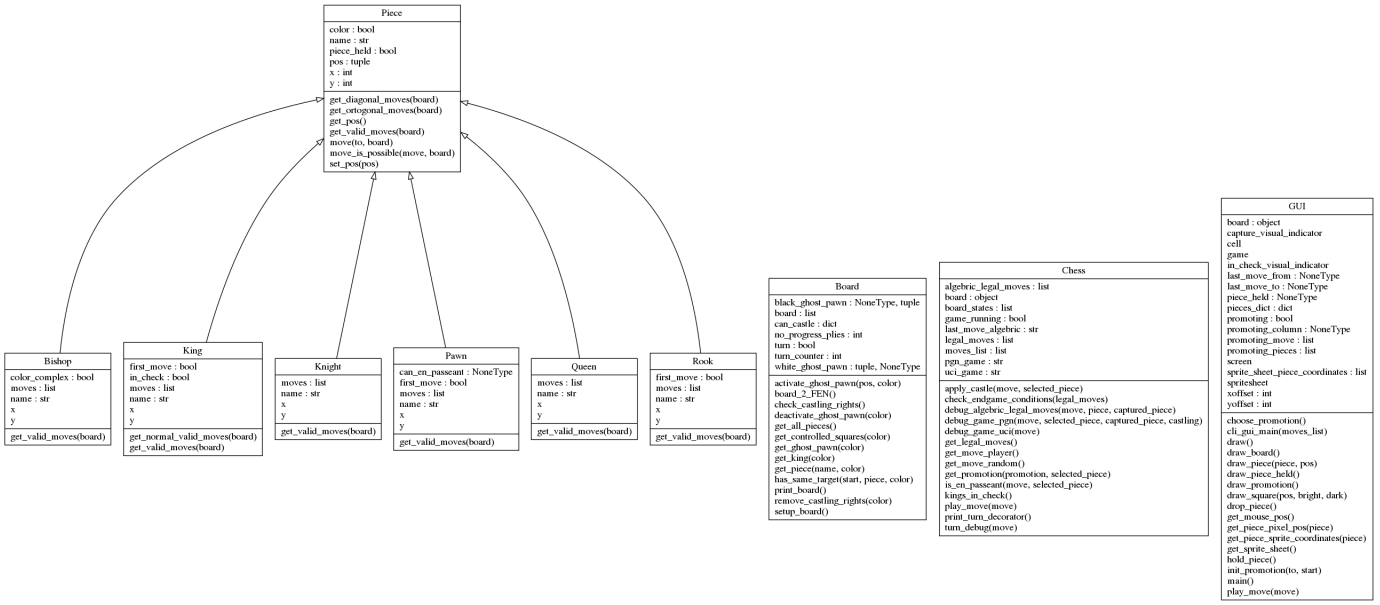
The project is constitued by four modules that contains in itself their respective major classe: The Piece, Board, Chess and GUI.

1. The Pieces module contains the Piece class, that is inherited by all the chess pieces, and specify how to get from each piece their own set of possible moves.

2. The Board module contains the Board class that is used to save all information relative to board state, such as pieces positions, castling rights, number of turns, en passeant possibility, etc.

3. The Chess module contains the Chess class that is used to process the Board information and create legal moves from which the player can chose to play.

4. The GUI module uses the Board and Chess classes to play the game in a graphical interface mode.



3 Testing

Number of plies (half-moves)	Number of possible games
1	20
2	400
3	8092
4	197,281
5	4,865,609
6	119,060,324
...	...
10	69,352,859,712,417

Tabela 1: Shannon's Calculation. Obs: A turn is composed by a white move and a black move. Five plies therefore stands for white playing three times and black two.

For basic operations accuracy, it was used the Shannon Number, which stands for all the possible moves that can be played until a certain ply(half-move). By the limitation of the computer power avaible for our disposal, and considering that the game was not written in a language nor written in a way for fast computation, we could only check the precision of the game until 5 ply, as we can see by the test log:

```
2022-01-22 00:00:36,742 Result of possible games with 1 ply: 20/20 - OK
2022-01-22 00:00:36,742 Elapsed time in 1 ply: 00h00m00s seconds
2022-01-22 00:00:37,312 Result of possible games with 2 ply: 400/400 - OK
2022-01-22 00:00:37,312 Elapsed time in 2 ply: 00h00m00s seconds
2022-01-22 00:00:52,137 Result of possible games with 3 ply: 8902/8902 - OK
2022-01-22 00:00:52,137 Elapsed time in 3 ply: 00h00m14s seconds
2022-01-22 00:07:11,715 Result of possible games with 4 ply: 197281/197281 - OK
2022-01-22 00:07:11,715 Elapsed time in 4 ply: 00h06m19s seconds
2022-01-22 08:45:00,073 Result of possible games with 5 ply: 4865609/4865609 - OK
2022-01-22 08:45:00,073 Elapsed time in 5 ply: 08h37m48s seconds
```

Although this is a good signal that basic operations are working, in 5 plies we cannot test all the complications that might arise during a chess game.

Depth	Captures	E.P	Castles	Promotions	Checks	Dscry Checks	Dbl Checks	Checkmates
1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0
3	34	0	0	0	12	0	0	0
4	1576	0	0	0	469	0	0	8
5	82,719	258	0	0	27,251	6	0	347

Tabela 2: Number of "special" moves by depth accordingly to `https://www.chessprogramming.org/Perft_Results`

By this table we can see that we need to concentrate our efforts in testing Castle, Promotions, Discovery Checks and Double Checks.

One way to test them, is loading "complex positions", brute forcing their possible moves, and comparing the results with a proof table. Using the positions recommended in `https://www.chessprogramming.org/Perft_Results`, we get the following result:

```
2022-01-24 00:03:34,134 ----------------------------------------------------
2022-01-24 00:03:34,134 Initiating move generation test on depth: 2
2022-01-24 00:03:34,268 Result of possible games with 1 ply: 48/48 - OK
2022-01-24 00:03:34,268 Elapsed time in 1 ply: 00h00m00s seconds
2022-01-24 00:03:40,177 Result of possible games with 2 ply: 2039/2039 - OK
2022-01-24 00:03:40,177 Elapsed time in 2 ply: 00h00m05s seconds
2022-01-24 00:03:40,177 Total Elapsed time: (00h00m06s)
```

```
2022-01-24 00:03:40,177 ----------------------------------------------------
2022-01-24 00:03:40,177 ----------------------------------------------------
2022-01-24 00:03:40,177 Initiating move generation test on depth: 3
2022-01-24 00:03:40,195 Result of possible games with 1 ply: 14/14 - OK
2022-01-24 00:03:40,195 Elapsed time in 1 ply: 00h00m00s seconds
2022-01-24 00:03:40,454 Result of possible games with 2 ply: 191/191 - OK
2022-01-24 00:03:40,454 Elapsed time in 2 ply: 00h00m00s seconds
2022-01-24 00:03:44,350 Result of possible games with 3 ply: 2812/2812 - OK
2022-01-24 00:03:44,351 Elapsed time in 3 ply: 00h00m03s seconds
2022-01-24 00:03:44,351 Total Elapsed time: (00h00m04s)
2022-01-24 00:03:44,351 ----------------------------------------------------
2022-01-24 00:03:44,358 ----------------------------------------------------
2022-01-24 00:03:44,359 Initiating move generation test on depth: 3
2022-01-24 00:03:44,382 Result of possible games with 1 ply: 6/6 - OK
2022-01-24 00:03:44,382 Elapsed time in 1 ply: 00h00m00s seconds
2022-01-24 00:03:45,135 Result of possible games with 2 ply: 264/264 - OK
2022-01-24 00:03:45,135 Elapsed time in 2 ply: 00h00m00s seconds
2022-01-24 00:04:12,876 Result of possible games with 3 ply: 9467/9467 - OK
2022-01-24 00:04:12,876 Elapsed time in 3 ply: 00h00m27s seconds
2022-01-24 00:04:12,876 Total Elapsed time: (00h00m28s)
2022-01-24 00:04:12,876 ----------------------------------------------------
2022-01-24 00:04:12,881 ----------------------------------------------------
2022-01-24 00:04:12,881 Initiating move generation test on depth: 3
2022-01-24 00:04:12,997 Result of possible games with 1 ply: 44/44 - OK
2022-01-24 00:04:12,997 Elapsed time in 1 ply: 00h00m00s seconds
2022-01-24 00:04:17,258 Result of possible games with 2 ply: 1486/1486 - OK
2022-01-24 00:04:17,258 Elapsed time in 2 ply: 00h00m04s seconds
2022-01-24 00:07:10,379 Result of possible games with 3 ply: 62379/62379 - OK
2022-01-24 00:07:10,379 Elapsed time in 3 ply: 00h02m53s seconds
2022-01-24 00:07:10,379 Total Elapsed time: (00h02m57s)
2022-01-24 01:06:43,597 ----------------------------------------------------
2022-01-24 01:06:43,598 Initiating move generation test on depth: 3
2022-01-24 01:06:43,724 Result of possible games with 1 ply: 46/46 - OK
2022-01-24 01:06:43,724 Elapsed time in 1 ply: 00h00m00s seconds
2022-01-24 01:06:49,460 Result of possible games with 2 ply: 2079/2079 - OK
2022-01-24 01:06:49,460 Elapsed time in 2 ply: 00h00m05s seconds
2022-01-24 01:11:14,405 Result of possible games with 3 ply: 89890/89890 - OK
2022-01-24 01:11:14,423 Elapsed time in 3 ply: 00h04m24s seconds
2022-01-24 01:11:14,424 Total Elapsed time: (00h04m30s)
2022-01-24 01:11:14,424 ----------------------------------------------------
```

By doing these tests, we can be certain that the program is working as desired.