Chapter 1

Testing

1.1 Iterative Testing

I have been playtesting the program throughout the development process to find any bugs and fix them accordingly. However, a few issues have required additional measures.

1.1.1 Minimax

Since minimax is recursive algorithm, debugging it has proven a challenge. I have therefore configured the Python `logging` library to help collect information on the minimax tree for every function call.

base.py

```
1     def print_stats(self, score, move):
2         """
3         Prints statistics after traversing tree.
4 
5         Args:
6             score (int): Final score obtained after traversal.
7             move (Move): Best move obtained after traversal.
8         """
9         if self._verbose is False:
10            return
11 
12        self._stats['time_taken'] = round(1000 * (time.time() - self._stats['
      time_taken']), 3)
13        self._stats['ms_per_node'] = round(self._stats['time_taken'] / self._stats
      ['nodes'], 3)
14 
15        # Prints stats across multiple lines
16        if self._verbose is True:
17            logger.info(f'{self.__str__()} Search Results:')
18            logger.info(printer.pformat(self._stats))
19            logger.info(f'Best score:  {score}   Best move: {move}')
20 
21        # Prints stats in a compacted format
22        elif self._verbose.lower() == 'compact':
23            logger.info(self._stats)
24            logger.info(f'Best score: {score}   Best move: {move}')
```

Listing 1.1: BaseCPU Method for logging minimax statistics

1.1.2 Migrations

To correct errors made to the `games` table, since recreating it would mean deleting all existing games, I have opted to use migrations to fix bugs by editing the table schema, as shown in Section ??.

1.2 Unit Tests

1.2.1 Board Evaluator

To test every aspect of the evaluation function, I have set up some unit tests with custom positions using my editor screen. These positions are designed to test every aspect of the evaluation, along with some obviously imbalanced positions to test the overall accuracy of the evaluation function. All positions are set up to give an advantage to the blue player.

Evaluating	FEN string	Score	Passed
Material	`sc9/10/10/4paPa4/5Pa4/10/10/9Sa b`	124	✓
Position	`sc9/4nanana3/10/10/10/4NaNaNa3/10/9Sa b`	66	✓
Mobility	See footnote[1]	196	✓
King Safety	`sc4fa3pa/10/10/10/10/10/10/5FaPa2Sa b`	3	✓
Combined	See footnote[2]	437	✓

Table 1.1: Board evaluator test results

1.2.2 CPU

Similarly, to evaluate the strength of my CPU, I have setup some custom positions that I already know the best continuation of, and run each CPU engine on them to test if they can solve it.

Description	FEN string	Best Move	Passe
Mate in 1	`sc9/pafa8/Fa9/10/10/10/10/9Sa b`	Rotate J3 clockwise	✓
Mate in 1	`sc9/10/10/10/8faRb/8FaRb/10/9Sa b`	Move J3 to J2	✓
Mate in 3	`sc9/10/10/8Ra1/7FaRafa/8RaRa/9Ra/9Sa b`	Move J2 to I1...[3]	✓
Mate in 3	`sc5fcnc2/4Pa4Pc/3pb6/2Pc2ra1pb2/10/pb9/7Pa2/2PdNaFa4Sa b`	Move E7 to F7...[4]	✓
Escape Mate	`sc9/10/10/10/8faRb/8FaRb/10/9Sa b`	Move J3 to J2	✓
Win Material	`sc9/10/10/10/8faRb/8FaRb/10/9Sa b`	Move J3 to J2	✓

Table 1.2: Transposition Table CPU test results

I have also personally played against the CPU engines to gauge its strength in a realistic setting. The results are shown below:

[1] scpapa7/papapa7/papapa1Pa1Pa1Pa1/10/4Pa1Pa1Pa1/10/4Pa1Pa3/9Sa b
[2] scnc1fcncpbpb3/pa9/pb1pc1rbpa3Pd/1Pc2Pd4Pc/2Pd1RaRb4/10/7Pa2/2PdNaFaNa3Sa b
[3] 2. `Move J4 to J5` 3. `Move J3 to I2`
[4] 2. `Rotate anticlockwise H8` 3. `Move F7 to G7`

Me	CPU
2	5

Table 1.3: Score of me vs CPU (I am very bad)

Mr Myslov	CPU
1	6

Table 1.4: Score of Mr Myslov vs CPU (Mr Myslov is even worse)

1.2.3 Shadow Mapping

To test the shadow mapping algorithm, I have set up some occluding objects togehter with a light source. Since visuals are subjective, me and my client have deemed the following results to be adequate.

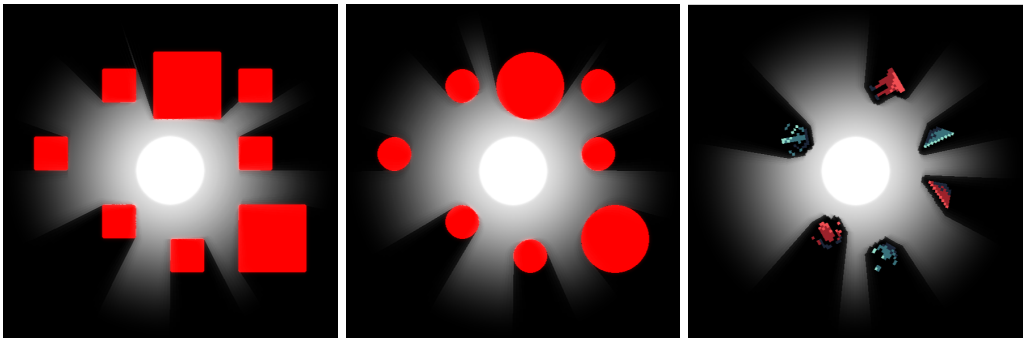

Figure 1.1: Shadow mapping algorithm test (softShadow=0.5, radius=0.5)

1.3 Final Tests

1.3.1 Objective 1

All laser chess game logic should be properly implemented.

No.	Input	Output	Passed
1	Laser fires on piece	Piece reflects laser	✓
2	Select piece and press rotate clockwise button	Piece rotates clockwise	✓
3	Select piece and adjacent empty square	Piece moves to adjacent square	✓
4	Blue player makes a move	Next move is made by red player	✓
5	Make move	Laser reflects off pieces correctly	✓
6	Position piece with non-reflecting side facing laser	Piece is destroyed	✓

No.	Input	Output	Passed
7	Move Pharoah in front of player	Game displays game over screen	✓
8	Repeat same position three times	Game displays game over screen	✓

1.3.2 Objective 2

Save or load game options should be implemented.

No.	Input	Output	Passed
9	Click on copy FEN string button in editor or browser screen	Position formatted as FEN string copied to clipboard	✓
10	Enter browser screen	Program shows list of past games to be scrolled through	✓
11	Click on previous or next move button in review screen	Board undoes or applies move	✓
12	Enter review screen	Program displays list of past moves, winner and move number	✓

1.3.3 Objective 3

Other board game requirements should be implemented.

No.	Input	Output	Passed
13	Click on draw button	Game ends and shows draw result	✓
14	Click on resign button	Game ends and shows win result for opposite player	✓
15	Enable timer	Timer appears on the left and decrements every second	✓
16	Pause the game	Timer stops decrementing	✓

1.3.4 Objective 4

Game settings and config should be customisable.

No.	Input	Output	Passed
17	Set primary board colour to blue	Alternating board squares appear blue	✓
18	Select CPU player	Every other move is played by the CPU	✓
19	Input timer duration in config screen	Inputted timer duration appears on the left of the game screen	✓
21	Press starting colour button to set starting colour to red	Starting move is played by red player	✓
21	Set up custom board layout in editor screen	Game starts with custom board position	✓

1.3.5 Objective 5

Game UI should improve player experience.

No.	Input	Output	Passed
27	Click the play button on the main menu screen	Program switches to the config screen	✓
28	Click main menu button	Program switches to the menu screen	✓
29	Click help button	Help overlay appears	✓
30	Resize program window	GUI Widgets resize continuously	✓
31	Drag program window	Program continues running	✓

1.4 Videos