Assignment 2 - Grading Criteria

1. GUI (10 points)

The GUI of the program has to contain the following elements:

1.1 Chess board and `Help`→`About` menu item (1 point)

The GUI has to include a chess board and a `Help`→`About` menu item, which is used to show a window (form) with the version of the application and some additional information about the program (e.g. your name).

Note: This dialog is used to test the responsiveness of the GUI for the multithreading task.

1.2 Area for captured pieces (1 point)

The GUI has to include an area to display the captured pieces.

1.3 History list (1 point)

The GUI has to include a history list that shows the moves of both players. These moves have to be represented in the algebraic notation.

Note: Take a look at the link provided in assignment1.pdf to check the algebraic notation format.

1.4 Command buttons (1 point)

The GUI has to include three buttons: Redo, Undo and Reset.

1.5 Menu item File→New and Reset button (1 point)

The File→New command and the reset button have to set the board to the initial positions. I.e. to the standard starting position in chess.

1.6 Menu item File→Load (1 point)

This command displays an open-file dialog box for loading a Forsyth file. The program shall then calculate the next move of the black player and render the result on the board.

Note: This command has the same purpose as the loading operation of the previous assignment. Instead of printing the result in a file, you have to render it on the board.

1.7 Menu item File→Open (2 points)

This command displays an open-file dialog box for loading a legal game file (*.chess) containing the positions of the pieces on the board and a history of moves.

Note: Contrary to the File→Load command, the program does not need to handle invalid game files.
Furthermore, you can assume that the move history is consistent and complete.

1.8 Menu item File→Save as … (1 point)

This command opens a save-file dialog box for storing the current state of the game and its history into a game file (*.chess).

1.9 Menu item File→Exit (1 Point)

This command exits the game without saving the state.

2. AI component (15 points)

Do not consider drawn situations.

It is not required to implement the En passant and the castling rules.

2.1 Find the optimal move when the white player can be checkmate in one step (10 points)

Reference: Section 3.1 in assignment2.pdf.

2.2 Find the optimal move when the white player can be defeated in two steps (5 points)

Reference: Section 3.1 in assignment2.pdf

3. Redo and Undo (10 Points)

The program has to implement the Redo and Undo operations.

If Undo has never been executed, Redo has no effect and must not throw any exception.
If no move was made, Undo has no effect and must not throw any exception.

The File→Load command opens a Forsyth file describing the state of the board. As the file does not contain a move history, pressing the Undo button has no effect.

The File→Open command opens a *.chess file. A *.chess file contains a move history, which has to be loaded. Therefore, the Undo command has to work after the loading of the file.

4. Multithreading (10 points)

The AI component and the GUI have to be put into different threads. While the AI component computes the next move, the user shall be able to use the GUI to access the help. To prevent race conditions, other components in the GUI shall not response to the user's activity.

5. Extra points

5.1 Check the consistence of the move history in the game file (5 points)

Check if the given history is consistent with the position given in the *.chess file. If not, the program should output a warning and should not throw an exception.

5.2 Good documentation and organized unit test process (5 points)

- A set of automatically generated HTML files generated based on your comments.
- Make use of a unit testing tool to test your classes.
- Unit test cases.

5.3 Implement an efficient search algorithm (10 points)

The efficiency of your AI component is decided by the steps to cause the white player in checkmate. Some typical chess samples are used to test your program to see how many steps your program needs to bring the white player in checkmate status. If your program can generate the least steps, you will get these extra points.

6. Reduction of points

Unhandled exceptions are considered as crashes and points will be deduced in such cases.