Budapest University of Technology and Economics
Faculty of Electrical Engineering and Informatics

Basics of Programming 2
(BMEVIIIAA03, CS16D)

CPPChess User Guide

Final Project

Instructor
Dr. Vaitkus Márton

Author
Diego Davidovich Gomes
NEPTUN Code: GHXLNI

May 13, 2025

Contents

1 Introduction

Chess is a board game for two players. It is an abstract strategy game which involves no hidden information and no elements of chance. It is played on a square board containing 64 squares arranged in an 8×8 grid. The players, referred to as "White" and "Black", each control sixteen pieces: one king, one queen, two rooks, two bishops, two knights, and eight pawns; each type of piece has a different pattern of movement. An enemy piece may be captured (removed from the board) by moving one's own piece onto the square it occupies; the object of the game is to "checkmate" (threaten with inescapable capture) the enemy king. There are also several ways a game can end in a draw.[1]

2 Getting Started

2.1 Requirements

The program is designed and tested to run in the **macOS** operating system with ARM processors, which is the same system of the author. Furthermore, the program requires the installation of the libraries **SDL2**, **SDL2_Image**, **SDL2_TTF** and **SDL2_Mixer**. The library **Dear ImGui** is included in the program folder.

2.2 Installation

To install and execute the file, read and execute the following steps carefully:

1. **Install Dependencies**
   Before executing the game, make sure you have the SDL2 libraries installed.

   - **On macOS:**
     Check if brew is installed, if not, install from homebrew:

     ```
     brew --version
     ```

     If it is installed, execute the following:

     ```
     brew install sdl2 sdl2_image sdl2_ttf sdl2_mixer
     ```

2. **Download the executable file**

   - The source code can be downloaded from GitHub
   - Place the folder in the desired location and do not delete any file

2.3 Running

As a final step to run the game, we will need to execute it from the terminal. We suggest that in the terminal, you should type **cd** and then drag the folder **cppchess** into the terminal. Something similar will appear and you should press **Enter**:

```
cd /home/username/Downloads/cppchess
```

[1] https://en.wikipedia.org/wiki/Chess

After pressing Enter, you will be inside the **cppchess** folder and ready to run the game. You can run the game by typing:

```
./build/cppchess
```

Warning: if you wish, you can run it by double clicking the **cppchess** file inside build, however do not expect the PGN file generator to work correctly due to limitations related to the file path.

After executing all the steps, it is expected to the chess board in the screen:

Figure 1: Default program window in the desktop

3 Game Instructions

In this section, you will get knowledge on how to play the Chess game and learn about the available features.

3.1 Basic Knowledge

After completing the steps to run the program, you should expect to the same window as in the previous figure to appear. Initially, you will see two different things: the chessboard itself and a floating menu with different options. This section is mostly dedicate to present some of the basic notions of Chess.

The first thing that is important to know is how the pieces moves. There are 6 pieces types on the board: **King**, **Queen**, **Rook**, **Knight**, **Bishop** and **Pawn**.

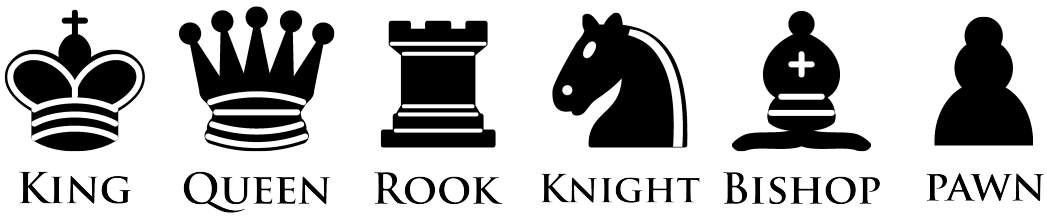

Figure 2: Chess Pieces

Below are the standard movement rules for each chess piece:

- **Pawn**: it moves forward one square or two on its first move. It only capture diagonally and if it reaches the end of the board, it will be promoted to a Queen
- **Knight**: moves in an **L-Shape**, two squares in one direction, then one square perpendicular. It has the special ability of jumping over pieces
- **Bishop**: it moves diagonally any number of squares and stays on the same color throughout the game (called light square bishop and black square bishop)
- **Queen**: it behaves like a rook and bishop together, as it can move horizontally, vertically or diagonally any number of squares
- **King**: moves one square in any direction and it can switch places with a rook if neither has moved and the path is clear. It cannot move into danger or where other piece attacks

The objective of the game is **checkmate** your opponent's king. This occurs when the king is under attack (called in **check**) and has no legal moves to escape, which can be blocking, capturing the attacker or moving to a safe square. The game ends immediately when checkmate is achieved. Other possible outcome include **stalemate** (no legal moves but the king is not in check), **repetition**, 50-move rule or insufficient material. Always watch for threats to your king while planning attacks.

3.2 Moving Pieces and Game Flow

Now that we know how our pieces move, we can start playing. To select a piece, we simply click in the desired square where the piece is placed. After clicking, every valid and legal move will be highlighted. Either you have the possibility to drag the piece (hold the left button of the mouse down, and then drag the piece) to a valid square, or simply click into the destination square. After the piece has been played, the game automatically changes turns, so if white played, the next move will be made by the black pieces. If you are waiting for something to happen, well, do not wait more, as the program is made for two players to play. If you wish to flip the board to see the perspective of the other player, you can press **F**.

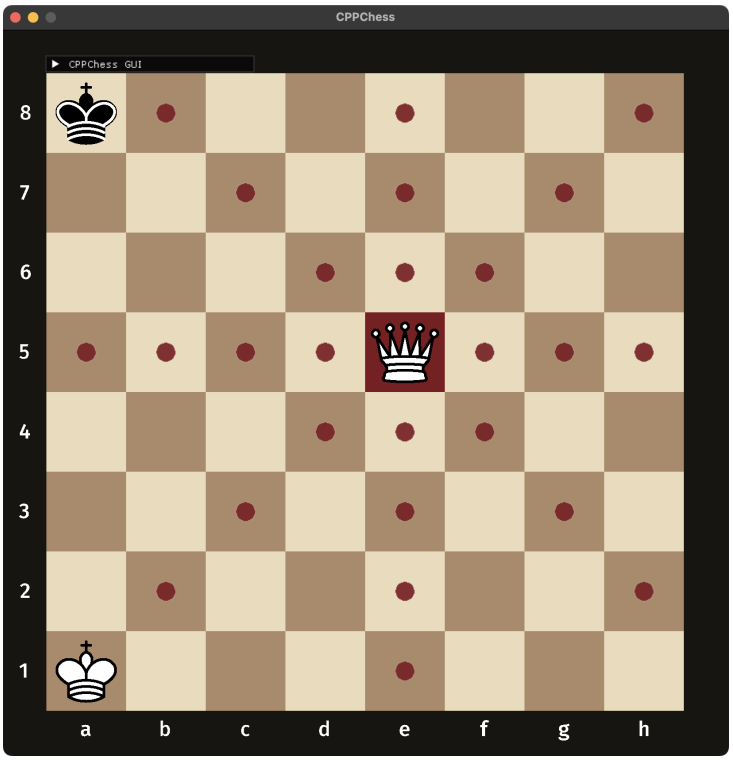

Figure 3: Legal moves of a white queen

If the King is checked/attacked, it will be highlighted with a red marking, which means that either you need to move to a safe square, capture the attacking piece or block it. You can click to see what the valid moves are.

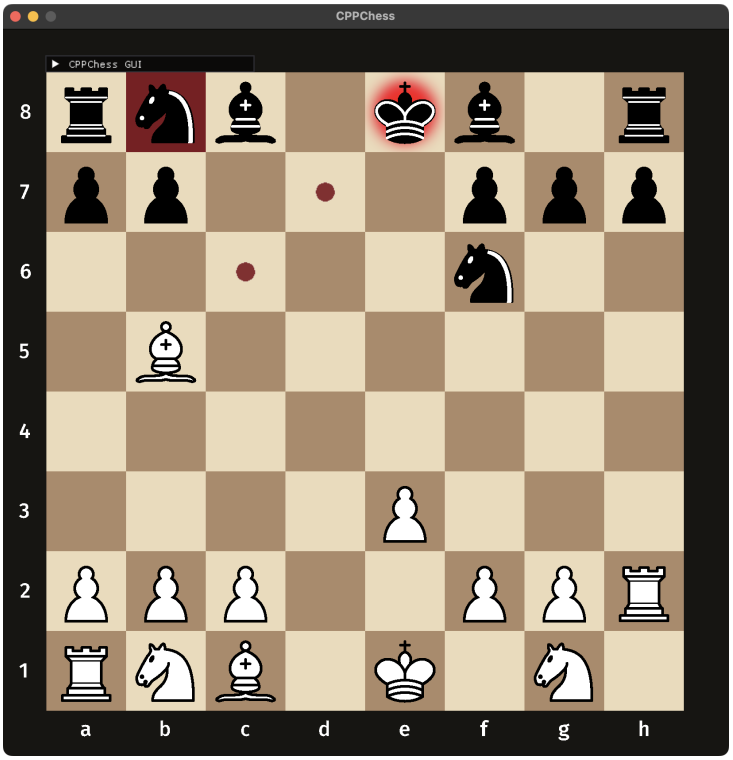

Figure 4: White bishop attacking the black knight. You can block with the Knight or move the King

If you reached a position where the enemy King is out of move and it is in check, congratulations, you **checkmated** the King and won the game. The game result will displayed accordingly:

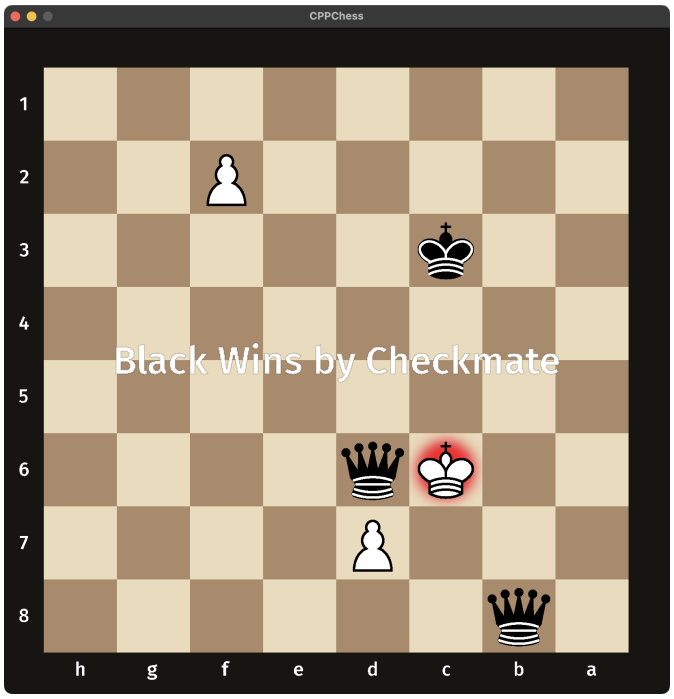

Figure 5: Black wins by checkmate

Then, after the game is finalized, you can either choose to terminate the program or restart the game:

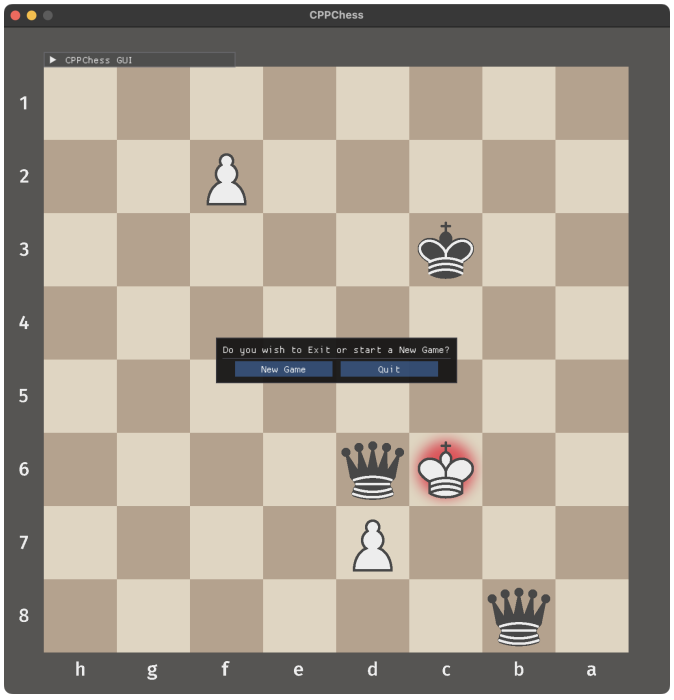

Figure 6: End of Game options

3.3 Navigating the Menu

The graphical user interface (GUI) provides centralized access to the program's advanced features through an intuitive menu system. Upon launching the application, users encounter a main menu with multiple options:

Figure 7: Game options

Some of the features are purely experimental, but the main are:

1. **Info**: some information regarding the turn, move clocks and en-passant square index.

2. **Load Position**: it is possible to import a chess position using a **FEN String** (Forsyth–Edwards Notation)[2], either from a book, online platform or other means. You just simply need to copy and paste it, and press **Load FEN**. If there is any problem with it, the default chess position will be loaded instead.

3. **Board Settings**: you can flip the board through here as well, disable or enable the markings on the board or change the speed of animation

4. **Board Colors**: it is possible to personalize the colors of the white square, black squares and the background color. If you wish to use the default colors, just press reset

5. **Move History**: you can check the played moves in order and if you wish so, save it as a **PGN** file (Portable Game Notation[3]). This allows to analyse the game in external platform such as www.chess.com or lichess.org.

6. **Debugger**: it is purely made for developers :D (it can be disabled)

7. **Others**: it comes from the implemented libraries and is also made for developers

[2]https://en.wikipedia.org/wiki/Forsyth–Edwards_Notation
[3]https://en.wikipedia.org/wiki/Portable_Game_Notation

3.4 File Management

To ensure the proper working of the game, it is important to discuss the files that will be used and generated. The source code has the following structure:

```
cppchess/
├── assets/
│   ├── fonts/
│   ├── icon/
│   ├── original/
│   ├── pieces/
│   ├── sounds/
│   └── ....png
├── build/
│   └── cppchess
├── games/
│   └── YYYY-MM-DD_HH-MM-SS.pgn
├── include/
│   └── ...
├── lib/
│   └── ...
├── src/
│   └── ...
├── makefile
└── README.md
```

Figure 8: File Structure

The executable file can be found at the **build** folder and the generated PGN files can be found at the **games** folder. Anything else are not relevant, but is not recommended to delete any of the files.

4 Troubleshooting and Support

This section is to provide possible solutions to some possible errors or unexpected behavior. Please note that due to the program being recently launched, some bugs might not be known.

1. **The pieces do not appear in the board**
   In this case, it is worth checking if you did not accidentally change or move the files around. All the piece textures can be found in the **assets** folder. If it is empty, it is worth re-downloading the project files.

2. **The PGN is not being generated**
   As it was written in the remark, this happens due to some limitations of the file paths. If you initialized the program by just opening the binary file, the path will not match the programmed location of where the files are written. The files are expected to be written into the **games** folder.

3. **The FEN does not work**
   That would be mostly due to a faulty FEN with invalid characters or invalid fields. Double ckeck if the FEN is correct.

4. **The pawn automatically gets promoted to queen**
   Currently, there is no support for under-promotion (to a Rook, Knight or Bishop).

5. **I accidentally deleted the compiled executable file**
   Download it again at the provided link in 2.2. Or if it is easy for you to use the terminal, open a new terminal, change directory to the cppchess folder and execute the **make** command, which will create a new executable

In case you find any other error/bug/problem in the program that is not documented here, please write an email to **diego.gomes@edu.bme.hu** with a screenshot of the problem and a description. Alternatively you can create an issue in GitHub.

5 Conclusion

Chess is a timeless, strategic, and deeply engaging board game that is both intellectually stimulating and rewarding. Its elegant design makes it universally accessible, allowing it to be played anywhere—from grand tournaments to casual matches at home. Chess not only provides a captivating mental challenge but also sharpens critical thinking, foresight, and decision-making skills. With CPPChess, you can enjoy this amazing game in your machine, fully playable with a graphical user interface, PGN support and customizable options.

6 About the Project

This program was created by the student **Diego Davidovich Gomes** as the final project for the **Basics of Programming 2** subject at **Budapest University of Technology and Economics**, Computer Engineering BSc, 2nd Semester, Academic Year 2024/25, under the supervision of the laboratory guide, **Dr. Vaitkus Márton**. The purpose of this project was to apply foundational programming concepts, including algorithm design, structured programming, problem-solving and object-oriented programming, in the creation of a complex functional and interactive program. The game is implemented in the C++ programming language, using the **SDL2** library for graphics and user interaction, and **Dear ImGui** library for the graphical user interface (GUI), both available in multiple platforms.