Abstract

This report examines the design, implementation, and execution of an expert system, which simulates a series of checkers games. Report covers how the program employs course concepts and how it makes use of data structures and algorithms.

Two player games - checkers

Project 2 – CS5346: Advanced Artificial Intelligence

Team members: David Torrente and Randal Henderson

Author: Borislav S. Sabotinov

bss64@txstate.edu | TXST Student ID: A04626934

Table of Contents

[1 Introduction 6](#_Toc70351974)

[1.1 Purpose of Report 6](#_Toc70351975)

[1.2 System Scope 6](#_Toc70351976)

[1.3 Terms and Definitions 7](#_Toc70351977)

[1.4 Report Overview 8](#_Toc70351978)

[1.5 Contributions 9](#_Toc70351979)

[1.5.1 Individual Contributions 9](#_Toc70351980)

[1.5.2 Group Member Contributions 10](#_Toc70351981)

[2 System Description 11](#_Toc70351982)

[2.1 System Perspective 11](#_Toc70351983)

[2.2 System Features 11](#_Toc70351984)

[2.3 Design and Implementation Constraints 11](#_Toc70351985)

[2.4 Assumptions and dependencies 12](#_Toc70351986)

[3 System Design and Specification 12](#_Toc70351987)

[3.1 How to Build and Run the Program 13](#_Toc70351988)

[3.2 Why use C++11? 15](#_Toc70351989)

[3.5 System Class Diagram 16](#_Toc70351990)

[3.6 Analysis of Results 18](#_Toc70351991)

[3.6.1 How Good Are the Results? 18](#_Toc70351992)

[3.6.2 Memory and Speed 18](#_Toc70351993)

[3.6.3 Optimizations 24](#_Toc70351994)

[3.7 Evaluation Functions 26](#_Toc70351995)

[3.7.1 Evolution of Evaluation Function Three (EF-3) 26](#_Toc70351996)

[3.7.2 Final state of EF-3 36](#_Toc70351997)

[3.7.3 Evaluation Function One (EF-1) 46](#_Toc70351998)

[3.7.4 Evaluation Function Two (EF-2) 46](#_Toc70351999)

[3.8 UI Design Considerations 47](#_Toc70352000)

[4 Classes Deep-Dive 50](#_Toc70352001)

[4.1 Pieces 50](#_Toc70352002)

[4.1.1 Data Structures Used 52](#_Toc70352003)

[4.2 Board 52](#_Toc70352004)

[4.2.1 Data Structures Used 54](#_Toc70352005)

[4.3 Player 57](#_Toc70352006)

[4.3.1 Data Structures and Algorithms 59](#_Toc70352007)

[4.4 Game 59](#_Toc70352008)

[4.4.1 Data Structures and Algorithms 61](#_Toc70352009)

[4.5 Algorithm 61](#_Toc70352010)

[4.5.1 Data Structures and Algorithms 63](#_Toc70352011)

[4.6 Simulation 66](#_Toc70352012)

[4.6.1 Data Structures and Algorithms 66](#_Toc70352013)

[4.7 Main 67](#_Toc70352014)

[5 Sample Runs 68](#_Toc70352015)

[5.1 Sample Run #1: ALPHA-BETA-SEARCH with EF-1 vs ALPHA-BETA-SEARCH with EF-3 at Depth 2 69](#_Toc70352016)

[5.2 Sample Run #2: ALPHA-BETA-SEARCH with EF-1 vs ALPHA-BETA-SEARCH with EF-3 at Depth 4 70](#_Toc70352017)

[5.3 Sample Run #3: Minimax-a-b with EF-3 vs Alpha-Beta-Search with EF-3 at Depth 6 (extra) 72](#_Toc70352018)

[5.4 Sample Run #4: ALPHA-BETA-SEARCH with EF-1 vs ALPHA-BETA-SEARCH with EF-3 at Depth 8 (extra) 73](#_Toc70352019)

[5.5 Sample Run #5: Human Player vs AI (Minmax-A-B with EF-3) 74](#_Toc70352020)

[5.6 Sample Run #6: Printing the Help Menu 75](#_Toc70352021)

[6 Conclusion 76](#_Toc70352022)

[References 77](#_Toc70352023)

[Appendix A: Source Code 78](#_Toc70352024)

[Pieces.hpp 78](#_Toc70352025)

[Pieces.cpp 79](#_Toc70352026)

[Board.hpp 84](#_Toc70352027)

[Board.cpp 86](#_Toc70352028)

[Player.hpp 115](#_Toc70352029)

[Player.cpp 117](#_Toc70352030)

[Game.hpp 121](#_Toc70352031)

[Game.cpp 123](#_Toc70352032)

[Algorithm.hpp 127](#_Toc70352033)

[Algorithm.cpp 130](#_Toc70352034)

[Simulation.hpp 167](#_Toc70352035)

[Simulation.cpp 170](#_Toc70352036)

[main.cpp 179](#_Toc70352037)

[Appendix B: Sample Output for 18 Games 189](#_Toc70352038)

[Game 1: (Minmax-A-B, EF-1) vs (A-B-Search, EF-1) – Depth 2 190](#_Toc70352039)

[Game 2: (Minmax-A-B, EF-2) vs (A-B-Search, EF-2) – Depth 2 252](#_Toc70352040)

[Game 3: (Minmax-A-B, EF-3) vs (A-B-Search, EF-3) – Depth 2 253](#_Toc70352041)

[Game 4: (Minmax-A-B, EF-1) vs (Minimax-A-B, EF-2) – Depth 2 254](#_Toc70352042)

[Game 5: (Minmax-A-B, EF-1) vs (Minimax-A-B, EF-3) – Depth 2 255](#_Toc70352043)

[Game 6: (Minmax-A-B, EF-2) vs (Minimax-A-B, EF-3) – Depth 2 256](#_Toc70352044)

[Game 7: (A-B-Search, EF-1) vs (A-B-Search, EF-2) – Depth 2 257](#_Toc70352045)

[Game 8: (A-B-Search, EF-1) vs (A-B-Search, EF-3) – Depth 2 258](#_Toc70352046)

[Game 9: (A-B-Search, EF-2) vs (A-B-Search, EF-3) – Depth 2 260](#_Toc70352047)

[Game 10: (Minmax-A-B, EF-1) vs (A-B-Search, EF-1) – Depth 4 262](#_Toc70352048)

[Game 11: (Minmax-A-B, EF-2) vs (A-B-Search, EF-2) – Depth 4 323](#_Toc70352049)

[Game 12: (Minmax-A-B, EF-3) vs (A-B-Search, EF-3) – Depth 4 324](#_Toc70352050)

[Game 13: (Minmax-A-B, EF-1) vs (Minimax-A-B, EF-2) – Depth 4 325](#_Toc70352051)

[Game 14: (Minmax-A-B, EF-1) vs (Minimax-A-B, EF-3) – Depth 4 326](#_Toc70352052)

[Game 15: (Minmax-A-B, EF-2) vs (Minimax-A-B, EF-3) – Depth 4 327](#_Toc70352053)

[Game 16: (A-B-Search, EF-1) vs (A-B-Search, EF-2) – Depth 4 328](#_Toc70352054)

[Game 17: (A-B-Search, EF-1) vs (A-B-Search, EF-3) – Depth 4 329](#_Toc70352055)

[Game 18: (A-B-Search, EF-2) vs (A-B-Search, EF-3) – Depth 4 330](#_Toc70352056)

**List of Tables**

[Table 1: Terms and Definitions 7](#_Toc70351259)

[Table 2: Timing Comparison for minmax-a-b vs a-b-search; the latter slightly outperforms 22](#_Toc70351260)

[Table 3: Results for 9 runs at depth 2 23](#_Toc70351261)

[Table 4: results for 9 runs at depth 4 24](#_Toc70351262)

[Table 5: 100mil copy test by David Torrente 25](#_Toc70351263)

[Table 6: 30mil comparisons test by David Torrente 25](#_Toc70351264)

**List of Figures**

[Figure 1: GitHub Contributions – 12 items split equally 3 ways for 4 items per team member 10](#_Toc70351923)

[Figure 2: Main Menu 13](#_Toc70351924)

[Figure 3: UML Class Diagram for the Checkers AI System 17](#_Toc70351925)

[Figure 4: Memory profile of Minmax with EF-1 vs ABS with EF-1 19](#_Toc70351926)

[Figure 5: Memory Profile of Minmax with EF-2 vs Minmax with EF-2 20](#_Toc70351927)

[Figure 6: Memory Profile of ABS with EF-2 vs ABS with EF-3 21](#_Toc70351928)

[Figure 7: CPU, RAM, and OS type used for testing 22](#_Toc70351929)

[Figure 8: EF-3 wins against an AI player who selects the 1st move by letting the enemy block itself in a no-moves state 29](#_Toc70351930)

[Figure 9: Welcome message 47](#_Toc70351931)

[Figure 10: Highlighting choices to guide the user 47](#_Toc70351932)

[Figure 11: A checkers board as represented on the CLI, with color enabled 48](#_Toc70351933)

[Figure 12: Draw end state ASCII art 48](#_Toc70351934)

[Figure 13: One player wins end state ASCII art 49](#_Toc70351935)

[Figure 14: End screen 49](#_Toc70351936)

[Figure 15: Pieces UML Class Node 51](#_Toc70351937)

[Figure 16: Board UML Class Node 53](#_Toc70351938)

[Figure 17: Moves and Jumps for Square 16 54](#_Toc70351939)

[Figure 18: A Board with a Black move from 25 to 18 (a jump) and a capture at 22 55](#_Toc70351940)

[Figure 19: Player UML Class Node 58](#_Toc70351941)

[Figure 20: Game UML Class Node 60](#_Toc70351942)

[Figure 21: Algorithm UML Class Node 62](#_Toc70351943)

[Figure 22: Minimax-a-b Futility Cutoff 64](#_Toc70351944)

[Figure 23: Alpha-Beta-Search Algorithm 65](#_Toc70351945)

[Figure 24: Simulation UML Class Node 66](#_Toc70351946)

[Figure 25: Main application driver 67](#_Toc70351947)

[Figure 26: Sample Run 1 69](#_Toc70351948)

[Figure 27: Sample Run 2, Turn 8 70](#_Toc70351949)

[Figure 28:Sample Run 2 End state 71](#_Toc70351950)

[Figure 29: Depth 6 - MAB EF-3 vs ABS EF-3, Draw 72](#_Toc70351951)

[Figure 30: Sample Run 4 Depth 8! 73](#_Toc70351952)

[Figure 31: Sample Run 5 - Manual Game Ending 74](#_Toc70351953)

[Figure 32: Sample Run #6 - Print the Help Menu 75](#_Toc70351954)

[Figure 33: Minimax-1 vs ABS-1 Depth 2 190](#_Toc70351955)

[Figure 34: Minimax-2 vs ABS-2 Depth 2 253](#_Toc70351956)

[Figure 35: Minimax-3 vs ABS-3 Depth 2 254](#_Toc70351957)

[Figure 36: Minimax-1 vs Minimax-2 Depth 2 255](#_Toc70351958)

[Figure 37: Minimax-1 vs Minimax-3 Depth 2 256](#_Toc70351959)

[Figure 38: Minimax-2 vs Minimax-3 Depth 2 257](#_Toc70351960)

[Figure 39: ABS-1 vs ABS-2 Depth 2 258](#_Toc70351961)

[Figure 40: ABS-1 vs ABS-3 Depth 2 259](#_Toc70351962)

[Figure 41: ABS-2 vs ABS-3 Depth 2 260](#_Toc70351963)

[Figure 42: Minimax-1 vs ABS-1 Depth 4 262](#_Toc70351964)

[Figure 43: Minimax-2 vs ABS-2 Depth 4 324](#_Toc70351965)

[Figure 44: Minimax-3 vs ABS-3 Depth 4 325](#_Toc70351966)

[Figure 45: Minimax-1 vs Minimax-2 Depth 4 326](#_Toc70351967)

[Figure 46: Minimax-1 vs Minimax-3 Depth 4 327](#_Toc70351968)

[Figure 47: Minimax-2 vs Minimax-3 Depth 4 328](#_Toc70351969)

[Figure 48: ABS-1 vs ABS-2 Depth 4 329](#_Toc70351970)

[Figure 49: ABS-1 vs ABS-3 Depth 4 330](#_Toc70351971)

[Figure 50: ABS-2 vs ABS-3 Depth 4 331](#_Toc70351972)

[Figure 51: Full Simulation of 72 total games! 332](#_Toc70351973)

# 1 Introduction

The application covered by this report serves as a two-player checkers game system using artificial intelligence to conduct the game. The application employs two main algorithms for this purpose – MINMAX-A-B as described by Rich and Knight and ALPHA-BETA-SEARCH as described by Russel and Norvig.

Checkers is a full deterministic two-player game; it currently has a mathematical “solution” that may be considered optimal. Jonathan Schaeffer and his team designed an AI algorithm, which, when pitted against an optimal opponent who does not make a mistake, ends the game in a draw. Thus, we may say that checkers is solved and if neither player makes a mistake, the game will end in a draw. Other two-player games, such as Chess and Go, also fall in this category and while they too likely have a solution, it is not yet discovered. The computational complexity of these games is larger than it is for checkers.

This system employs three different static evaluation functions to evaluate a given state of a checkers board and determine whether it is a good state for a player to move into or not. These functions are the “heart” of the AI engine in this program and are what enable the minmax-a-b and alpha-beta-search algorithms to find the “best” move a player can make. The game of checkers is evaluated using the two algorithms, three evaluation functions, and several measures of game complexity – game-tree size and complexity and decision and computational complexity to name a few.

For additional evaluation and flexibility, the program for Project 2 also offers several simulation and game options, which will be covered in detail throughout the report. In short, the application allows a user to conduct a full simulation of a series of games, a single game with custom configuration, test their skills against and AI player, or play a game manually. The application is thorough and goes beyond the scope of the requirements in some areas to provide additional results for analysis and consideration. Results obtained from the simulations show that minmax-a-b and alpha-beta-search expand the same number of nodes and reach the same conclusion, given the same initial configuration. For example, a Red player using minimax-a-b and the first evaluation function versus Black player using minimax-a-b and evaluation 2 simulation reaches the same conclusion as a simulation where both players use alpha-beta-search and the same evaluation function. This is in line with our expectations and confirm that the algorithms are correctly implemented.

## 1.1 Purpose of Report

The intent of this report is to demonstrate a thorough understanding of class concepts and provide a detailed explanation of the Project 2 checkers artificial intelligence application. This report will attempt to exhaustively cover every aspect of the design, implementation, execution, and analysis of the program.

## 1.2 System Scope

The application for Project 2 is an intelligent system for playing checkers. It uses either minmax-a-b or alpha-beta-search to determine the best move, which is evaluated and scored by three unique evaluation functions.

The system shall:

* Display a menu to the user with prompts for input.
* Read in user input for the simulation they desire and the configuration of a game, if applicable.
* Provide users the ability to print verbose output or suppress it as needed.
* Implement the required algorithms and evaluation functions as described on Canvas.
* Play a checkers game using AI through completion.
  + Either one of the players wins or the game ends in a draw after a predefined number of turns.

The system is designed to compile and run on the Texas State Linux hosts. It was additionally tested on Windows – provided certain prerequisites are met, it can execute on the Windows OS as well.

Complete code and documentation, as well as this report, may be found on GitHub here: <https://github.com/TXST-CS5346-AI/project-two>

## 1.3 Terms and Definitions

Table : Terms and Definitions

|  |  |
| --- | --- |
| **Term** | **Definition** |
| The system | “The system” in this report shall refer to the application designed and implemented for Project 2, an intelligent two-player checkers game engine. The application can do a full simulation given the project requirements, a single game with a custom configuration, a player-vs-AI game, or a manual player-vs-player game. |
| EF | Evaluation Function, abbreviated to EF for conciseness |
| EF-1 | Evaluation function #1, developed by Davit Torrente |
| EF-2 | Evaluation function #2, developed by Randall Henderson |
| EF-3 | Evaluation function #3, developed by Borislav Sabotinov |
| Black wins | Either the opposing player (Red) is out of moves or pieces |
| Red wins | Either the opposing player (Black) is out of moves or pieces |
| Draw | After 80 rounds, where each player has 80 moves for a total of 160 maximum allowed moves per game, end in a draw |
| Ply / Turn | Used interchangeably throughout the report. In a two-player game, a ply is one turn taken by one of the players. |
| CLI | Command line interface, a text-based user interface for interacting with this application on Texas State’s Linux servers. |
| ABS | Alpha-Beta-Search algorithm |
| MAB / Minimax | Minmax-A-B algorithm |
| R / r | An upper-case R shall represent a red KING piece, lower-case r will represent a MAN |
| B / b | An upper-case B shall represent a black KING piece, lower-case b will represent a MAN |
| MAN | A standard piece in checkers, also known as a Man. May only move forward diagonally. |
| KING | Once a player’s Man reaches the opposite side of the board (i.e., opponent’s back row) the piece becomes a King and can move backwards on the board. |
| MSB | Most significant bit |
| LSB | Least significant bit |
| OS | Operating System |

## 1.4 Report Overview

The Table of Contents shows where each section is located and contains a list of tables and figures used throughout the report for ease of reference.

**Section 1** covers general, introductory information about the system created for this project. It shows terms and definitions used in this report.

**Section 2** describes the system at a high level. It defines the intended audience and the types of users served by this system, along with a list of features. Any constraints encountered, or assumptions made, during design and implementation are also outlined here.

**Section 3** dives into the system design in greater detail. It shows how to build and run the program, providing a complete listing of all available commands. We examine the overall UML class diagram. We analyze the results of the program and the program’s efficiency. The evaluation functions are examined; I focus on the function I designed and implemented – EF-3 – in greater detail, showing the evolution of the function over time and how I solved problems encountered. Finally, we briefly cover UI design considerations for providing a pleasant user experience.

**Section 4** dives into the source code and outlines each class, as well as any noteworthy data structures or algorithms employed.

**Section 5** comprehensively covers the system by exercising it against all unique available options. Due to the size of certain outputs, only screenshots of the final state of a game are provided in the project report. A complete text output is made available externally, in the **Project2\_depth\_#\_results** directories (where # is the depth) – there are two such directories, one for depth two and another for depth four.

**The appendices** provide large images of all diagrams (where necessary) for readability, as well as the complete source code. Any diagram too large to be readable in its entirety is dissected into sections and each section is presented separately, expanded to fill the screen. Appendix A provides the source code while Appendix B provides sample output for all required 18 runs and lists which files contain the full output.

## 1.5 Contributions

### 1.5.1 Individual Contributions

In this project, our team ensured **all** code was equally distributed. There were 12 items (i.e., GitHub “issues” from a project management perspective) – each team member implemented 4 items equally. We designed the application together as a team and engaged in troubleshooting activities equally.

My contributions to this project include:

* Four main items out of the twelve total: Main.cpp, Simulation.hpp and Simulation.cpp, the alpha-beta-search algorithm and all of its helper functions, and evaluation function three.
* Assisted the team to collectively design the application, including how a Player takes a turn and how the main game loop runs.
* Implemented other functions in Algorithm.cpp, which is over 1K lines of code long.
* Came up with the idea to number the squares on the printed board to improve readability and with ANSII color scheme.
* Participated in group troubleshooting activities, helping to fix code issues across all headers and class implementation files.
* Created the GitHub repo and initial workspace.
* Lead the UML class diagram design – in this project, we designed up-front more than the last and this investment paid off.
* Facilitated pull requests/merges to ensure code synchronization; served as a project manager to facilitate meetings.

Please note that in this project, all team members really did contribute equally to design, implementation, and troubleshooting of the project code base.

|  |  |
| --- | --- |
| Figure : GitHub Contributions – 12 items split equally 3 ways for 4 items per team member |  |

### 1.5.2 Group Member Contributions

I felt all members were active, equal participants in this project and came together to make it a success. All members participated in planning, design, implementation, team meetings, and debugging activities.

Randy worked on four out of twelve items (equally split three ways): Game.hpp and Game.cpp, Player.hpp and Player.cpp, the minimax-a-b algorithm and all its helper functions, and evaluation function two. Randy also assisted in troubleshooting code throughout the entirety of the application, came up with the idea of adding checks before couts for debug output (allowing us to disable it), and implemented helper functions in Algorithm.

David worked on four out of twelve items (equally split three ways): Pieces.hpp and Pieces.cpp, Board.hpp and Board.cpp, the MoveGen function to generate a list of available moves for a given Player, and evaluation function one. He compared a traditional 2D matrix against a bitmap representation of the checkers board and advocated for the latter to improve performance. David also assisted in troubleshooting code throughout the entirety of the application.

There were **142** commits and **40** closed pull requests into the code repository, which is **3,803** lines long.

# 2 System Description

## 2.1 System Perspective

The system is designed to closely simulate a real-life checkers game. We visually display the board and the pieces on it and re-print it each time a player makes a move to show the new state. The sequence of boards therefore constitutes the full path of the game. While the system perspective is focused on playing the game of checkers, it also provides additional verbose output by default to show what is transpiring under the surface. The minmax-a-b and alpha-beta-search algorithms print out (within reason) what they are doing internally. Some of the evaluation functions also print to the screen what decisions they are making at certain points of interest (e.g., sacrificing a MAN to capture an enemy KING receives a higher score).

## 2.2 System Features

The following lists shall comprise a complete listing of the system’s available features:

1. Display a help menu on request.
2. Display a welcome message to the user on start-up.
3. Display a main menu with option codes for the user to select.
4. On option 1, run a full, exhaustive simulation of 72 games.
5. On option 2, prompt the user for the algorithms, evaluation functions, and depth for a single game between two AI players.
6. On option 3, allow the player to play a manual game against themselves or another human player.
7. On option 4, prompt the user for an algorithm, EF, and depth of an AI player and play a human vs AI game.
8. The Ctrl + C command will terminate the program at any time.

## 2.3 Design and Implementation Constraints

The system was built with Texas State Linux servers in mind. While it may operate properly in other environments (e.g., Windows-10, Ubuntu, SLES, etc.) it is guaranteed to run on either the Eros or Zeus TXST servers. Section 3.1 outlines detailed build and execution instructions, which are also available in a shortened format in the Project2-README-A04626934.txt file submitted alongside the source code.

Our team created a pleasant user interface, opting to fully display the entire checkers board, as close to its real-life counterpart as possible, making use of supported ANSII colors on the Texas State hosts. The program was designed to present the user with some common simulation scenarios – a full, exhaustive simulation, a manual game between two human players, an option to play against an AI player, and most importantly – the ability to manually run a single game with a custom configuration. This option is very useful, as it allows a user to focus on what they want. While the first option is useful to exhaustively simulate all possible combinations (including reversing the starting positions for the same algorithm and EF combination), the output is quite large.

The output size was a constraint and full output for each simulated game will be provided in a separate file along with the report. It should also be noted that our team opted to focus on efficiency of the data structures, to minimize space utilization and reduce the size of the game tree (not the number of nodes expanded but the size of each one).

Another approach explored with storing each piece as a full class. Again, to stay lightweight and save space, we opted for a bitmap and representing a piece with the value of one. This means, however, that we lose the ability to add additional contextual information along with a “Piece” as they are no longer represented with a full-fledged class but as a single bit instead. The object-oriented approach of representing a piece as a class would allow us to add information to each piece. For example, we notice after a certain number of turns pieces start going “back-and-forth” between a few spots – jiggering if you will. If we could add a Boolean variable “visitedHomeRow” for each piece that is a King, the piece could “remember” a certain contextual action. Namely that it became a King and travelled from the enemy back row all the way back to our own. We can then use this information to change how the piece behaves by treating it differently – say by using the enemy’s weighted move table instead of our own to force it to traverse across the board in the opposite direction. This specific problem of jittering will be examined in detail below.

## 2.4 Assumptions and dependencies

We assume users of this program will be familiar with using a command line interface (CLI) on the Texas State Linux servers.

The program is dependent on:

* A C++ compiler, such as the GNU C++ compiler for Linux (g++).
* The C++11 language standard.

# 3 System Design and Specification

The program is initially user-input driven. At the beginning a main menu will be displayed (Figure below), prompting the user to select an option. Option one is an exhaustive simulation of each algorithm and each depth, including swapping the player starting positions. This option is provided because results may vary between the same combination of algorithm and evaluation function for different starting positions. For example, Minimax with EF-1 as Red vs Minimax with EF-3 Black may be a draw whereas Minimax EF-3 Red vs Minimax EF-1 Black is a win for black. This is because in our program, Black always moves first. In checkers, just as in chess, moving first is an advantage. It may not be a sizeable advantage, but it does permit the first player to make a move to, say, take the center of the board early.

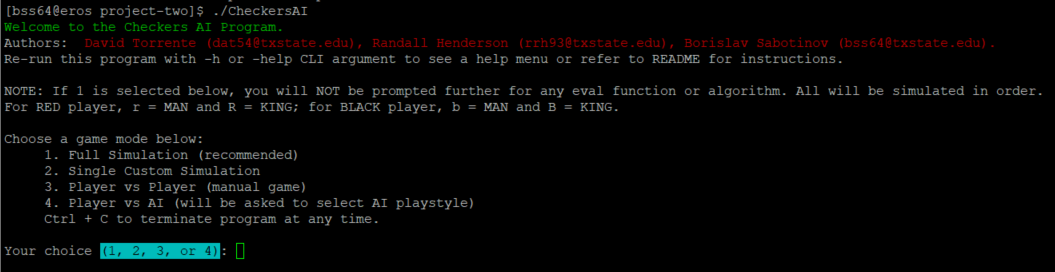


Figure : Main Menu

Regarding the game of checkers itself, a "piece" is a checker piece used in the game. It may be either red or black. Color is an immutable property.

A player has 12 starting pieces of a given color. A standard checkers piece may move only forwards in a diagonal manner. If the piece reaches the opposite end of the board, it becomes a "king" piece. This means it may now move backwards diagonally. A piece from player A may take a piece from player B by "jumping" over it diagonally if the square on the opposite side is empty. Jumps are mandatory - a player must make a jump if one is available to them. If multiple jumps are available, the player simply selects among the choices. A jump may be continuous - the piece must continue capturing enemy pieces if more are available for capture. It may not stop in the middle of a jump. Our program follows the standard rules for checkers as outlined above.

## 3.1 How to Build and Run the Program

The submission for this project consists of 9 files:

1. Project2-A04626934.cpp: source code file.
2. Project2-Report- A04626934.docx: the project report in Word format
3. Project2-Report- A04626934.pdf: the project report in PDF format
4. Project2-README-A04626934.txt: instructions for building and running the program.
5. Project2-ResultLogs-A04626934.zip:
   1. This ZIP contains two directories: Project2\_depth\_2\_results\_A04626934 and Project2\_depth\_4\_results\_A04626934. Each directory contains 18 log files, two files per each of the 9 games played at the respective depth.
   2. The two files per game in these directories are as follows:
      1. abs-1-abs-2-2-full.log and abs-1-abs-2-2-short.log. The first is with full debug output, the second is only the Board and move displayed for brevity. The naming convention for the output files is: <alg>-<eval>-<alg>-<eval>-<depth>-<verbosity>.log, where alg can be either min (for minimax-a-b) or abs (for alpha-beta-search), eval can be either 1, 2 or 3 (representing the evaluation functions), depth is either 2 or 4, and verbosity is either “short” or “full.”
6. Project2-fullSimulation- A04626934.log: 72 games with debug output suppressed.
7. Project2-fullSimulationVerbose- A04626934.zip: a very large file, compressed as a ZIP. It contains verbose output for all 72 games simulated.
8. Project2-Results- A04626934.xlsx: results of the 18 games outlining number of nodes expanded, number of leaf nodes, total nodes, memory usage, and more details. It is also included directly in this report in tables two and three.
9. Project2-UMLDiagram-A04626934.png: the UML Class diagram image for ease of access and readability; zooming in will display details more clearly.

This application is primarily designed to run on Texas State (TXST) Linux servers.

Eros: EROS.CS.TXSTATE.EDU (147.26.231.153)

Zeus: ZEUS.CS.TXSTATE.EDU (147.26.231.156)

You may use WinSCP, FileZilla, or equivalent FTP software to transfer the project files to a TXST Linux host.

1. Project2-A04626934.cpp must be in a directory you own on a TXST host.
2. Build with this command:

**g++ -o Project2 Project2-A04626934.cpp -std=c++11**

1. Run with this command (will use default options):

**./Project2**

1. To see help menu (optional):

**./Project2 -h**

1. To run with no color:

**./Project2 -nc**

1. To run with no debug output:

**./Project2 -no**

1. To run with no debug output and no color:

**./Project2 -ncno**

1. To put std output in a log file:

**./Project2 -nc | tee myFile.log**

It is recommended to use option 2 from the main menu and run a game with one configuration at a time.

## 3.2 Why use C++11?

A small aside but worth mentioning. Why use the C++11 compiler? C++11 now supports:

- lambda expressions,

- automatic type deduction of objects,

- uniform initialization syntax,

- delegating constructors,

- deleted and defaulted function declarations,

- nullptr,

- rvalue references

The language was overhauled with new container classes, algorithms, smart pointers, async capability, and multithreading support, in addition to other useful features. The C++11 compiler is readily available on Texas State Linux servers.

## 3.5 System Class Diagram

Figure 10 below shows a class diagram for the program. I intentionally do not use a simplified UML diagram but instead opt to present full method signatures, including the arguments the member functions will expect. Each class is reviewed in detail in Section 4.

This program is longer and more complex than the one presented in Project-1. It is 3,803 lines long and contains several unique data structures to represent the checkers pieces and the board on which the game is played.

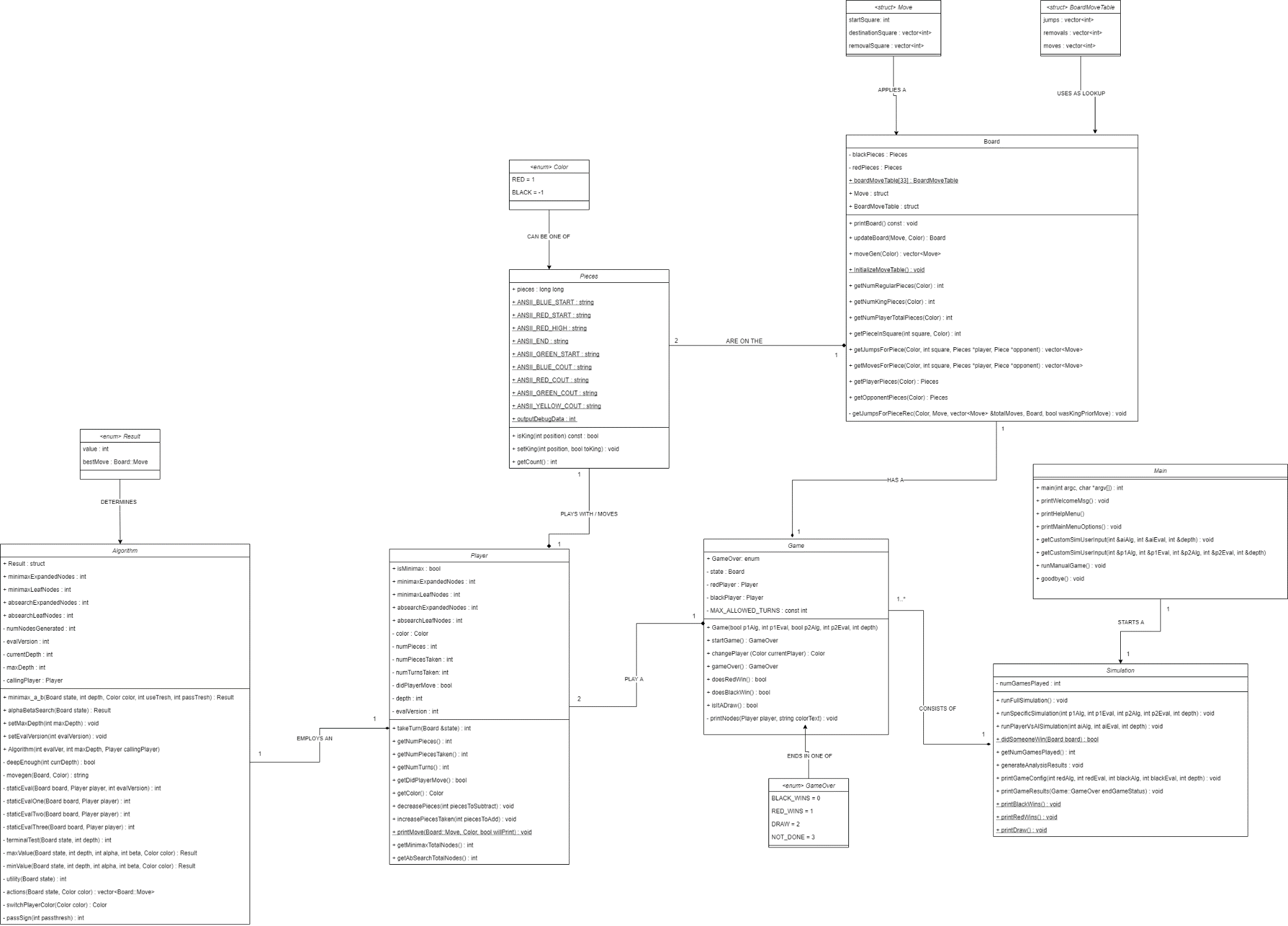


Figure : UML Class Diagram for the Checkers AI System

## 3.6 Analysis of Results

### 3.6.1 How Good Are the Results?

The program is successful in producing correct, repeatable results. It is also robust – it anticipates several edge cases and error scenarios and handles them appropriately. As we will see below, the two main algorithms produce the same game path, reach the same conclusion, and expand the same number of nodes given the same configuration (e.g., comparing Minimax-a-b with alpha-beta-search, both using EF-1). The program was thoroughly tested by the team and is generally free of bugs and flaws. The overall simulation results are in line with our expectation – minimax-a-b and alpha-beta-search are implemented in two different ways, but they produce identical results. Regarding the time it takes for each algorithm to perform the work, it appears alpha-beta-search is slightly faster at depths two and four, although the difference is negligible.

My personal preference is alpha-beta-search. It is very clear in the implementation who is being called (MIN or MAX). There is no need to flip the value, invert or swap the alpha and beta variables (known as passThresh and useThresh in minimax-a-b).

I ran the program 18 times – once for each required simulation (double if we include the runs I did with no output versus with verbose output). Additionally, I ran the program as follows:

* + with the “Player vs AI” option to test myself against EF-2 and how it felt to play against the AI
  + with a manual game to test that it works
  + with depths 6 and again with 8 with the following configuration:
    1. ALPHA-BETA-SEARCH with EF-1 vs ALPHA-BETA-SEARCH with EF-3
  + twice for each of the three rows in table 2 below for a total of six runs
  + once for the full simulation, option 1

Total 29 runs, double if we include the runs I did for capturing verbose and silent output log files included along with this report.

### 3.6.2 Memory and Speed

**Disclaimer**: just as in Project-1, only one student had Visual Studio, which contained a memory profiler. Our team collectively used this tool to profile the application’s memory usage and take some snapshots. The screenshots below are from a team activity that was collectively performed during a Zoom meeting.

The program performs well and there are no noticeable slow-downs or performance issues.

The three subsequent figures below show a captured recording of the program memory profile using Visual Studio 2019. We can see that the program is relatively lightweight and efficient at depth 4. We did test with higher depths and the amount of time the program took to complete a game increased significantly beyond depth 8, likely due to the size of the game tree being explored by the algorithms even with alpha and beta cutoffs.

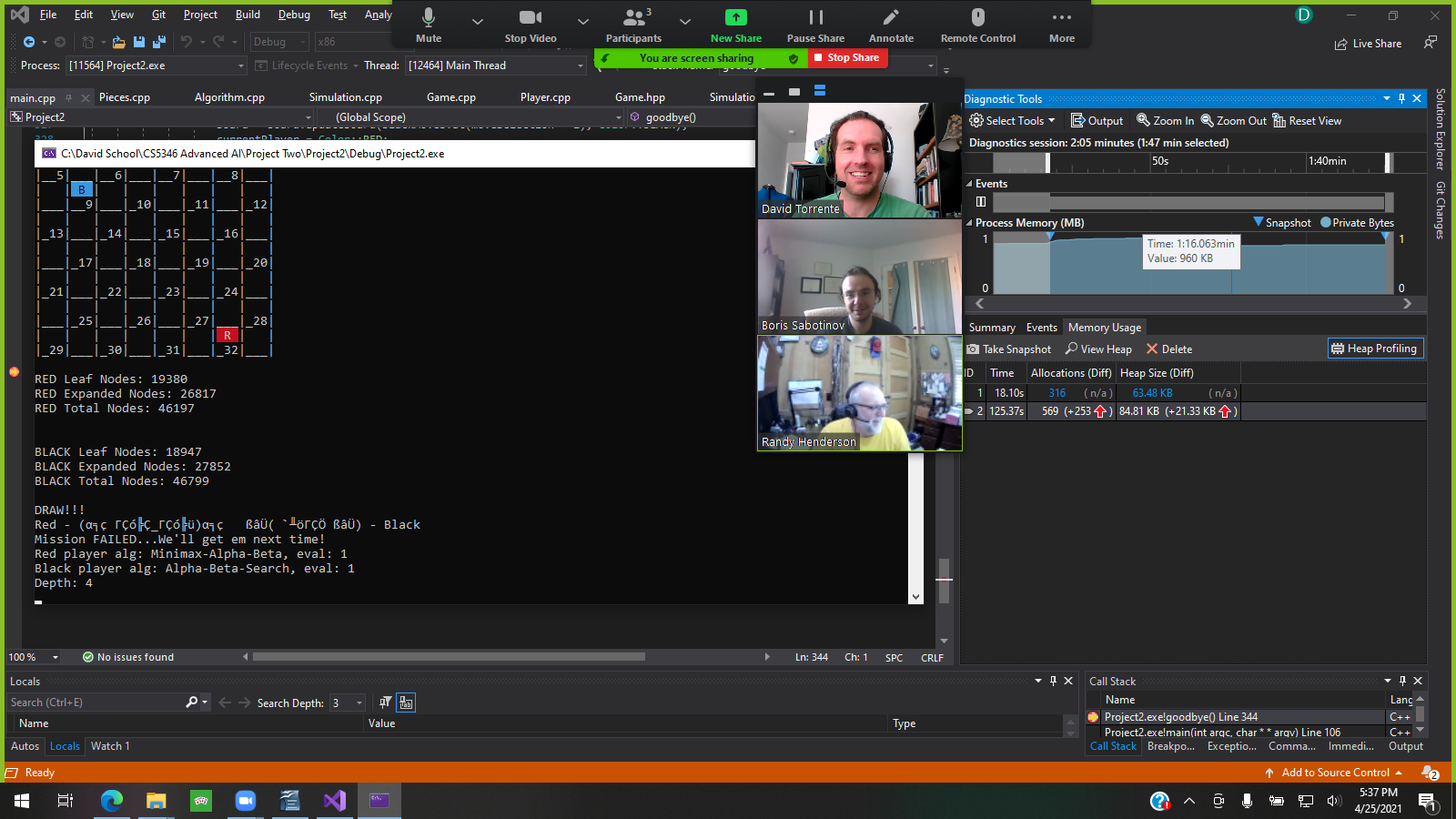


Figure : Memory profile of Minmax with EF-1 vs ABS with EF-1

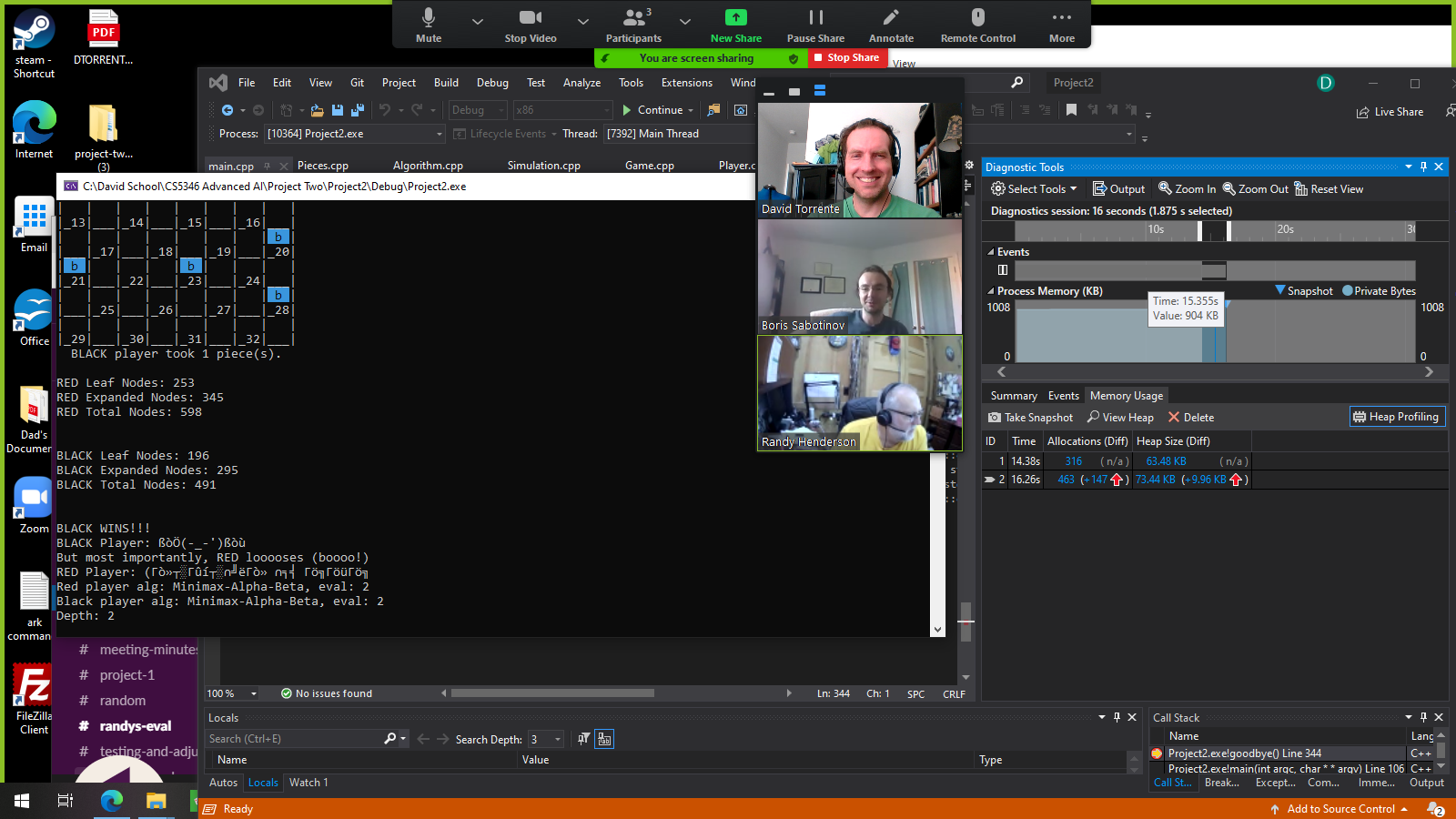


Figure : Memory Profile of Minmax with EF-2 vs Minmax with EF-2

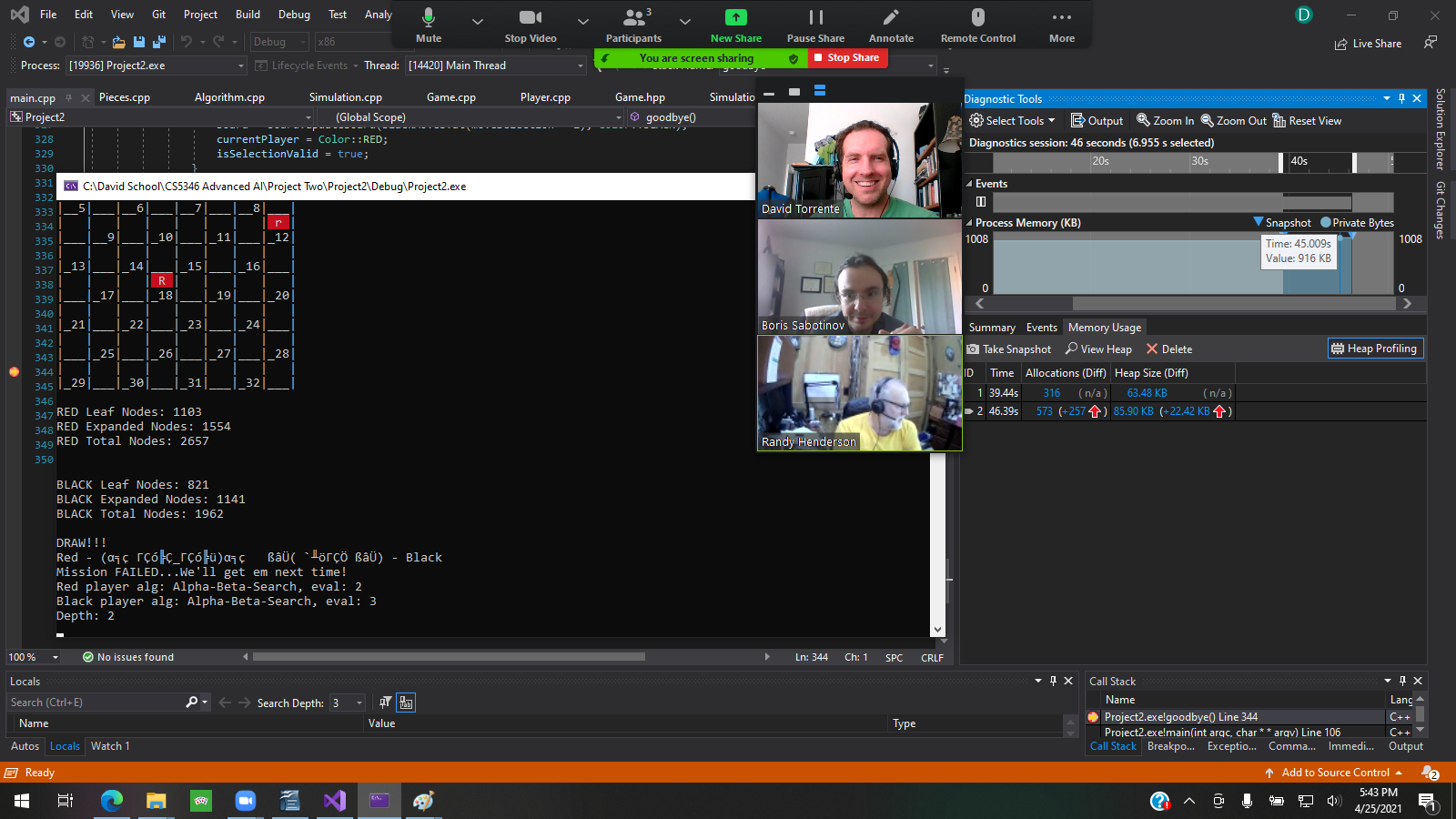


Figure : Memory Profile of ABS with EF-2 vs ABS with EF-3

Table : Timing Comparison for minmax-a-b vs a-b-search; the latter slightly outperforms

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Algo 1** | **Algo 2** | **Eval 1 r** | **Eval 2 b** | **Min Time Depth 2** | **ABS Time Depth 2** | **Min Time**  **Depth 4** | **ABS Time Depth 4** | **Min Time**  **Depth 6** | **ABS Time**  **Depth 6** |
| Minmax | ABS | 1 | 1 | 74.7621ms | 83.1409ms | 470.561ms | 507.63ms | 3167.95ms | 3902.60ms |
| Minmax | ABS | 2 | 2 | 2.8452ms | 2.7658ms | 217.269ms | 191.182ms | 6683.54ms | 3662.31ms |
| Minmax | ABS | 3 | 3 | 29.7645ms | 31.11ms | 115.978ms | 187.228ms | 4389.26ms | 4281.88ms |
|  |  |  |  |  |  |  |  |  |  |

From Table 2 above, we can see that the alpha-beta-search algorithm slightly outperforms minmax-a-b as it has a lower average runtime at higher depths, but at both depths two and four the algorithms perform similarly with minimax-a-b faring slightly better. Green values indicate the “winner” in terms of timing. At depth 6 alpha-beta-search did better overall in timing. However, we must keep in mind that the results heavily depend on the evaluation function being used, as its implementation may be as lightweight or as heavy as the author desires for evaluation of how good a given state may be. The results may also be influenced by the hardware used for testing (CPU and RAM used may be seen in the figure below). Also, what other programs were running at the time of the test may also influence the timing, depending on how available the CPU was at the time.

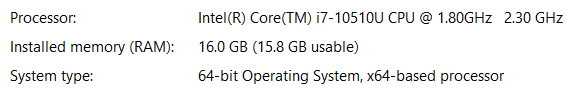


Figure : CPU, RAM, and OS type used for testing

In the two tables below (3 and 4), we see an interesting result. Minimax-a-b and alpha-beta-search perform exactly the same in terms of the number of nodes each algorithm expands. For example, we see that MAB, EF-1 vs MAB, EF-2 expands 3616 red nodes and 4209 black nodes in total. Comparing to ABS, EF-1 vs ABS, EF-2, the results are the same – 3616 red and 4209 black nodes were expanded. This makes intuitive sense. Both algorithms use recursion, MAB directly via a recursive call to itself and ABS indirectly by having two helper functions minValue() and maxValue() call each other. Both functions use alpha and beta values to determine where to perform a cutoff and prune a branch that does not need to be explored further (as a “best” value has already been found elsewhere).

Seeing the results and the two different algorithms perform the same, giving us the same game path, the same conclusion, and the same number of expanded nodes shows us that the algorithms are correctly implemented and behaving as expected.

At depth two, we do not go above 12K generated nodes in total. The game is rather limited in options and the moves are simple and straightforward. All games end in a Draw for us, except Minmax with EF-2 vs ABS with EF-2 (same EF against itself); we get a win for Black, likely because of the slight “first move” advantage Black has.

Table : Results for 9 runs at depth 2

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Runs at Depth 2 (r = red, b = black)** | | | | | | | | | | |
| **Algo 1** | **Algo 2** | **Eval 1 r** | **Eval 2 b** | **Path Len** | **Leaf Nodes** | **Expanded Nodes** | **Total Nodes (leaf)** | **Time (sec)** | **Peak mem** | **Win / Loss** |
| Minmax | ABS | 1 | 1 | 80 | 2596r 3004b | 3295r 3675b | 5891r 6679b | 17.9 | 968 KB | Draw |
| Minmax | ABS | 2 | 2 | 25 | 253r 196b | 345r 295b | 598r 491b | 1.6 | 904 KB | Black |
| Minmax | ABS | 3 | 3 | 80 | 1158r 1202b | 1674r 1533b | 2832r 2735b | 9.07 | 924 KB | Draw |
|  |  |  |  |  |  |  |  |  |  |  |
| Minmax | Minmax | 1 | 2 | 80 | 1648r 1620b | 1968r 2589b | 3616r 4209b | 10.18 | 940 KB | Draw |
| Minmax | Minmax | 1 | 3 | 80 | 1604r 1640b | 2511r 1950b | 4115r 3590b | 13.37 | 932 KB | Draw |
| Minmax | Minmax | 2 | 3 | 80 | 1103r 821b | 1554r 1141b | 2657r 1962b | 6.18 | 916 KB | Draw |
|  |  |  |  |  |  |  |  |  |  |  |
| ABS | ABS | 1 | 2 | 80 | 1648r 1620b | 1968r 2589b | 3616r 4209b | 10.15 | 924 KB | Draw |
| ABS | ABS | 1 | 3 | 80 | 1604r 1640b | 2511r 1950b | 4115r 3590b | 13.25 | 936 KB | Draw |
| ABS | ABS | 2 | 3 | 80 | 1103r 821b | 1554r 1141b | 2657r 1962b | 6.27 | 928 KB | Draw |

At depth four, we are already seeing a tremendous increase in the number of nodes we expand. The game tree explodes in complexity from under 7K nodes at depth 2 (which only is observed on row 1, Min w/ EF-1 vs ABS w/ EF-1) to over 63K nodes at depth 4, a near 10x increase. I tested at depth 6 and saw the same thing, about a 10x increase in nodes generated, where the average was around 250,000 leaf nodes for one player (compared to 25,000 at depth 4). We can infer that increasing the depth by two leads to a tenfold increase in the number of nodes we can expect to generate. One notable exception is my evaluation function, EF-3, when tested against itself at depth 4 led to fewer expanded nodes. Perhaps because each player made the expected or “rational” move from each other’s perspective.

Table : results for 9 runs at depth 4

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Runs at Depth 4 (r = red, b = black)** | | | | | | | | | | |
| **Algo 1** | **Algo 2** | **Eval 1 r** | **Eval 2 b** | **Path Len** | **Leaf Nodes** | **Expanded Nodes** | **Total Nodes (leaf)** | **Time (sec)** | **Peak mem** | **Win / Loss** |
| Minmax | ABS | 1 | 1 | 80 | 19380r 18947b | 26817r 27852b | 46197r 46799b | 95.44 | 960 KB | Draw |
| Minmax | ABS | 2 | 2 | 80 | 28641r 29419b | 44847r 39043b | 73488r 68462b | 39.27 | 964 KB | Draw |
| Minmax | ABS | 3 | 3 | 22 | 3052r 4894b | 4814r 7203b | 7866r 12097b | 30.08 | 952 KB | Red |
|  |  |  |  |  |  |  |  |  |  |  |
| Minmax | Minmax | 1 | 2 | 80 | 19139r 14910b | 25786r 26728b | 44925r 41638b | 69.42 | 980 KB | Draw |
| Minmax | Minmax | 1 | 3 | 80 | 24683r 24683r | 37489r 51299b | 62172r 89789b | 185.51 | 972 KB | Draw |
| Minmax | Minmax | 2 | 3 | 80 | 23521r 14272b | 39900r 18544b | 63421r 32816b | 58.69 | 984 KB | Draw |
|  |  |  |  |  |  |  |  |  |  |  |
| ABS | ABS | 1 | 2 | 80 | 19139r 14910b | 25786r 26728b | 44925r 41638b | 68.07 | 960 KB | Draw |
| ABS | ABS | 1 | 3 | 80 | 24683r 24683r | 37489r 51299b | 62172r 89789b | 187.04 | 948 KB | Draw |
| ABS | ABS | 2 | 3 | 80 | 23521r 14272b | 39900r 18544b | 63421r 32816b | 59.69 | 968 KB | Draw |

### 3.6.3 Optimizations

Our team used object-oriented design principles while keeping some data structures (for the board) low-level to save time and space. The minimax-a-b and alpha-beta-search algorithms are implemented as-is. They are an exact implementation of the algorithms provided on Canvas and the respective book referenced for each algorithm.

A notable improvement is our decision to forego a traditional 2D array representation of a board and use a bitmap.

We initially examined a “traditional” approach for representing the checkers board and its pieces – a class Piece to represent each piece and the board represented as a 2D array with row and column. After simulating this approach against a bitmap representation of the board, we quickly decided to go with the latter as it provided a sizeable difference in performance and runtime.

David Torrente simulated copying a board 100 million times, comparing an int[8][4] vs char[8][4] vs a long long bitfield.

The results from this test are shown below (only the first 4 runs, though 10 were performed).

Table : 100mil copy test by David Torrente

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test # (time in sec): | 1 | 2 | 3 | 4 | Avg (for 10 runs) |
| int[8][4] | 6.08 | 6.05 | 5.92 | 5.83 | 5.863 |
| char[8][4] | 5.73 | 5.80 | 5.83 | 5.80 | 5.8 |
| long long | 0.28 | 0.29 | 0.28 | 0.28 | .288 |

Before we started implementing this project, we wanted to decide on the data structure to represent the Board and pieces. In the table above, we observed that a bitmap is 20 times more efficient on average than either an int or a char 2D array. Bitmaps were the clear winner. In the table below, however, we see a drawback – checking the position of a piece in an array is O(1) because we simply access it directly via the index (e.g., board[1][3]). Whereas with bitmaps we need to perform a shift and compare. When tested with 30 million comparisons, bitmaps were about 4 times slower than the competition.

Table : 30mil comparisons test by David Torrente

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test # (time in sec): | 1 | 2 | 3 | 4 | Avg (for 10 runs) |
| int[8][4] | 2.63 | 2.5 | 2.6 | 2.5 | 2.605 |
| char[8][4] | 2.56 | 2.51 | 2.31 | 2.48 | 2.528 |
| long long | 11.8 | 11.51 | 11.48 | 11.47 | 11.566 |

Our team opted to sacrifice a bit of speed to gain a significant improvement in memory utilization and save on space.

## 3.7 Evaluation Functions

There are three EFs implemented for this program, EF-1 as written by David Torrente, EF-2 as written by Randall Henderson, and EF-2 as written by Borislav Sabotinov. Each evaluation function takes in a Board and a Color and returns an integer score, which signifies how good we think is the provided state.

I will begin by discussing EF-3, as it is the function I designed and implemented for this project.

### 3.7.1 Evolution of Evaluation Function Three (EF-3)

The third evaluation function was the first to be implemented and pushed to the repository for testing in a way that returned an integer value with some meaning behind it. Initially we simply always returned 1, which would create a good predictable game – AI players would always take the first move available to them. In the initial state of EF-3 referenced below, we can see I began by assigning each position on the board a value or weight. The intent was to guide the behavior of the pieces using a strategy I employed when I played checkers against an AI online to get a feel for the game.

My strategy was to keep two checker pieces on the back row, in squares that had two available moves. That way, I could “defend” my back row against enemy checkers that tried to get a King piece while remaining safe (back row cannot be captured). While I placed some value on the sides due to their relative safety from captures, I placed a higher value on squares in the center of the board. I also calculated a “mobility” score – given the provided state of the game, where could my pieces move to – would I be able to control squares with a higher value like in the center? The final value returned by this initial EF was a summation of the number of pieces I had including kings, the mobility score, and the score of my current position. As we did not have other evaluation functions to test against at the time, I tested it against itself. Results were either a draw or a win for black, who moves first. When I played against a player who used a “always take the first available move” simple strategy, the algorithm would lose at both depths.

#### Initial state of EF-3

1. int Algorithm::evalFunctThree(Board state, Color color)
2. {
3. int squareValuesForRed[] = { 100, 1, 100, 1,
4. 2, 1, 1, 2,
5. 1, 1, 1, 2,
6. 2, 3, 3, 1,
7. 2, 3, 3, 4,
8. 2, 3, 3, 1,
9. 3, 3, 3, 4,
10. 10, 10, 10, 10 };
11. int squareValuesForBlack[] = { 10, 10, 10, 10,
12. 4, 3, 3, 3,
13. 1, 3, 3, 2,
14. 4, 3, 3, 2,
15. 1, 3, 3, 2,
16. 2, 1, 1, 1,
17. 2, 1, 1, 2,
18. 1, 100, 1, 100 };
20. int numPieces = state.getNumPlayerTotalPieces(color);
21. int numKingsScore = state.getNumKingPieces(color) \* 10;
22. std::vector<Board::Move> moves = state.moveGen(color);
23. int advancementScore = 0;
24. for (int move = 0; move < moves.size(); move++)
25. {
26. if (color == Color::RED)
27. {
28. advancementScore += squareValuesForRed[moves.at(move).destinationSquare.back()];
29. }
30. else if (color == Color::BLACK)
31. {
32. advancementScore += squareValuesForBlack[moves.at(move).destinationSquare.back()];
33. }
34. }
35. int positionScore = 0;
36. Pieces playerPieces = state.getPlayerPieces(color);
38. for (int piece = 0; piece < 32; piece++)
39. {
40. if (color == Color::RED)
41. {
42. int pieceBit = (playerPieces.pieces >> piece) & 1;
43. if (pieceBit == 1)
44. {
45. positionScore += squareValuesForRed[piece];
46. }
48. }
49. else if (color == Color::BLACK)
50. {
51. int pieceBit = (playerPieces.pieces >> piece) & 1;
52. if (pieceBit == 1)
53. {
54. positionScore += squareValuesForBlack[piece];
55. }
56. }
58. }
59. int compositeScore = numPieces + numKingsScore + advancementScore + positionScore;
60. return compositeScore;
61. }

After some trial and error, I added additional logic to the function and arrived at the second snapshot of EF-3 as seen below.

After testing, I discovered that this second version of EF-3 had a flaw. Starting on line 115 and again on 152 in the second snapshot below, I was attempting to determine if the player could retaliate after losing a piece. I would look for the surrounding squares, but my calculations were off as I did not notice the mathematical difference between squares differs depending on whether we are on an odd or even row. Suppose an enemy takes our piece and lands on the second row in square 6. We would add 3 to get the bottom left square of 9 and add 4 to get right square of 10. But suppose now the enemy landed on 10 instead, which is on row 3 – we would have to add 4 to get the bottom left square and 5 to get the bottom right. I had not noticed this when designing the algorithm, leading to unpredictable scoring. The algorithm would assume we could retaliate when in fact we could not, for instance.

My strategy now was to use two additional tables to drive the behavior of King pieces. The values for each player are mirror opposites and the intent was to play very aggressively – simply move all the pieces forward, towards the opponent’s back row in the hopes of getting a king, then move the king to the center of the board and score some captures on the way. I had a captures section to create a score of how many pieces, if any, the player could hope to capture on this state. Similarly, I had a section for casualties – how many pieces the player would lose. I decided in an aggressive checkers strategy losing a piece is not always a bad thing – if we do not lose a king, if the enemy does not gain a king by landing on our back row, and if we can retaliate immediately on the next turn, we would want to trade the piece for a position advantage.

At this stage we had preliminary implementations for both EF-1 and EF-2 and despite its flaws this version of EF-1 forced a Draw against both. It also forced a Draw against my arch-nemesis, the simple “take the first move” strategy of our test player mentioned earlier. However, it won against this test player in a unique way at depth four! When I first saw the board, I thought there was a bug but the win for Black in Figure 8 below is legitimate. Red has a 10 to 1 advantage in pieces and 4 kings to my one remaining king piece. But because my piece was able to sneak behind enemy lines so to speak, the simple AI enemy boxed itself in and ran out of moves at turn 43, giving Black the win.

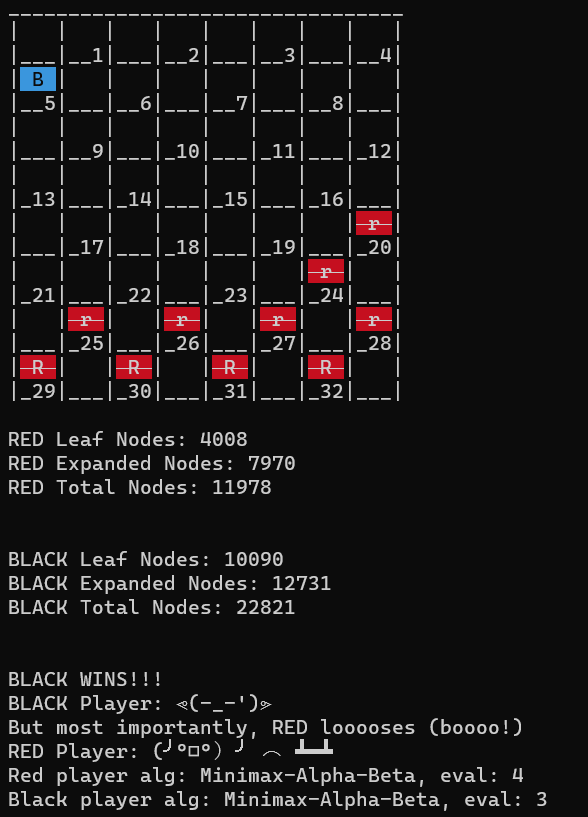


Figure : EF-3 wins against an AI player who selects the 1st move by letting the enemy block itself in a no-moves state

#### Second snapshot of EF-3 evolution in design and implementation

1. int Algorithm::evalFunctThree(Board state, Color color)
2. {
3. int squareValuesForRed[] = { 100, 0, 100, 0,
4. 1, 1, 1, 1,
5. 1, 1, 1, 1,
6. 1, 9, 9, 9,
7. 10, 10, 10, 1,
8. 1, 11, 11, 11,
9. 12, 12, 12, 10,
10. 110, 110, 110, 110};
11. int squareValuesForBlack[] = { 110, 110, 110, 110,
12. 10, 12, 12, 12,
13. 11, 11, 11, 1,
14. 1, 10, 10, 10,
15. 9, 9, 9, 1,
16. 1, 1, 1, 1,
17. 1, 1, 1, 1,
18. 0, 100, 0, 100};
19. int squareValuesForRedKing[] = { 3, 1, 1, 5,
20. 5, 2, 2, 3,
21. 3, 10, 10, 5,
22. 5, 10, 10, 3,
23. 3, 10, 10, 5,
24. 5, 10, 10, 10,
25. 3, 2, 2, 5,
26. 5, 1, 1, 3};
27. int squareValuesForBlackKing[] = { 3, 1, 1, 5,
28. 5, 2, 2, 3,
29. 3, 10, 10, 5,
30. 5, 10, 10, 3,
31. 3, 10, 10, 5,
32. 5, 10, 10, 10,
33. 3, 2, 2, 5,
34. 5, 1, 1, 3};
35. const int KING = 2, MAN = 1;
36. int numPieces = state.getNumPlayerTotalPieces(color);
37. int numKingsScore = state.getNumKingPieces(color);
38. int casualtyScore = 0, captureScore = 0, positionScore = 0, playerPiece = 0, enemyPiece = 0;
39. std::vector<Board::Move> playerMoves = state.moveGen(color);
40. std::vector<Board::Move> enemyMoves = state.moveGen(switchPlayerColor(color));
41. Pieces playerPieces = state.getPlayerPieces(color);
42. Pieces opponentPieces = state.getOpponentPieces(color);
43. Pieces \*p\_playerPieces = &playerPieces;
44. Pieces \*p\_opponentPieces = &opponentPieces;
45. std::vector<Board::Move> playerJumpsForPiece;
46. std::vector<Board::Move> opponentJumpsForPiece;
47. const long long redBackRowGrp = (1LL << 1) | (1LL << 2) | (1LL << 3) | (1LL << 4);
48. const long long blackBackRowGrp = (1LL << 29) | (1LL << 30) | (1LL << 31) | (1LL << 32);
49. const long long sideColumnGrp = (1LL << 5) | (1LL << 13) | (1LL << 21) | (1LL << 12) | (1LL << 20) | (1LL << 28);
50. // CHECK TERMINAL STATE - Enemy is out of moves, this is a game ending move
51. if (enemyMoves.size() == 0)
52. return std::numeric\_limits<int>::max() - 1000;
53. else if (playerMoves.size() == 0)
54. return 0; // bad move for us!
55. // BEGIN ADVANCEMENT SCORE SECTION
56. int advancementScore = 0;
57. for (int move = 0; move < playerMoves.size(); move++)
58. {
59. if (color == Color::RED)
60. advancementScore += squareValuesForRed[playerMoves.at(move).destinationSquare.back()];
61. else if (color == Color::BLACK)
62. advancementScore += squareValuesForBlack[playerMoves.at(move).destinationSquare.back()];
63. }
64. // END ADVANCEMENT SCORE SECTION
65. /\* BEGIN CAPTURES SECTION
66. If for any given piece we can make more than one capture, we are done.
67. That is a very good move and we prioritize it.
68. Otherwise if we can make at least one capture, we give this state a high score
69. but we still want to examine others to see if they may be bettr
70. \*/
71. for (int i = 0; i < playerMoves.size(); i++)
72. {
73. if (playerMoves.at(i).removalSquare.size() > 1)
74. {
75. if (Pieces::ouputDebugData)
76. std::cout << "We can capture multiple pieces!" << std::endl;
77. return std::numeric\_limits<int>::max() - 10; // great move!
78. }
79. else if (playerMoves.at(i).removalSquare.size() == 1)
80. {
81. return std::numeric\_limits<int>::max() / 2; // great move but can we do better?
82. }
83. }
84. for (int j = 0; j < enemyMoves.size(); j++)
85. {
86. if (enemyMoves.at(j).removalSquare.size() > 1)
87. {
88. if (Pieces::ouputDebugData)
89. std::cout << "Enemy can capture multiple pieces!" << std::endl;
90. return 0; // bad move!
91. }
92. else if (enemyMoves.at(j).removalSquare.size() == 1)
93. {
94. int capturedPieceType = state.getPieceInSquare(enemyMoves.at(j).removalSquare.at(0), color);
95. if (capturedPieceType == KING)
96. return 0; // bad move!
97. else if (capturedPieceType == MAN)
98. {
99. int opponentDestinationSqr = enemyMoves.at(j).destinationSquare.at(0);
100. if (color == Color::RED)
101. {
102. // if we are RED, opponent is Black; if BLACK enemy lands on our back row, avoid at all cost
103. // we are therefore trying to minimize the chance of an enemy getting a KING
104. if ((1 << opponentDestinationSqr) & redBackRowGrp)
105. return 0;
106. // opponent lands on their own back row; not so bad but we can't retaliate so avoid
107. else if ((1 << opponentDestinationSqr) & blackBackRowGrp)
108. return 0;
109. // opponent lands on one of the side squares, where we cannot retaliate. Avoid as well
110. else if ((1 << opponentDestinationSqr) & sideColumnGrp)
111. return 0;
112. // finally, we exhaust negative scenarios and are where we want to be
113. else
114. {
115. // get the square numbers around the enemy
116. int playerTopLeftSqr = opponentDestinationSqr - 4;
117. int playerTopRightSqr = opponentDestinationSqr - 3;
118. int playerBottomLeftSqr = opponentDestinationSqr + 4;
119. int playerBottomRightSqr = opponentDestinationSqr + 5;
120. // get our pieces in those squares, if we have any
121. int playerTopLeftPiece = state.getPieceInSquare(playerTopLeftSqr, Color::RED);
122. int playerTopRightPiece = state.getPieceInSquare(playerTopRightSqr, Color::RED);
123. int playerBottomLeftPiece = state.getPieceInSquare(playerBottomLeftSqr, Color::RED);
124. int playerBottomRightPiece = state.getPieceInSquare(playerBottomRightSqr, Color::RED);
125. // Since we are RED, our MEN can only advance forward top to bottom. Top pieces adjacent to where the opponent
126. // landed can be MAN or KING. Bottom pieces however must be KING, otherwise we can't move backwards
127. if (playerTopLeftPiece == MAN || playerTopRightPiece == MAN ||
128. playerTopLeftPiece == KING || playerTopRightPiece == KING ||
129. playerBottomLeftPiece == KING || playerBottomRightPiece == KING)
130. {
131. casualtyScore += 500;
132. }
133. else
134. return 0; // we lost a piece and there is nothing we can do about it, avoid this state
135. }
136. }
137. else
138. {
139. // if we are BLACK, opponent is Red; if RED enemy lands on our back row, avoid at all cost
140. // we are therefore trying to minimize the chance of an enemy getting a KING
141. if ((1 << opponentDestinationSqr) & blackBackRowGrp)
142. return 0;
143. // opponent lands on their own back row; not so bad but we can't retaliate so avoid
144. else if ((1 << opponentDestinationSqr) & redBackRowGrp)
145. return 0;
146. // opponent lands on one of the side squares, where we cannot retaliate. Avoid as well
147. else if ((1 << opponentDestinationSqr) & sideColumnGrp)
148. return 0;
149. // finally, we exhaust negative scenarios and are where we want to be
150. else
151. {
152. // get the square numbers around the enemy
153. int playerTopLeftSqr = opponentDestinationSqr - 4;
154. int playerTopRightSqr = opponentDestinationSqr - 3;
155. int playerBottomLeftSqr = opponentDestinationSqr + 4;
156. int playerBottomRightSqr = opponentDestinationSqr + 5;
157. // get our pieces in those squares, if we have any
158. int playerTopLeftPiece = state.getPieceInSquare(playerTopLeftSqr, Color::RED);
159. int playerTopRightPiece = state.getPieceInSquare(playerTopRightSqr, Color::RED);
160. int playerBottomLeftPiece = state.getPieceInSquare(playerBottomLeftSqr, Color::RED);
161. int playerBottomRightPiece = state.getPieceInSquare(playerBottomRightSqr, Color::RED);
162. // Since we are BLACK, our MEN can only advance forward bottom to top. Top pieces adjacent to where the opponent
163. // landed can KING only or we cannot move backwards. Bottom pieces can be either MAN or KING
164. if (playerTopLeftPiece == KING || playerTopRightPiece == KING ||
165. playerBottomLeftPiece == MAN || playerBottomRightPiece == MAN ||
166. playerBottomLeftPiece == KING || playerBottomRightPiece == KING)
167. {
168. casualtyScore += 500;
169. }
170. else
171. return 0; // we lost a piece and there is nothing we can do about it, avoid this state
172. }
173. }
174. }
175. }
176. }
177. // MAIN LOOP FOR SCORING CALCULATIONS
178. for (int piece = 0; piece < 32; piece++)
179. {
180. // BEGIN POSITION SCORE SECTION
181. if (color == Color::RED)
182. {
183. playerPiece = state.getPieceInSquare(piece, color);
184. if (playerPiece == MAN)
185. positionScore += squareValuesForRed[piece];
186. else if (playerPiece == KING)
187. positionScore += squareValuesForRedKing[piece];
188. enemyPiece = state.getPieceInSquare(piece, Color::BLACK);
189. if (enemyPiece == MAN)
190. positionScore -= squareValuesForBlack[piece];
191. else if (enemyPiece == KING)
192. positionScore -= squareValuesForBlackKing[piece];
193. }
194. else if (color == Color::BLACK)
195. {
196. playerPiece = state.getPieceInSquare(piece, color);
197. if (playerPiece == MAN)
198. positionScore += squareValuesForBlack[piece];
199. else if (playerPiece == KING)
200. positionScore += squareValuesForBlackKing[piece];
201. }
202. // END POSITION SCORE SECTION
203. }
204. int compositeScore = numPieces + numKingsScore + advancementScore + positionScore + captureScore + casualtyScore;
205. return compositeScore;
206. }

### 3.7.2 Final state of EF-3

Below is the final version of EF-3. I redesigned it section by section, applying lessons learned from previous iterations. I removed the many returns at different locations and opted to simply provide an integer value for those situations. I removed one of the king arrays, so both colors now use the same position value table. I also marked the first row of the player with a higher score than the one in front of it. From Red’s perspective it is the 3rd row from the top of the displayed board (squares 9, 10, 11, and 12). From Black’s perspective it is the 3rd row from the bottom (squares 21, 22, 23, and 24). Because only this row has moves available, the first row will advance. But once this happens, the second row now has a piece which can move into the gap left by the first piece. This ensures we advance somewhat in waves – one piece from the front row moves but then the gap is filled by a piece behind it and so forth. I still opted to keep two pieces on defense in the back row, giving me 10 pieces to use offensively.

The values get progressively higher as we get towards the opponent back row – I try to attack the “double corner” of the enemy, the square on their home row at the edge, which connects to two other squares. I try to stay away from getting boxed in the other corner which is only accessible via one adjacent square. This is an effective strategy at preserving pieces and advancing systematically towards a goal of obtaining a king. The drawback is that the king may get boxed in if we get lucky and our Men behind score some captures and follow closely behind, limiting our King’s mobility.

This final version of EF-3 has a more intelligent method for scoring the number of pieces on the board. Each piece has an associated weight equal to the number of moves it may perform. Thus, a Man is worth 2 and a King is worth 4 points. We look at our pieces – their number and type – in relation to the enemy and have the option to assign a bonus to boards where we have a numerical advantage. This section starting on line 57 and ending on 65 is currently commented out as it led to worse performance against other AI players using EF-2 and EF-3, the reason for which needs to be investigated.

A clever use of bit shifting is used to quickly check if a piece is in a particular area of the board. Suppose we are Red, and we want to check if an enemy Black piece may land anywhere on our back row.

redBackRowGrp = (1LL << 1) | (1LL << 2) | (1LL << 3) | (1LL << 4);

On line 83 we declare the redBackRowGrp as shown above. It is of type long long, thus 64 bits. We shift left once, bitwise-OR with a shift left of two. Repeat for two and three. If we start with 0001 and shift left 1, we get 0010. We bitwise-OR this with 0001 shifted twice (0100), giving us 0110. For this example, we shall stop there as this provides the general idea. We can use this to check if an opponent has landed on our back row:

if ((1 << opponentDestinationSqr) & redBackRowGrp)

If the opponent landed on 2, we shift and get 0100. We then bitwise-AND with 0110 (our group value) and we get 0100. This is a non-zero value, which will evaluate to true in our if condition, so we know the opponent has landed somewhere on our back row.

I now also multiply the value of a square by the weight of the piece if we have a piece on that square. So, a state leading to a piece in square 1 would lead to 1000 for the position \* 4 (value of a king) = 4000, a desirable state. Redesigned also are the captures and casualties sections – less returns, replaced with scores. Also, the method for determining if it is safe to conduct a trade and sacrifice a Man, where we can retaliate on the next turn, corrects the flaw described in the second snapshot. We still return an integer value to signal how good this state is but the components we sum up are greater this time and the method by which we arrive at our score more complex.

What I learned while implementing the evaluation function is that even though we are returning a single number, much information can be conveyed in that one value if we arrive at it intelligently. Because the algorithms will look for a “good” state and prune unpromising branches, the more accurate the score provided by the EF is, the better we can perform.

One limitation, however, is that the AI player only knows their own evaluation function – we cannot read the enemy’s mind just as in real life. If we are playing against an AI using the same EF, results and the enemy behavior are more predictable. If, on the other hand, we are playing with EF-1 and the enemy with EF-2, our results simulate the enemy moves as if they were using EF-1. We are only as good as our evaluation function, so we must take great care to anticipate scenarios and evaluate the state of the board and its many possibilities.

One notable limitation all three evaluation functions encountered is a back-and-forth jiggering towards the late game, leading to a draw. This is because we can drive the pieces towards a destination relatively easily but once they get there, getting them to behave in a different manner is more complex. One possible solution was mentioned earlier – introduce a Piece class for each piece and have each checker piece remember if it has performed some action or visited some area. If it has, treat it differently from other pieces to drive it to traverse the board. Yet another solution is to remember the last few states of the game because a real human player would detect they are going back and forth and try something different. Especially when an AI player has a large piece advantage, there is really no need for the game to end in a Draw as the player can start moving towards the enemy. But here we are limited by the “horizon effect” of the defined depth. At depth 2, we can only see two moves ahead. If the enemy pieces are beyond two moves away, we do not see a capture possible and thus we have no reason to make different moves. At higher depths (8 and beyond) it is possible we may detect the enemy and a possible capture. Finally, a simple strategy of taking a random move if a back-and-forth jitter is detected may work as well, forcing the player into more unique states and possibly closer to the opposing player.

#### Final version of EF-3

1. int Algorithm::evalFunctThree(Board state, Color color)
2. {
3. /\*  Declaration of "boards" with a weight for each square
4. Two for regular pieces of each color and two for kings of each color
5. Encourage player to keep two pieces in the back for defense
6. leaving 10 pieces for offense. Advance to the center
7. but somewhat in waves as a cluster, to avoid suicidal pieces that expose themselves
8. Try to attack the opponent's "double corner" from where a kinged piece can escape faster
9. \*/
10. int squareValuesForRed[] = {7, 1, 7, 1,
11. 1, 2, 2, 2,
12. 1, 5, 5, 5,
13. 1, 3, 3, 3,
14. 1, 4, 4, 4,
15. 1, 5, 250, 250,
16. 1, 250, 500, 500,
17. 50, 100, 100, 1000};
18. int squareValuesForBlack[] = {1000, 100, 100, 50,
19. 500, 500, 250, 1,
20. 250, 250, 5, 1,
21. 4, 4, 4, 1,
22. 3, 3, 3, 1,
23. 5, 5, 5, 1,
24. 2, 2, 2, 1,
25. 1, 7, 1, 7};
26. // Kings preference for center, with some traversal lines
27. // to attempt and avoid a "back-and-forth" pattern
28. int squareValuesForKing[] = {1, 1, 1, 1,
29. 1, 5, 5, 55,
30. 5, 15, 45, 1,
31. 1, 5, 35, 5,
32. 5, 25, 25, 1,
33. 1, 15, 5, 15,
34. 5, 5, 5, 10,
35. 1, 1, 1, 1};
36. std::string colorTxt = (color == Color::RED) ? " (RED is Friendly) " : " (BLACK is Friendly) ";
37. // KING has 4 moves max, so value is 4; MAN has 2 moves max so values is 2
38. const int KING = 2, MAN = 1, KING\_VALUE = 4, MAN\_VALUE = 2;
39. int numPlayerTotalPieces = state.getNumPlayerTotalPieces(color);
40. int numEnemyTotalPieces = state.getNumPlayerTotalPieces(switchPlayerColor(color));
41. int numPlayerTotalKings = state.getNumKingPieces(color);
42. int numEnemyTotalKings = state.getNumKingPieces(switchPlayerColor(color));
43. int numPlayerTotalMen = numPlayerTotalPieces - numPlayerTotalKings;
44. int numEnemyTotalMen = numEnemyTotalPieces - numEnemyTotalKings;
45. int numKingsScore = numPlayerTotalKings \* KING\_VALUE;
46. int numMenScore = numPlayerTotalMen \* MAN\_VALUE;
47. int diffInNumPieces = numPlayerTotalPieces - numEnemyTotalPieces;
48. int diffInNumKings = numPlayerTotalKings - numEnemyTotalKings;
49. int diffInNumMen = numPlayerTotalMen - numEnemyTotalMen;
50. // PIECE BONUS/PENALTY - UNCOMMENT TO ACTIVATE
51. // if diff in kings is positive, score is amplified with a bonus
52. // if diff in kings is negative, however, score is penalized accordingly (by adding a negative)
53. // numKingsScore += (40 \* diffInNumKings);
54. // if diff in men is positive, score is amplified with a bonus
55. // if diff in men is negative, however, score is penalized accordingly (by adding a negative)
56. // numMenScore += (20 \* diffInNumMen);
57. // END PIECE BONUS/PENALTY
58. int casualtyScore = 0, captureScore = 0, positionScore = 0, playerPiece = 0, enemyPiece = 0, advancementScore = 0;
59. std::vector<Board::Move> playerMoves = state.moveGen(color);
60. std::vector<Board::Move> enemyMoves = state.moveGen(switchPlayerColor(color));
61. Pieces playerPieces = state.getPlayerPieces(color);
62. Pieces opponentPieces = state.getOpponentPieces(color);
63. Pieces \*p\_playerPieces = &playerPieces;
64. Pieces \*p\_opponentPieces = &opponentPieces;
65. std::vector<Board::Move> playerJumpsForPiece;
66. std::vector<Board::Move> opponentJumpsForPiece;
67. // helper values to quickly check if a piece is in a certain notable location
68. // back rows for each color player to determine KING-ing
69. // sides indicate limited moves
70. const long long redBackRowGrp = (1LL << 1) | (1LL << 2) | (1LL << 3) | (1LL << 4);
71. const long long blackBackRowGrp = (1LL << 29) | (1LL << 30) | (1LL << 31) | (1LL << 32);
72. const long long sideColumnGrp = (1LL << 5) | (1LL << 13) | (1LL << 21) | (1LL << 12) | (1LL << 20) | (1LL << 28);
73. // CHECK TERMINAL STATE
74. if (enemyMoves.size() == 0)
75. return 7999999; // good for us if enemy has no moves left!
76. else if (playerMoves.size() == 0)
77. return -7999999; // bad for us if we're out of moves!
78. // MAIN LOOP FOR SCORING POSITION
79. for (int piece = 0; piece < 32; piece++)
80. {
81. // BEGIN POSITION SCORE SECTION
82. if (color == Color::RED)
83. {
84. playerPiece = state.getPieceInSquare(piece, color);
85. if (playerPiece == MAN)
86. positionScore += (squareValuesForRed[piece] \* MAN\_VALUE);
87. else if (playerPiece == KING)
88. positionScore += (squareValuesForKing[piece] \* KING\_VALUE);
89. }
90. else if (color == Color::BLACK)
91. {
92. playerPiece = state.getPieceInSquare(piece, color);
93. if (playerPiece == MAN)
94. positionScore += (squareValuesForBlack[piece] \* MAN\_VALUE);
95. else if (playerPiece == KING)
96. positionScore += (squareValuesForKing[piece] \* KING\_VALUE);
97. }
98. // END POSITION SCORE SECTION
99. }
100. // Check our moves; 1000 points for a safe capture, 2000 points for a multi-jump
101. for (int i = 0; i < playerMoves.size(); i++)
102. {
103. if (playerMoves.at(i).removalSquare.size() > 1)
104. {
105. if (Pieces::ouputDebugData)
106. std::cout << " INSIDE EVAL-3: We " << colorTxt << " can capture multiple pieces on this state!  "
107. << "Start: " << playerMoves.at(i).startSquare << "End: " << playerMoves.at(i).destinationSquare.back()
108. << " " << std::endl;
109. captureScore += 2000;
110. }
111. else if (playerMoves.at(i).removalSquare.size() == 1)
112. {
113. int enemyCaptureSqr = playerMoves.at(i).removalSquare.back();
114. int enemyCaptureType = state.getPieceInSquare(enemyCaptureSqr, switchPlayerColor(color)); // it's an enemy piece, what is it's type?
115. if (enemyCaptureType == KING)
116. {
117. if (Pieces::ouputDebugData)
118. std::cout << " INSIDE EVAL-3: We can capture enemy KING! " << colorTxt << std::endl;
119. captureScore += 400;
120. }
121. else if (enemyCaptureType == MAN)
122. captureScore += 200;
123. int destSqr = playerMoves.at(i).destinationSquare.back();
124. std::vector<int> adjMoves = state.boardMoveTable[destSqr].moves;
125. if (color == Color::RED)
126. {
127. for (int j = 0; j < adjMoves.size(); j++)
128. {
129. if (adjMoves.at(j) > destSqr) // check enemy MEN and KING below
130. {
131. int enemyPiece = state.getPieceInSquare(adjMoves.at(j), switchPlayerColor(color));
132. if (enemyPiece == MAN || enemyPiece == KING)
133. captureScore -= 100; // not safe
134. }
135. else if (adjMoves.at(j) < destSqr) // we're red, anyting above us can only capture if enemy KING
136. {
137. int enemyPiece = state.getPieceInSquare(adjMoves.at(j), switchPlayerColor(color));
138. if (enemyPiece == KING)
139. captureScore -= 100; // not safe
140. }
141. else
142. captureScore += 1000; // we're safe to capture
143. }
144. }
145. else
146. {
147. for (int j = 0; j < adjMoves.size(); j++)
148. {
149. if (adjMoves.at(j) < destSqr) // check enemy MEN and KING above
150. {
151. int enemyPiece = state.getPieceInSquare(adjMoves.at(j), switchPlayerColor(color));
152. if (enemyPiece == MAN || enemyPiece == KING)
153. captureScore -= 100; // not safe
154. }
155. else if (adjMoves.at(j) > destSqr) // we're black, anyting below us can only capture if enemy KING
156. {
157. int enemyPiece = state.getPieceInSquare(adjMoves.at(j), switchPlayerColor(color));
158. if (enemyPiece == KING)
159. captureScore -= 100; // not safe
160. }
161. captureScore += 1000; // we're safe to capture
162. }
163. }
164. }
165. }
166. // BEGIN CASUALTY SECTION
167. for (int j = 0; j < enemyMoves.size(); j++)
168. {
169. //std::cout << "we are here" << std::endl;
170. if (enemyMoves.at(j).removalSquare.size() > 1)
171. {
172. if (Pieces::ouputDebugData)
173. std::cout << " INSIDE EVAL-3: Enemy can capture multiple pieces, avoid!" << colorTxt << std::endl;
174. casualtyScore -= 40000; // we lose too much, really bad
175. }
176. else if (enemyMoves.at(j).removalSquare.size() == 1)
177. {
178. // friendly piece is captured, what is it's type?
179. int capturedPieceType = state.getPieceInSquare(enemyMoves.at(j).removalSquare.at(0), color);
180. if (capturedPieceType == KING)
181. {
182. if (Pieces::ouputDebugData)
183. std::cout << " INSIDE EVAL-3: Enemy can capture a KING, avoid!" << colorTxt << std::endl;
184. casualtyScore -= 4000; // we lose a KING, a valuable piece
185. }
186. else if (capturedPieceType == MAN) // we lose one MAN
187. {
188. int opponentDestinationSqr = enemyMoves.at(j).destinationSquare.at(0);
189. std::vector<int> adjMoves = state.boardMoveTable[opponentDestinationSqr].moves;
190. if (color == Color::RED)
191. {
192. // if we are RED, opponent is Black; if BLACK enemy lands on our back row, avoid at all cost
193. // we are therefore trying to minimize the chance of an enemy getting a KING
194. if ((1 << opponentDestinationSqr) & redBackRowGrp)
195. casualtyScore -= 5000;
196. // opponent lands on their own back row; not so bad but we can't retaliate so avoid
197. else if ((1 << opponentDestinationSqr) & blackBackRowGrp)
198. casualtyScore -= 2000;
199. // opponent lands on one of the side squares, where we cannot retaliate. Avoid as well
200. else if ((1 << opponentDestinationSqr) & sideColumnGrp)
201. casualtyScore -= 2000;
202. else
203. {
204. // if (diffInNumMen >= 1)
205. // {
206. for (int j = 0; j < adjMoves.size(); j++)
207. {
208. int ourPiece = state.getPieceInSquare(adjMoves.at(j), switchPlayerColor(color));
209. // check if we have a king below; we're red, only our king can go upwards
210. if (adjMoves.at(j) > opponentDestinationSqr)
211. {
212. if (ourPiece == KING)
213. captureScore += 1000; // can retaliate
214. }
215. // we're red, we can retaliate with MAN or KING if enemy is above
216. else if (adjMoves.at(j) < opponentDestinationSqr)
217. {
218. if (ourPiece == MAN || ourPiece == KING)
219. captureScore += 1000; // can retaliate
220. }
221. else
222. captureScore += 0; // we cannot capture
223. }
224. // }
225. }
226. }
227. else
228. {
229. // if we are BLACK, opponent is Red; if RED enemy lands on our back row, avoid at all cost
230. // we are therefore trying to minimize the chance of an enemy getting a KING
231. if ((1 << opponentDestinationSqr) & blackBackRowGrp)
232. casualtyScore -= 5000;
233. // opponent lands on their own back row; not so bad but we can't retaliate so avoid
234. else if ((1 << opponentDestinationSqr) & redBackRowGrp)
235. casualtyScore -= 2000;
236. // opponent lands on one of the side squares, where we cannot retaliate. Avoid as well
237. else if ((1 << opponentDestinationSqr) & sideColumnGrp)
238. casualtyScore -= 2000;
239. // if we've gotten this far, we lose one MAN and opponent lands somewhere we can retaliate
240. // We ask - Can we? If yes, do it if we have piece parity or an advantage of more pieces
241. else
242. {
243. // if (numPlayerTotalPieces >= numEnemyTotalPieces)
244. // {
245. for (int j = 0; j < adjMoves.size(); j++)
246. {
247. int ourPiece = state.getPieceInSquare(adjMoves.at(j), switchPlayerColor(color));
248. if (adjMoves.at(j) < opponentDestinationSqr) // check our KING above
249. {
250. if (ourPiece == KING)
251. captureScore += 1000; // can retaliate
252. }
253. // we're black, if enemy is below we can retaliate with MAN or KING
254. else if (adjMoves.at(j) > opponentDestinationSqr)
255. {
256. if (ourPiece == MAN || ourPiece == KING)
257. captureScore += 1000; // can retaliate
258. }
259. captureScore += 0; // we cannot retaliate
260. }
261. // }
262. }
263. }
264. }
265. }
266. }
267. //END CASUALTY SECTION
268. int compositeScore = numMenScore + numKingsScore + advancementScore + positionScore + captureScore + casualtyScore;
269. return compositeScore;
270. }

### 3.7.3 Evaluation Function One (EF-1)

Please refer to Appendix A, Algorithm.cpp for a complete implementation of this evaluation function, which was implemented by David Torrente. While this evaluation function will be covered in more detail in the respective authors report, I want to take a moment to provide a brief description of its goals using the final state.

This EF makes a more careful examination of the opponent’s score than others. It tries to assign different weights to pieces and scores depending on the state of the game and the disparity between pieces, to adjust the playstyle. This is an interesting approach and can lead to unique behavior – for example it values kings more highly in the early stages and tries to be more aggressive. Depending on whether the player has a lead in number of pieces, the game is either more aggressive or more defensive (starting around line 195 and onward).

### 3.7.4 Evaluation Function Two (EF-2)

Please refer to Appendix A, Algorithm.cpp for a complete implementation of this evaluation function, which was implemented by Randall Henderson. While this evaluation function will be covered in more detail in the respective authors report, I want to take a moment to provide a brief description of its goals using the final state.

EF-2 appears to value Kings quite highly (3000 for King vs 560 for a piece on our side, 1000 vs 100 respectively for the opponent). It also makes use of an array containing values for each square on the board. It encourages kings to move to the center while regular pieces head for the opponent’s back row.

## 3.8 UI Design Considerations

Games can be thought provoking, challenging, used as a tool for learning, or a form of competition. But fundamentally – games are fun!

We wanted to provide a pleasant user experience with this program, where we approximate a real-life checkers game as much as possible. The application provides several additional options to the user where they may either play against another human or test themselves against one of the evaluation functions.

While we are limited in our options on a CLI terminal, we do have some avenues to make the user interface and game experience fun. The Texas State Linux hosts support ANSII colors; special colors and color combinations are used throughout the program to both guide the user and brighten the experience that is normally just black and white text.

We display a colorful welcome message with the authors’ names highlighted for visibility.

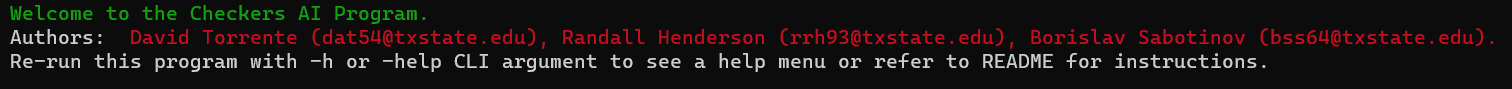


Figure : Welcome message

Whenever the user is prompted for a choice, the choices are highlighted in a pleasant blue color that is also readable. This guides the user by drawing their eye to their available options.



Figure : Highlighting choices to guide the user

The checkers board itself is visually represented, with all pieces in their color, in full. This provides a good user experience as it allows the player to immediately and intuitively understand the state of the game. They will know by looking at the printed board where their pieces are in relation to the opponent in the same way as they would looking at a real checkers board (or a fancy HTML web interface). The squares are also numbered, providing players with instant access to the location of their pieces on the board. Initially we discussed having two boards – one with pieces on it and another as a lookup table with square numbers. I advocated for the expanded board with layered, contextual information.

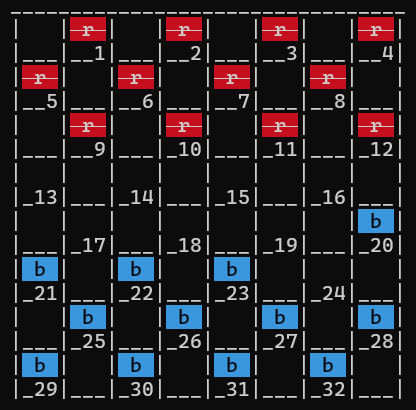


Figure : A checkers board as represented on the CLI, with color enabled

End game states are also considered – reporting the outcome of the game in a fun manner can improve the experience. Should the game end in a draw, as most games in checkers and in our program do, we display a fun graphic using ASCII characters. The first line in figure 10 is simply text, stating the outcome. The second line may require some imagination, but it depicts two boxers with their fists up squaring against one another. The third line is a reference to Call of Duty (a famous video game) and its well-known “Mission Failed” soundbite.

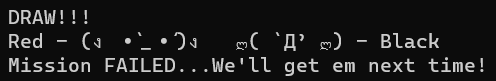


Figure : Draw end state ASCII art

Now suppose one of the players wins the game – that is a pretty exciting outcome, and we can have some fun here. The first line in figure 11 again states the outcome using text. The second line shows the winning player (in this case Red) sweating and posing with their muscles flexed. The third line taunts the losing player. And the fourth and last line shows the losing player’s supposed reaction – flipping a table!

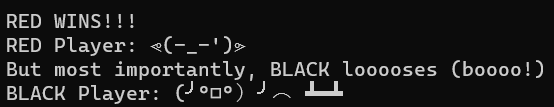


Figure : One player wins end state ASCII art

At the end of the game, a simple but pleasant graphic saying “Goodbye!” to the user is displayed.

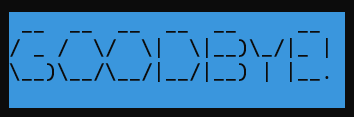


Figure : End screen

# 4 Classes Deep-Dive

Refer to Appendix A for complete source code, which contains helpful comments. I will refer to file name and line number below.

We may say that class Simulation has one or more Games. A Game has two Players and one Board. A Board has two Pieces. A Player has an Algorithm and one Pieces instance (to represent the player’s pieces for the game).

## 4.1 Pieces

This class represents a set of 12 initial pieces in their starting positions. It contains an enum class Color, which helps us avoid “magic numbers” – for example 1 is Red and -1 is Black but using an enum we can refer to them by name. We decided to use an enum class instead of an enum due to stronger type checking. Suppose we had two enums, one Color and another Fruit; if red were equal to 1 and apple also to 1, they would be equal. But with an enum class we avoid this potential issue as they type would be different.

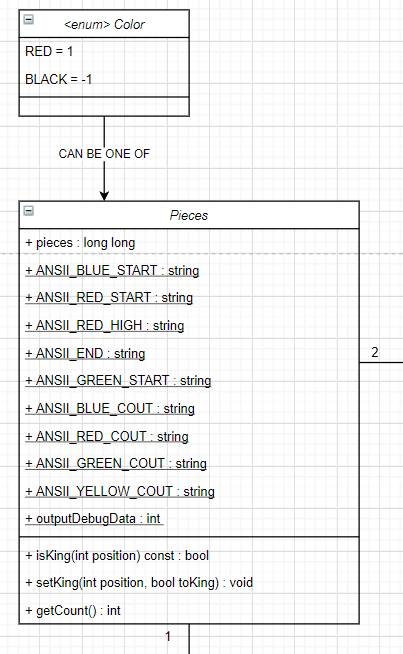


Figure : Pieces UML Class Node

### 4.1.1 Data Structures Used

The pieces are represented using a long long type, which is 64 bits. The lower 32 bits are used to determine the location of the piece. The upper 32 bits tell us if a piece is a king (1) or not (0). This is accessed by using an offset of 31 plus the number of the piece.

**MSB** 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 **LSB**

King bits Piece position bits

Starting position for Red player, who is on top of the board, is 4095. All the lower 12 bits are set and they are not a King yet, so upper 32 bits are empty.   
We can intuitively see this is correct, two to the power of 12 is 4096 if the 12th bit (using 0 indexing) is set. Since all bits below (0 to 11) are set, we simply do 4096 – 1 = 4095, which is what we expect. Black is a much higher starting value (4,293,918,720) as it occupies the upper 12 bits of the lower 32 bits available.

Red Player starting value: **MSB** 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 1111 1111 1111 **LSB**

Black Player starting value: **MSB** 0000 0000 0000 0000 0000 0000 0000 0000 1111 1111 1111 0000 0000 0000 0000 0000 **LSB**

The Pieces class contains a few simple helper functions and some static constant strings to assist us with displaying color (if desirable) throughout the application. This is because the Pieces header is the most fundamental and at the top of the hierarchy. All other classes either directly import it or end up having access to it indirectly by importing a class which uses it.

Of import is the isKing() function which allows us to tell if a piece is of type King. We first must determine if the position is occupied via a separate call; if it is, we can call this helper function. We use the position of the piece, let us say 4, plus an offset of 31 for a total of 35. We shift right 35 bits and we bitwise-AND with 1 – if the result is 1, the piece is a King. Otherwise, it is a standard piece.

## 4.2 Board

Class board is used to represent the entire board and its current state. It is based on an 8 x 8 grid, with 32 possible spaces. Two piece data types are the primary memory consumers of this class. This class also includes a static member that acts as a move guide. This move guide determines the possible moves for a square on the board, not for a piece.

Board contains two helper structs – Move and BoardMoveTable. Move is used to track the moves we make on the board. It contains a starting square, a vector of squares to move across (if jumping), where the last square is the destination of the piece we are moving. It also contains a vector listing the squares of the enemy pieces we captured, if any. BoardMoveTable, on the other hand, contains static data types to significantly speed up how we search for moves given a position.

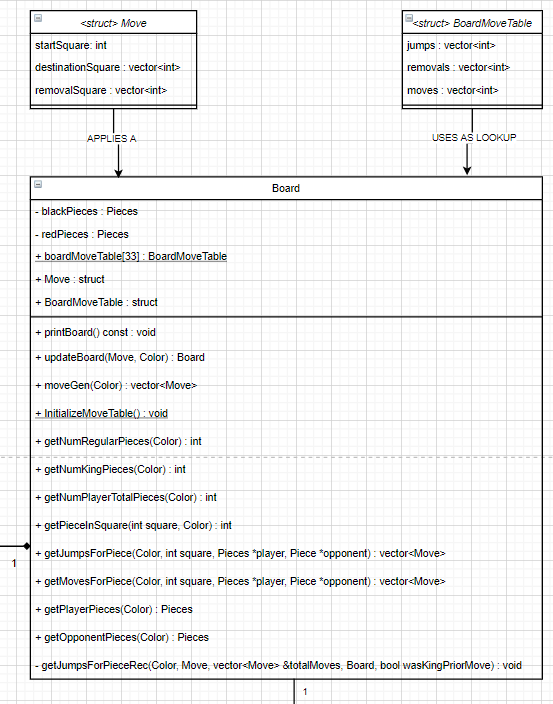


Figure : Board UML Class Node

### 4.2.1 Data Structures Used

The Board class makes clever use of a static move table – boardMoveTable. It provides us with O(1) lookup time for the possible moves and jumps for each square.

Here is an example:

boardMoveTable[16].jumps.push\_back(7);

boardMoveTable[16].removals.push\_back(11);

boardMoveTable[16].jumps.push\_back(23);

boardMoveTable[16].removals.push\_back(19);

boardMoveTable[16].moves.push\_back(11);

boardMoveTable[16].moves.push\_back(12);

boardMoveTable[16].moves.push\_back(19);

boardMoveTable[16].moves.push\_back(20);

We see that the lookup table contains the following information accessible for square 16:

* It can move to squares 11, 12, 19, 20 (depending on the color and type of the piece)
* It can jump to 7, which would remove the piece in square 11
* It can jump to 23, removing piece in square 19

If we closely examine the figure below, we see this is true because 16 is the 2nd to last row from the right, so it cannot jump to the right. It may only jump across 11 to or across 19 to 23. In this example, we would determine the type of piece that is on the board at 16 (RED, MAN), which would allow us to use the look up table properly. Because the piece is red and a standard piece, it may only move downward diagonally to either 19 or 20 below.

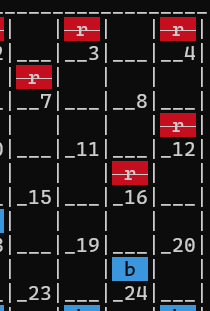


Figure : Moves and Jumps for Square 16

A Board is represented as an 8x8 grid though we only ever use 32 spots. When we print the board, we make the individual squares large enough to contain a colored piece and the square number, the latter of which does not change. A move is represented by struct Move, which has a starting square, a vector of squares to move to in sequence, and a vector of captured/removed enemy pieces. If we examine the figure below, we see Black moved from 25 to 18 and captured 1 piece (on 25). Thus move.startSquare = 25, move.destinationSquare.at(0) = 18, and move.removalSquare.at(0) = 22. These values would be determined by calling getMovesForPiece() or getJumpsForPiece(), which would in turn leverage the O(1) lookup table to determine where a piece may move to or jump. Jumps are mandatory so we must take them if offered, therefore moves for a piece are not generated if a jump is available.

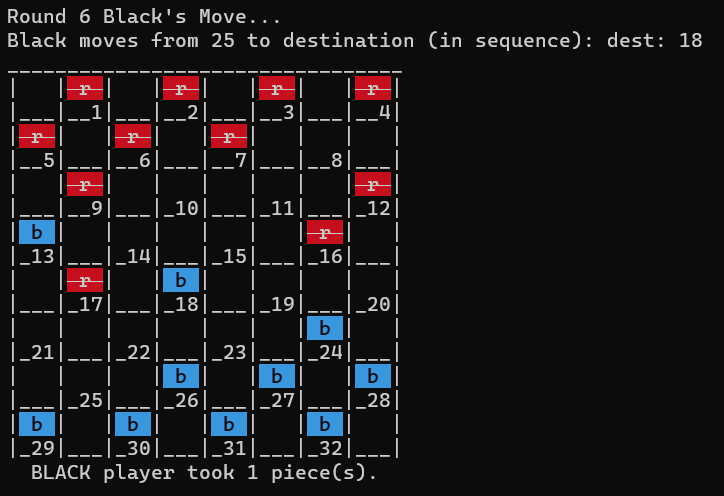


Figure : A Board with a Black move from 25 to 18 (a jump) and a capture at 22

Of particular note is the getJumpsForPieceRec(), a recursive function which gets a full jump chain (in case more than one is available in succession). The function keeps track if we are starting a jump or in the middle of it by looking at the size of the destination vector, which contains all the squares on which the player will land. We determine if a piece is of type King to be able to tell which moves are available. Then in the main for-loop we calculate if a jump is available. The loop appears to be O(n) but is actually O(1) complexity because we only loop up to boardMoveTable[piece].jumps.size(). Recall from previous discussions above, Board uses a lengthy look-up table to store all possible moves and jumps for each square individually. Thus, this size is known up-front and will never grow, giving us a constant rather than variable look-up time!

The moveGen function is important because it is how we get a list of available moves for a player. We pass in the color of the player and use it to check Pieces, where the player’s pieces are stored as a long long bitmap. We go through each piece the player has on the board and use the O(1) lookup table to determine where the pieces may move. It is important to note that if one or more jumps are available, only the jumps will be returned and not any other moves.

Here is the main for-loops, which drive this process. Both only loop up to and including 32. A bit offset based on the piece counter is used, so we can check each position in turn by shifting the bitmap and using bitwise-AND to compare if the square is occupied.

1. // Go through all 32 squares and see if it is one
2. // of the appropriate pieces belonging to player.
3. for (int pieceIter = 1; pieceIter <= 32; pieceIter++)
4. {
5. // The offset is used to align to 0 - 31, but
6. // the board in checkers is 1 - 32. Use an offset
7. // here to align to the position bits properly.
8. bitOffset = pieceIter - 1;
9. if ((((playerPieces->pieces) >> bitOffset) & 1) == 1)
10. {
11. // Check for possible jump mpves first.
12. returnedMoves = getJumpsForPiece(color, pieceIter, playerPieces, opponentPieces);
13. totalMoves.insert(totalMoves.end(), returnedMoves.begin(), returnedMoves.end());
14. }
15. }
16. // Go through all 32 squares and see if it is one
17. // of the appropriate pieces belonging to player.
18. // Do this only if no jumps are possible. This is controlled
19. // by the if condition here. No need to get moves if there are
20. // jumps already found.
21. if (totalMoves.size() == 0)
22. {
23. for (int pieceIter = 1; pieceIter <= 32; pieceIter++)
24. {
25. // The offset is used to align to 0 - 31, but
26. // the board in checkers is 1 - 32. Use an offset
27. // here to align to the position bits properly.
28. bitOffset = pieceIter - 1;
29. // Check if player is in this space
30. if ((((playerPieces->pieces) >> bitOffset) & 1) == 1)
31. {
32. // Check for non jump moves here.
33. returnedMoves = getMovesForPiece(color, pieceIter, playerPieces, opponentPieces);
34. totalMoves.insert(totalMoves.end(), returnedMoves.begin(), returnedMoves.end());
35. }
36. }
37. }

## 4.3 Player

The Player class defines a virtual player, in the case of this program an AI; no human player is used – that is to say, no manual input is sought from the user at any time when this class is used. The “player” will use an algorithm in combination with an evaluation function to determine the best move it wants to make. The player class has several straightforward helper functions to get the number of pieces a player has, or has taken, to get the player’s color, and so on. Of particular note is the overloaded constructor, which allows us to set what algorithm, evaluation function, and depth the player will leverage for the game. We also have four variables to keep track of the number of expanded and leaf nodes, two for each algorithm.

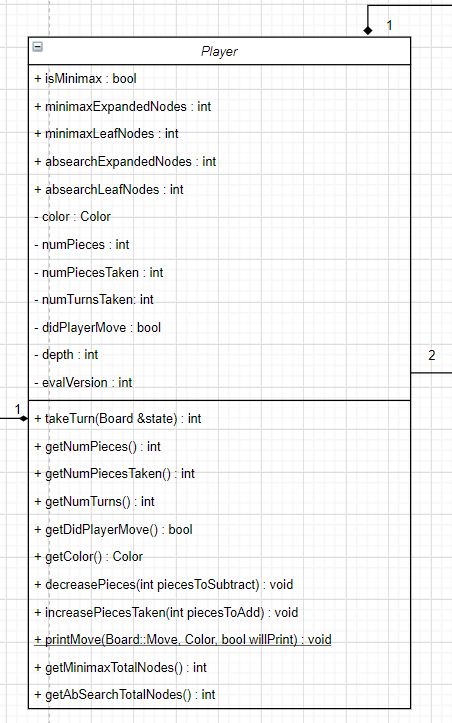


Figure : Player UML Class Node

### 4.3.1 Data Structures and Algorithms

The takeTurn(Board &state) method is where the interesting logic takes place. Here we instantiate an Algorithm with a depth, an evaluation function version, and a reference to the calling player. We check if isMinimax is set – if it is, the player will use minimax-a-b, otherwise it will use alpha-beta-search. We call the appropriate algorithm with it’s required initial values and get back a Result structure, which contains a move the player will make along with a value. If the player is not out of moves, we update the game state with the player’s move returned to us by one of the algorithms. Note that class Game calls player.takeTurn(Board &state) and the state is updated directly as we passed it in by reference and not by value. When a player makes their move, we increment the turn counter and print the new board, which contains the players move.

## 4.4 Game

The Game class represents a checkers Game. A game consists of a board, two players (red and black), and 12 checkers pieces for each player. A game ends when either player loses all their pieces or is blocked and has no further moves available. The startGame() function will trigger the process and no input from the user is required. Red and Black players will be created. Each player will be given 12 checker pieces. The pieces will be appropriately placed on the board. Each player will execute their strategy using the Algorithm class. We use a GameOver enum class to avoid magic numbers and define what constitutes a “game over” condition. In our case, either one of the players may win (BLACK\_WINS or RED\_WINS), the game may end in a DRAW, or the game is NOT\_DONE yet.

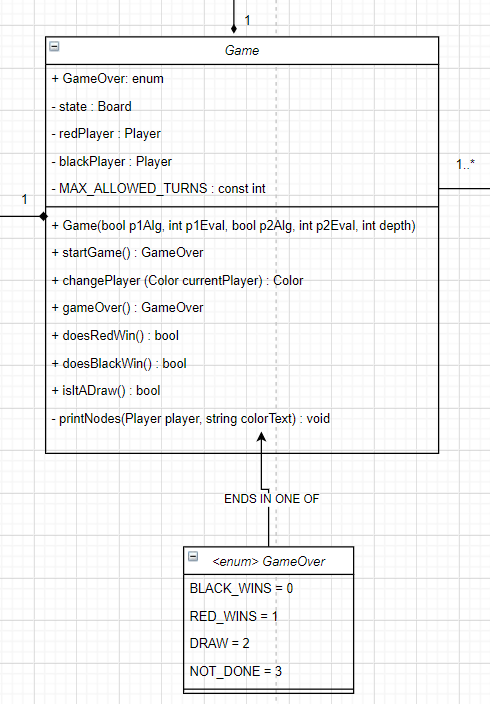


Figure : Game UML Class Node

### 4.4.1 Data Structures and Algorithms

Game’s overloaded constructor allows us to pass in the player’s respective algorithm, evaluation function, and the overall depth from main.cpp (where we obtain the user’s input) to Simulation. From Simulation, which may start a series of games, to Game. From Game to Player. And from Player to Algorithm (which only needs the evaluation version and depth).

There are no special data structures in this class. It has a few straightforward helper functions to either change the color of the current player, check win conditions, and print some statistics about the game after it concludes.

Of interest is the startGame() function – it returns a GameOver struct. It has one main while(true) loop. We only break out of this loop when a game reaches a satisfactory conclusion – either one player wins or we exceed the number of permitted turns (80 per player for a total of 160 per game), in which case we get a draw. In this function, Black moves first. After the Black player takes a turn using the Player::takeTurn(Board &state) function discussed in section 4.3, we increase the player’s captured pieces counter and decrease the opponent’s piece counter if any jumps were present. We then unconditionally check all win conditions. This is because even though only Black moved, they may have boxed themselves in and ran out of moves leading to a game over event or it may be their last permitted turn. Next the Red player takes a turn and again we check win and loss conditions for all players.

## 4.5 Algorithm

This class encapsulates the algorithmic approach the AI uses to play Checkers. There are only two major algorithms supported.

1. Minimax-a-b: a depth first, depth limited search procedure from the Richard and Knight book. The minimax function has a heuristic value for leaf nodes (end nodes and nodes at the maximum permitted depth). Intermediate nodes get their value from a child/successor leaf node. It is a recursive function and instead of alpha and beta we use passThreshold and useThreshold values, which are swapped and inverted at each recursive call.
2. Alpha-Beta-Search: uses indirect recursion and does not require us to swap or negate values. There are two helper functions, one MAX and one MIN for each respective player. We stop evaluating a branch when at least one option is found to be worse than a previously examined move.

Note that both algorithms return the same result given the same evaluation function. This means that while the internal implementation of the two algorithms is quite different, they behave in the same way and will play a game the same turn by turn, leading to the same outcome and the same number of expanded nodes. This is a good thing and exactly what we expect to see; it indicates that the two algorithms are correctly implemented within the context of this program.

Class Algorithm uses a Result structure, which consists of an integer value (the score we give a particular state) and a Board::Move bestMove struct (the move the player needs to make to head in that direction in the game tree).

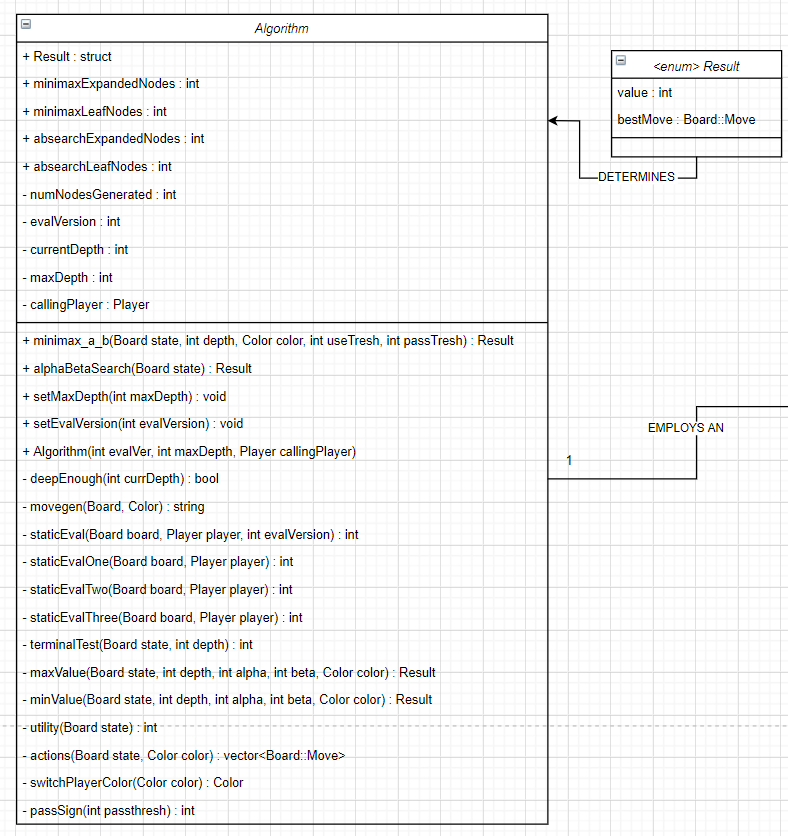


Figure : Algorithm UML Class Node

### 4.5.1 Data Structures and Algorithms

The evaluation functions are covered in more detail in section 3.7 above.

In minimax-a-b, we increment the number of leaf nodes if we are deep enough or at a terminal state. The number of intermediate expanded nodes is captured before we make the recursive call. In the for-loop, we must first create a new Board (i.e., state) by applying the move that is at successors.at(successorsIndex). This is the move we pass into the recursive call of minimax-a-b. We decrement the depth, switch the player color, and both swap and invert passThresh and useThresh. Once this call returns, we invert the value that was obtained and store it in newValue. We first check if this is strictly greater (better) than passThresh, if it is, we found a better move. Next, we check if passThresh is greater or equal to useThresh and if it is, we already have the best move on the branch and we need not explore the branch deeper, so we terminate the for loop and return with the value and move. To deal with odd depths and keep values in the right order, we check if the original calling player’s color is the same as the color passed into the function. If it is, there is no issue. Otherwise, we must negate the result.value we obtain. The algorithm remains correct for odd depths; but failure to guard against even depth inversion leads to undesirable behavior at even depths greater than four. In minimax-a-b, we use 8 million and negative 8 million for the best and worst cases instead of INT\_MAX and INT\_MIN maximum values. This is because inside the algorithm, we use negation. Because 2’s complement is used, negating INT\_MIN leads to undefined behavior. We use the lower reasonable values to get around this.

Alpha-Beta-Search does not have this risk and we can use INT\_MIN and INT\_MAX, as there is no negation. I use std::numeric\_limits<int>::min() and std::numeric\_limits<int>::max() respectively. The minValue() and maxValue() functions are effectively the same, just inverses of each other. We check if we are deep enough. Otherwise we generate a list of actions and check if we are at a terminal state. We set result.value to either min (if inside maxValue, as that is the worst case) or max (if inside minValue, as that is now the worst case). Inside the for-loop we create a new hypothetical board by applying the move from our list of available moves (or actions) by using listOfActions.at(actionIndex). We do not negate or swap alpha and beta, they are passed in by value as-is. Alpha is associated with MAX nodes and cannot decrease; it starts out as the worst-case for MAX. Beta tracks MIN nodes and cannot increase; it starts out as the worst case for MIN.

This was the algorithm I implemented for this project; below are snapshots from both referenced textbooks. It is worth mentioning that the effectiveness of pruning depends on the order in which we examine available moves. If our luck is bad, the first move we examine is the worst available and each subsequent move is slightly better, where the best move is at the end. In this case, we are essentially performing a depth firs search with no pruning as we keep updating the best move and going through the tree. And if the nodes are perfectly ordered, “the number of terminal nodes considered by a search to depth d using … pruning is … equal to twice the number of terminal nodes generated by a search to depth d/2 without alpha-beta.” [Knuth and Moore, 1975]

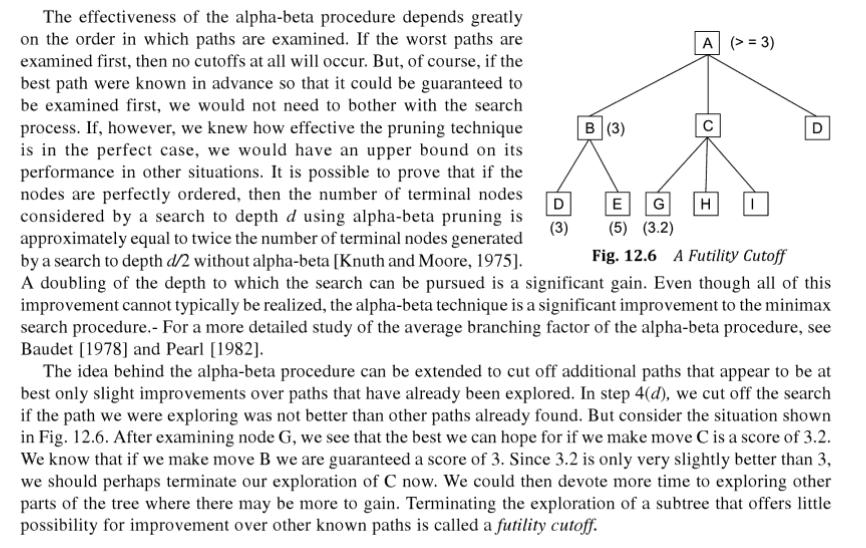


Figure : Minimax-a-b Futility Cutoff

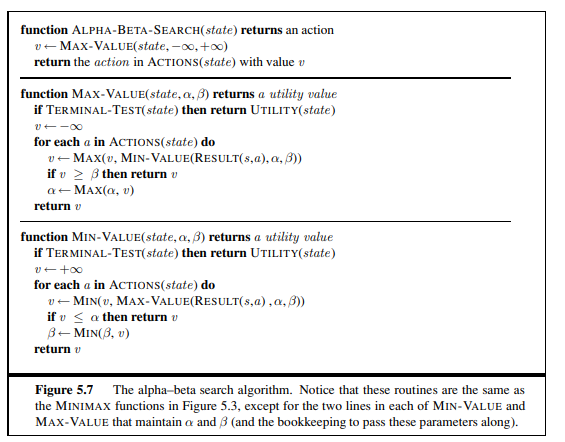


Figure : Alpha-Beta-Search Algorithm

Both minimax-a-b and alpha-beta-search correctly prune a branch that is not worth exploring as described in the textbook.

## 4.6 Simulation

The simulation class hands information to Game as needed and instantiates (and deletes when done) Games as needed. It gets back the GameOver struct from class Game once a game is complete, then prints the results along with some game statistics to the screen.

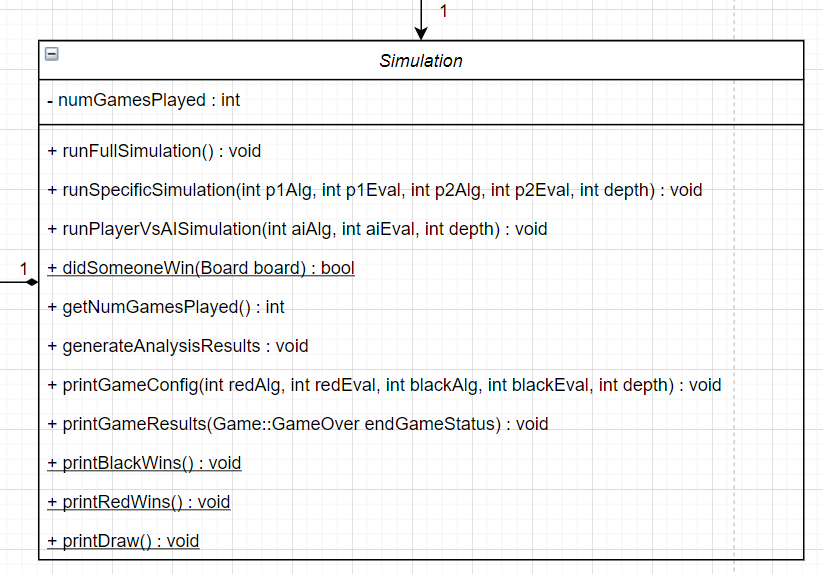


Figure : Simulation UML Class Node

### 4.6.1 Data Structures and Algorithms

To perform a full, exhaustive simulation (with limited redundancy), five nested for loops are used. The outer drives the depth. The first player’s algorithm, the third the second. Fourth and fifth loops are for the evaluation functions of the first and second player, respectively. The playerVsAI function is interesting because we mix and match strategies – we use both a player with an algorithm for the AI and obtain user input from the console for the human player’s moves.

## 4.7 Main

This is the main driver of the application and where we obtain the user’s input for the simulated games. Helper functions are used to print prompts to the CLI and obtain user input. We mostly obtain user input and send to Simulation, which then takes care of the rest. In the case of a manual game, however, we cannot use the Game or Player classes. We interface directly with Board, get the list of available moves, prompt the user for a selection, and update the board manually. The game has no limit – it continues until one player wins or the program is terminated.

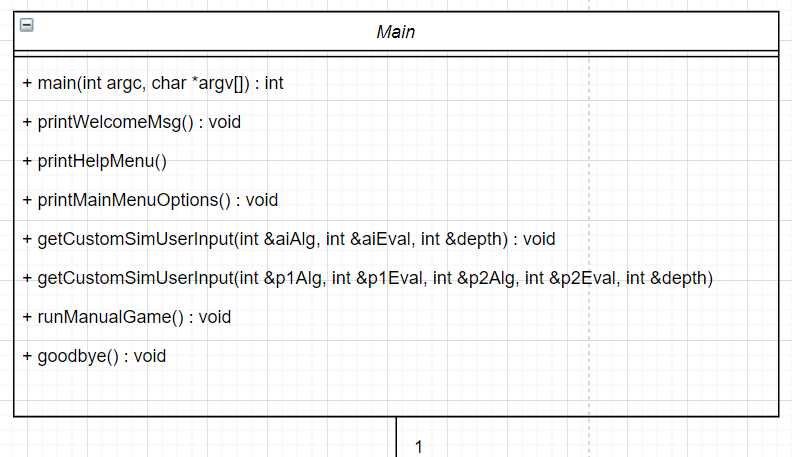


Figure : Main application driver

# 5 Sample Runs

The program prints the same content to the console at the beginning of each run. Both the welcome message and the CLI messages printed while parsing the knowledge base file do not differ between each subsequent run. A complete listing of this output may be found at the end of this report, under “**Appendix B: Complete Sample Output**.” Games 1 and 10 contain a complete game path from start to finish; game 1 is with depth two and game 10 with depth 4. The output for all other games is similar enough that these examples are enough to provide an accurate representation of the game path generated by the application. The full output for all 18 simulated games would be too long to include in the report, so they are provided separately as log files. The program always displays a welcome message to the user, parses the KB text file and prints contents to the console as it does so, and pauses. After the user hits Enter, it asks if the user wants to display the KB in human readable output. As such, the beginning section is only available in Appendix B to avoid repeating hundreds of lines.

Complete output of this program was obtained on eros.cs.txstate.edu. Program was invoked normally but standard output was piped to tee. Tee is a command which reads our standard I/O and writes it to both standard output and a text file, in our case a log file.

This section aims to comprehensively cover all options of the implemented system. The complete output of a game or a simulation of a series of games is quite large (tens of thousands of lines long). As such, full output files are provided along with the report in two directories – one for runs at depth two and another for runs at depth four.

## 5.1 Sample Run #1: ALPHA-BETA-SEARCH with EF-1 vs ALPHA-BETA-SEARCH with EF-3 at Depth 2

The screenshots capture only the relevant portion of the output for this sample run. For complete program output, please refer to Appendix B.

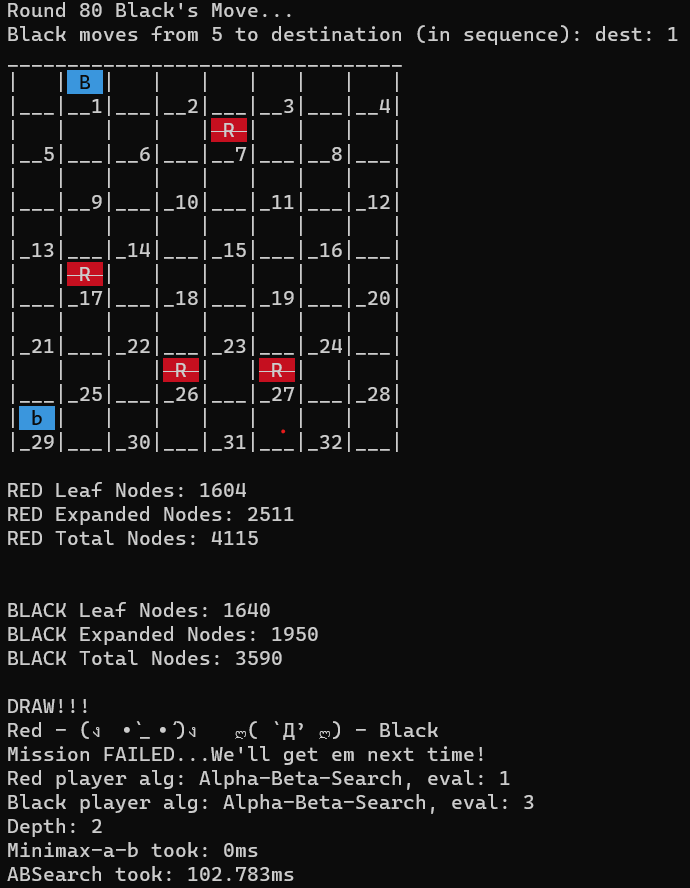


Figure : Sample Run 1

## 5.2 Sample Run #2: ALPHA-BETA-SEARCH with EF-1 vs ALPHA-BETA-SEARCH with EF-3 at Depth 4

In this sample run we run a game with ABS, EF-1 vs EF-3. End up in the back-and-forth state mentioned earlier, where black moves between 10 and 7 repeatedly and Red between 17 and 13. At around turn 8 we see Red can capture black by jumping over 19 and land on 23, which is what it does (as jumps are mandatory).

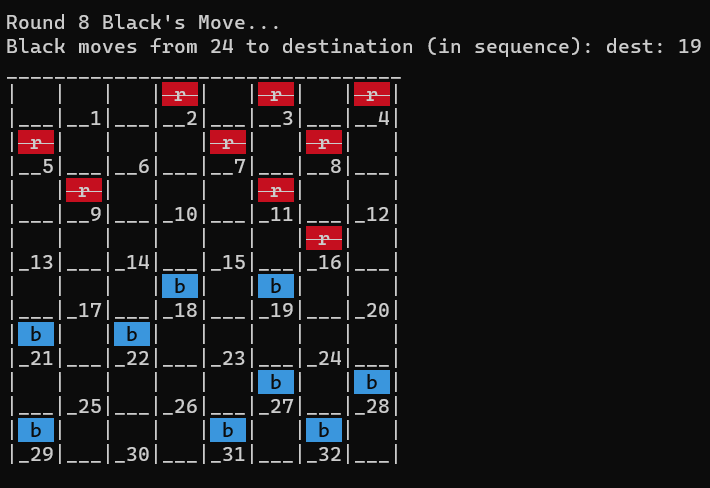


Figure : Sample Run 2, Turn 8

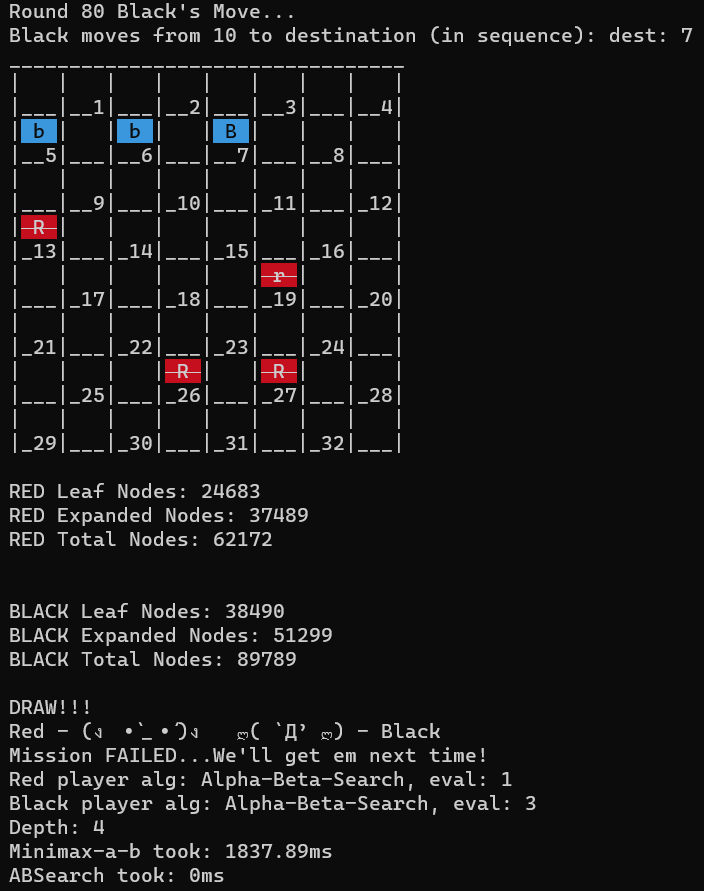


Figure :Sample Run 2 End state

## 5.3 Sample Run #3: Minimax-a-b with EF-3 vs Alpha-Beta-Search with EF-3 at Depth 6 (extra)

We now get into higher depths. At depth 6, we see a 10x explosion once again in the number of nodes generated though the game still ends in a draw after a back-and-forth stalemate for several turns. As we are using EF-3, which tends to attack Red’s “double corner” on the top left, we see a cluster of black pieces which traveled to those squares, which are valued higher; this is in line with our expectations.

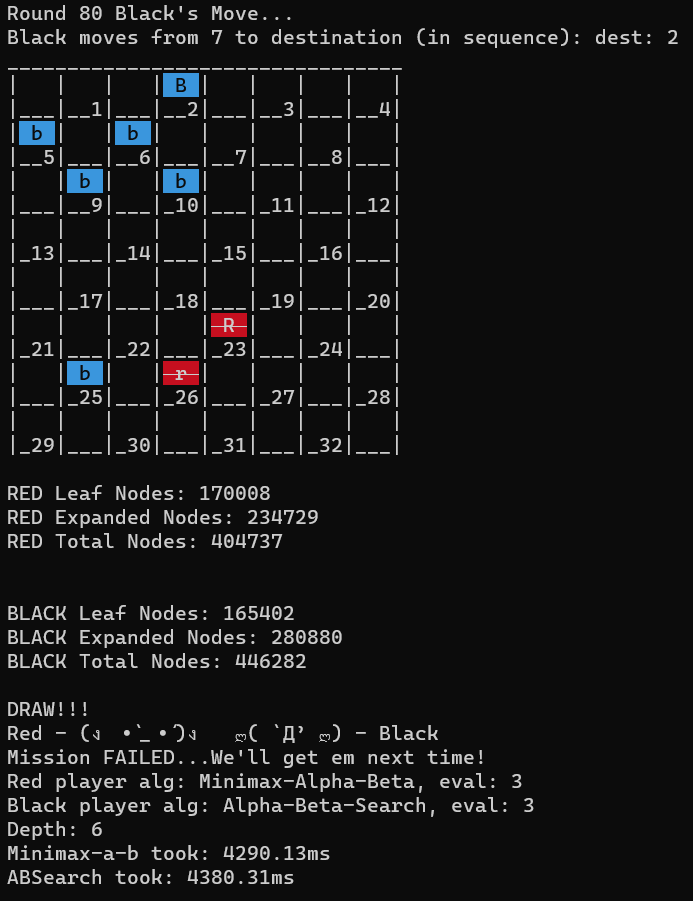


Figure : Depth 6 - MAB EF-3 vs ABS EF-3, Draw

## 5.4 Sample Run #4: ALPHA-BETA-SEARCH with EF-1 vs ALPHA-BETA-SEARCH with EF-3 at Depth 8 (extra)

In this sample run, we go to depth 8; in the beginning when the AI player is “thinking” each turn takes longer than at lower depths. The game significantly speeds up towards the end.

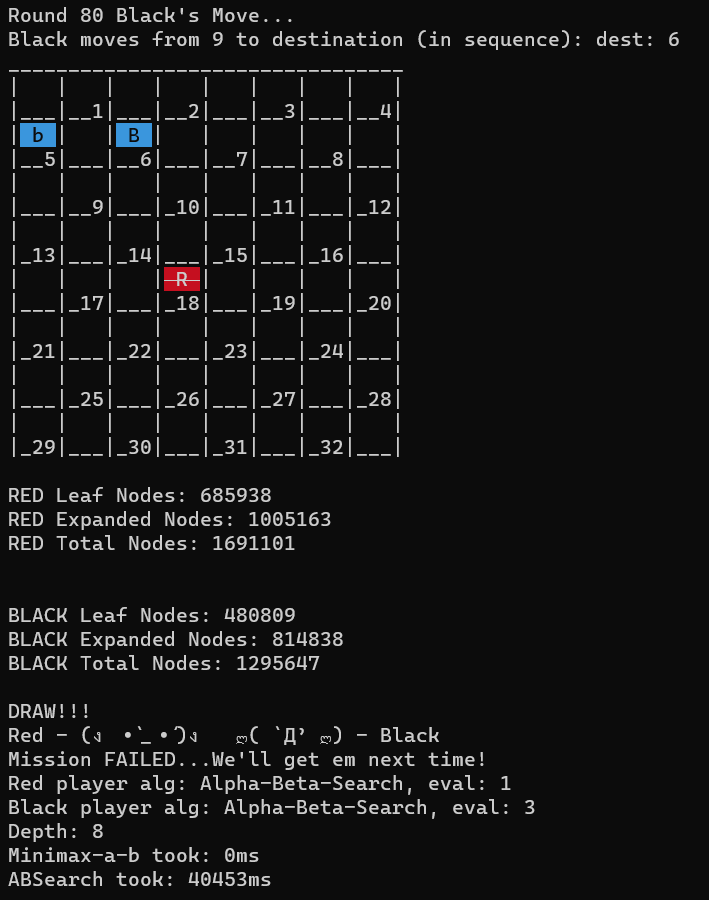


Figure : Sample Run 4 Depth 8!

## 5.5 Sample Run #5: Human Player vs AI (Minmax-A-B with EF-3)

I tested playing against the AI at depth 4 and opted to select a simple strategy of always selecting the first move presented. I ended up winning by boxing in the Red player after a few turns, which was surprisingly easy. I had at one-point optimized EF-3 to win or draw against this style of play, but it led to losses against EF 2 and opted to ignore the “take first move” strategy.”   


Figure : Sample Run 5 - Manual Game Ending

## 5.6 Sample Run #6: Printing the Help Menu

The system also offers a help menu should the user invoke the program with the following command: ./Project2 -h

This is a relatively brief help message, giving the user guidance on how to interact with the program and where to refer to accompanying documentation for more guidance.

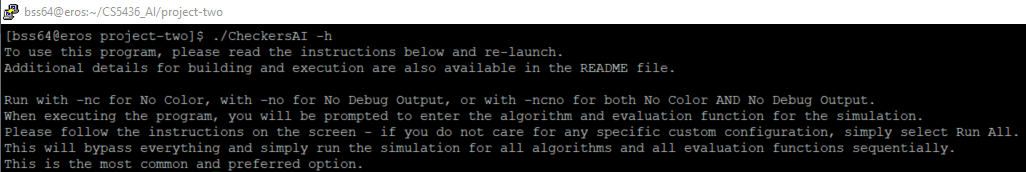


Figure : Sample Run #6 - Print the Help Menu

# 6 Conclusion

This was a highly successful project. We correctly implemented all algorithms and learned a great deal from implementing the evaluation functions, which are the heart of the AI engine. We saw that for the same evaluation function and depth, both algorithms are identical – they reach the same end-game state via the same game path while generating the same number of nodes. Performance wise, alpha-beta-search appears to fare slightly better at higher depths, while at low depths the two are comparable. At depths two and four minimax-a-b is slightly faster, though the difference is not significant. Seeing how the paths are generated and different states of the board evaluated, it now makes sense why a game like checkers may be mathematically “solved” – we can choose an optimal path in relation to the opponent’s moves. Checkers is currently considered a “solved” game in that an optimal solution exists. If the opponent makes no mistakes, the game should always end in a Draw. Our evaluation functions, when pitted either against each other or themselves, lead mostly to a Draw as well, albeit for slightly different reasons. While the evaluation functions are detailed and intelligent, towards the end-game state we encounter the horizon-effect, especially if the opponent is farther away than our depth limit. We observe a back-and-forth jitter of a few pieces, which quickly exhausts the permitted number of turns, leading to a draw. Several possibilities for resolving this were explored – including but not limited to: introducing the concept of memory for each piece, so they may behave differently when needed; remembering the last several moves of the game to detect a back-and-forth repetition and break out of it; or simply giving a random score to force the piece to move and explore elsewhere.

We observed that the program is very fast at depths two and four, tolerable at depths six and eight, and quite time consuming at higher depths due to the size of the game tree and the vast number of nodes being explored. The team strived to not only meet the project requirements but exceed them in several ways. Designing a fun and pleasant user interface, analyzing the program at depths higher than four, allowing the user to play a manual game against a friend or challenge themselves against one of the AI evaluation functions are all extra items our team incorporated to stretch ourselves and make the program fun and usable. We provide options for enabling or disabling the color and verbose output at will. By default, the application runs with verbose output enabled as it is optimized to be used as a tool for learning and for evaluation from a computer science perspective.

# References

1. Gaddis, Tony. "Starting out with C++ From Control Structures through Objects, Ninth Edition." Chapter 10 (c-strings & the string class), Chapter 17.3 the Vector Class.

2. Schaeffer, Jonathan & Björnsson, Yngvi & Kishimoto, Akihiro & Müller, Martin & Lake, Robert & Lu, Paul & Sutphen, Steve. (2007). Checkers Is Solved. Science. 317. 1518-1522. 10.1126/science.1144079.

3. C++03 Standard [2.1.1.2]. <https://gcc.gnu.org/legacy-ml/gcc/2001-07/msg01120.html>

4. Resources, PPT slides, and course materials on Canvas.

5. Knuth and Moore, 1975. An Analysis of Alpha-Beta Priming.

# Appendix A: Source Code

## Pieces.hpp

1. #ifndef PIECES\_H
2. #define PIECES\_H
3. #include <string>
4. /\*\*
5. \* The Color enumerator is used in place of values 1 and -1
6. \* in order to make the code easier to read. Note the values
7. \* of 1 and -1. This allows for alternating play by multiplying the current
8. \* by 1.
9. \*/
10. enum class Color
11. {
12. RED = 1,
13. BLACK = -1
14. };
15. /\*\*
16. \* The Pieces class represents all pieces for a particular player. Each object
17. \* of the class contains a single instance of pieces. In addition to this,
18. \* ANSII code strings were included here in order to make the output
19. \* of the code easier to read. These codes serve no functional purpose
20. \* aside from changing text color.
21. \*/
22. class Pieces
23. {
24. public:
25. Pieces();
26. Pieces(Color color);
27. // ANSII codes for colored text, to improve UI and readability
28. static std::string ANSII\_BLUE\_START;
29. static std::string ANSII\_RED\_START;
30. static std::string ANSII\_RED\_HIGH;
31. static std::string ANSII\_END;
32. static std::string ANSII\_GREEN\_START;
33. static std::string ANSII\_BLUE\_COUT;
34. static std::string ANSII\_RED\_COUT;
35. static std::string ANSII\_GREEN\_COUT;
36. static std::string ANSII\_YELLOW\_COUT;
37. // Debug reporting level 3 == display all debug/status lines 2== important, 1 == basic, 0 == none
38. static int ouputDebugData;
39. bool isKing(int position) const;
40. void setKing(int poisition, bool toKing);
41. // This contains 64 bits to represent the entire piece set and king status for one side.
42. long long pieces;
43. };
44. #endif // !PIECES\_H

## Pieces.cpp

1. #include "Pieces.hpp"
2. // ANSII codes for colored text, to improve UI and readability
3. std::string Pieces::ANSII\_BLUE\_START = "\033[0;30;46m";
4. std::string Pieces::ANSII\_RED\_START = "\033[0;31m";
5. std::string Pieces::ANSII\_RED\_HIGH = "\033[9;37;41m";
6. std::string Pieces::ANSII\_END = "\033[0m";
7. std::string Pieces::ANSII\_GREEN\_START = "\033[0;32m";
8. std::string Pieces::ANSII\_BLUE\_COUT = "\033[0;30;46m";
9. std::string Pieces::ANSII\_RED\_COUT = "\033[41;1m";
10. std::string Pieces::ANSII\_GREEN\_COUT = "\033[0;30;42m";
11. std::string Pieces::ANSII\_YELLOW\_COUT = "\033[30;48;5;3m";
12. int Pieces::ouputDebugData = 3;
13. /\*\*
14. \* Constructor | Pieces | Pieces
15. \*
16. \* Summary :    Unused. Constructor with
17. \*              a parameter is always required.
18. \*
19. \* @author : David Torrente
20. \*
21. \*/
22. Pieces::Pieces()
23. {
24. }
25. /\*\*
26. \* Constructor | Pieces | Pieces
27. \*
28. \* Summary :    Sets up the player pieces depending on
29. \*              the color passed in.
30. \*
31. \* @author : David Torrente
32. \*
33. \* @param Color color   : The color to assign the pieces to.
34. \*
35. \*/
36. Pieces::Pieces(Color color)
37. {
38. // The following are bit fields to be used for the board. Each value
39. // turns on or off a bit. For example, 4095 looks like:
40. // 1111 1111 1111, which will position a piece in squares 1 - 12
41. // top squares. The middle is blank, so 0000 0000, which covers
42. // squares 13 - 20. Lastly,
43. // black is placed in the lowest bits similar to white's pattern,
44. // but in spaces 21 - 32, hence, the large number. Neither start with
45. // kings, so bits 33 - 64 are all zeros on both sides.
46. // Note that the commented out values are primarily for debugging
47. // special move cases or interesting situations. They are retained here as
48. // pairs. If used, each pair must be uncommented.
49. if (color == Color::RED) // red
50. //pieces = 1;
51. //pieces = 19455;
52. //pieces = 1152921504875282432;
53. pieces = 4095; // Initial board state
54. else               // black
55. //pieces = 4291952640;
56. //pieces = 128;
57. //pieces = 16974848;
58. pieces = 4293918720; // Initial board state
59. }
60. /\*\*
61. \* Member Function | Pieces | isKing
62. \*
63. \* Summary :    Determines if the piece in a given position is
64. \*              a king or not. Prior to calling this, the position
65. \*              must be checked to see if the position is occupied
66. \*              by a player.
67. \*
68. \* @author : David Torrente
69. \*
70. \* @param int position  :   The position on the board to
71. check to see if it is a king.
72. \*
73. \* @return bool         :   Returns true if it is a king piece,
74. \*                          false otherwise.
75. \*
76. \*/
77. bool Pieces::isKing(int position) const
78. {
79. // NOTE: it is 31, since the offset is 0 - 31
80. // Could have written +32 -1, but better to just combine.
81. // Checks the high bit (33 - 64) which is aligned with
82. // the position bit (1 - 32).
83. int kingBit = position + 31;
84. bool isKing = false;
85. if (((pieces >> kingBit) & 1) == 1)
86. {
87. isKing = true;
88. }
89. return isKing;
90. }
91. /\*\*
92. \* Member Function | Pieces | setKing
93. \*
94. \* Summary :    Sets the king bit for a given position to either
95. \*              be on or off.
96. \*
97. \* @author : David Torrente
98. \*
99. \* @param int position  :   The position on the board to
100. \*                          toggle to a king or not king.
101. \*
102. \* @param bool toKing   :   True if the piece is to become
103. \*                          a king, false if it is to be reduced
104. \*                          to not a king. Reducing the piece type
105. \*                          is primarily done to keep the board clean.
106. \*
107. \*/
108. void Pieces::setKing(int position, bool toKing)
109. {
110. int kingBit = position + 31;
111. if (toKing == true)
112. {
113. pieces = pieces | (1ULL << kingBit);
114. }
115. else
116. {
117. pieces = pieces & ~(1ULL << kingBit);
118. }
119. }

## Board.hpp

1. #ifndef BOARD\_H
2. #define BOARD\_H
3. #include <vector>
4. #include <string>
5. #include "Pieces.hpp"
6. /\*\*
7. \* Class board is used to represent the entire board and its current state.
8. \* It is based on an 8 x 8 grid, with 32 possible spaces. Two piece data
9. \* types are the primary memory consumers of this class. This class also includes
10. \* a static member that acts as a move guide. This move guide determines the possible
11. \* moves for a square on the board, not for a piece.
12. \*/
13. class Board
14. {
15. public:
16. /\*\*
17. \* Struct Nove is used to track moves made on the board. It contains
18. \* a single starting location, a vector of steps to a move,
19. \* as well as all of the pieces to remove when this move is made.
20. \*/
21. struct Move
22. {
23. int startSquare;
24. std::vector<int> destinationSquare;
25. std::vector<int> removalSquare;
26. };
27. /\*\*
28. \* Struct BoardMoveTable is a static data type that is used to speed
29. \* up searching for moves for each position. It is shared among all
30. \* boards.
31. \*/
32. struct BoardMoveTable
33. {
34. std::vector<int> jumps;
35. std::vector<int> removals;
36. std::vector<int> moves;
37. };
38. Board();
39. ~Board();
40. static void InitializeMoveTable();
41. std::vector<Move> moveGen(Color color);
42. void printBoard() const;
43. Board updateBoard(Move move, Color color);
44. int getNumRegularPieces(Color color);
45. int getNumKingPieces(Color color);
46. int getNumPlayerTotalPieces(Color color);
47. int getPieceInSquare(int square, Color color);
48. std::vector<Move> getJumpsForPiece(Color color, int square, Pieces \*playerPieces, Pieces \*opponentPieces);
49. std::vector<Move> getMovesForPiece(Color color, int square, Pieces \*playerPieces, Pieces \*opponentPieces);
50. Pieces getPlayerPieces(Color color);
51. Pieces getOpponentPieces(Color color);
52. static BoardMoveTable boardMoveTable[33];
53. private:
54. Pieces blackPieces;
55. Pieces redPieces;
56. void getJumpsForPieceRec(Color color, Board::Move move, std::vector<Board::Move> &totalMoves, Board board, bool wasKingPriorMove);
57. };
58. #endif

## Board.cpp

1. #include "Board.hpp"
2. #include "Pieces.hpp"
3. #include <iostream>
4. #include <iomanip>
5. // Forward declare the static data member.
6. Board::BoardMoveTable Board::boardMoveTable[33];
7. /\*\*
8. \* Constructor | Board | Board
9. \*
10. \* Summary : Creates both the red player and black player
11. \*              piece list. If not already done, it also
12. \*             instantiates the static move table.
13. \*
14. \* @author : David Torrente
15. \*
16. \*/
17. Board::Board()
18. {
19. // Guarded internally to prevent from having moves added to it.
20. InitializeMoveTable();
21. // Assign the proper pieces to the player, either red or black.
22. redPieces = Pieces(Color::RED);
23. blackPieces = Pieces(Color::BLACK);
24. }
25. /\*\*
26. \* Destructor | Board | ~Board
27. \*
28. \* Summary : Class destructor. No special code required.
29. \*
30. \* @author : David Torrente
31. \*
32. \*/
33. Board::~Board()
34. {
35. }
36. /\*\*
37. \* Member Function | Board | getPlayerPieces
38. \*
39. \* Summary :    Gets the pieces that belong to a particular player.
40. \*
41. \* @author : David Torrente
42. \*
43. \* @param Color color :  The player color of pieces to get.
44. \*
45. \* @return Pieces : The pieces belonging to the player.
46. \*
47. \*/
48. Pieces Board::getPlayerPieces(Color color)
49. {
50. if (color == Color::RED)
51. {
52. return redPieces;
53. }
54. else
55. {
56. return blackPieces;
57. }
58. }
59. /\*\*
60. \* Member Function | Board | getOpponentPieces
61. \*
62. \* Summary :    Gets the pieces that belong to the
63. \*              opposing player.
64. \*
65. \* @author : David Torrente
66. \*
67. \* @param Color color :  The player color of pieces to
68. \*                       get the opponent of.
69. \*
70. \* @return Pieces : The pieces belonging to the
71. \*                      opposing player.
72. \*
73. \*/
74. Pieces Board::getOpponentPieces(Color color)
75. {
76. if (color == Color::RED)
77. {
78. return blackPieces;
79. }
80. else
81. {
82. return redPieces;
83. }
84. }
85. /\*\*
86. \* Member Function | Board | getNumRegularPieces
87. \*
88. \* Summary :    Gets the count of man pieces that belong
89. \*              to the player.
90. \*
91. \* @author : David Torrente
92. \* @param Color color :  The player color of pieces to
93. \*                       get the count for.
94. \*
95. \* @return int :    A value 0 to 12 which is the count
96. \*                  of man pieces.
97. \*
98. \*/
99. int Board::getNumRegularPieces(Color color)
100. {
101. return getNumPlayerTotalPieces(color) - getNumKingPieces(color);
102. }
103. /\*\*
104. \* Member Function | Board | getNumKingPieces
105. \*
106. \* Summary :    Gets the count of king pieces that belong
107. \*              to the player.
108. \*
109. \* @author : David Torrente
110. \*
111. \* @param Color color :  The player color of pieces to
112. \*                       get the count for.
113. \*
114. \* @return int :    A value 0 to 12 which is the count
115. \*                  of king pieces.
116. \*
117. \*/
118. int Board::getNumKingPieces(Color color)
119. {
120. {
121. int kingPieceCount = 0;
122. long long \*playerPieces;
123. if (color == Color::RED)
124. {
125. playerPieces = &redPieces.pieces;
126. }
127. else
128. {
129. playerPieces = &blackPieces.pieces;
130. }
131. for (int iter = 32; iter < 64; iter++)
132. {
133. if (((\*playerPieces >> iter) & 1) == 1)
134. {
135. kingPieceCount++;
136. }
137. }
138. return kingPieceCount;
139. }
140. }
141. /\*\*
142. \* Member Function | Board | getNumPlayerTotalPieces
143. \*
144. \* Summary :    Gets the count of total pieces that belong
145. \*              to the player.
146. \*
147. \* @author : David Torrente
148. \*
149. \* @param Color color :  The player color of pieces to
150. \*                       get the count for.
151. \*
152. \* @return int :    A value 0 to 12 which is the count
153. \*                  of total pieces.
154. \*
155. \*/
156. int Board::getNumPlayerTotalPieces(Color color)
157. {
158. int totalPieceCount = 0;
159. long long \*playerPieces;
160. if (color == Color::RED)
161. {
162. playerPieces = &redPieces.pieces;
163. }
164. else
165. {
166. playerPieces = &blackPieces.pieces;
167. }
168. for (int iter = 0; iter < 32; iter++)
169. {
170. if (((\*playerPieces >> iter) & 1) == 1)
171. {
172. totalPieceCount++;
173. }
174. }
175. return totalPieceCount;
176. }
177. /\*\*
178. \* Member Function | Board | moveGen
179. \*
180. \* Summary :    Gets all of the possible moves for the player
181. \*              based on the color parameter. Note that this
182. \*              is bound by the mandatory jump rule, meaning
183. \*              that a player who can jump will not be given
184. \*              the option to move to a adjacent space. These
185. \*              adjacent space moves will not even be computed
186. \*              if there is a jump possible.
187. \*
188. \* @author : David Torrente
189. \*
190. \* @param Color color :         The player color to get the moves
191. \*                              for.
192. \*
193. \* @return vector<Board::Move> :    A vector of possible moves
194. \*                                  for the player.
195. \*
196. \*/
197. std::vector<Board::Move> Board::moveGen(Color color)
198. {
199. std::vector<Move> totalMoves;
200. std::vector<Move> returnedMoves;
201. Pieces \*playerPieces;
202. Pieces \*opponentPieces;
203. int bitOffset = 0;
204. if (color == Color::RED)
205. {
206. playerPieces = &redPieces;
207. opponentPieces = &blackPieces;
208. }
209. else
210. {
211. playerPieces = &blackPieces;
212. opponentPieces = &redPieces;
213. }
214. // Go through all 32 squares and see if it is one
215. // of the appropriate pieces belonging to player.
216. for (int pieceIter = 1; pieceIter <= 32; pieceIter++)
217. {
218. // The offset is used to align to 0 - 31, but
219. // the board in checkers is 1 - 32. Use an offset
220. // here to align to the position bits properly.
221. bitOffset = pieceIter - 1;
222. if ((((playerPieces->pieces) >> bitOffset) & 1) == 1)
223. {
224. // Check for possible jump mpves first.
225. returnedMoves = getJumpsForPiece(color, pieceIter, playerPieces, opponentPieces);
226. totalMoves.insert(totalMoves.end(), returnedMoves.begin(), returnedMoves.end());
227. }
228. }
229. // Go through all 32 squares and see if it is one
230. // of the appropriate pieces belonging to player.
231. // Do this only if no jumps are possible. This is controlled
232. // by the if condition here. No need to get moves if there are
233. // jumps already found.
234. if (totalMoves.size() == 0)
235. {
236. for (int pieceIter = 1; pieceIter <= 32; pieceIter++)
237. {
238. // The offset is used to align to 0 - 31, but
239. // the board in checkers is 1 - 32. Use an offset
240. // here to align to the position bits properly.
241. bitOffset = pieceIter - 1;
242. // Check if player is in this space
243. if ((((playerPieces->pieces) >> bitOffset) & 1) == 1)
244. {
245. // Check for non jump moves here.
246. returnedMoves = getMovesForPiece(color, pieceIter, playerPieces, opponentPieces);
247. totalMoves.insert(totalMoves.end(), returnedMoves.begin(), returnedMoves.end());
248. }
249. }
250. }
251. return totalMoves;
252. }
253. /\*\*
254. \* Member Function | Board | getJumpsForPiece
255. \*
256. \* Summary :    Gets all of the possible jumps for the specific
257. \*              piece based on the color parameter and location.
258. \*              This is only called after confirming that the player
259. \*              has a piece in the proper location.
260. \*
261. \* @author : David Torrente
262. \*
263. \* @param Color color :             The player color of pieces to
264. \*                                  get the jumps for.
265. \*
266. \* @param int piece   :             The board location to get the jumps
267. \*                                  for.
268. \*
269. \* @param Pieces \*playerPieces :    A pointer to the player pieces.
270. \*                                  Needed since these can block a possible
271. \*                                  jump.
272. \*
273. \* @param Pieces \*opponentPieces :  A pointer to the opponent pieces.
274. \*                                  Needed since these can block a possible
275. \*                                  jump as well as to confirm that a jump
276. \*                                  can happen.
277. \*
278. \* @return vector<Board::Move> :    A vector of possible jumps
279. \*                                  for the particular piece.
280. \*
281. \*/
282. std::vector<Board::Move> Board::getJumpsForPiece(Color color, int piece, Pieces \*playerPieces, Pieces \*opponentPieces)
283. {
284. // The returned vector of all possible jumps carried out completely
285. std::vector<Move> finalMoves;
286. Board board;
287. bool wasKingPriorMove = false;
288. if (color == Color::RED)
289. {
290. board.redPieces = \*playerPieces;
291. board.blackPieces = \*opponentPieces;
292. }
293. else
294. {
295. board.redPieces = \*opponentPieces;
296. board.blackPieces = \*playerPieces;
297. }
298. // In order to start the recursion, this is needed.
299. Move move;
300. move.startSquare = piece;
301. wasKingPriorMove = board.getPlayerPieces(color).isKing(move.startSquare);
302. getJumpsForPieceRec(color, move, finalMoves, board, wasKingPriorMove);
303. return finalMoves;
304. }
305. /\*\*
306. \* Member Function | Board | getJumpsForPieceRec
307. \*
308. \* Summary :    Gets all of the possible jumps in a single jump attempt.
309. \*              Often called "double" jumping. This creates a full
310. \*              jump chain. a recursive call.
311. \*
312. \* @author : David Torrente
313. \*
314. \* @param Color color :         The player color of pieces to
315. \*                              get the jump chain for.
316. \*
317. \* @param Move move   :         A starter move. Contains a starting
318. \*                              space, with an empty destination vector.
319. \*                              when passed in to a recursive call,
320. \*                              this destination vector may contain
321. \*                              moves, signifying that it is in the middle
322. \*                              of a jump chain.
323. \*
324. \* @param vector<Board> & totalMovesAccumulator : Accumulates all possible
325. \*                                      jumps at the end of each jump chain. Passed
326. \*                                      in by reference and returned to calling
327. \*                                      function.
328. \*
329. \* @param Board board :         The current state of the board, updated for each jump.
330. \*
331. \* @param bool wasKingPriorMove : A value used to determine if the king was a king prior
332. \*                                to this move. Becoming a king stops a jump chain.
333. \*
334. \*/
335. void Board::getJumpsForPieceRec(Color color, Board::Move move, std::vector<Board::Move> &totalMovesAccumulator, Board board, bool wasKingPriorMove)
336. {
337. int piece;
338. bool endOfJumpChain = true;
339. bool isKing = false;
340. int bitOffset = 0;
341. int squareJumped = 0;
342. // This determines if we are starting the jump sequence or in the middle of it.
343. if (move.destinationSquare.size() == 0)
344. {
345. piece = move.startSquare;
346. // use the current board as it is.
347. }
348. else
349. {
350. piece = move.destinationSquare.back();
351. board = updateBoard(move, color);
352. }
353. // First, determine if it is a king. This is needed to see which moves
354. // are valid for this piece/player. Move either up/down at first.
355. isKing = board.getPlayerPieces(color).isKing(piece);
356. if (isKing == wasKingPriorMove)
357. {
358. // Check if a jump position is open for this piece. This goes through all of the jumps.
359. for (int jumpIter = 0; jumpIter < Board::boardMoveTable[piece].jumps.size(); jumpIter++)
360. {
361. // Get the position of the jump, reduce it by one for an offset. Note
362. // that while it is one less than the position, the direction check still works
363. // since a jump will always be greater than the distance needed to determine
364. // direction.
365. bitOffset = (Board::boardMoveTable[piece].jumps.at(jumpIter) - 1);
366. // Check to see which direction it is going, and if it can go that direction.
367. // King goes both ways      Red not a king goes "down"            Black not a king goes "up"
368. if (isKing || ((piece - bitOffset) < 0 && color == Color::RED) || ((piece - bitOffset) > 0 && color == Color::BLACK))
369. {
370. // Combine both bit fields into one and check if the space is empty.
371. if ((((board.getPlayerPieces(color).pieces | board.getOpponentPieces(color).pieces) >> bitOffset) & 1) == 0)
372. {
373. // If it is open, get the space between. We will need to check it.
374. // Again with the -1 to help with the offset
375. squareJumped = Board::boardMoveTable[piece].removals.at(jumpIter) - 1;
376. if (((board.getOpponentPieces(color).pieces >> squareJumped) & 1) == 1)
377. {
378. // Yup, we are good!!!! this means that an opponent was in this spot and
379. // we can jump them.
380. // Since we found a new jump, we need to keep going with the jump chain
381. endOfJumpChain = false;
382. // Do not change start square. This is set to the overall start of the jump.
383. // Commented out to show the thought process behind setting the move.
384. // move.startSquare = piece;
385. move.destinationSquare.push\_back(bitOffset + 1);
386. move.removalSquare.push\_back(squareJumped + 1);
387. //make recursive call here
388. getJumpsForPieceRec(color, move, totalMovesAccumulator, board, wasKingPriorMove);
389. // Remove the jump we just added to the chain. We've already made the recursive call
390. move.destinationSquare.pop\_back();
391. move.removalSquare.pop\_back();
392. }
393. }
394. }
395. }
396. }
397. if (endOfJumpChain == true)
398. {
399. // Once here, it means we are at the leaf of a jump or
400. // that we became a king due to this jump.
401. // We add it to the list at that point.
402. // It will rise back up the chain when the stack unwinds.
403. if (move.destinationSquare.size() != 0)
404. {
405. totalMovesAccumulator.push\_back(move);
406. }
407. }
408. }
409. /\*\*
410. \* Member Function | Board | getMovesForPiece
411. \*
412. \* Summary :    Gets all of the possible moves for a piece. This call knows that the
413. \*              piece is a proper piece for this color to play on prior to this call.
414. \*              Also note that prior to this call, jumps should be checked. If there
415. \*              are jumps, do not allow these moves.
416. \*
417. \* @author : David Torrente
418. \*
419. \* @param Color color :             The player color of pieces to
420. \*                                  get the moves for.
421. \*
422. \* @param int piece   :             The board location to get the moves
423. \*                                  for.
424. \*
425. \* @param Pieces \*playerPieces :    A pointer to the player pieces.
426. \*                                  Needed since these can block a possible
427. \*                                  jump.
428. \*
429. \* @param Pieces \*opponentPieces :  A pointer to the opponent pieces.
430. \*                                  Needed since these can block a possible
431. \*                                  jump as well as to confirm that a jump
432. \*                                  can happen.
433. \*
434. \* @return vector<Board::Move> :    A vector of possible moves
435. \*                                  for the particular piece.
436. \*
437. \*/
438. std::vector<Board::Move> Board::getMovesForPiece(Color color, int piece, Pieces \*playerPieces, Pieces \*opponentPieces)
439. {
440. std::vector<Move> moves;
441. Move move;
442. bool isKing = false;
443. int squareJumped = 0;
444. int bitOffset = 0;
445. // First, determine if it is a king. This is needed to see which moves
446. // are valid for this piece/player. Move either up/down at first.
447. if (playerPieces->isKing(piece))
448. {
449. isKing = true;
450. }
451. else
452. {
453. isKing = false;
454. }
455. for (int moveIter = 0; moveIter < Board::boardMoveTable[piece].moves.size(); moveIter++)
456. {
457. bitOffset = Board::boardMoveTable[piece].moves.at(moveIter) - 1;
458. if (isKing || ((piece - bitOffset) < 0 && color == Color::RED) || ((piece - bitOffset) > 0 && color == Color::BLACK))
459. {
460. if ((((playerPieces->pieces | opponentPieces->pieces) >> bitOffset) & 1) == 0)
461. {
462. move.startSquare = piece;
463. move.destinationSquare.push\_back(bitOffset + 1);
464. moves.push\_back(move);
465. move.destinationSquare.clear();
466. }
467. }
468. }
469. return moves;
470. }
471. /\*\*
472. \* Member Function | Board | printBoard
473. \*
474. \* Summary :    Prints the current state of the board. It is a constant function.
475. \*
476. \* @author : David Torrente
477. \*
478. \*/
479. void Board::printBoard() const
480. {
481. int squareOffset = 0;
482. // The board toggles back and forth in regards to the positioning
483. // of pieces. This either adds the offset to the beginning of the
484. // row, or to the end of it. Odd rows start with a space,
485. // Even rows end with a space.
486. bool oddRow = true;
487. // A basic bar to go across the top. Cosmetic.
488. std::cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << std::endl;
489. for (int rowIter = 0; rowIter <= 7; rowIter++)
490. {
491. std::cout << "|";
492. for (int colIter = 0; colIter <= 3; colIter++)
493. {
494. if (oddRow)
495. {
496. std::cout << "   |";
497. }
498. squareOffset = (rowIter \* 4 + colIter);
499. if (((redPieces.pieces >> squareOffset) & 1) == 1)
500. {
501. if (redPieces.isKing(squareOffset + 1))
502. {
503. std::cout << Pieces::ANSII\_RED\_HIGH << " R " << Pieces::ANSII\_END << "|";
504. }
505. else
506. {
507. std::cout << Pieces::ANSII\_RED\_HIGH << " r " << Pieces::ANSII\_END << "|";
508. }
509. }
510. else if (((blackPieces.pieces >> squareOffset) & 1) == 1)
511. {
512. if (blackPieces.isKing(squareOffset + 1))
513. {
514. std::cout << Pieces::ANSII\_BLUE\_START << " B " << Pieces::ANSII\_END << "|";
515. }
516. else
517. {
518. std::cout << Pieces::ANSII\_BLUE\_START << " b " << Pieces::ANSII\_END << "|";
519. }
520. }
521. else
522. {
523. std::cout << "   |";
524. }
525. if (!oddRow)
526. {
527. std::cout << "   |";
528. }
529. }
530. // Now print the numeric values of the squares.
531. std::cout << std::endl;
532. std::cout << "|";
533. for (int colIterNum = 0; colIterNum <= 3; colIterNum++)
534. {
535. if (oddRow)
536. {
537. std::cout << "\_\_\_|";
538. }
539. squareOffset = (rowIter \* 4 + colIterNum);
540. std::cout << std::setfill('\_') << std::setw(3) << squareOffset + 1 << "|";
541. if (!oddRow)
542. {
543. std::cout << "\_\_\_|";
544. }
545. }
546. // Toggle between even and odd rows for the offset.
547. oddRow = !oddRow;
548. std::cout << std::endl;
549. }
550. }
551. /\*\*
552. \* Member Function | Board | getPieceInSquare
553. \*
554. \* Summary :    Gets the type of piece at a particular location for a particular color.
555. \*              Note that the position does not need to be checked for a piece prior to this
556. \*              call. If the position does not contain a piece of the matching color type,
557. \*              it will report as empty for that color.
558. \*
559. \* @author : David Torrente
560. \*
561. \* @param int piece   :             The board location to get the piece type
562. \*                                  for.
563. \*
564. \* @param Color color :             The player color to get the piece type for.
565. \*
566. \* @return int  :                   a value, 0 to 2, specifying the piece type of the given
567. \*                                  color. 0 = no piece for this color. 1 = man piece
568. \*                                  for this color. 2 = king piece for this color.
569. \*
570. \*/
571. int Board::getPieceInSquare(int position, Color color)
572. {
573. int pieceType = 0;
574. Pieces playerPieces;
575. if (Color::RED == color)
576. {
577. playerPieces = redPieces;
578. }
579. else
580. {
581. playerPieces = blackPieces;
582. }
583. if (((playerPieces.pieces >> (position - 1)) & 1) == 1)
584. {
585. pieceType = 1;
586. if (playerPieces.isKing(position))
587. {
588. pieceType = 2;
589. }
590. }
591. return pieceType;
592. }
593. /\*\*
594. \* Member Function | Board | updateBoard
595. \*
596. \* Summary :    Updates the entire board based on a given move. Note that this updates the complete
597. \*              board for both sides. This will always operate on a single properly formatted move.
598. \*              It will also convert pieces into kings if they reach the back row of the opposing
599. \*              player.
600. \*
601. \* @author : David Torrente
602. \*
603. \* @param Move move   :             The move to apply to the board. Includes the destination
604. \*                                  and pieces to remove.
605. \*
606. \* @param Color color :             The player color that is applying this move. Needed primarily
607. \*                                  to determine if a piece needs to be kinged.
608. \*
609. \* @return Board    :               Returns a new copy of the board. Note that this is only
610. \*                                  two bitfields (small).
611. \*
612. \*/
613. Board Board::updateBoard(Move move, Color color)
614. {
615. Board updatedBoard;
616. updatedBoard.blackPieces = blackPieces;
617. updatedBoard.redPieces = redPieces;
618. Pieces \*playerPieces;
619. Pieces \*opponentPieces;
620. if (color == Color::RED)
621. {
622. playerPieces = &updatedBoard.redPieces;
623. opponentPieces = &updatedBoard.blackPieces;
624. }
625. else
626. {
627. playerPieces = &updatedBoard.blackPieces;
628. opponentPieces = &updatedBoard.redPieces;
629. }
630. // Position in final destination spot - probably check if it needs to be kinged here.
631. // Do this first since you will need to clear the king flag.
632. playerPieces->pieces = playerPieces->pieces | (1LL << (move.destinationSquare.back() - 1));
633. if (playerPieces->isKing(move.startSquare))
634. {
635. playerPieces->setKing(move.startSquare, false);
636. playerPieces->setKing(move.destinationSquare.back(), true);
637. }
638. else
639. {
640. // Check to see if we've landed in the kinging row, the back row opposite the starting side.
641. if (color == Color::RED && (move.destinationSquare.back() >= 29) || (color == Color::BLACK && (move.destinationSquare.back() <= 4)))
642. {
643. playerPieces->setKing(move.destinationSquare.back(), true);
644. }
645. }
646. // Remove start pos
647. if (move.startSquare != move.destinationSquare.back())
648. {
649. playerPieces->pieces = playerPieces->pieces & ~(1LL << (move.startSquare - 1));
650. }
651. // Remove all jumped spots and set them back to not a king.
652. for (int jumpedSpaceIter = 0; jumpedSpaceIter < move.removalSquare.size(); jumpedSpaceIter++)
653. {
654. opponentPieces->pieces = opponentPieces->pieces & ~(1LL << (move.removalSquare.at(jumpedSpaceIter) - 1));
655. opponentPieces->pieces = opponentPieces->pieces & ~(1LL << (move.removalSquare.at(jumpedSpaceIter) + 31));
656. }
657. return updatedBoard;
658. }
659. /\*\*
660. \* Member Function | Board | InitializeMoveTable
661. \*
662. \* Summary :    Sets up the static move table which is shared among all
663. \*              boards. Note that it is a move table, not a piece tracker.
664. \*              pieces are stored in the board. In order to prevent adding
665. \*              moves each time a constructor is called, the move table
666. \*              will not add any additional values after it is set up
667. \*              the first time.
668. \*
669. \* @author : David Torrente
670. \*
671. \*/
672. void Board::InitializeMoveTable()
673. {
674. // In order to support faster look up,
675. // The index of the position is used as
676. // the initial key to an index. Each
677. // position has only a few moves,
678. // even less if there is a mandatory jump.
679. // The table is big, but never changes.
680. if (boardMoveTable[1].jumps.size() == 0)
681. {
682. boardMoveTable[1].jumps.push\_back(10);
683. boardMoveTable[1].removals.push\_back(6);
684. boardMoveTable[1].moves.push\_back(5);
685. boardMoveTable[1].moves.push\_back(6);
686. boardMoveTable[2].jumps.push\_back(9);
687. boardMoveTable[2].removals.push\_back(6);
688. boardMoveTable[2].jumps.push\_back(11);
689. boardMoveTable[2].removals.push\_back(7);
690. boardMoveTable[2].moves.push\_back(6);
691. boardMoveTable[2].moves.push\_back(7);
692. boardMoveTable[3].jumps.push\_back(10);
693. boardMoveTable[3].removals.push\_back(7);
694. boardMoveTable[3].jumps.push\_back(12);
695. boardMoveTable[3].removals.push\_back(8);
696. boardMoveTable[3].moves.push\_back(7);
697. boardMoveTable[3].moves.push\_back(8);
698. boardMoveTable[4].jumps.push\_back(11);
699. boardMoveTable[4].removals.push\_back(8);
700. boardMoveTable[4].moves.push\_back(8);
701. boardMoveTable[5].jumps.push\_back(14);
702. boardMoveTable[5].removals.push\_back(9);
703. boardMoveTable[5].moves.push\_back(1);
704. boardMoveTable[5].moves.push\_back(9);
705. boardMoveTable[6].jumps.push\_back(13);
706. boardMoveTable[6].removals.push\_back(9);
707. boardMoveTable[6].jumps.push\_back(15);
708. boardMoveTable[6].removals.push\_back(10);
709. boardMoveTable[6].moves.push\_back(1);
710. boardMoveTable[6].moves.push\_back(2);
711. boardMoveTable[6].moves.push\_back(9);
712. boardMoveTable[6].moves.push\_back(10);
713. boardMoveTable[7].jumps.push\_back(14);
714. boardMoveTable[7].removals.push\_back(10);
715. boardMoveTable[7].jumps.push\_back(16);
716. boardMoveTable[7].removals.push\_back(11);
717. boardMoveTable[7].moves.push\_back(2);
718. boardMoveTable[7].moves.push\_back(3);
719. boardMoveTable[7].moves.push\_back(10);
720. boardMoveTable[7].moves.push\_back(11);
721. boardMoveTable[8].jumps.push\_back(15);
722. boardMoveTable[8].removals.push\_back(11);
723. boardMoveTable[8].moves.push\_back(3);
724. boardMoveTable[8].moves.push\_back(4);
725. boardMoveTable[8].moves.push\_back(11);
726. boardMoveTable[8].moves.push\_back(12);
727. boardMoveTable[9].jumps.push\_back(2);
728. boardMoveTable[9].removals.push\_back(6);
729. boardMoveTable[9].jumps.push\_back(18);
730. boardMoveTable[9].removals.push\_back(14);
731. boardMoveTable[9].moves.push\_back(5);
732. boardMoveTable[9].moves.push\_back(6);
733. boardMoveTable[9].moves.push\_back(13);
734. boardMoveTable[9].moves.push\_back(14);
735. boardMoveTable[10].jumps.push\_back(1);
736. boardMoveTable[10].removals.push\_back(6);
737. boardMoveTable[10].jumps.push\_back(3);
738. boardMoveTable[10].removals.push\_back(7);
739. boardMoveTable[10].jumps.push\_back(17);
740. boardMoveTable[10].removals.push\_back(14);
741. boardMoveTable[10].jumps.push\_back(19);
742. boardMoveTable[10].removals.push\_back(15);
743. boardMoveTable[10].moves.push\_back(6);
744. boardMoveTable[10].moves.push\_back(7);
745. boardMoveTable[10].moves.push\_back(14);
746. boardMoveTable[10].moves.push\_back(15);
747. boardMoveTable[11].jumps.push\_back(2);
748. boardMoveTable[11].removals.push\_back(7);
749. boardMoveTable[11].jumps.push\_back(4);
750. boardMoveTable[11].removals.push\_back(8);
751. boardMoveTable[11].jumps.push\_back(18);
752. boardMoveTable[11].removals.push\_back(15);
753. boardMoveTable[11].jumps.push\_back(20);
754. boardMoveTable[11].removals.push\_back(16);
755. boardMoveTable[11].moves.push\_back(7);
756. boardMoveTable[11].moves.push\_back(8);
757. boardMoveTable[11].moves.push\_back(15);
758. boardMoveTable[11].moves.push\_back(16);
759. boardMoveTable[12].jumps.push\_back(3);
760. boardMoveTable[12].removals.push\_back(8);
761. boardMoveTable[12].jumps.push\_back(19);
762. boardMoveTable[12].removals.push\_back(16);
763. boardMoveTable[12].moves.push\_back(8);
764. boardMoveTable[12].moves.push\_back(16);
765. boardMoveTable[13].jumps.push\_back(6);
766. boardMoveTable[13].removals.push\_back(9);
767. boardMoveTable[13].jumps.push\_back(22);
768. boardMoveTable[13].removals.push\_back(17);
769. boardMoveTable[13].moves.push\_back(9);
770. boardMoveTable[13].moves.push\_back(17);
771. boardMoveTable[14].jumps.push\_back(5);
772. boardMoveTable[14].removals.push\_back(9);
773. boardMoveTable[14].jumps.push\_back(7);
774. boardMoveTable[14].removals.push\_back(10);
775. boardMoveTable[14].jumps.push\_back(21);
776. boardMoveTable[14].removals.push\_back(17);
777. boardMoveTable[14].jumps.push\_back(23);
778. boardMoveTable[14].removals.push\_back(18);
779. boardMoveTable[14].moves.push\_back(9);
780. boardMoveTable[14].moves.push\_back(10);
781. boardMoveTable[14].moves.push\_back(17);
782. boardMoveTable[14].moves.push\_back(18);
783. boardMoveTable[15].jumps.push\_back(6);
784. boardMoveTable[15].removals.push\_back(10);
785. boardMoveTable[15].jumps.push\_back(8);
786. boardMoveTable[15].removals.push\_back(11);
787. boardMoveTable[15].jumps.push\_back(22);
788. boardMoveTable[15].removals.push\_back(18);
789. boardMoveTable[15].jumps.push\_back(24);
790. boardMoveTable[15].removals.push\_back(19);
791. boardMoveTable[15].moves.push\_back(10);
792. boardMoveTable[15].moves.push\_back(11);
793. boardMoveTable[15].moves.push\_back(18);
794. boardMoveTable[15].moves.push\_back(19);
795. boardMoveTable[16].jumps.push\_back(7);
796. boardMoveTable[16].removals.push\_back(11);
797. boardMoveTable[16].jumps.push\_back(23);
798. boardMoveTable[16].removals.push\_back(19);
799. boardMoveTable[16].moves.push\_back(11);
800. boardMoveTable[16].moves.push\_back(12);
801. boardMoveTable[16].moves.push\_back(19);
802. boardMoveTable[16].moves.push\_back(20);
803. boardMoveTable[17].jumps.push\_back(10);
804. boardMoveTable[17].removals.push\_back(14);
805. boardMoveTable[17].jumps.push\_back(26);
806. boardMoveTable[17].removals.push\_back(22);
807. boardMoveTable[17].moves.push\_back(13);
808. boardMoveTable[17].moves.push\_back(14);
809. boardMoveTable[17].moves.push\_back(21);
810. boardMoveTable[17].moves.push\_back(22);
811. boardMoveTable[18].jumps.push\_back(9);
812. boardMoveTable[18].removals.push\_back(14);
813. boardMoveTable[18].jumps.push\_back(11);
814. boardMoveTable[18].removals.push\_back(15);
815. boardMoveTable[18].jumps.push\_back(25);
816. boardMoveTable[18].removals.push\_back(22);
817. boardMoveTable[18].jumps.push\_back(27);
818. boardMoveTable[18].removals.push\_back(23);
819. boardMoveTable[18].moves.push\_back(14);
820. boardMoveTable[18].moves.push\_back(15);
821. boardMoveTable[18].moves.push\_back(22);
822. boardMoveTable[18].moves.push\_back(23);
823. boardMoveTable[19].jumps.push\_back(10);
824. boardMoveTable[19].removals.push\_back(15);
825. boardMoveTable[19].jumps.push\_back(12);
826. boardMoveTable[19].removals.push\_back(16);
827. boardMoveTable[19].jumps.push\_back(26);
828. boardMoveTable[19].removals.push\_back(23);
829. boardMoveTable[19].jumps.push\_back(28);
830. boardMoveTable[19].removals.push\_back(24);
831. boardMoveTable[19].moves.push\_back(15);
832. boardMoveTable[19].moves.push\_back(16);
833. boardMoveTable[19].moves.push\_back(23);
834. boardMoveTable[19].moves.push\_back(24);
835. boardMoveTable[20].jumps.push\_back(11);
836. boardMoveTable[20].removals.push\_back(16);
837. boardMoveTable[20].jumps.push\_back(27);
838. boardMoveTable[20].removals.push\_back(24);
839. boardMoveTable[20].moves.push\_back(16);
840. boardMoveTable[20].moves.push\_back(24);
841. boardMoveTable[21].jumps.push\_back(14);
842. boardMoveTable[21].removals.push\_back(17);
843. boardMoveTable[21].jumps.push\_back(30);
844. boardMoveTable[21].removals.push\_back(25);
845. boardMoveTable[21].moves.push\_back(17);
846. boardMoveTable[21].moves.push\_back(25);
847. boardMoveTable[22].jumps.push\_back(13);
848. boardMoveTable[22].removals.push\_back(17);
849. boardMoveTable[22].jumps.push\_back(15);
850. boardMoveTable[22].removals.push\_back(18);
851. boardMoveTable[22].jumps.push\_back(29);
852. boardMoveTable[22].removals.push\_back(25);
853. boardMoveTable[22].jumps.push\_back(31);
854. boardMoveTable[22].removals.push\_back(26);
855. boardMoveTable[22].moves.push\_back(17);
856. boardMoveTable[22].moves.push\_back(18);
857. boardMoveTable[22].moves.push\_back(25);
858. boardMoveTable[22].moves.push\_back(26);
859. boardMoveTable[23].jumps.push\_back(14);
860. boardMoveTable[23].removals.push\_back(18);
861. boardMoveTable[23].jumps.push\_back(16);
862. boardMoveTable[23].removals.push\_back(19);
863. boardMoveTable[23].jumps.push\_back(30);
864. boardMoveTable[23].removals.push\_back(26);
865. boardMoveTable[23].jumps.push\_back(32);
866. boardMoveTable[23].removals.push\_back(27);
867. boardMoveTable[23].moves.push\_back(18);
868. boardMoveTable[23].moves.push\_back(19);
869. boardMoveTable[23].moves.push\_back(26);
870. boardMoveTable[23].moves.push\_back(27);
871. boardMoveTable[24].jumps.push\_back(15);
872. boardMoveTable[24].removals.push\_back(19);
873. boardMoveTable[24].jumps.push\_back(31);
874. boardMoveTable[24].removals.push\_back(27);
875. boardMoveTable[24].moves.push\_back(19);
876. boardMoveTable[24].moves.push\_back(20);
877. boardMoveTable[24].moves.push\_back(27);
878. boardMoveTable[24].moves.push\_back(28);
879. boardMoveTable[25].jumps.push\_back(18);
880. boardMoveTable[25].removals.push\_back(22);
881. boardMoveTable[25].moves.push\_back(21);
882. boardMoveTable[25].moves.push\_back(22);
883. boardMoveTable[25].moves.push\_back(29);
884. boardMoveTable[25].moves.push\_back(30);
885. boardMoveTable[26].jumps.push\_back(17);
886. boardMoveTable[26].removals.push\_back(22);
887. boardMoveTable[26].jumps.push\_back(19);
888. boardMoveTable[26].removals.push\_back(23);
889. boardMoveTable[26].moves.push\_back(22);
890. boardMoveTable[26].moves.push\_back(23);
891. boardMoveTable[26].moves.push\_back(30);
892. boardMoveTable[26].moves.push\_back(31);
893. boardMoveTable[27].jumps.push\_back(18);
894. boardMoveTable[27].removals.push\_back(23);
895. boardMoveTable[27].jumps.push\_back(20);
896. boardMoveTable[27].removals.push\_back(24);
897. boardMoveTable[27].moves.push\_back(23);
898. boardMoveTable[27].moves.push\_back(24);
899. boardMoveTable[27].moves.push\_back(31);
900. boardMoveTable[27].moves.push\_back(32);
901. boardMoveTable[28].jumps.push\_back(19);
902. boardMoveTable[28].removals.push\_back(24);
903. boardMoveTable[28].moves.push\_back(24);
904. boardMoveTable[28].moves.push\_back(32);
905. boardMoveTable[29].jumps.push\_back(22);
906. boardMoveTable[29].removals.push\_back(25);
907. boardMoveTable[29].moves.push\_back(25);
908. boardMoveTable[30].jumps.push\_back(21);
909. boardMoveTable[30].removals.push\_back(25);
910. boardMoveTable[30].jumps.push\_back(23);
911. boardMoveTable[30].removals.push\_back(26);
912. boardMoveTable[30].moves.push\_back(25);
913. boardMoveTable[30].moves.push\_back(26);
914. boardMoveTable[31].jumps.push\_back(22);
915. boardMoveTable[31].removals.push\_back(26);
916. boardMoveTable[31].jumps.push\_back(24);
917. boardMoveTable[31].removals.push\_back(27);
918. boardMoveTable[31].moves.push\_back(26);
919. boardMoveTable[31].moves.push\_back(27);
920. boardMoveTable[32].jumps.push\_back(23);
921. boardMoveTable[32].removals.push\_back(27);
922. boardMoveTable[32].moves.push\_back(27);
923. boardMoveTable[32].moves.push\_back(28);
924. }
925. }

## Player.hpp

1. #ifndef PLAYER\_H
2. #define PLAYER\_H
3. #include "Pieces.hpp"
4. #include "Board.hpp"
5. /\*\*
6. \* Header definition for class Player.
7. \*
8. \* The Player class defines a virtual player, in the case of this program an AI.
9. \* No human players are used in this game - meaning no manual input is sought from the user.
10. \* Project 2, Requirement #7: "Your program should play with a computer to a computer."
11. \*
12. \*/
13. class Player
14. {
15. private:
16. Color color;        // represents player's color in the game, either RED or BLACK
17. int numPieces;      // how many pieces does Player have left
18. int numPiecesTaken; // Player's current score based on captured enemy pieces
19. int numTurnsTaken;  // counter for Player's turns taken
20. bool didPlayerMove; //  EndGame Condition
21. int depth, evalVersion;
22. public:
23. Player();  // constructor
24. ~Player(); // destructor
25. bool isMinimax; // if false use AB Prune, if true use Minimax. Allows control over alg player uses
26. Player(bool minMax, Color color, int depth, int evalVersion); // overloaded constructor to set player color, which is IMMUTABLE
27. int takeTurn(Board &state);
28. int getNumPieces();
29. int getNumPiecesTaken();
30. int getNumTurns();
31. bool getDidPlayerMove();
32. Color getColor();
33. void decreaseNumPieces(int numPiecesToDecreaseCount);
34. void increaseNumPiecesTaken(int numPiecesToIncreaseScore);
35. static void printMove(Board::Move, Color color, bool willPrintAlways);
36. int minimaxExpandedNodes;  // how many nodes we expand
37. int minimaxLeafNodes;      // how many nodes we expand
38. int absearchExpandedNodes; // how many nodes we expand
39. int absearchLeafNodes;     // how many nodes we expand
40. int getMinimaxTotalNodes()
41. {
42. return minimaxExpandedNodes + minimaxLeafNodes;
43. }
44. int getAbSearchTotalNodes()
45. {
46. return absearchExpandedNodes + absearchLeafNodes;
47. }
48. };
49. #endif // !PLAYER\_H

## Player.cpp

1. #include "Player.hpp"
2. #include "Algorithm.hpp"
3. #include <iostream>
4. Player::Player()
5. {
6. }
7. Player::~Player()
8. {
9. }
10. Player::Player(bool minMaxState, Color color, int depth, int evalVersion)
11. {
12. this->color = color;
13. numPieces = 12;     // how many pieces does Player have left
14. numPiecesTaken = 0; // Player's current score based on captured enemy pieces
15. numTurnsTaken = 0;  // counter for Player's turns taken
16. isMinimax = minMaxState;
17. this->depth = depth;
18. this->evalVersion = evalVersion;
19. this->minimaxExpandedNodes = 0;
20. this->minimaxLeafNodes = 0;
21. this->absearchExpandedNodes = 0;
22. this->absearchLeafNodes = 0;
23. }
24. int Player::takeTurn(Board &state)
25. {
26. Algorithm::Result result;
27. Algorithm \*algorithm = new Algorithm(evalVersion, depth, \*this);
28. if (isMinimax)
29. {
30. result = algorithm->minimax\_a\_b(state, this->depth, this->color, 9000000, -8000000);
31. this->minimaxExpandedNodes += algorithm->minimaxExpandedNodes;
32. this->minimaxLeafNodes += algorithm->minimaxLeafNodes;
33. }
34. else
35. {
36. result = algorithm->alphaBetaSearch(state);
37. this->absearchExpandedNodes += algorithm->absearchExpandedNodes;
38. this->absearchLeafNodes += algorithm->absearchLeafNodes;
39. }
40. if (result.bestMove.destinationSquare.size() == 0)
41. {
42. didPlayerMove = false; // Player did not make a turn
43. }
44. else
45. {
46. state = state.updateBoard(result.bestMove, this->color);
47. printMove(result.bestMove, this->color, true);
48. numTurnsTaken++;      // incremente Player's own turn counter
49. didPlayerMove = true; // return true as player did make a turn
50. state.printBoard();
51. }
52. // return how many pieces the player took during their turn
53. return result.bestMove.removalSquare.size();
54. }
55. int Player::getNumPieces()
56. {
57. return numPieces;
58. }
59. int Player::getNumPiecesTaken()
60. {
61. return numPiecesTaken;
62. }
63. int Player::getNumTurns()
64. {
65. return numTurnsTaken;
66. }
67. bool Player::getDidPlayerMove()
68. {
69. return didPlayerMove;
70. }
71. Color Player::getColor()
72. {
73. return color;
74. }
75. void Player::decreaseNumPieces(int numPiecesToDecreaseCount)
76. {
77. numPieces -= numPiecesToDecreaseCount;
78. }
79. void Player::increaseNumPiecesTaken(int numPiecesToIncreaseScore)
80. {
81. numPiecesTaken += numPiecesToIncreaseScore;
82. }
83. void Player::printMove(Board::Move move, Color color, bool alwaysPrint)
84. {
85. std::string colorStr;
86. if (color == Color::BLACK)
87. colorStr = "Black";
88. else
89. colorStr = "Red";
90. int conditionalPrint;
91. if (alwaysPrint)
92. conditionalPrint = 1;
93. else
94. conditionalPrint = Pieces::ouputDebugData;
95. if (conditionalPrint)
96. {
97. std::cout << colorStr << " moves from " << move.startSquare << " to destination (in sequence): ";
98. for (int i = 0; i < move.destinationSquare.size(); i++)
99. std::cout << "dest: " << move.destinationSquare.at(i) << std::endl;
100. }
101. }

## Game.hpp

1. #ifndef GAME\_H
2. #define GAME\_H
3. #include "Player.hpp"
4. #include "Board.hpp"
5. /\*\*
6. \* Header definition for class Game.
7. \*
8. \* The Game class represents a checkers Game.
9. \*
10. \* A game consists of a board, two players (red and black), and
11. \* 12 checkers pieces for each player.
12. \*
13. \* A game ends when either player loses all their pieces or is blocked and
14. \* has no further moves available.
15. \*
16. \* The startGame() function will trigger the process and no input from the user is required.
17. \* Red and Black players will be created. Each player will be given 12 checker pieces.
18. \* The pieces will be appropriately placed on the board.
19. \* Each player will execute their strategy.
20. \*
21. \*/
22. class Game
23. {
24. private:
25. Board state;
26. Player redPlayer;
27. Player blackPlayer;
28. const int MAX\_ALLOWED\_TURNS = 80;
29. void printNodes(Player player, std::string colorText);
30. public:
31. Game();                          // constructor
32. ~Game();                         // destructor
33. Game(bool, int, bool, int, int); // player1 algo, eval version, player2 algo, eval version, depth
34. enum class GameOver
35. {
36. BLACK\_WINS = 0,
37. RED\_WINS = 1,
38. DRAW = 2,
39. NOT\_DONE = 3
40. };
41. // The only initialization function needed, as the game will
42. // be played automatically by 2 AI players (MIN and MAX).
43. // while gameOver == NOT\_DONE keep playing
44. GameOver startGame();
45. // simple function to invert the enum value, thus determine who's turn is it next.
46. // E.g., if currentPlayer is RED (1), function returns BLACK (-1)
47. Color changePlayer(Color currentPlayer);
48. GameOver gameOver(); // Have end game conditions been met?
49. bool doesRedWin();
50. bool doesBlackWin();
51. bool isItADraw();
52. };
53. #endif // !GAME\_H

## Game.cpp

1. #include "Game.hpp"
2. #include <iostream>
3. Game::Game()
4. {
5. }
6. Game::~Game()
7. {
8. }
9. Game::Game(bool player1MinMax, int evalVersionP1, bool player2MinMax, int evalVersionP2, int depth)
10. {
11. state = Board();
12. state.InitializeMoveTable();
13. redPlayer = Player(player1MinMax, Color::RED, depth, evalVersionP1);
14. blackPlayer = Player(player2MinMax, Color::BLACK, depth, evalVersionP2);
15. }
16. Game::GameOver Game::startGame()
17. {
18. int piecesTaken;
19. while (true)
20. {
21. std::cout << "\n\nRound " << blackPlayer.getNumTurns() + 1 << " Black's Move..." << std::endl;
22. piecesTaken = blackPlayer.takeTurn(state);
23. blackPlayer.increaseNumPiecesTaken(piecesTaken);
24. redPlayer.decreaseNumPieces(piecesTaken);
25. if (piecesTaken > 0)
26. std::cout << "  BLACK player took " << piecesTaken << " piece(s)." << std::endl;
27. if (doesBlackWin())
28. {
29. printNodes(redPlayer, "RED");
30. printNodes(blackPlayer, "BLACK");
31. return GameOver::BLACK\_WINS;
32. }
33. if (doesRedWin())
34. {
35. printNodes(redPlayer, "RED");
36. printNodes(blackPlayer, "BLACK");
37. return GameOver::RED\_WINS;
38. }
39. if (isItADraw())
40. {
41. printNodes(redPlayer, "RED");
42. printNodes(blackPlayer, "BLACK");
43. return GameOver::DRAW;
44. }
45. std::cout << "\n\nRound " << redPlayer.getNumTurns() + 1 << " Red's Move..." << std::endl;
46. piecesTaken = redPlayer.takeTurn(state);
47. redPlayer.increaseNumPiecesTaken(piecesTaken);
48. blackPlayer.decreaseNumPieces(piecesTaken);
49. if (piecesTaken > 0)
50. std::cout << "  RED player took " << piecesTaken << " piece(s)." << std::endl;
51. if (doesBlackWin())
52. {
53. printNodes(redPlayer, "RED");
54. printNodes(blackPlayer, "BLACK");
55. return GameOver::BLACK\_WINS;
56. }
57. if (doesRedWin())
58. {
59. printNodes(redPlayer, "RED");
60. printNodes(blackPlayer, "BLACK");
61. return GameOver::RED\_WINS;
62. }
63. if (isItADraw())
64. {
65. printNodes(redPlayer, "RED");
66. printNodes(blackPlayer, "BLACK");
67. return GameOver::DRAW;
68. }
69. }
70. }
71. void Game::printNodes(Player player, std::string colorText)
72. {
73. std::cout << std::endl;
74. if (player.isMinimax)
75. {
76. std::cout << colorText << " Leaf Nodes: " << player.minimaxLeafNodes << std::endl;
77. std::cout << colorText << " Expanded Nodes: " << player.minimaxExpandedNodes << std::endl;
78. std::cout << colorText << " Total Nodes: " << player.getMinimaxTotalNodes() << std::endl;
79. }
80. else
81. {
82. std::cout << colorText << " Leaf Nodes: " << player.absearchLeafNodes << std::endl;
83. std::cout << colorText << " Expanded Nodes: " << player.absearchExpandedNodes << std::endl;
84. std::cout << colorText << " Total Nodes: " << player.getAbSearchTotalNodes() << std::endl;
85. }
86. std::cout << std::endl;
87. }
88. Color Game::changePlayer(Color currentPlayer)
89. {
90. if (currentPlayer == Color::BLACK)
91. return Color::RED;
92. else
93. return Color::BLACK;
94. }
95. bool Game::doesBlackWin()
96. {
97. std::vector<Board::Move> redMoves = state.moveGen(Color::RED);
98. return (redMoves.size() == 0);
99. }
100. bool Game::doesRedWin()
101. {
102. std::vector<Board::Move> blackMoves = state.moveGen(Color::BLACK);
103. return (blackMoves.size() == 0);
104. }
105. bool Game::isItADraw()
106. {
107. if (redPlayer.getNumTurns() >= MAX\_ALLOWED\_TURNS ||
108. blackPlayer.getNumTurns() >= MAX\_ALLOWED\_TURNS)
109. return true;
110. return false;
111. }

## Algorithm.hpp

1. #ifndef ALGORITHM\_H
2. #define ALGORITHM\_H
3. #include <vector>
4. #include <string>
5. #include "Player.hpp"
6. #include "Board.hpp"
7. /\*\*
8. \* Header definition for class Algorithm.
9. \* @author multiple - David, Boris, and Randy
10. \*
11. \* This class will encapsulate the algorithmic approach the AI uses to play Checkers.
12. \*
13. \* There are only two major algorithms supported.
14. \*    1) Minimax - a depth first, depth limited search procedure. From the Richard and Knight book.
15. \*       The minimax function has a heuristic value for leaf nodes (end nodes and nodes at the maximum permitted depth).
16. \*       Intermediate nodes get their value from a child/successor leaf node.
17. \*    2) Alpha-Beta Pruning - a search algorithm that decreases the number of nodes evaluated by minimax in it's search tree.
18. \*       We stop evaluating a possible move when at least one option is found to be worse than a previously examined move.
19. \*       NOTE: It should return the SAME result as minimax, just "prunes" branches that will not affect the final outcome,
20. \*             thus improving performance.
21. \*
22. \* Three evaluation functions will be used in conjunction with the two search algorithms.
23. \*
24. \*/
25. class Algorithm
26. {
27. public:
28. Algorithm();  // constructor
29. ~Algorithm(); // destructor
30. Algorithm(int evalVersion, int maxDepth, Player callingPlayer);
31. struct Result
32. {
33. int value;
34. Board::Move bestMove;
35. };
36. int minimaxExpandedNodes;  // how many nodes we expand
37. int minimaxLeafNodes;      // how many nodes we expand
38. int absearchExpandedNodes; // how many nodes we expand
39. int absearchLeafNodes;     // how many nodes we expand
40. // minimax algorithm returns the position of the best move
41. Result minimax\_a\_b(Board board, int depth, Color color, int useThresh, int passThresh);
42. // AB Prune algorithm
43. Result alphaBetaSearch(Board state);
44. void setEvalVersion(int evalVersion);
45. void setMaxDepth(int maxDepth);
46. private:
47. int numNodesGenerated;
48. int evalVersion;
49. int currentDepth, maxDepth;
50. Player callingPlayer;
51. // plausible move generator, returns a list of positions that can be made by player
52. std::vector<Board::Move> movegen(Board board, Color color);
53. /\* static evaluation functions return a number representing the
54. \* goodness of Position from the standpoint of Player
55. \* A helper function staticEval is used to determine which evalFunction to use
56. \*/
57. int evalFunctOne(Board state, Color color);
58. int evalFunctTwo(Board state, Color color);
59. int evalFunctThree(Board state, Color color);
60. // wrapper function that will decide which of the actual three eval functions to call
61. int staticEval(Board state, Color color, int evalVersion);
62. // if true, return the structure
63. bool deepEnough(int currentDepth);
64. bool terminalTest(Board state, int depth); // terminal test for alpha-beta-search
65. Result maxValue(Board state, int depth, int alpha, int beta, Color color);
66. Result minValue(Board state, int depth, int alpha, int beta, Color color);
67. int utility(Board state);
68. std::vector<Board::Move> actions(Board state, Color color);
69. Color switchPlayerColor(Color color);
70. int passSign(int passthresh);
71. };
72. #endif // !ALGORITHM\_H

## Algorithm.cpp

1. #include "Algorithm.hpp"
2. #include <limits>
3. #include <stdexcept>
4. #include <iostream>
5. Algorithm::Algorithm()
6. {
7. this->minimaxLeafNodes = 0;
8. this->minimaxExpandedNodes = 0;
9. this->absearchLeafNodes = 0;
10. this->absearchExpandedNodes = 0;
11. }
12. Algorithm::~Algorithm()
13. {
14. }
15. /\*\*
16. \* Overloaded constructor for Algorithm to set internal member variables
17. \*/
18. Algorithm::Algorithm(int evalVersion, int maxDepth, Player callingPlayer)
19. {
20. this->evalVersion = evalVersion;
21. this->maxDepth = maxDepth;
22. this->callingPlayer = callingPlayer;
23. this->minimaxLeafNodes = 0;
24. this->minimaxExpandedNodes = 0;
25. this->absearchLeafNodes = 0;
26. this->absearchExpandedNodes = 0;
27. }
28. /\*\*
29. \* movegen function gets a list of all possible moves for a player as a vector of moves.
30. \* This function is essentially a wrapper, which calls the movegen function in class Board.
31. \* @author Borislav Sabotinov
32. \*
33. \* @param Board board
34. \* @param Player board
35. \*
36. \* @return vector<Board::Move> listofPossibleMoves
37. \*/
38. std::vector<Board::Move> Algorithm::movegen(Board board, Color color)
39. {
40. return board.moveGen(color);
41. }
42. /\*\*
43. \* Member Function | Algorithm | evalFunctionOne
44. \*
45. \* Summary :    Evaluates the current board and produces a score for it based on
46. \*              the current player of this ply. Note that it toggles back and forth
47. \*              from maximizing to minimizing.
48. \*
49. \* @author : David Torrente
50. \*
51. \* @param State state   :           The state of the board to be used in evaluation.
52. \*
53. \* @param Color color   :           The player color that is considered the current player.
54. \*
55. \* @return int          :           Returns an evaluated score for the board.
56. \*
57. \*/
58. int Algorithm::evalFunctOne(Board state, Color color)
59. {
60. int gameTurn = callingPlayer.getNumTurns();
61. int finalScore = 0;
62. bool criticalPoint = false;
63. Color opponentColor;
64. int backRowDefense = 0;
65. int opponentBackRowDefense = 0;
66. int centerPiece;
67. int centerControl = 0;
68. // These are the weights for the opening series of turns.
69. // They later adjust to better fit end game play.
70. int manWeight = 40;
71. int kingWeight = 90;
72. int mobilityWeight = 5;
73. int backRowDefenseWeight = 5;
74. int centerControlWeight = 6;
75. int playerScore = 0;
76. int playerBonus = 0;
77. int opponentManWeight = 40;
78. int opponentKingWeight = 90;
79. int opponentMobilityWeight = 5;
80. int opponentBackRowDefenseWeight = 2;
81. int opponentScore = 0;
82. int opponentBonus = 0;
83. std::vector<Board::Move> playerMoves;
84. std::vector<Board::Move> opponentMoves;
85. //=======================================================================
86. // Set up a few items based on the player color. Opponent and player
87. // defense row.
88. //=======================================================================
89. if (Color::RED == color)
90. {
91. opponentColor = Color::BLACK;
92. // Calculate back row defense
93. for (int redIter = 1; redIter <= 4; redIter++)
94. {
95. if (state.getPieceInSquare(redIter, color) == 1)
96. {
97. backRowDefense++;
98. }
99. if (state.getPieceInSquare(redIter + 28, opponentColor) == 1)
100. {
101. opponentBackRowDefense++;
102. }
103. }
104. }
105. else
106. {
107. opponentColor = Color::RED;
108. for (int blackIter = 29; blackIter <= 32; blackIter++)
109. {
110. if (state.getPieceInSquare(blackIter, color) == 1)
111. {
112. backRowDefense++;
113. }
114. if (state.getPieceInSquare(blackIter - 28, opponentColor) == 1)
115. {
116. opponentBackRowDefense++;
117. }
118. }
119. }
120. int manCount = state.getNumRegularPieces(color);
121. int kingCount = state.getNumKingPieces(color);
122. int opponentManCount = state.getNumRegularPieces(opponentColor);
123. int opponentKingCount = state.getNumKingPieces(opponentColor);
124. //=======================================================================
125. // Check for the two obvious ones, a winning state or a losing state.
126. // Used to simply exit out of the eval function without calculating
127. // a bunch of extra values.
128. //=======================================================================
129. playerMoves = state.moveGen(color);
130. int mobility = playerMoves.size();
131. // This is a losing state, avoid it.
132. // Note that vs. other evaluation functions, this may not happen.
133. // This means that while it sees a losing state, the other eval
134. // may take a different path and void out these results.
135. if (!criticalPoint && mobility == 0)
136. {
137. opponentBonus = opponentBonus + 6000000;
138. criticalPoint = true;
139. }
140. opponentMoves = state.moveGen(opponentColor);
141. int opponentMobility = opponentMoves.size();
142. // This is a winning state, try to get it.
143. // Note that vs. other evaluation functions, this may not happen.
144. // This means that while it sees a losing state, the other eval
145. // may take a different path and void out these results.
146. int totalJumped = 0;
147. int totalToJump = opponentKingCount + opponentManCount;
148. for (int moveIter = 0; !criticalPoint && moveIter < playerMoves.size(); moveIter++)
149. {
150. totalJumped = playerMoves.at(moveIter).destinationSquare.size();
151. if ((totalToJump - totalJumped) == 0)
152. {
153. playerBonus = playerBonus + 6000000;
154. criticalPoint = true;
155. }
156. }
157. if (!criticalPoint)
158. {
159. //=======================================================================
160. // Set some values based on the current turn to adjust play styles.
161. // Games tend to run a little longer, so the tunr numbers are set up
162. // higher than one would expect.
163. //=======================================================================
164. if (gameTurn >= 40)
165. {
166. manWeight = 20;
167. kingWeight = 80;
168. mobilityWeight = 3;
169. backRowDefenseWeight = 2;
170. centerControlWeight = 2;
171. opponentManWeight = 20;
172. opponentKingWeight = 80;
173. opponentMobilityWeight = 3;
174. opponentBackRowDefenseWeight = 1;
175. }
176. else if (gameTurn >= 60)
177. {
178. manWeight = 10;
179. kingWeight = 30;
180. mobilityWeight = 3;
181. backRowDefenseWeight = 0;
182. centerControlWeight = 1;
183. opponentManWeight = 40;
184. opponentKingWeight = 90;
185. opponentMobilityWeight = 1;
186. opponentBackRowDefenseWeight = 0;
187. }
188. //=======================================================================
189. // Check for pretty bad states, such as a large difference in the number
190. // of pieces between players. This typically means multiple jumps.
191. // Intended to avoid fine tuning specific calculations.
192. //=======================================================================
193. int totalPieceDisparityCount = ((manCount + kingCount) - (opponentManCount + opponentKingCount));
194. // This means attempt to reduce the opponent
195. // at almost any price. Even if it means losing
196. // a few pieces.
197. if (totalPieceDisparityCount >= 2)
198. {
199. playerBonus = playerBonus + 1000;
200. opponentManWeight = 600;
201. opponentKingWeight = 1600;
202. manWeight = 1;
203. kingWeight = 1;
204. }
205. // This is a larger disparity. Be even more agressive.
206. if (totalPieceDisparityCount >= 4)
207. {
208. playerBonus = playerBonus + 10000;
209. opponentManWeight = 6000;
210. opponentKingWeight = 16000;
211. manWeight = 1;
212. kingWeight = 1;
213. }
214. // This means that you are trailing in pieces.
215. // The objective is to now set up a more defensive
216. // mode of play. Typically it is bad to get here.
217. else if (totalPieceDisparityCount <= -2)
218. {
219. opponentBonus = opponentBonus + 1000;
220. manWeight = 60;
221. kingWeight = 160;
222. }
223. // A difference in kings can lead to a loss.
224. // This is a small adjustment to monitor this.
225. if ((kingCount - opponentKingCount) >= 2)
226. {
227. playerBonus = playerBonus + 100;
228. }
229. else if (kingCount - opponentKingCount <= -2)
230. {
231. opponentBonus = opponentBonus + 100;
232. }
233. //=======================================================================
234. // Check for a specific location bonus.
235. // 14,15,17,18,19,22,23 are the center locations.
236. // Add a bonus if you occupy them.
237. //=======================================================================
238. for (int centerIter = 14; centerIter <= 23; centerIter++)
239. {
240. centerPiece = state.getPieceInSquare(centerIter, color);
241. // Double the control value if it is a king in that location.
242. if (centerPiece == 1)
243. {
244. centerControl = centerControl + 1;
245. }
246. else if (centerPiece == 2)
247. {
248. centerControl = centerControl + 2;
249. }
250. // Skip the edges
251. if (centerIter == 15 || centerIter == 19)
252. {
253. centerIter++;
254. if (centerIter == 20)
255. {
256. centerIter++;
257. }
258. }
259. }
260. }
261. // Tally up the player overall score and then the opponent overall score.
262. playerScore =
263. (manCount \* manWeight) +
264. (kingCount \* kingWeight) +
265. (mobility \* mobilityWeight) +
266. (backRowDefense \* backRowDefenseWeight) +
267. playerBonus;
268. opponentScore =
269. (opponentManCount \* opponentManWeight) +
270. (opponentKingCount \* opponentKingWeight) +
271. (opponentMobility \* opponentMobilityWeight) +
272. (opponentBackRowDefense \* opponentBackRowDefenseWeight) +
273. opponentBonus;
274. finalScore = playerScore - opponentScore;
275. // Clamp the final score based on the current scoring
276. // system. Note that these values can later be adjusted
277. // to use a wider range.
278. if (finalScore <= -8000000)
279. {
280. finalScore = -7999999;
281. }
282. if (finalScore >= 9000000)
283. {
284. finalScore = 7999999;
285. }
286. return finalScore;
287. }
288. /\*\*
289. \* Second evaluation function
290. \* @author Randall Henderson
291. \*
292. \* @param Board State
293. \* @param Color color
294. \*
295. \* @return an integer score of how good we think the state is
296. \*/
297. int Algorithm::evalFunctTwo(Board state, Color color)
298. {
299. const int KING = 2,
300. MAN = 1;
301. std::string indentValue; // Tracking header in cout statements.  Shows who is player being processed
302. std::string playerColor; // Used for debugging output
303. // these four variables are the number of pieces each player has on the board in this state.
304. int numPieces = state.getNumRegularPieces(color),                              //  how many regular pieces are on the board
305. numKingPieces = state.getNumKingPieces(color),                             //  how many Kings are on the board
306. numOpponentPieces = state.getNumRegularPieces(switchPlayerColor(color)),   //  same as above
307. numOpponentKingsPieces = state.getNumKingPieces(switchPlayerColor(color)), // except opponent values
308. // Total Pieces on the board
309. totalPieces = numPieces + numKingPieces + numOpponentPieces + numOpponentKingsPieces,
310. // This places a value on the player's pieces.  This weight can be modified to encourage defensive
311. // piece preservation strategy
312. pieceValue = 560,
313. kingValue = 3000,
314. //  total value of the piece preservation portion of the equation
315. preservePlayersPieces = numPieces \* pieceValue + numKingPieces \* kingValue,
316. // total value of the opponents piece value
317. preserveOpponentsPieces = numOpponentPieces \* pieceValue + numOpponentKingsPieces \* kingValue,
318. //  This places a value on taking opponent pieces,  if a move removes an opponent piece this results in
319. // a higher return value for the move
320. opponentValue = 100,
321. opponentKingValue = 1000,
322. opponentPieces = (numOpponentPieces \* opponentValue + numOpponentKingsPieces \* opponentKingValue),        // Value of all opponent's pieces
323. reduceOpponentPieceValue = (opponentPieces - numOpponentKingsPieces \* opponentKingValue) / opponentValue, // value of each man
324. reduceOpponentKingValue = opponentPieces - reduceOpponentPieceValue,                                      // value of each king
325. crossOfPainValue = 50,                                                                                    //This is the square immediately above, below, to the right or left of the king
326. // When in end game (less than 8 pieces) this value is used to alter strategy between taking pieces
327. // and preserving pieces
328. endGameAdjust = 1,
329. playerPiece,
330. opponentPiece,
331. positionAdder = 0,
332. opponentPositionAdder = 0,
333. playerScore,
334. opponentScore,
335. moveScore,
336. currentTerminal = 0,
337. opponentTerminal = 0;
338. //  The opponent's board will be scored the same as the current player.  This percentage allows for the
339. // adjustment in the impact the player's board has on the final value
340. double opponentEvaluationWeight = .55;
341. // the values Red squares.  This encourages the Red men to move towards the opponents base line
342. // the two high values for Red's base line encourages a base defense strategy
343. // startup Board
344. int squareValuesForRed[] = {10, 1, 10, 1,
345. 8, 5, 5, 2,
346. 5, 3, 3, 5,
347. 5, 3, 5, 3,
348. 5, 8, 5, 8,
349. 15, 8, 15, 8,
350. 15, 25, 15, 25,
351. 75, 75, 75, 75};
352. if (numOpponentPieces == 3) // only Kings left, encourage all men to move to the back row
353. {
354. for (int i = 0; i < 32; ++i)
355. {
356. squareValuesForRed[i] = (i % 4) \* 3;
357. }
358. }
359. int squareValuesForBlack[32]; // this board will be initialized below
360. // black values are the opposite of Red
361. for (int i = 31; i >= 0; --i)
362. squareValuesForBlack[31 - i] = squareValuesForRed[i];
363. // this array is used to encourage the King to move towards the center of the board
364. int squareValuesForKing[] = {1, 1, 1, 1,
365. 3, 5, 5, 3,
366. 7, 9, 9, 7,
367. 8, 13, 13, 8,
368. 8, 13, 13, 8,
369. 7, 9, 9, 7,
370. 3, 5, 5, 3,
371. 1, 1, 1, 1};
372. if (color == Color::RED)
373. {
374. playerColor = Pieces::ANSII\_RED\_START;
375. playerColor.append("RED ");
376. playerColor.append(Pieces::ANSII\_END);
377. }
378. else
379. playerColor = "BLACK ";
380. indentValue = playerColor;
381. if (Pieces::ouputDebugData > 1)
382. std::cout << indentValue << Pieces::ANSII\_GREEN\_COUT << " Evaluating Current Player:  Pieces-> "
383. << Pieces::ANSII\_END << numPieces << " Kings-> " << numKingPieces << " Opponent Pieces-> "
384. << numOpponentPieces << " Kings-> " << numOpponentKingsPieces << " total pieces-> "
385. << totalPieces << std::endl;
386. if (numPieces + numKingPieces == 0) // if the current player is out of pieces this is a terminal state
387. {
388. if (Pieces::ouputDebugData > 1)
389. std::cout << indentValue << Pieces::ANSII\_RED\_COUT << "Current Player is in TERMINAL STATE!! "
390. << Pieces::ANSII\_END << std::endl;
391. currentTerminal = -877775; //  returning a high value.  the current player is about to lose
392. // Avoid this move!
393. }
394. if (numOpponentPieces + numOpponentKingsPieces == 0) // if the current player is out of pieces this is a terminal state
395. {
396. //if (Pieces::ouputDebugData > 1)
397. std::cout << indentValue << Pieces::ANSII\_RED\_COUT << "Opponent is in TERMINAL STATE!!  "
398. << Pieces::ANSII\_END << std::endl;
399. opponentTerminal = 777775; //  The opponent's is about to lose
400. //  Go for it if possible
401. }
402. // Piece preservation and the taking of opponent pieces are  global values. In other words, if no pieces are lost
403. // or taken, all moves that have this result will have the same evaluation value.
404. // This loop looks at each player's piece and the board generates a value that will separate certain moves from
405. // the rest.
406. for (int i = 1; i <= 32; ++i)
407. {
408. // is a player or opponent in this square?  RV (0 = empty, 1 = man, 2 = king)
409. playerPiece = state.getPieceInSquare(i, color);
410. opponentPiece = state.getPieceInSquare(i, switchPlayerColor(color));
411. // get value of current player's board
412. if (playerPiece == KING) // piece is a King
413. {
414. // this encourages a king to move towards the center of the board
415. positionAdder += squareValuesForKing[i - 1] \* 10;
416. // Cross of Pain Calculation
417. // look for opponents in column + or - 1 or row + or - 8.  if true this is a good place to be
418. // Current piece is not on the right edge!
419. if ((i - 1) % 4 != 0 && state.getPieceInSquare((i - 1) % 4, switchPlayerColor(color)) != 0)
420. positionAdder += crossOfPainValue;
421. // Current piece is not on the left edge!  i.e. they are in the same row
422. if (((i - 1) / 4 + 1) == (i / 4 + 1) && state.getPieceInSquare((i + 1) % 4, switchPlayerColor(color)) != 0)
423. positionAdder += crossOfPainValue;
424. // Current piece is not on the top row!
425. if (i - 8 > 0 && state.getPieceInSquare((i - 8), switchPlayerColor(color)) != 0)
426. positionAdder += crossOfPainValue;
427. // Current piece is not on the bottom row!
428. if (i + 8 < 33 && state.getPieceInSquare((i + 8), switchPlayerColor(color)) != 0)
429. positionAdder += crossOfPainValue;
430. }
431. else if (playerPiece == MAN) // Regular pieces are encouraged to move towards the opponent's back row.
432. {
433. if (color == Color::RED)
434. {
435. positionAdder += squareValuesForRed[i - 1];
436. }
437. else if (color == Color::BLACK)
438. {
439. positionAdder += squareValuesForBlack[i - 1];
440. }
441. }
442. //Get value of opponent player's board using the same method as player
443. if (opponentPiece == KING) // piece is a King
444. {
445. // this encourages a king to move towards the center of the board
446. opponentPositionAdder += squareValuesForKing[i - 1] \* 10;
447. // Does the current player end up in the Cross of Pain Calculation
448. // look for opponents in column + or - 1 or row + or - 8.  if true this is a good place to be
449. // Current piece is not on the right edge!
450. if ((i - 1) % 4 != 0 && state.getPieceInSquare((i - 1) % 4, switchPlayerColor(color)) != 0)
451. opponentPositionAdder += crossOfPainValue;
452. // Current piece is not on the left edge!
453. if (((i - 1) / 4 + 1) == (i / 4 + 1) && state.getPieceInSquare((i + 1) % 4, switchPlayerColor(color)) != 0)
454. opponentPositionAdder += crossOfPainValue;
455. // Current piece is not on the top row!
456. if (i - 8 > 0 && state.getPieceInSquare((i - 8), switchPlayerColor(color)) != 0)
457. opponentPositionAdder += crossOfPainValue;
458. // Current piece is not on the bottom row!
459. if (i + 8 < 33 && state.getPieceInSquare((i + 8), switchPlayerColor(color)) != 0)
460. opponentPositionAdder += crossOfPainValue;
461. }
462. else if (playerPiece == MAN) // Opponent's Regular pieces are encouraged to move towards the back row.
463. {
464. if (color == Color::RED)
465. {
466. opponentPositionAdder += squareValuesForRed[i - 1];
467. }
468. else if (color == Color::BLACK)
469. {
470. opponentPositionAdder += squareValuesForBlack[i - 1];
471. }
472. }
473. }
474. playerScore = preservePlayersPieces + positionAdder + currentTerminal;
475. opponentScore = (preserveOpponentsPieces + opponentPositionAdder + reduceOpponentKingValue) \* opponentEvaluationWeight + opponentTerminal;
476. moveScore = playerScore - opponentScore;
477. if (Pieces::ouputDebugData > 1)
478. std::cout << indentValue << Pieces::ANSII\_GREEN\_COUT << " Evaluated Move:  moveScore-> " << Pieces::ANSII\_END
479. << moveScore << std::endl;
480. // Clamp the final score based on the current scoring
481. // system. Note that these values can later be adjusted
482. // to use a wider range.
483. if (moveScore <= -8000000)
484. {
485. moveScore = -7999999;
486. }
487. if (moveScore >= 9000000)
488. {
489. moveScore = 7999999;
490. }
491. return moveScore;
492. }
493. /\*\*
494. \* Third evaluation function
495. \* @author Borislav Sabotinov
496. \*
497. \* @param Board State
498. \* @param Color color
499. \*
500. \* @return an integer score of how good we think the state is
501. \*/
502. int Algorithm::evalFunctThree(Board state, Color color)
503. {
504. /\*  Declaration of "boards" with a weight for each square
505. Two for regular pieces of each color and two for kings of each color
506. Encourage player to keep two pieces in the back for defense
507. leaving 10 pieces for offense. Advance to the center
508. but somewhat in waves as a cluster, to avoid suicidal pieces that expose themselves
509. Try to attack the opponent's "double corner" from where a kinged piece can escape faster
510. \*/
511. int squareValuesForRed[] = {7, 1, 7, 1,
512. 1, 2, 2, 2,
513. 1, 5, 5, 5,
514. 1, 3, 3, 3,
515. 1, 4, 4, 4,
516. 1, 5, 250, 250,
517. 1, 250, 500, 500,
518. 50, 100, 100, 1000};
519. int squareValuesForBlack[] = {1000, 100, 100, 50,
520. 500, 500, 250, 1,
521. 250, 250, 5, 1,
522. 4, 4, 4, 1,
523. 3, 3, 3, 1,
524. 5, 5, 5, 1,
525. 2, 2, 2, 1,
526. 1, 7, 1, 7};
527. // Kings preference for center, with some traversal lines
528. // to attempt and avoid a "back-and-forth" pattern
529. int squareValuesForKing[] = {1, 1, 1, 1,
530. 1, 5, 5, 55,
531. 5, 15, 45, 1,
532. 1, 5, 35, 5,
533. 5, 25, 25, 1,
534. 1, 15, 5, 15,
535. 5, 5, 5, 10,
536. 1, 1, 1, 1};
537. std::string colorTxt = (color == Color::RED) ? " (RED is Friendly) " : " (BLACK is Friendly) ";
538. // KING has 4 moves max, so value is 4; MAN has 2 moves max so values is 2
539. const int KING = 2, MAN = 1, KING\_VALUE = 4, MAN\_VALUE = 2;
540. int numPlayerTotalPieces = state.getNumPlayerTotalPieces(color);
541. int numEnemyTotalPieces = state.getNumPlayerTotalPieces(switchPlayerColor(color));
542. int numPlayerTotalKings = state.getNumKingPieces(color);
543. int numEnemyTotalKings = state.getNumKingPieces(switchPlayerColor(color));
544. int numPlayerTotalMen = numPlayerTotalPieces - numPlayerTotalKings;
545. int numEnemyTotalMen = numEnemyTotalPieces - numEnemyTotalKings;
546. int numKingsScore = numPlayerTotalKings \* KING\_VALUE;
547. int numMenScore = numPlayerTotalMen \* MAN\_VALUE;
548. int diffInNumPieces = numPlayerTotalPieces - numEnemyTotalPieces;
549. int diffInNumKings = numPlayerTotalKings - numEnemyTotalKings;
550. int diffInNumMen = numPlayerTotalMen - numEnemyTotalMen;
551. // PIECE BONUS/PENALTY - UNCOMMENT TO ACTIVATE
552. // if diff in kings is positive, score is amplified with a bonus
553. // if diff in kings is negative, however, score is penalized accordingly (by adding a negative)
554. // numKingsScore += (40 \* diffInNumKings);
555. // if diff in men is positive, score is amplified with a bonus
556. // if diff in men is negative, however, score is penalized accordingly (by adding a negative)
557. // numMenScore += (20 \* diffInNumMen);
558. // END PIECE BONUS/PENALTY
559. int casualtyScore = 0, captureScore = 0, positionScore = 0, playerPiece = 0, enemyPiece = 0, advancementScore = 0;
560. std::vector<Board::Move> playerMoves = state.moveGen(color);
561. std::vector<Board::Move> enemyMoves = state.moveGen(switchPlayerColor(color));
562. Pieces playerPieces = state.getPlayerPieces(color);
563. Pieces opponentPieces = state.getOpponentPieces(color);
564. Pieces \*p\_playerPieces = &playerPieces;
565. Pieces \*p\_opponentPieces = &opponentPieces;
566. std::vector<Board::Move> playerJumpsForPiece;
567. std::vector<Board::Move> opponentJumpsForPiece;
568. // helper values to quickly check if a piece is in a certain notable location
569. // back rows for each color player to determine KING-ing
570. // sides inidcate limited moves
571. const long long redBackRowGrp = (1LL << 1) | (1LL << 2) | (1LL << 3) | (1LL << 4);
572. const long long blackBackRowGrp = (1LL << 29) | (1LL << 30) | (1LL << 31) | (1LL << 32);
573. const long long sideColumnGrp = (1LL << 5) | (1LL << 13) | (1LL << 21) | (1LL << 12) | (1LL << 20) | (1LL << 28);
574. // CHECK TERMINAL STATE
575. if (enemyMoves.size() == 0)
576. return 7999999; // good for us if enemy has no moves left!
577. else if (playerMoves.size() == 0)
578. return -7999999; // bad for us if we're out of moves!
579. // MAIN LOOP FOR SCORING POSITION
580. for (int piece = 0; piece < 32; piece++)
581. {
582. // BEGIN POSITION SCORE SECTION
583. if (color == Color::RED)
584. {
585. playerPiece = state.getPieceInSquare(piece, color);
586. if (playerPiece == MAN)
587. positionScore += (squareValuesForRed[piece] \* MAN\_VALUE);
588. else if (playerPiece == KING)
589. positionScore += (squareValuesForKing[piece] \* KING\_VALUE);
590. }
591. else if (color == Color::BLACK)
592. {
593. playerPiece = state.getPieceInSquare(piece, color);
594. if (playerPiece == MAN)
595. positionScore += (squareValuesForBlack[piece] \* MAN\_VALUE);
596. else if (playerPiece == KING)
597. positionScore += (squareValuesForKing[piece] \* KING\_VALUE);
598. }
599. // END POSITION SCORE SECTION
600. }
601. // Check our moves; 1000 points for a safe capture, 2000 points for a multi-jump
602. for (int i = 0; i < playerMoves.size(); i++)
603. {
604. if (playerMoves.at(i).removalSquare.size() > 1)
605. {
606. if (Pieces::ouputDebugData)
607. std::cout << " INSIDE EVAL-3: We " << colorTxt << " can capture multiple pieces on this state!  "
608. << "Start: " << playerMoves.at(i).startSquare << "End: " << playerMoves.at(i).destinationSquare.back()
609. << " " << std::endl;
610. captureScore += 2000;
611. }
612. else if (playerMoves.at(i).removalSquare.size() == 1)
613. {
614. int enemyCaptureSqr = playerMoves.at(i).removalSquare.back();
615. int enemyCaptureType = state.getPieceInSquare(enemyCaptureSqr, switchPlayerColor(color)); // it's an enemy piece, what is it's type?
616. if (enemyCaptureType == KING)
617. {
618. if (Pieces::ouputDebugData)
619. std::cout << " INSIDE EVAL-3: We can capture enemy KING! " << colorTxt << std::endl;
620. captureScore += 400;
621. }
622. else if (enemyCaptureType == MAN)
623. captureScore += 200;
624. int destSqr = playerMoves.at(i).destinationSquare.back();
625. std::vector<int> adjMoves = state.boardMoveTable[destSqr].moves;
626. if (color == Color::RED)
627. {
628. for (int j = 0; j < adjMoves.size(); j++)
629. {
630. if (adjMoves.at(j) > destSqr) // check enemy MEN and KING below
631. {
632. int enemyPiece = state.getPieceInSquare(adjMoves.at(j), switchPlayerColor(color));
633. if (enemyPiece == MAN || enemyPiece == KING)
634. captureScore -= 100; // not safe
635. }
636. else if (adjMoves.at(j) < destSqr) // we're red, anyting above us can only capture if enemy KING
637. {
638. int enemyPiece = state.getPieceInSquare(adjMoves.at(j), switchPlayerColor(color));
639. if (enemyPiece == KING)
640. captureScore -= 100; // not safe
641. }
642. else
643. captureScore += 1000; // we're safe to capture
644. }
645. }
646. else
647. {
648. for (int j = 0; j < adjMoves.size(); j++)
649. {
650. if (adjMoves.at(j) < destSqr) // check enemy MEN and KING above
651. {
652. int enemyPiece = state.getPieceInSquare(adjMoves.at(j), switchPlayerColor(color));
653. if (enemyPiece == MAN || enemyPiece == KING)
654. captureScore -= 100; // not safe
655. }
656. else if (adjMoves.at(j) > destSqr) // we're black, anyting below us can only capture if enemy KING
657. {
658. int enemyPiece = state.getPieceInSquare(adjMoves.at(j), switchPlayerColor(color));
659. if (enemyPiece == KING)
660. captureScore -= 100; // not safe
661. }
662. captureScore += 1000; // we're safe to capture
663. }
664. }
665. }
666. }
667. // BEGIN CASUALTY SECTION
668. for (int j = 0; j < enemyMoves.size(); j++)
669. {
670. //std::cout << "we are here" << std::endl;
671. if (enemyMoves.at(j).removalSquare.size() > 1)
672. {
673. if (Pieces::ouputDebugData)
674. std::cout << " INSIDE EVAL-3: Enemy can capture multiple pieces, avoid!" << colorTxt << std::endl;
675. casualtyScore -= 40000; // we lose too much, really bad
676. }
677. else if (enemyMoves.at(j).removalSquare.size() == 1)
678. {
679. // friendly piece is captured, what is it's type?
680. int capturedPieceType = state.getPieceInSquare(enemyMoves.at(j).removalSquare.at(0), color);
681. if (capturedPieceType == KING)
682. {
683. if (Pieces::ouputDebugData)
684. std::cout << " INSIDE EVAL-3: Enemy can capture a KING, avoid!" << colorTxt << std::endl;
685. casualtyScore -= 4000; // we lose a KING, a valuable piece
686. }
687. else if (capturedPieceType == MAN) // we lose one MAN
688. {
689. int opponentDestinationSqr = enemyMoves.at(j).destinationSquare.at(0);
690. std::vector<int> adjMoves = state.boardMoveTable[opponentDestinationSqr].moves;
691. if (color == Color::RED)
692. {
693. // if we are RED, opponent is Black; if BLACK enemy lands on our back row, avoid at all cost
694. // we are therefore trying to minimize the chance of an enemy getting a KING
695. if ((1 << opponentDestinationSqr) & redBackRowGrp)
696. casualtyScore -= 5000;
697. // opponent lands on their own back row; not so bad but we can't retaliate so avoid
698. else if ((1 << opponentDestinationSqr) & blackBackRowGrp)
699. casualtyScore -= 2000;
700. // opponent lands on one of the side squares, where we cannot retaliate. Avoid as well
701. else if ((1 << opponentDestinationSqr) & sideColumnGrp)
702. casualtyScore -= 2000;
703. else
704. {
705. // if (diffInNumMen >= 1)
706. // {
707. for (int j = 0; j < adjMoves.size(); j++)
708. {
709. int ourPiece = state.getPieceInSquare(adjMoves.at(j), switchPlayerColor(color));
710. // check if we have a king below; we're red, only our king can go upwards
711. if (adjMoves.at(j) > opponentDestinationSqr)
712. {
713. if (ourPiece == KING)
714. captureScore += 1000; // can retaliate
715. }
716. // we're red, we can retaliate with MAN or KING if enemy is above
717. else if (adjMoves.at(j) < opponentDestinationSqr)
718. {
719. if (ourPiece == MAN || ourPiece == KING)
720. captureScore += 1000; // can retaliate
721. }
722. else
723. captureScore += 0; // we cannot capture
724. }
725. // }
726. }
727. }
728. else
729. {
730. // if we are BLACK, opponent is Red; if RED enemy lands on our back row, avoid at all cost
731. // we are therefore trying to minimize the chance of an enemy getting a KING
732. if ((1 << opponentDestinationSqr) & blackBackRowGrp)
733. casualtyScore -= 5000;
734. // opponent lands on their own back row; not so bad but we can't retaliate so avoid
735. else if ((1 << opponentDestinationSqr) & redBackRowGrp)
736. casualtyScore -= 2000;
737. // opponent lands on one of the side squares, where we cannot retaliate. Avoid as well
738. else if ((1 << opponentDestinationSqr) & sideColumnGrp)
739. casualtyScore -= 2000;
740. // if we've gotten this far, we lose one MAN and opponent lands somewhere we can retaliate
741. // We ask - Can we? If yes, do it if we have piece parity or an advantage of more pieces
742. else
743. {
744. // if (numPlayerTotalPieces >= numEnemyTotalPieces)
745. // {
746. for (int j = 0; j < adjMoves.size(); j++)
747. {
748. int ourPiece = state.getPieceInSquare(adjMoves.at(j), switchPlayerColor(color));
749. if (adjMoves.at(j) < opponentDestinationSqr) // check our KING above
750. {
751. if (ourPiece == KING)
752. captureScore += 1000; // can retaliate
753. }
754. // we're black, if enemy is below we can retaliate with MAN or KING
755. else if (adjMoves.at(j) > opponentDestinationSqr)
756. {
757. if (ourPiece == MAN || ourPiece == KING)
758. captureScore += 1000; // can retaliate
759. }
760. captureScore += 0; // we cannot retaliate
761. }
762. // }
763. }
764. }
765. }
766. }
767. }
768. //END CASUALTY SECTION
769. int compositeScore = numMenScore + numKingsScore + advancementScore + positionScore + captureScore + casualtyScore;
770. return compositeScore;
771. }
772. /\*\*
773. \* wrapper function that will decide which of the actual three eval functions to call
774. \* @author Borislav Sabotinov
775. \*
776. \* @param Board position
777. \* @param Player player
778. \* @param int evalVersion - used to determine which of the 3 eval functions to call
779. \*
780. \* @return a Result struct, which consists of a value and a move.
781. \*/
782. int Algorithm::staticEval(Board state, Color color, int evalVersion)
783. {
784. int scoreOfGoodness;
785. switch (evalVersion)
786. {
787. case 1:
788. scoreOfGoodness = evalFunctOne(state, color);
789. break;
790. case 2:
791. scoreOfGoodness = evalFunctTwo(state, color);
792. break;
793. case 3:
794. scoreOfGoodness = evalFunctThree(state, color);
795. break;
796. default:
797. {
798. scoreOfGoodness = 1; // default and debug value.  Player takes first option everytime
799. if (Pieces::ouputDebugData > 1)
800. std::cout << Pieces::ANSII\_GREEN\_COUT << " TEST/DEBUG EVALUATION BRANCH RETURN VALUE = 1 " << Pieces::ANSII\_END << std::endl;
801. //throw std::runtime\_error("Error: eval function # may only be 1, 2, or 3!");
802. }
803. }
804. return scoreOfGoodness;
805. }
806. /\*\*
807. \* deepEnough - Basic if check, if currentDepth >= maxDepth, true; otherwise false
808. \*
809. \* @author Randall Henderson
810. \*
811. \*/
812. bool Algorithm::deepEnough(int currentDepth)
813. {
814. if (currentDepth <= 0)
815. return true;
816. else
817. return false;
818. }
819. /\*\*
820. \* minimax algorithm returns the position of the best move
821. \* @author Randall Henderson
822. \*
823. \* @param Board board
824. \* @param int depth
825. \* @param Color color
826. \* @param int passThresh = 9000000
827. \* @param int useThresh = -8000000
828. \*
829. \* @return a Result struct, which consists of a value and a Move
830. \*/
831. Algorithm::Result Algorithm::minimax\_a\_b(Board state, int depth, Color color, int useThresh, int passThresh)
832. {
833. Algorithm::Result result; // Return structure for MiniMaxAB
834. std::string indentValue;  // Tracking header in cout statments.  Shows level in recursion
835. std::string playerColor;  // Used for debugging ouput
836. Board::Move bestPath;     // best move struct - starts as a null move
837. int newValue;
838. if (color == Color::RED)
839. {
840. playerColor = Pieces::ANSII\_RED\_START;
841. playerColor.append("  RED ");
842. playerColor.append(Pieces::ANSII\_END);
843. }
844. else
845. {
846. playerColor = "BLACK ";
847. }
848. indentValue = playerColor;
849. indentValue.append("1.");
850. //   Debug code for values passed into the function
851. if (Pieces::ouputDebugData > 0 && depth == 0)
852. std::cout << indentValue << Pieces::ANSII\_GREEN\_COUT << "Entering MINIMAX\_A\_B Value:  " << result.value
853. << " Depth-> " << depth << " useThresh-> " << useThresh << " passThresh-> " << passThresh
854. << Pieces::ANSII\_END << std::endl;
855. indentValue.append("2.");
856. if (deepEnough(depth)) // deep enough and Terminal could be combined.  Separated for error tracking
857. {
858. result.value = staticEval(state, color, evalVersion);
859. if (Pieces::ouputDebugData > 0)
860. std::cout << indentValue << Pieces::ANSII\_RED\_COUT << "Deep Enough, Move Evaluated.  Returning -> "
861. << result.value << " Bestmove destination size " << result.bestMove.destinationSquare.size() << Pieces::ANSII\_END << std::endl;
862. minimaxLeafNodes++;
863. // to deal with odd depths and keep it in the right order
864. if (callingPlayer.getColor() != color)
865. result.value = -result.value;
866. return result;
867. }
868. indentValue.append("3.");
869. std::vector<Board::Move> successors = movegen(state, color);
870. //  Current Player has no moves.  This is the equivalent to deep enough or terminal move
871. if (successors.size() == 0)
872. {
873. result.value = staticEval(state, color, evalVersion);
874. if (Pieces::ouputDebugData)
875. std::cout << indentValue << Pieces::ANSII\_RED\_COUT << "Player has no moves.  Returning -> "
876. << result.value << Pieces::ANSII\_END << std::endl;
877. minimaxLeafNodes++;
878. // to deal with odd depths and keep it in the right order
879. if (callingPlayer.getColor() != color)
880. result.value = -result.value;
881. return result;
882. }
883. for (int successorIndex = 0; successorIndex < successors.size(); successorIndex++)
884. {
885. //Create a board at the current iteration of successors
886. Board tmpState = state.updateBoard(successors.at(successorIndex), color);
887. minimaxExpandedNodes++;
888. indentValue.append(">.");
889. if (Pieces::ouputDebugData)
890. std::cout << indentValue << Pieces::ANSII\_BLUE\_COUT << "Checking Moves:  Move #-> " << successorIndex + 1
891. << " Passed in parameters:  Start-> " << successors.at(successorIndex).startSquare << " Move to-> "
892. << successors.at(successorIndex).destinationSquare.back() << " Depth-> " << depth
893. << Pieces::ANSII\_END << std::endl;
894. // recursive call
895. Result resultSucc = minimax\_a\_b(tmpState, depth - 1, switchPlayerColor(color), -passThresh, -useThresh);
896. if (Pieces::ouputDebugData)
897. std::cout << indentValue << Pieces::ANSII\_GREEN\_START << "Recursive Return:  Just checked-> "
898. << successors.at(successorIndex).startSquare << " -> "
899. << successors.at(successorIndex).destinationSquare.back()
900. << " New Value-> " << resultSucc.value << " Depth-> "
901. << depth << " useThresh-> " << useThresh << " passThresh-> " << passThresh
902. << Pieces::ANSII\_END << std::endl;
903. newValue = -resultSucc.value;
904. if (newValue > passThresh) // Found the Best Move
905. {
906. if (Pieces::ouputDebugData > 0 && depth == this->maxDepth)
907. std::cout << indentValue << Pieces::ANSII\_YELLOW\_COUT << "New Best Move.  From-> " << successors.at(successorIndex).startSquare
908. << " to-> " << successors.at(successorIndex).destinationSquare.back()
909. << " Change PassThresh  Old: " << passThresh
910. << " to  New: " << newValue << Pieces::ANSII\_END << std::endl;
911. passThresh = newValue;
912. bestPath = successors.at(successorIndex);
913. }
914. if (passThresh >= useThresh) // Best move on the branch.  No need to look anymore
915. {
916. if (Pieces::ouputDebugData)
917. std::cout << indentValue << Pieces::ANSII\_YELLOW\_COUT << "AB-CUTOFF!!  Best Move on the Branch.  PassThresh -> "
918. << Pieces::ANSII\_END << passThresh << " UseThresh-> " << useThresh << " Returning " << std::endl;
919. result.value = passThresh;
920. result.bestMove = successors.at(successorIndex);
921. return result;
922. }
923. }
924. result.value = passThresh;
925. result.bestMove = bestPath;
926. return result;
927. }
928. /\*\*
929. \* Alpha Beta Search
930. \* @author Borislav Sabotinov
931. \*
932. \* @param Board state
933. \*
934. \* @return a Result struct, which consists of a value and a Move
935. \*/
936. Algorithm::Result Algorithm::alphaBetaSearch(Board state)
937. {
938. if (Pieces::ouputDebugData)
939. {
940. if (callingPlayer.getColor() == Color::RED)
941. std::cout << "RED ";
942. else
943. std::cout << "BLACK ";
944. if (Pieces::ouputDebugData > 2)
945. std::cout << "In alphaBetaSearch...." << std::endl;
946. }
947. int alpha = std::numeric\_limits<int>::min(); // tracks best value for max, initialized to WORST case
948. int beta = std::numeric\_limits<int>::max();  // tracks best value for min, initialized to WORST case
949. return maxValue(state, maxDepth, alpha, beta, callingPlayer.getColor());
950. }
951. /\*\*
952. \* maxValue - algorithm from Russel & Norvig, implemented to fit this program
953. \* This function would determine the score of goodness for a state passed in to the MAX player
954. \* There is indirect recursion as MAX calls MIN, which calls MAX again and so on.
955. \* @author Borislav Sabotinov
956. \*
957. \* @param Board state
958. \* @param int alpha
959. \* @param int beta
960. \*
961. \* @return int utilityValue
962. \*/
963. Algorithm::Result Algorithm::maxValue(Board state, int depth, int alpha, int beta, Color color)
964. {
965. if (Pieces::ouputDebugData)
966. {
967. if (color == Color::RED)
968. std::cout << "\nRED ";
969. else
970. std::cout << "\nBLACK ";
971. std::cout << "In maxValue()! Depth is " << depth << std::endl;
972. }
973. Algorithm::Result result;
974. Board::Move bestMove;
975. if (deepEnough(depth))
976. {
977. absearchLeafNodes++;
978. result.value = staticEval(state, color, evalVersion);
979. return result;
980. }
981. std::vector<Board::Move> listOfActions = actions(state, color);
982. // terminal check
983. if (listOfActions.size() == 0)
984. {
985. absearchLeafNodes++;
986. result.value = staticEval(state, color, evalVersion); // eval acts as utility funct
987. return result;
988. }
989. if (Pieces::ouputDebugData)
990. std::cout << "Not yet at a terminal state...." << std::endl;
991. result.value = std::numeric\_limits<int>::min();
992. for (int actionIndex = 0; actionIndex < listOfActions.size(); actionIndex++)
993. {
994. Player::printMove(listOfActions.at(actionIndex), color, false);
995. absearchExpandedNodes++;
996. Board tmpState = state.updateBoard(listOfActions.at(actionIndex), color);
997. Algorithm::Result minValueResult = minValue(tmpState, depth - 1, alpha, beta, switchPlayerColor(color));
998. if (minValueResult.value > result.value) // Best move located
999. {
1000. result.value = minValueResult.value;
1001. bestMove = listOfActions.at(actionIndex);
1002. }
1003. if (result.value >= beta) // no need to examine branch
1004. {
1005. result.bestMove = listOfActions.at(actionIndex);
1006. return result;
1007. }
1008. alpha = std::max(alpha, result.value);
1009. }
1010. if (Pieces::ouputDebugData)
1011. {
1012. std::cout << "alpha: " << alpha << " beta: " << beta << " val: " << result.value << " move start: " << result.bestMove.startSquare << std::endl;
1013. for (int i = 0; i < result.bestMove.destinationSquare.size(); i++)
1014. std::cout << "dest: " << result.bestMove.destinationSquare.at(i) << std::endl;
1015. }
1016. result.bestMove = bestMove;
1017. return result;
1018. }
1019. /\*\*
1020. \* minValue function - algorithm from Russel & Norvig, implemented to fit this program
1021. \* This function would determine the score of goodness for a state passed in to the MIN player
1022. \* There is indirect recursion as MAX calls MIN, which calls MAX again and so on.
1023. \* @author Borislav Sabotinov
1024. \*
1025. \* @param Board state
1026. \* @param int alpha
1027. \* @param int beta
1028. \*
1029. \* @return Result structure, which contains a value score and a Board::Move bestMove structure
1030. \*/
1031. Algorithm::Result Algorithm::minValue(Board state, int depth, int alpha, int beta, Color color)
1032. {
1033. if (Pieces::ouputDebugData)
1034. {
1035. if (color == Color::RED)
1036. std::cout << "\nRED ";
1037. else
1038. std::cout << "\nBLACK ";
1039. std::cout << "In minValue()! Depth is " << depth << std::endl;
1040. }
1041. Result result;
1042. Board::Move bestMove;
1043. if (deepEnough(depth))
1044. {
1045. absearchLeafNodes++;
1046. result.value = staticEval(state, color, evalVersion);
1047. return result;
1048. }
1049. std::vector<Board::Move> listOfActions = actions(state, color);
1050. // terminal check
1051. if (listOfActions.size() == 0)
1052. {
1053. absearchLeafNodes++;
1054. result.value = staticEval(state, color, evalVersion); // eval acts as utility funct
1055. return result;
1056. }
1057. if (Pieces::ouputDebugData)
1058. std::cout << "Not yet at a terminal state...." << std::endl;
1059. result.value = std::numeric\_limits<int>::max();
1060. for (int actionIndex = 0; actionIndex < listOfActions.size(); actionIndex++)
1061. {
1062. Player::printMove(listOfActions.at(actionIndex), color, false);
1063. absearchExpandedNodes++;
1064. Board tmpState = state.updateBoard(listOfActions.at(actionIndex), color);
1065. Result maxValueResult = maxValue(tmpState, depth - 1, alpha, beta, switchPlayerColor(color));
1066. if (maxValueResult.value < result.value) // Best move located
1067. {
1068. result.value = maxValueResult.value;
1069. bestMove = listOfActions.at(actionIndex);
1070. }
1071. if (result.value <= alpha)
1072. {
1073. result.bestMove = listOfActions.at(actionIndex);
1074. return result;
1075. }
1076. beta = std::min(beta, result.value);
1077. }
1078. if (Pieces::ouputDebugData)
1079. {
1080. std::cout << "alpha: " << alpha << " beta: " << beta << " val: " << result.value << " move start: " << result.bestMove.startSquare << std::endl;
1081. for (int i = 0; i < result.bestMove.destinationSquare.size(); i++)
1082. std::cout << "dest: " << result.bestMove.destinationSquare.at(i) << std::endl;
1083. }
1084. result.bestMove = bestMove;
1085. return result;
1086. }
1087. /\*\*
1088. \* Helper function to switch a color, so if we are RED and we want our opponent, we would get BACK
1089. \* @author Borislav Sabotinov
1090. \*
1091. \* @param Color color - the color we want to invert
1092. \*
1093. \* @return Color - the color opposite of what we passed in
1094. \*/
1095. Color Algorithm::switchPlayerColor(Color color)
1096. {
1097. if (color == Color::RED)
1098. return Color::BLACK;
1099. else
1100. return Color::RED;
1101. }
1102. /\*\*
1103. \* terminalTest function is used by alpha-beta-search to determine if
1104. \* the move either ends the game or leads to a leaf node.
1105. \* @author Borislav Sabotinov
1106. \*
1107. \* First checks if we're at maxDepth (i.e. at a leaf/terminal node).
1108. \* If yes, return true. Otherwise, next we check if the move ends the game.
1109. \* For example - if Red player ends up with zero pieces, they lost and the game is over.
1110. \*
1111. \* @param Board state
1112. \*
1113. \* @return bool isTerminalState
1114. \*/
1115. bool Algorithm::terminalTest(Board state, int depth)
1116. {
1117. bool isTerminalState = false;
1118. std::vector<Board::Move> redMoves = state.moveGen(Color::RED);
1119. std::vector<Board::Move> blackMoves = state.moveGen(Color::BLACK);
1120. if (Pieces::ouputDebugData)
1121. std::cout << "Red Moves " << redMoves.size() << "  Black Moves  " << blackMoves.size() << std::endl;
1122. if (redMoves.size() == 0 || blackMoves.size() == 0)
1123. isTerminalState = true;
1124. return isTerminalState;
1125. }
1126. /\*\*
1127. \* Utility function to determine the best move
1128. \* Essentially a wrapper function that calls staticEval
1129. \* @author Borislav Sabotinov
1130. \*/
1131. int Algorithm::utility(Board state)
1132. {
1133. //return staticEval(state, this->callingPlayer, this->evalVersion);
1134. return 1;
1135. }
1136. /\*\*
1137. \* actions method determines the list of possible actions, or moves, a player can make
1138. \* Essentially a wrapper function to call Board's movegen function.
1139. \* @author Borislav Sabotinov
1140. \*
1141. \* @param Board state
1142. \*
1143. \* @return vector<Board::Move> - a list of possible moves for a player of a given color
1144. \*/
1145. std::vector<Board::Move> Algorithm::actions(Board state, Color color)
1146. {
1147. return state.moveGen(color);
1148. }
1149. /\*\*
1150. \* Set the evaluation function version - 1, 2, or 3
1151. \*/
1152. void Algorithm::setEvalVersion(int evalVersion)
1153. {
1154. this->evalVersion = evalVersion;
1155. }
1156. /\*\*
1157. \* Set the max depth that will be used
1158. \*/
1159. void Algorithm::setMaxDepth(int maxDepth)
1160. {
1161. this->maxDepth = maxDepth;
1162. }
1163. int Algorithm::passSign(int passThresh)
1164. {
1165. if (passThresh < 0)
1166. return -1;
1167. else
1168. return 1;
1169. }

## Simulation.hpp

1. #ifndef SIMULATION\_H
2. #define SIMULATION\_H
3. #include "Board.hpp"
4. #include "Game.hpp"
5. /\*\*
6. \* Header definition for class Simulation.
7. \* @author Borislav Sabotinov
8. \*
9. \* This class is responsible for managing the series of Games that AI players will play for Project Two.
10. \* It persists during the execution of the program and keeps track of the number of games played.
11. \* It also allows to aggregate and print simulation analysis details.
12. \*
13. \* There are THREE (3) evaluation functions, one for each team member; two algorithms (minimax and Alpha-Beta).
14. \* The simulation will execute each in turn.
15. \*
16. \* Fifteen runs with depth 2:
17. \*    1. MinMax-A-B with Evl. Function #1  Verses Alpha-Beta with Evl. Function #1
18. \*    2. MinMax-A-B with Evl. Function #1  Verses Alpha-Beta with Evl. Function #2
19. \*    3. MinMax-A-B with Evl. Function #1  Verses Alpha-Beta with Evl. Function #3
20. \*
21. \*    4. MinMax-A-B with Evl. Function #2  Verses Alpha-Beta with Evl. Function #1
22. \*    5. MinMax-A-B with Evl. Function #2  Verses Alpha-Beta with Evl. Function #2
23. \*    6. MinMax-A-B with Evl. Function #2  Verses Alpha-Beta with Evl. Function #3
24. \*
25. \*    7. MinMax-A-B with Evl. Function #3  Verses Alpha-Beta with Evl. Function #1
26. \*    8. MinMax-A-B with Evl. Function #3  Verses Alpha-Beta with Evl. Function #2
27. \*    9. MinMax-A-B with Evl. Function #3  Verses Alpha-Beta with Evl. Function #3
28. \*
29. \*   10. MinMax-A-B with Evl. Function #1  Verses MinMax-A-B with Evl. Function #2
30. \*   11. MinMax-A-B with Evl. Function #1  Verses MinMax-A-B with Evl. Function #3
31. \*   12. MinMax-A-B with Evl. Function #2  Verses MinMax-A-B with Evl. Function #3
32. \*
33. \*   13. Alpha-Beta with Evl. Function #1  Verses Alpha-Beta with Evl. Function #2
34. \*   14. Alpha-Beta with Evl. Function #1  Verses Alpha-Beta with Evl. Function #3
35. \*   15. Alpha-Beta with Evl. Function #2  Verses Alpha-Beta with Evl. Function #3
36. \*
37. \* Fifteen runs with depth 4:
38. \*    1. MinMax-A-B with Evl. Function #1  Verses Alpha-Beta with Evl. Function #1
39. \*    2. MinMax-A-B with Evl. Function #1  Verses Alpha-Beta with Evl. Function #2
40. \*    3. MinMax-A-B with Evl. Function #1  Verses Alpha-Beta with Evl. Function #3
41. \*
42. \*    4. MinMax-A-B with Evl. Function #2  Verses Alpha-Beta with Evl. Function #1
43. \*    5. MinMax-A-B with Evl. Function #2  Verses Alpha-Beta with Evl. Function #2
44. \*    6. MinMax-A-B with Evl. Function #2  Verses Alpha-Beta with Evl. Function #3
45. \*
46. \*    7. MinMax-A-B with Evl. Function #3  Verses Alpha-Beta with Evl. Function #1
47. \*    8. MinMax-A-B with Evl. Function #3  Verses Alpha-Beta with Evl. Function #2
48. \*    9. MinMax-A-B with Evl. Function #3  Verses Alpha-Beta with Evl. Function #3
49. \*
50. \*   10. MinMax-A-B with Evl. Function #1  Verses MinMax-A-B with Evl. Function #2
51. \*   11. MinMax-A-B with Evl. Function #1  Verses MinMax-A-B with Evl. Function #3
52. \*   12. MinMax-A-B with Evl. Function #2  Verses MinMax-A-B with Evl. Function #3
53. \*
54. \*   13. Alpha-Beta with Evl. Function #1  Verses Alpha-Beta with Evl. Function #2
55. \*   14. Alpha-Beta with Evl. Function #1  Verses Alpha-Beta with Evl. Function #3
56. \*   15. Alpha-Beta with Evl. Function #2  Verses Alpha-Beta with Evl. Function #3
57. \*
58. \* TOTAL 30 RUNS/GAMES WILL BE SIMULATED.
59. \*/
60. class Simulation
61. {
62. private:
63. int numGamesPlayed;
64. // runs only games using Minimax algorithm
65. void runMinimaxOnly();
66. // runs only games using AB Prune algorithm
67. void runABPruneOnly();
68. public:
69. Simulation();  // constructor
70. ~Simulation(); // destructor
71. // runs all games runs as delineated above
72. void runFullSimulation();
73. // public method for specific simulations
74. void runSpecificSimulation(int playerOneAlg, int playerOneEvalFunct, int playerTwoAlg, int PlayerTwoEvalFunct, int depth);
75. // public method for player vs AI simulation
76. void runPlayerVsAISimulation(int playerAlg, int playerEvalFunct, int depth);
77. // helper function to determine winner and break out of game loop
78. static bool didSomeoneWin(Board board);
79. // returns a count of the number of games played in a simulation
80. // each of the 3 run functions.
81. int getNumGamesPlayed();
82. // creates a table with results for analysis.
83. // how many nodes were created, etc.
84. void generateAnalysisResults();
85. void printGameConfig(int redPlayerAlg, int redPlayerEvalFunct, int blackPlayerAlg, int blackPlayerEvalFunct, int depth);
86. void printGameResults(Game::GameOver endGameStatus);
87. // helper print methods
88. static void printBlackWins();
89. static void printRedWins();
90. static void printDraw();
91. };
92. #endif // !SIMULATION\_H

## Simulation.cpp

1. #include "Simulation.hpp"
2. #include <iostream>
3. #include <stdexcept>
4. /\*\*
5. \* Simulation implementation
6. \* @author Borislav Sabotinov
7. \*
8. \* Responsible for driving the simulation based on user preference provided in Main
9. \*/
10. Simulation::Simulation()
11. {
12. this->numGamesPlayed = 0;
13. }
14. Simulation::~Simulation()
15. {
16. }
17. /\*\*
18. \* Runs all games runs as delineated in Simulation.hpp
19. \* R = redundant game simulated
20. \*
21. \* 1.  p1\_alg: 1 p1\_eval: 1 p2\_alg: 1 p2\_eval: 1  R
22. \* 2.  p1\_alg: 1 p1\_eval: 1 p2\_alg: 1 p2\_eval: 2
23. \* 3.  p1\_alg: 1 p1\_eval: 1 p2\_alg: 1 p2\_eval: 3
24. \* 4.  p1\_alg: 1 p1\_eval: 2 p2\_alg: 1 p2\_eval: 1  R
25. \* 5.  p1\_alg: 1 p1\_eval: 2 p2\_alg: 1 p2\_eval: 2  R
26. \* 6.  p1\_alg: 1 p1\_eval: 2 p2\_alg: 1 p2\_eval: 3
27. \* 7.  p1\_alg: 1 p1\_eval: 3 p2\_alg: 1 p2\_eval: 1  R
28. \* 8.  p1\_alg: 1 p1\_eval: 3 p2\_alg: 1 p2\_eval: 2  R
29. \* 9.  p1\_alg: 1 p1\_eval: 3 p2\_alg: 1 p2\_eval: 3
30. \* 10. p1\_alg: 1 p1\_eval: 1 p2\_alg: 2 p2\_eval: 1
31. \* 11. p1\_alg: 1 p1\_eval: 1 p2\_alg: 2 p2\_eval: 2
32. \* 12. p1\_alg: 1 p1\_eval: 1 p2\_alg: 2 p2\_eval: 3
33. \* 13. p1\_alg: 1 p1\_eval: 2 p2\_alg: 2 p2\_eval: 1
34. \* 14. p1\_alg: 1 p1\_eval: 2 p2\_alg: 2 p2\_eval: 2
35. \* 15. p1\_alg: 1 p1\_eval: 2 p2\_alg: 2 p2\_eval: 3
36. \* 16. p1\_alg: 1 p1\_eval: 3 p2\_alg: 2 p2\_eval: 1
37. \* 17. p1\_alg: 1 p1\_eval: 3 p2\_alg: 2 p2\_eval: 2
38. \* 18. p1\_alg: 1 p1\_eval: 3 p2\_alg: 2 p2\_eval: 3
39. \* 19. p1\_alg: 2 p1\_eval: 1 p2\_alg: 2 p2\_eval: 1  R
40. \* 20. p1\_alg: 2 p1\_eval: 1 p2\_alg: 2 p2\_eval: 2  R
41. \* 21. p1\_alg: 2 p1\_eval: 1 p2\_alg: 2 p2\_eval: 3
42. \* 22. p1\_alg: 2 p1\_eval: 2 p2\_alg: 2 p2\_eval: 1
43. \* 23. p1\_alg: 2 p1\_eval: 2 p2\_alg: 2 p2\_eval: 2
44. \* 24. p1\_alg: 2 p1\_eval: 2 p2\_alg: 2 p2\_eval: 3
45. \* 25. p1\_alg: 2 p1\_eval: 3 p2\_alg: 2 p2\_eval: 1  R
46. \* 26. p1\_alg: 2 p1\_eval: 3 p2\_alg: 2 p2\_eval: 2
47. \* 27. p1\_alg: 2 p1\_eval: 3 p2\_alg: 2 p2\_eval: 3  R
48. \*
49. \*/
50. void Simulation::runFullSimulation()
51. {
52. std::cout << "\033[0;32mRunning a FULL simulation!\033[0m" << std::endl;
53. for (int depth = 2; depth <= 4; depth += 2)
54. {
55. std::cout << "Depth: " << depth << std::endl;
56. for (int p1\_alg = 0; p1\_alg < 2; p1\_alg++)
57. {
58. for (int p2\_alg = 0; p2\_alg < 2; p2\_alg++)
59. {
60. for (int p1\_eval = 1; p1\_eval < 4; p1\_eval++)
61. {
62. for (int p2\_eval = 1; p2\_eval < 4; p2\_eval++)
63. {
64. // omit duplicates to save time - we only care about unique runs
65. // player 1 need not use alg 2. It's already covered by Player 2
66. if (p1\_alg == 2 && p2\_alg == 1)
67. continue;
68. std::cout << "p1\_alg: " << p1\_alg << " p1\_eval: " << p1\_eval << " p2\_alg: "
69. << p2\_alg << " p2\_eval: " << p2\_eval << std::endl;
70. Game \*game = new Game(p1\_alg, p1\_eval, p2\_alg, p2\_eval, depth);
71. Game::GameOver endGameStatus = game->startGame();
72. numGamesPlayed++;
73. printGameResults(endGameStatus);
74. delete game;
75. } // p2\_eval
76. }     // p1\_eval
77. }         // p2\_alg
78. }             // p1\_alg
79. }                 // depth
80. }
81. /\*\*
82. \* Allows the user to run a specific, custom simulation based on their preference
83. \* @param int redPlayerAlg - If 1, minimax; if 0, AB Prune
84. \* @param int redPlayerEvalFunct - 1,2, 3, or 4
85. \* @param int blackPlayerAlg - If 1, minimax; if 0, AB Prune
86. \* @param int blackPlayerEvalFunct - 1,2, 3, or 4
87. \* @param int depth - 2 to 15, preferrably 2 or 4 as per project requirements
88. \*/
89. void Simulation::runSpecificSimulation(int redPlayerAlg, int redPlayerEvalFunct, int blackPlayerAlg, int blackPlayerEvalFunct, int depth)
90. {
91. std::cout << Pieces::ANSII\_GREEN\_START << "Running a SINGLE game, specific simulation!" << Pieces::ANSII\_END << std::endl;
92. // Validate algorithm selections
93. if ((redPlayerAlg < 0 || redPlayerAlg > 1) && (blackPlayerAlg < 0 || blackPlayerAlg > 1))
94. throw std::runtime\_error("Error: algorithm may only be 1 (minimax-a-b) or 0 (ab-prune)!");
95. // Validate evaluation function selections
96. if ((redPlayerEvalFunct <= 0 || redPlayerEvalFunct > 4) && (blackPlayerEvalFunct <= 0 || blackPlayerEvalFunct > 4))
97. throw std::runtime\_error("Error: evalFunction may only be 1, 2, 3, or 4!");
98. // Validate depth
99. if (depth <= 1 || depth > 15)
100. throw std::runtime\_error("Error: depth must be > 1 and <= 15. ");
101. Game \*game = new Game(redPlayerAlg, redPlayerEvalFunct, blackPlayerAlg, blackPlayerEvalFunct, depth);
102. Game::GameOver endGameStatus = game->startGame();
103. printGameResults(endGameStatus);
104. printGameConfig(redPlayerAlg, redPlayerEvalFunct, blackPlayerAlg, blackPlayerEvalFunct, depth);
105. delete game;
106. }
107. /\*\*
108. \* Helper function to print overall Game results, depending on who won
109. \* @param Game::GameOver endGameStatus - the game will return to us who was the winner (if any)
110. \*/
111. void Simulation::printGameResults(Game::GameOver endGameStatus)
112. {
113. if (endGameStatus == Game::GameOver::BLACK\_WINS)
114. printBlackWins();
115. else if (endGameStatus == Game::GameOver::RED\_WINS)
116. printRedWins();
117. else if (endGameStatus == Game::GameOver::DRAW)
118. printDraw();
119. else
120. std::cout << "Oops, something went wrong!" << std::endl;
121. }
122. /\*\*
123. \* runPlayerVsAISimulation - play a game with a human against a computer player
124. \* This is a fun option, it allows a person to test themselves against the AI
125. \* @param int playerAlg - what alg will the AI use
126. \* @param int playerEvalFunct - what eval function will AI use
127. \* @param int depth - what depth will AI use
128. \*/
129. void Simulation::runPlayerVsAISimulation(int playerAlg, int playerEvalFunct, int depth)
130. {
131. Player computerPlayer = Player(playerAlg, Color::RED, depth, playerEvalFunct);
132. bool gameOver = false;
133. int moveSelection;
134. Color computerPlayerColor = Color::RED;
135. Color humanPlayerColor = Color::BLACK;
136. Color currentPlayerColor = humanPlayerColor;
137. Board board;
138. board.printBoard();
139. while (!gameOver)
140. {
141. if (currentPlayerColor == humanPlayerColor) // BLACK
142. {
143. std::vector<Board::Move> blackMoves = board.moveGen(humanPlayerColor);
144. // PRINT OUT BLACK'S MOVES
145. std::cout << "Black's moves (b/B): ";
146. for (int blackMoveIter = 0; blackMoveIter < blackMoves.size(); blackMoveIter++)
147. {
148. std::cout << "<" << blackMoveIter + 1 << "> " << blackMoves.at(blackMoveIter).startSquare;
149. for (int destinationIter = 0; destinationIter < blackMoves.at(blackMoveIter).destinationSquare.size(); destinationIter++)
150. {
151. std::cout << " to " << blackMoves.at(blackMoveIter).destinationSquare.at(destinationIter);
152. }
153. std::cout << ", ";
154. }
155. std::cout << std::endl;
156. // GET HUMAN PLAYER MOVE
157. bool isSelectionValid = false;
158. while (!isSelectionValid)
159. {
160. std::cout << "Select BLACK (Human) move: ";
161. std::cin >> moveSelection;
162. if (moveSelection > blackMoves.size() || moveSelection < 0)
163. {
164. std::cerr << "Out of range; please enter a valid choice!" << std::endl;
165. }
166. else
167. {
168. board = board.updateBoard(blackMoves.at(moveSelection - 1), Color::BLACK);
169. currentPlayerColor = computerPlayerColor; // RED
170. isSelectionValid = true;
171. }
172. }
173. }
174. else if (currentPlayerColor == computerPlayerColor) // RED
175. {
176. // AI TAKES TURN AND PRINTS BOARD
177. int numPiecesTakenByAI = computerPlayer.takeTurn(board);
178. currentPlayerColor = humanPlayerColor; // BLACK
179. }
180. // CHECK WIN-LOSS CONDITIONS
181. gameOver = didSomeoneWin(board); // if true, game will end
182. }
183. board.printBoard(); // print final board after someone wins
184. }
185. /\*\*
186. \* At the end of the game, print the game configuration the user provided for ease of reference
187. \* @param int redPlayerAlg - If 1, minimax; if 0, AB Prune
188. \* @param int redPlayerEvalFunct - 1,2, 3, or 4
189. \* @param int blackPlayerAlg - If 1, minimax; if 0, AB Prune
190. \* @param int blackPlayerEvalFunct - 1,2, 3, or 4
191. \* @param int depth - 2 to 15, preferrably 2 or 4 as per project requirements
192. \*/
193. void Simulation::printGameConfig(int redPlayerAlg, int redPlayerEvalFunct, int blackPlayerAlg, int blackPlayerEvalFunct, int depth)
194. {
195. std::string algs[2] = {"Alpha-Beta-Search", "Minimax-Alpha-Beta"};
196. std::cout << "Red player alg: " << algs[redPlayerAlg] << ", eval: " << redPlayerEvalFunct << std::endl;
197. std::cout << "Black player alg: " << algs[blackPlayerAlg] << ", eval: " << blackPlayerEvalFunct << std::endl;
198. std::cout << "Depth: " << depth << std::endl;
199. }
200. /\*\*
201. \* didSomeoneWin - returns true if one player won, to break out of game loops
202. \* @param Board board
203. \*
204. \* @return true if someone won, otherwise false
205. \*/
206. bool Simulation::didSomeoneWin(Board board)
207. {
208. bool isGameOver = false;
209. std::vector<Board::Move> redMoves = board.moveGen(Color::RED);
210. std::vector<Board::Move> blackMoves = board.moveGen(Color::BLACK);
211. if (blackMoves.size() == 0)
212. {
213. isGameOver = true;
214. printRedWins();
215. }
216. else if (redMoves.size() == 0)
217. {
218. isGameOver = true;
219. printBlackWins();
220. }
221. return isGameOver;
222. }
223. /\*\*
224. \* Helper function to display if Red wins
225. \*/
226. void Simulation::printRedWins()
227. {
228. std::cout << "\nRED WINS!!!" << std::endl;
229. std::cout << "RED Player: ᕙ(-\_-')ᕗ" << std::endl;
230. std::cout << "But most importantly, BLACK looooses (boooo!)" << std::endl;
231. std::cout << "BLACK Player: (╯°□°）╯︵ ┻━┻" << std::endl;
232. }
233. /\*\*
234. \* Helper function to display if Black wins
235. \*/
236. void Simulation::printBlackWins()
237. {
238. std::cout << "\nBLACK WINS!!!" << std::endl;
239. std::cout << "BLACK Player: ᕙ(-\_-')ᕗ" << std::endl;
240. std::cout << "But most importantly, RED looooses (boooo!)" << std::endl;
241. std::cout << "RED Player: (╯°□°）╯ ︵ ┻━┻" << std::endl;
242. }
243. /\*\*
244. \* Helper function to display if we run out of permitted turns and get a draw
245. \*/
246. void Simulation::printDraw()
247. {
248. std::cout << "DRAW!!!" << std::endl;
249. std::cout << "Red - (ง •̀\_•́)ง   ლ( `Д’ ლ) - Black" << std::endl;
250. std::cout << "Mission FAILED...We'll get em next time!" << std::endl;
251. }
252. /\*\*
253. \* getNumGamesPlayed - returns a count of the number of games played in a simulation
254. \*
255. \*/
256. int Simulation::getNumGamesPlayed()
257. {
258. return numGamesPlayed;
259. }
260. /\*\*
261. \* generateAnalsysisResults creates a table with results for analysis how many nodes were created, etc.
262. \*/
263. void Simulation::generateAnalysisResults()
264. {
265. }

## main.cpp

1. #include <iostream>
2. #include <string.h> // used by strcmp method
3. #include "Simulation.hpp"
4. #include "Game.hpp"
5. #include "Board.hpp"
6. #include "Player.hpp"
7. #include "Pieces.hpp"
8. #include "Algorithm.hpp"
9. /\*\*
10. \* Main entry way into the application via main() method.
11. \* @author Borislav Sabotinov
12. \*
13. \* The user can display a help menu.
14. \* The user will be prompted to select how they wish to interact with the program.
15. \* Available options are:
16. \*    1. a full simulation,
17. \*    2. partial (single game) simulation,
18. \*    3. player vs. player, or
19. \*    4. player vs. AI.
20. \*/
21. // helper functions to make main() more readable and conscise
22. void printWelcomeMsg();
23. void printHelpMenu();
24. void printMainMenuOptions();
25. void executeRunBasedOnUserInput(int userInput, bool &isInputValid);
26. void getCustomSimUserInput(int &computerPlayerAlg, int &computerPlayerEval, int &depth);
27. void getCustomSimUserInput(int &playerOneAlg, int &playerOneEvalFunct, int &playerTwoAlg, int &playerTwoEvalFunct, int &depth);
28. void runManualGame();
29. void goodbye();
30. /\*\*
31. \* Main function, which serves as an entry point to the Checkers application.
32. \* @author Borislav Sabotinov
33. \*
34. \* User may invoke a help menu by passing in either -h or -help as a CLI parameter when launching the program.
35. \*
36. \* @param int argc - count of the number of CLI arguments provided
37. \* @param char\* argv[] - char array of the CLI arguments
38. \*
39. \* @return EXIT\_SUCCESS if the program completes successfully
40. \*/
41. int main(int argc, char \*argv[])
42. {
43. // display help menu
44. if (argc >= 2)
45. {
46. std::string cliArg = argv[1];
47. if (cliArg == "-h" || cliArg == "-help")
48. {
49. printHelpMenu();
50. return EXIT\_SUCCESS;
51. }
52. if (cliArg == "-nc") // disable color
53. {
54. // ANSII codes for colored text, to improve UI and readability
55. Pieces::ANSII\_BLUE\_START = "";
56. Pieces::ANSII\_RED\_START = "";
57. Pieces::ANSII\_RED\_HIGH = "";
58. Pieces::ANSII\_END = "";
59. Pieces::ANSII\_GREEN\_START = "";
60. Pieces::ANSII\_BLUE\_COUT = "";
61. Pieces::ANSII\_RED\_COUT = "";
62. Pieces::ANSII\_GREEN\_COUT = "";
63. Pieces::ANSII\_YELLOW\_COUT = "";
64. }
65. else if (cliArg == "-no") //
66. {
67. Pieces::ouputDebugData = 0;
68. }
69. else if (cliArg == "-ncno")
70. {
71. // ANSII codes for colored text, to improve UI and readability
72. Pieces::ANSII\_BLUE\_START = "";
73. Pieces::ANSII\_RED\_START = "";
74. Pieces::ANSII\_RED\_HIGH = "";
75. Pieces::ANSII\_END = "";
76. Pieces::ANSII\_GREEN\_START = "";
77. Pieces::ANSII\_BLUE\_COUT = "";
78. Pieces::ANSII\_RED\_COUT = "";
79. Pieces::ANSII\_GREEN\_COUT = "";
80. Pieces::ANSII\_YELLOW\_COUT = "";
81. Pieces::ouputDebugData = 0;
82. }
83. }
84. printWelcomeMsg();
85. printMainMenuOptions();
86. bool isInputValid = false;
87. while (!isInputValid)
88. {
89. int userInput;
90. std::cout << "Your choice " << Pieces::ANSII\_BLUE\_START << "(1, 2, 3, or 4)" << Pieces::ANSII\_END << ": ";
91. std::cin >> userInput;
92. executeRunBasedOnUserInput(userInput, isInputValid);
93. }
94. goodbye();
95. return EXIT\_SUCCESS;
96. }
97. /\*\*
98. \* Prints a welcome message to the console, along with the authors' names and emails.
99. \*/
100. void printWelcomeMsg()
101. {
102. std::cout << Pieces::ANSII\_GREEN\_START << "Welcome to the Checkers AI Program." << Pieces::ANSII\_END << std::endl;
103. std::cout << "Authors: " << Pieces::ANSII\_RED\_START << " David Torrente (dat54@txstate.edu), Randall Henderson (rrh93@txstate.edu), "
104. << "Borislav Sabotinov (bss64@txstate.edu)." << Pieces::ANSII\_END << std::endl;
105. std::cout << "Re-run this program with -h or -help CLI argument to see a help menu or refer to README for instructions."
106. << std::endl;
107. std::cout << std::endl;
108. }
109. /\*\*
110. \* Prints a help message to the console if -h or -help are provided as CLI arguments when invoking the program
111. \*/
112. void printHelpMenu()
113. {
114. std::cout << "To use this program, please read the instructions below and re-launch." << std::endl;
115. std::cout << "Additional details for building and execution are also available in the README file." << std::endl;
116. std::cout << std::endl;
117. std::cout << "Run with -nc for No Color, with -no for No Debug Output, or with -ncno for both No Color AND No Debug Output." << std::endl;
118. std::cout << "When executing the program, you will be prompted to enter the algorithm and evaluation "
119. << "function for the simulation." << std::endl;
120. std::cout << "Please follow the instructions on the screen - if you do not care for any specific custom "
121. << "configuration, simply select Run All." << std::endl;
122. std::cout << "This will bypass everything and simply run the simulation for all algorithms and all "
123. << "evaluation functions sequentially."
124. << "\nThis is the most common and preferred option." << std::endl;
125. }
126. /\*\*
127. \* Prints the main menu with option codes.
128. \*/
129. void printMainMenuOptions()
130. {
131. std::cout << "NOTE: If 1 is selected below, you will NOT be prompted further for any eval function or algorithm. "
132. << "All will be simulated in order." << std::endl;
133. std::cout << "For RED player, r = MAN and R = KING; for BLACK player, b = MAN and B = KING." << std::endl;
134. std::cout << std::endl;
135. std::cout << "Choose a game mode below: " << std::endl;
136. std::cout << "     1. Full Simulation (recommended)" << std::endl;
137. std::cout << "     2. Single Custom Simulation" << std::endl;
138. std::cout << "     3. Player vs Player (manual game)" << std::endl;
139. std::cout << "     4. Player vs AI (will be asked to select AI playstyle)" << std::endl;
140. std::cout << "     Ctrl + C to terminate program at any time." << std::endl;
141. std::cout << std::endl;
142. }
143. /\*\*
144. \* Given the user's choice in the main menu, execute the program accordingly.
145. \*/
146. void executeRunBasedOnUserInput(int userInput, bool &isInputValid)
147. {
148. Simulation \*simulation = new Simulation();
149. switch (userInput)
150. {
151. case 1: // full sim
152. isInputValid = true;
153. simulation->runFullSimulation();
154. std::cout << "# of Games Played: " << simulation->getNumGamesPlayed() << std::endl;
155. break;
156. case 2: // one custom sim
157. isInputValid = true;
158. int playerOneAlg, playerOneEvalFunct, playerTwoAlg, playerTwoEvalFunct, depth;
159. getCustomSimUserInput(playerOneAlg, playerOneEvalFunct, playerTwoAlg, playerTwoEvalFunct, depth);
160. simulation->runSpecificSimulation(playerOneAlg, playerOneEvalFunct, playerTwoAlg, playerTwoEvalFunct, depth);
161. break;
162. case 3: // player vs. player
163. isInputValid = true;
164. runManualGame();
165. break;
166. case 4: // player vs. ai
167. isInputValid = true;
168. getCustomSimUserInput(playerOneAlg, playerOneEvalFunct, depth);
169. simulation->runPlayerVsAISimulation(playerOneAlg, playerOneEvalFunct, depth);
170. break;
171. default:
172. std::cerr << "Invalid option selected! Valid choices are 1, 2, 3, and 4" << std::endl;
173. }
174. }
175. /\*\*
176. \* For a custom simulation, obtains user input and returns via parameters
177. \* @param int &computerPlayerAlg
178. \* @param int &computerPlayerEval
179. \* @param int &depth
180. \*
181. \* @return int &computerPlayerAlg passed in by reference
182. \* @return int &computerPlayerEval passed in by reference
183. \* @return int &depth passed in by reference
184. \*/
185. void getCustomSimUserInput(int &computerPlayerAlg, int &computerPlayerEval, int &depth)
186. {
187. std::cout << "Please select the type of simulation you wish to run by entering in it's number." << std::endl;
188. std::cout << "1. Run Minimax-A-B algorithm" << std::endl;
189. std::cout << "0. Run Alpha-Beta-Search algorithm" << std::endl;
190. // PLAYER CHOICES
191. std::cout << "Algorithm for RED - Player 1 " << Pieces::ANSII\_BLUE\_START << "(1 for minimax, 0 for ab-Search)" << Pieces::ANSII\_END << ":";
192. std::cin >> computerPlayerAlg;
193. std::cout << std::endl;
194. std::cout << "Evaluation for RED - Player 1 " << Pieces::ANSII\_BLUE\_START << "(1 (David's), 2 (Randy's), 3 (Boris'), 4 (returns 1st available move))" << Pieces::ANSII\_END << ": ";
195. std::cin >> computerPlayerEval;
196. // DEPTH
197. std::cout << "Enter the depth for the search tree " << Pieces::ANSII\_BLUE\_START << "(2 or 4 recommended; min = 2, max = 15)" << Pieces::ANSII\_END << ": ";
198. std::cin >> depth;
199. }
200. /\*\*
201. \* getCustomSimUserInput is a helper function to obtain the algorithm and eval function for Player 1 and Player 2
202. \* One game will be simulated only using this input.
203. \*/
204. void getCustomSimUserInput(int &playerOneAlg, int &playerOneEvalFunct, int &playerTwoAlg, int &playerTwoEvalFunct, int &depth)
205. {
206. std::cout << "Please select the type of simulation you wish to run by entering in it's number." << std::endl;
207. std::cout << "1. Run Minimax-A-B algorithm" << std::endl;
208. std::cout << "0. Run Alpha-Beta-Search algorithm" << std::endl;
209. // PLAYER ONE CHOICES
210. std::cout << "Algorithm for RED - Player 1 " << Pieces::ANSII\_BLUE\_START << "(1 for minimax, 0 for ab-Search)" << Pieces::ANSII\_END << ":";
211. std::cin >> playerOneAlg;
212. std::cout << std::endl;
213. std::cout << "Evaluation for RED - Player 1 " << Pieces::ANSII\_BLUE\_START << "(1 (David's), 2 (Randy's), 3 (Boris'), 4 (returns 1st available move))" << Pieces::ANSII\_END << ": ";
214. std::cin >> playerOneEvalFunct;
215. // PLAYER TWO CHOICES
216. std::cout << "Algorithm for BLACK - Player 2 " << Pieces::ANSII\_BLUE\_START << "(1 for minimax, 0 for ab-Search)" << Pieces::ANSII\_END << ": ";
217. std::cin >> playerTwoAlg;
218. std::cout << std::endl;
219. std::cout << "Evaluation for BLACK - Player 2 " << Pieces::ANSII\_BLUE\_START << "(1 (David's), 2 (Randy's), 3 (Boris'), 4 (returns 1st available move))" << Pieces::ANSII\_END << ": ";
220. std::cin >> playerTwoEvalFunct;
221. // DEPTH
222. std::cout << "Enter the depth for the search tree " << Pieces::ANSII\_BLUE\_START << "(2 or 4 recommended; min = 2, max = 15)" << Pieces::ANSII\_END << ": ";
223. std::cin >> depth;
224. }
225. /\*\*
226. \* runManualGame function provides a human user the ability to play checkers with another human.
227. \* It is a manual, input based game where a player must enter the number of the turn they wish to execute.
228. \* @author David Torrente
229. \* @author Borislav Sabotinov
230. \*/
231. void runManualGame()
232. {
233. bool gameOver = false;
234. int moveSelection;
235. Color currentPlayer = Color::BLACK;
236. std::vector<Board::Move> redMoves;
237. std::vector<Board::Move> blackMoves;
238. Board board;
239. board.printBoard();
240. while (!gameOver)
241. {
242. std::vector<Board::Move> redMoves = board.moveGen(Color::RED);
243. std::vector<Board::Move> blackMoves = board.moveGen(Color::BLACK);
244. std::cout << std::endl;
245. std::cout << "Red (r = MAN / R = KING) moves: ";
246. for (int redMoveIter = 0; redMoveIter < redMoves.size(); redMoveIter++)
247. {
248. std::cout << "<" << redMoveIter + 1 << "> " << redMoves.at(redMoveIter).startSquare;
249. for (int destinationIter = 0; destinationIter < redMoves.at(redMoveIter).destinationSquare.size(); destinationIter++)
250. {
251. std::cout << " to " << redMoves.at(redMoveIter).destinationSquare.at(destinationIter);
252. }
253. std::cout << ", ";
254. }
255. std::cout << std::endl;
256. std::cout << "Black (b = MAN / B = KING) moves: ";
257. for (int blackMoveIter = 0; blackMoveIter < blackMoves.size(); blackMoveIter++)
258. {
259. std::cout << "<" << blackMoveIter + 1 << "> " << blackMoves.at(blackMoveIter).startSquare;
260. for (int destinationIter = 0; destinationIter < blackMoves.at(blackMoveIter).destinationSquare.size(); destinationIter++)
261. {
262. std::cout << " to " << blackMoves.at(blackMoveIter).destinationSquare.at(destinationIter);
263. }
264. std::cout << ", ";
265. }
266. std::cout << std::endl;
267. bool isSelectionValid = false;
268. if (currentPlayer == Color::RED)
269. {
270. while (!isSelectionValid)
271. {
272. std::cout << "Select RED move: ";
273. std::cin >> moveSelection;
274. if (moveSelection > redMoves.size() || moveSelection < 0)
275. {
276. std::cerr << "Out of range; please enter a valid choice!" << std::endl;
277. }
278. else
279. {
280. board = board.updateBoard(redMoves.at(moveSelection - 1), Color::RED);
281. currentPlayer = Color::BLACK;
282. isSelectionValid = true;
283. }
284. }
285. }
286. else  // BLACK's turn
287. {
288. isSelectionValid = false;
289. while (!isSelectionValid)
290. {
291. std::cout << "Select BLACK move: ";
292. std::cin >> moveSelection;
293. if (moveSelection > blackMoves.size() || moveSelection < 0)
294. {
295. std::cerr << "Out of range; please enter a valid choice!" << std::endl;
296. }
297. else
298. {
299. board = board.updateBoard(blackMoves.at(moveSelection - 1), Color::BLACK);
300. currentPlayer = Color::RED;
301. isSelectionValid = true;
302. }
303. }
304. }
305. board.printBoard();
306. gameOver = Simulation::didSomeoneWin(board);
307. }
308. }
309. /\*\*
310. \* Displays a compatible, pleasant graphic to the user as a goodbye
311. \*/
312. void goodbye()
313. {
314. std::cout << std::endl;
315. std::cout << Pieces::ANSII\_BLUE\_START << " \_\_  \_\_  \_\_  \_\_  \_\_     \_\_  " << Pieces::ANSII\_END << std::endl;
316. std::cout << Pieces::ANSII\_BLUE\_START << "/ \_ /  \\/  \\|  \\|\_\_)\\\_/|\_ | " << Pieces::ANSII\_END << std::endl;
317. std::cout << Pieces::ANSII\_BLUE\_START << "\\\_\_)\\\_\_/\\\_\_/|\_\_/|\_\_) | |\_\_. " << Pieces::ANSII\_END << std::endl;
318. std::cout << Pieces::ANSII\_BLUE\_START << "                            " << Pieces::ANSII\_END << std::endl;
319. std::cout << std::endl;
320. }

# Appendix B: Sample Output for 18 Games

Due to space constraints, complete output is available in individual files that accompany this project. Where possible, complete output with verbose printout is provided. These files, even the short ones with debug output disabled, are too long to include in this report. To provide an example – for Game 1 below, the short file is 3,397 lines long and the detailed file is 28,147 and 1.64 MB in size. The files are very readable, and the output is structured and well organized, so viewing the logs is preferrable regardless.

This report will include a screenshot of the last state of the board and the game summary, which details how many nodes were expanded.

The naming convention for the output files is: <alg>-<eval>-<alg>-<eval>-<depth>-<verbosity>.log, where alg can be either min (for minimax-a-b) or abs (for alpha-beta-search), eval can be either 1, 2 or 3 (representing the evaluation functions), depth is either 2 or 4, and verbosity is either “short” or “full.”

A full game path is shown only for Game 1 and Game 10 for depths 2 and 4, respectively. This is due to size limitations – the output is simply too long to include in the report in full.

## Game 1: (Minmax-A-B, EF-1) vs (A-B-Search, EF-1) – Depth 2

Refer to files min-1-abs-1-2-full and short.log for full output.

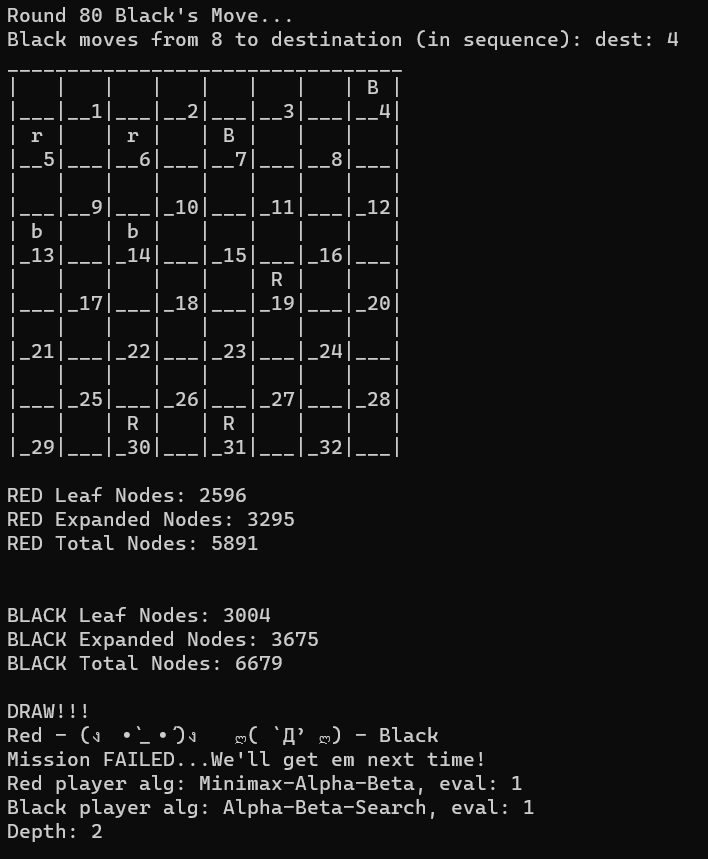


Figure : Minimax-1 vs ABS-1 Depth 2

Welcome to the Checkers AI Program.

Authors: David Torrente (dat54@txstate.edu), Randall Henderson (rrh93@txstate.edu), Borislav Sabotinov (bss64@txstate.edu).

Re-run this program with -h or -help CLI argument to see a help menu or refer to README for instructions.

NOTE: If 1 is selected below, you will NOT be prompted further for any eval function or algorithm. All will be simulated in order.

For RED player, r = MAN and R = KING; for BLACK player, b = MAN and B = KING.

Choose a game mode below:

1. Full Simulation (recommended)

2. Single Custom Simulation

3. Player vs Player (manual game)

4. Player vs AI (will be asked to select AI playstyle)

Ctrl + C to terminate program at any time.

Your choice (1, 2, 3, or 4): Please select the type of simulation you wish to run by entering in its number.

1. Run Minimax-A-B algorithm

0. Run Alpha-Beta-Search algorithm

Algorithm for RED - Player 1 (1 for minimax, 0 for ab-Search):

Evaluation for RED - Player 1 (1 (David's), 2 (Randy's), 3 (Boris'), 4 (returns 1st available move)): Algorithm for BLACK - Player 2 (1 for minimax, 0 for ab-Search):

Evaluation for BLACK - Player 2 (1 (David's), 2 (Randy's), 3 (Boris'), 4 (returns 1st available move)): Enter the depth for the search tree (2 or 4 recommended; min = 2, max = 15): Running a SINGLE game, specific simulation!

Round 1 Black's Move...

Black moves from 21 to destination (in sequence): dest: 17

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | r | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | b | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 1 Red's Move...

Red moves from 12 to destination (in sequence): dest: 16

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | r | | r | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | b | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 2 Black's Move...

Black moves from 17 to destination (in sequence): dest: 13

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | r | | r | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | b | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 2 Red's Move...

Red moves from 11 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | r | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | | | r | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | b | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 3 Black's Move...

Black moves from 25 to destination (in sequence): dest: 21

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | r | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | | | r | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | b | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 3 Red's Move...

Red moves from 10 to destination (in sequence): dest: 14

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | r | | r | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | b | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 4 Black's Move...

Black moves from 22 to destination (in sequence): dest: 17

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | r | | r | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | b | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 4 Red's Move...

Red moves from 7 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | r | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | r | | r | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | b | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 5 Black's Move...

Black moves from 26 to destination (in sequence): dest: 22

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | r | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | r | | r | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 5 Red's Move...

Red moves from 2 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | r | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | r | | r | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 6 Black's Move...

Black moves from 24 to destination (in sequence): dest: 20

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | r | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | r | | r | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 6 Red's Move...

Red moves from 7 to destination (in sequence): dest: 11

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | r | | r | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | r | | r | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 7 Black's Move...

Black moves from 27 to destination (in sequence): dest: 24

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | r | | r | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | r | | r | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 7 Red's Move...

Red moves from 3 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | r | | r | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | r | | r | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 8 Black's Move...

Black moves from 29 to destination (in sequence): dest: 25

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | r | | r | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | r | | r | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | | | | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 8 Red's Move...

Red moves from 8 to destination (in sequence): dest: 12

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | r | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | r | | r | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | | | | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 9 Black's Move...

Black moves from 30 to destination (in sequence): dest: 26

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | r | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | r | | r | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | b | | | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 9 Red's Move...

Red moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | r | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | r | | r | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | b | | | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 10 Black's Move...

Black moves from 31 to destination (in sequence): dest: 27

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | r | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | r | | r | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | b | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 10 Red's Move...

Red moves from 14 to destination (in sequence): dest: 18

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | r | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | | | r | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | r | | | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | b | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 11 Black's Move...

Black moves from 23 to destination (in sequence): dest: 14

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | r | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | r | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | b | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

BLACK player took 1 piece(s).

Round 11 Red's Move...

Red moves from 9 to destination (in sequence): dest: 18

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | r | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | | | r | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | r | | | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | b | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

RED player took 1 piece(s).

Round 12 Black's Move...

Black moves from 26 to destination (in sequence): dest: 23

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | r | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | | | r | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | r | | | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 12 Red's Move...

Red moves from 6 to destination (in sequence): dest: 9

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | r | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | | | r | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | r | | | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 13 Black's Move...

Black moves from 13 to destination (in sequence): dest: 6

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | b | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | r | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | r | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | r | | | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

BLACK player took 1 piece(s).

Round 13 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | b | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | r | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | r | | r | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 14 Black's Move...

Black moves from 22 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | b | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | r | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | b | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | r | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

BLACK player took 1 piece(s).

Round 14 Red's Move...

Red moves from 19 to destination (in sequence): dest: 26

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | b | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | r | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | b | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | r | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

RED player took 1 piece(s).

Round 15 Black's Move...

Black moves from 6 to destination (in sequence): dest: 2

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | B | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | r | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | b | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | r | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 15 Red's Move...

Red moves from 10 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | B | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | r | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | r | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

RED player took 1 piece(s).

Round 16 Black's Move...

Black moves from 24 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | B | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | b | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | r | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

BLACK player took 1 piece(s).

Round 16 Red's Move...

Red moves from 11 to destination (in sequence): dest: 18

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | B | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | r | | | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | r | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

RED player took 1 piece(s).

Round 17 Black's Move...

Black moves from 2 to destination (in sequence): dest: 11

dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | r | | | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | r | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

BLACK player took 2 piece(s).

Round 17 Red's Move...

Red moves from 26 to destination (in sequence): dest: 30

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | r | | | | b |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 18 Black's Move...

Black moves from 20 to destination (in sequence): dest: 11

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | r | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

BLACK player took 1 piece(s).

Round 18 Red's Move...

Red moves from 30 to destination (in sequence): dest: 26

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | r | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | R | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 19 Black's Move...

Black moves from 25 to destination (in sequence): dest: 22

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | r | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | R | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 19 Red's Move...

Red moves from 18 to destination (in sequence): dest: 25

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | r | | R | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

RED player took 1 piece(s).

Round 20 Black's Move...

Black moves from 27 to destination (in sequence): dest: 23

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | r | | R | | | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 20 Red's Move...

Red moves from 26 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | r | | | | | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

RED player took 1 piece(s).

Round 21 Black's Move...

Black moves from 28 to destination (in sequence): dest: 24

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | r | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 21 Red's Move...

Red moves from 19 to destination (in sequence): dest: 28

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | r | | | | | | R |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

RED player took 1 piece(s).

Round 22 Black's Move...

Black moves from 17 to destination (in sequence): dest: 14

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | r | | | | | | R |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 22 Red's Move...

Red moves from 25 to destination (in sequence): dest: 29

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | R |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| R | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 23 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | R |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| R | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 23 Red's Move...

Red moves from 1 to destination (in sequence): dest: 6

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | R |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| R | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 24 Black's Move...

Black moves from 8 to destination (in sequence): dest: 3

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | B | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | R |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| R | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 24 Red's Move...

Red moves from 28 to destination (in sequence): dest: 24

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | B | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | R | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| R | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 25 Black's Move...

Black moves from 3 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | R | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| R | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 25 Red's Move...

Red moves from 29 to destination (in sequence): dest: 25

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | R | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | R | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 26 Black's Move...

Black moves from 7 to destination (in sequence): dest: 3

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | B | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | R | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | R | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 26 Red's Move...

Red moves from 12 to destination (in sequence): dest: 16

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | B | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | R | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | R | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 27 Black's Move...

Black moves from 3 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | R | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | R | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 27 Red's Move...

Red moves from 25 to destination (in sequence): dest: 30

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | R | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 28 Black's Move...

Black moves from 7 to destination (in sequence): dest: 3

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | B | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | R | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 28 Red's Move...

Red moves from 16 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | B | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | r | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | R | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 29 Black's Move...

Black moves from 3 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | r | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | R | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 29 Red's Move...

Red moves from 19 to destination (in sequence): dest: 23

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | r | | R | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 30 Black's Move...

Black moves from 7 to destination (in sequence): dest: 3

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | B | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | r | | R | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 30 Red's Move...

Red moves from 23 to destination (in sequence): dest: 26

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | B | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | R | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | r | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 31 Black's Move...

Black moves from 32 to destination (in sequence): dest: 27

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | B | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | R | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | r | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 31 Red's Move...

Red moves from 24 to destination (in sequence): dest: 31

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | B | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | r | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

RED player took 1 piece(s).

Round 32 Black's Move...

Black moves from 3 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | r | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 32 Red's Move...

Red moves from 31 to destination (in sequence): dest: 27

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | r | | R | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 33 Black's Move...

Black moves from 11 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | r | | R | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 33 Red's Move...

Red moves from 26 to destination (in sequence): dest: 31

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 34 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 34 Red's Move...

Red moves from 27 to destination (in sequence): dest: 23

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | R | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 35 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | R | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 35 Red's Move...

Red moves from 23 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 36 Black's Move...

Black moves from 21 to destination (in sequence): dest: 17

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 36 Red's Move...

Red moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 37 Black's Move...

Black moves from 17 to destination (in sequence): dest: 13

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 37 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 38 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 38 Red's Move...

Red moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 39 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 39 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 40 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 40 Red's Move...

Red moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 41 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 41 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 42 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 42 Red's Move...

Red moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 43 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 43 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 44 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 44 Red's Move...

Red moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 45 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 45 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 46 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 46 Red's Move...

Red moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 47 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 47 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 48 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 48 Red's Move...

Red moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 49 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 49 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 50 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 50 Red's Move...

Red moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 51 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 51 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 52 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 52 Red's Move...

Red moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 53 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 53 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 54 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 54 Red's Move...

Red moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 55 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 55 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 56 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 56 Red's Move...

Red moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 57 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 57 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 58 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 58 Red's Move...

Red moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 59 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 59 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 60 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 60 Red's Move...

Red moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 61 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 61 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 62 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 62 Red's Move...

Red moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 63 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 63 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 64 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 64 Red's Move...

Red moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 65 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 65 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 66 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 66 Red's Move...

Red moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 67 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 67 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 68 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 68 Red's Move...

Red moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 69 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 69 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 70 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 70 Red's Move...

Red moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 71 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 71 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 72 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 72 Red's Move...

Red moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 73 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 73 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 74 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 74 Red's Move...

Red moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 75 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 75 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 76 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 76 Red's Move...

Red moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 77 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 77 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 78 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 78 Red's Move...

Red moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 79 Black's Move...

Black moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 79 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | B | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 80 Black's Move...

Black moves from 8 to destination (in sequence): dest: 4

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | B |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | B | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| b | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | R | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | R | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

RED Leaf Nodes: 2596

RED Expanded Nodes: 3295

RED Total Nodes: 5891

BLACK Leaf Nodes: 3004

BLACK Expanded Nodes: 3675

BLACK Total Nodes: 6679

DRAW!!!

Red - (ง •̀\_•́)ง ლ( `Д’ ლ) - Black

Mission FAILED...We'll get em next time!

Red player alg: Minimax-Alpha-Beta, eval: 1

Black player alg: Alpha-Beta-Search, eval: 1

Depth: 2

\_\_ \_\_ \_\_ \_\_ \_\_ \_\_

/ \_ / \/ \| \|\_\_)\\_/|\_ |

\\_\_)\\_\_/\\_\_/|\_\_/|\_\_) | |\_\_.

## Game 2: (Minmax-A-B, EF-2) vs (A-B-Search, EF-2) – Depth 2

Refer to files min-2-abs-2-2-full and short.log for full output.

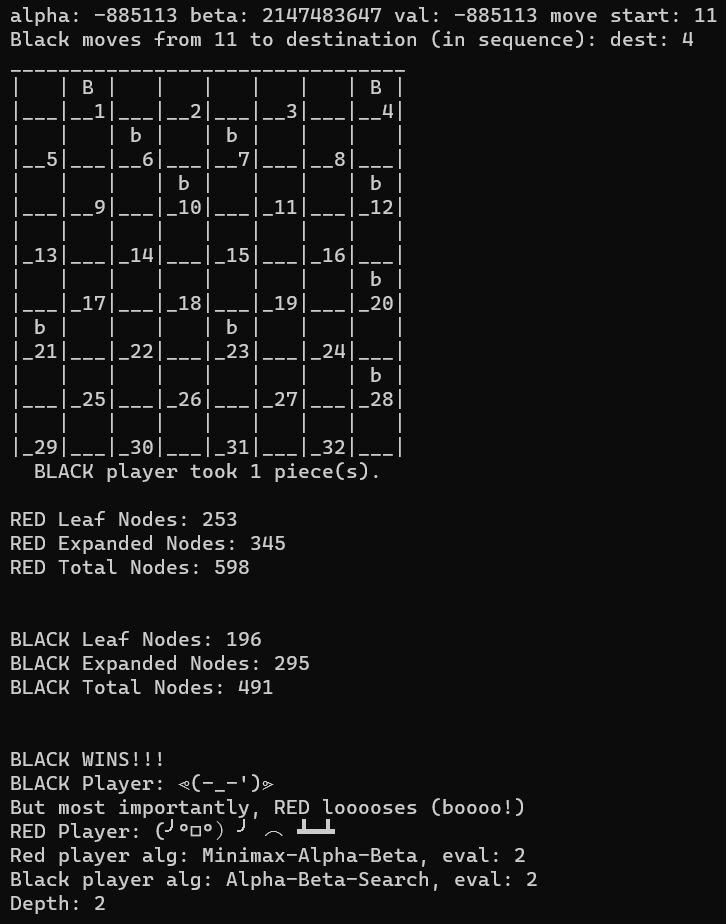


Figure : Minimax-2 vs ABS-2 Depth 2

## Game 3: (Minmax-A-B, EF-3) vs (A-B-Search, EF-3) – Depth 2

Refer to min-3-abs-3-2-short.log and min-3-abs-3-2-full.log files for complete output.

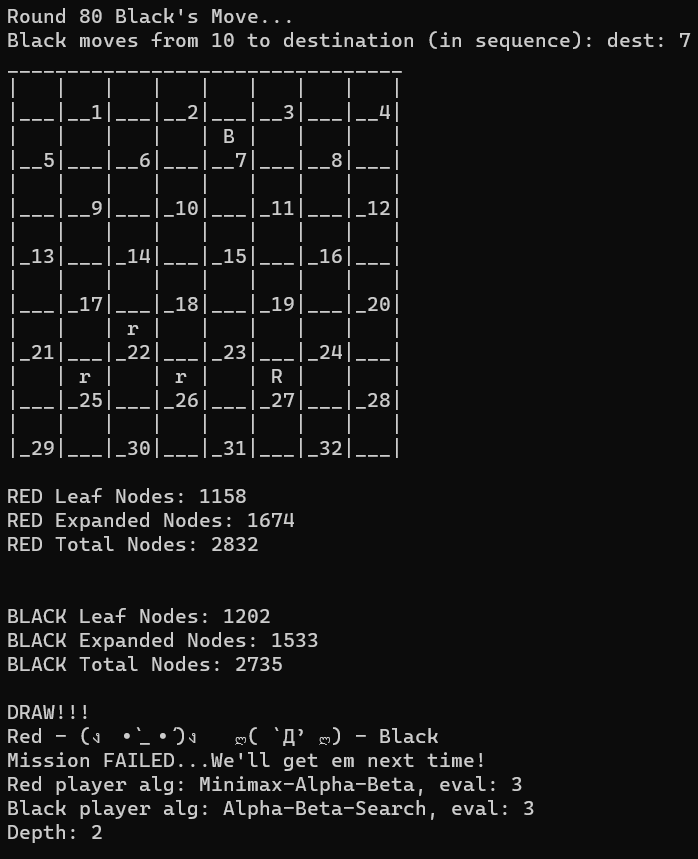


Figure : Minimax-3 vs ABS-3 Depth 2

## Game 4: (Minmax-A-B, EF-1) vs (Minimax-A-B, EF-2) – Depth 2

Refer to files min-1-min-2-2-full and short.log for full output.

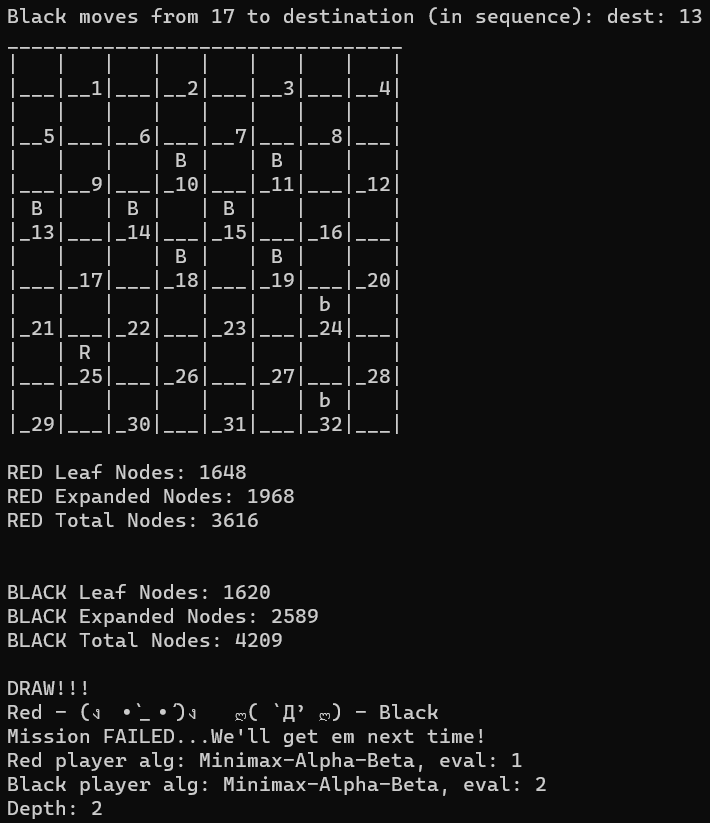


Figure : Minimax-1 vs Minimax-2 Depth 2

## Game 5: (Minmax-A-B, EF-1) vs (Minimax-A-B, EF-3) – Depth 2

Refer to files min-1-min-3-2-full and short.log for full output.

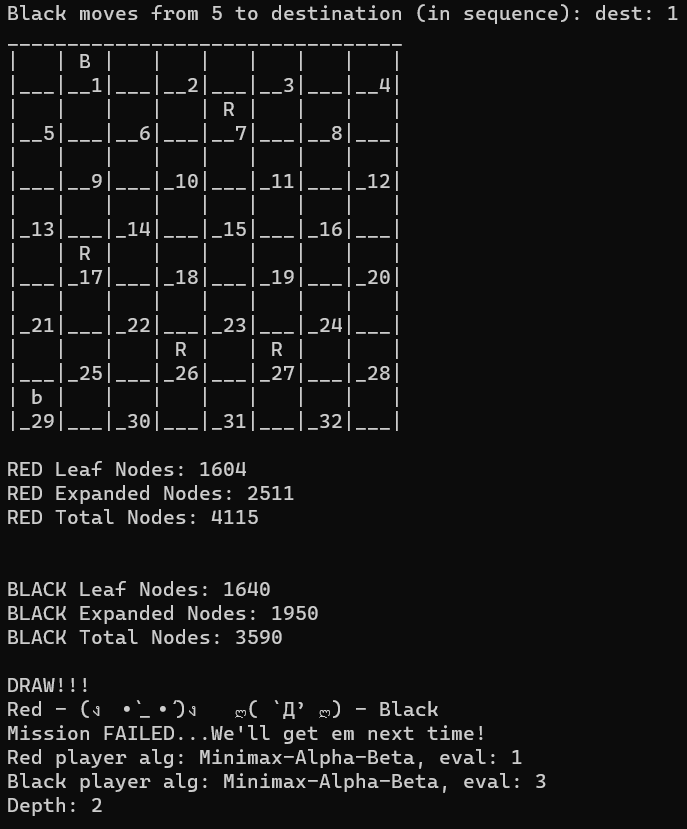


Figure : Minimax-1 vs Minimax-3 Depth 2

## Game 6: (Minmax-A-B, EF-2) vs (Minimax-A-B, EF-3) – Depth 2

Refer to files min-2-min-3-2-full and short.log for full output.

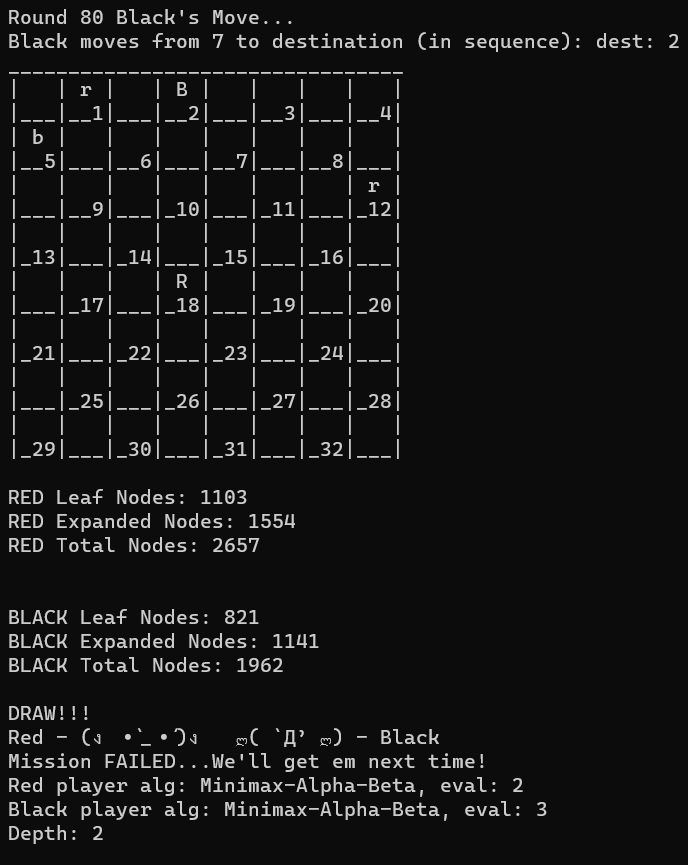


Figure : Minimax-2 vs Minimax-3 Depth 2

## Game 7: (A-B-Search, EF-1) vs (A-B-Search, EF-2) – Depth 2

Refer to files abs-1-abs-2-2-full and short.log for full output.

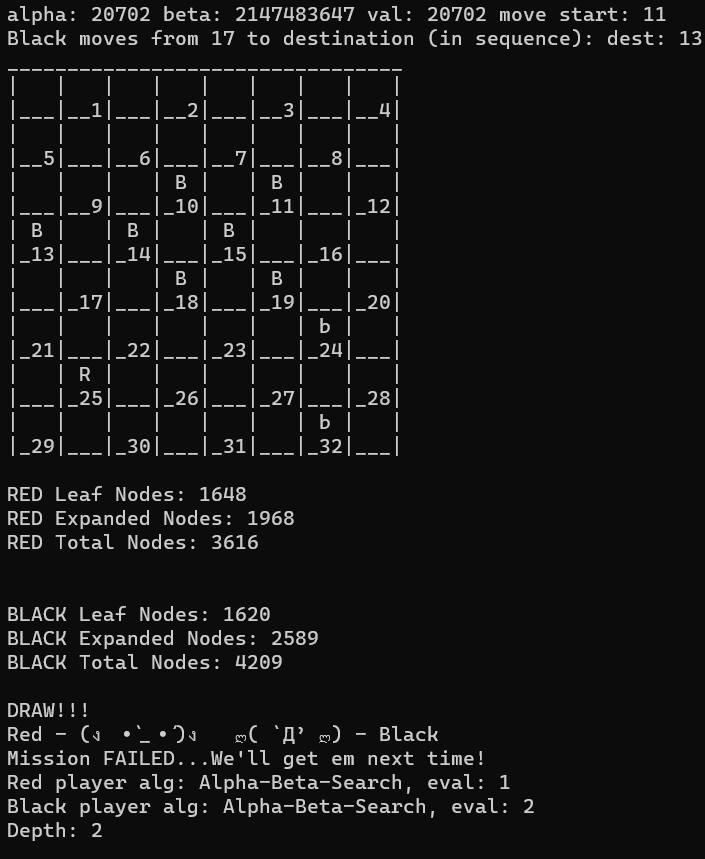


Figure : ABS-1 vs ABS-2 Depth 2

## Game 8: (A-B-Search, EF-1) vs (A-B-Search, EF-3) – Depth 2

Refer to files abs-1-abs-3-2-full and short.log for full output.

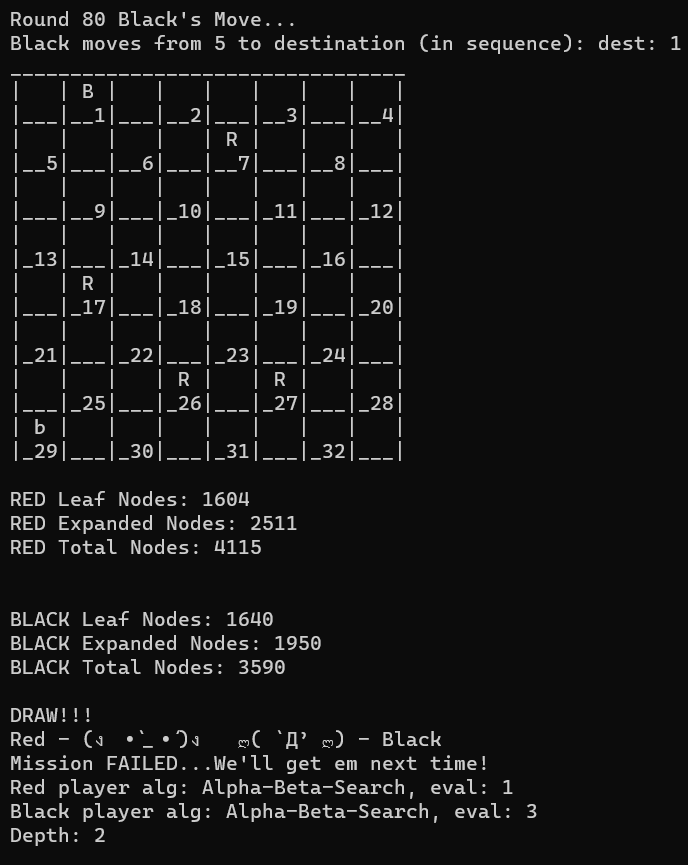


Figure : ABS-1 vs ABS-3 Depth 2

## Game 9: (A-B-Search, EF-2) vs (A-B-Search, EF-3) – Depth 2

Refer to files abs-2-abs-3-2-full and short.log for full output.

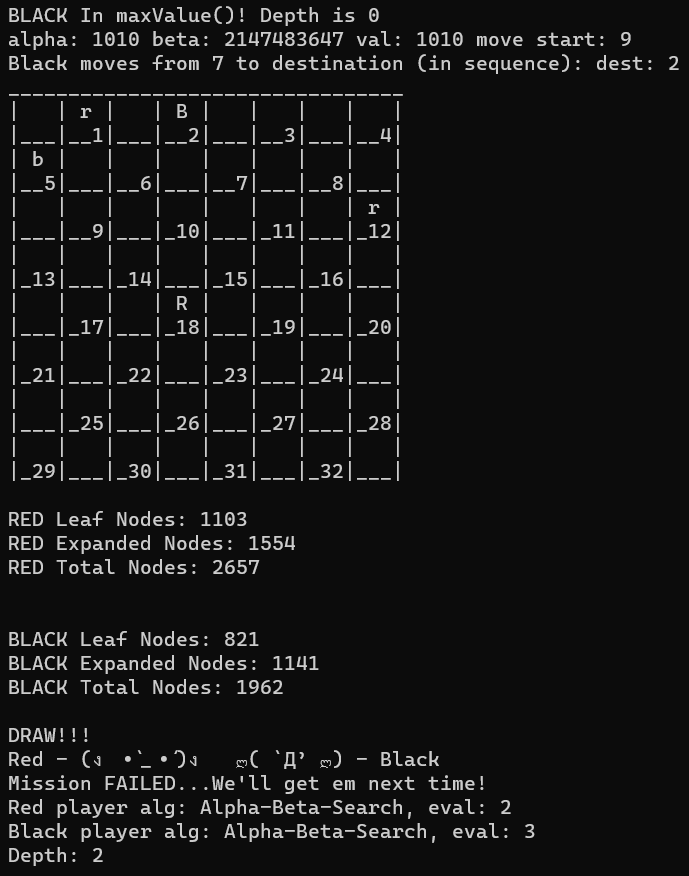


Figure : ABS-2 vs ABS-3 Depth 2

## Game 10: (Minmax-A-B, EF-1) vs (A-B-Search, EF-1) – Depth 4

Refer to files min-1-abs-1-4-full and short.log for full output.

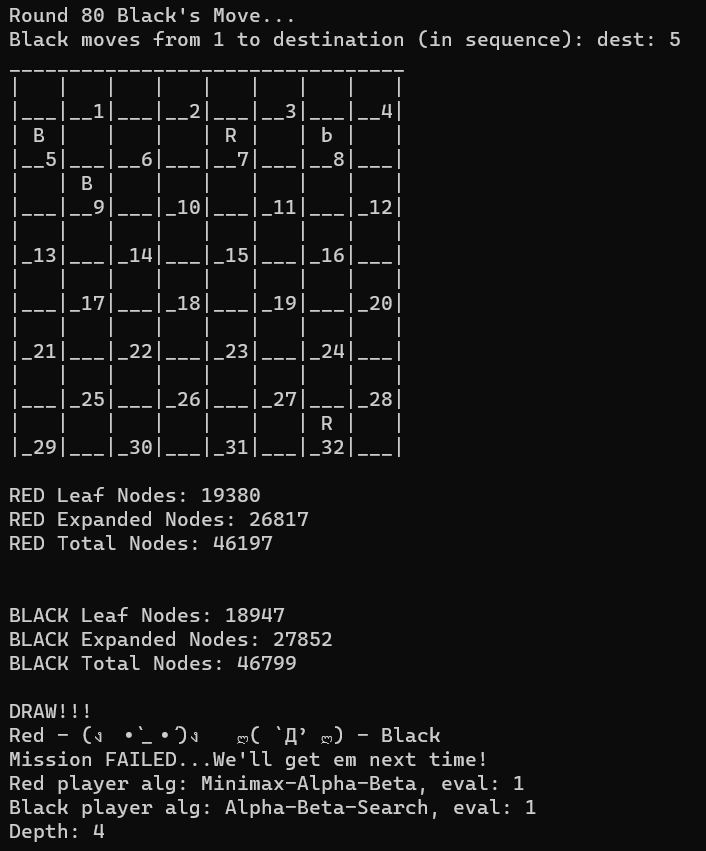


Figure : Minimax-1 vs ABS-1 Depth 4

Welcome to the Checkers AI Program.

Authors: David Torrente (dat54@txstate.edu), Randall Henderson (rrh93@txstate.edu), Borislav Sabotinov (bss64@txstate.edu).

Re-run this program with -h or -help CLI argument to see a help menu or refer to README for instructions.

NOTE: If 1 is selected below, you will NOT be prompted further for any eval function or algorithm. All will be simulated in order.

For RED player, r = MAN and R = KING; for BLACK player, b = MAN and B = KING.

Choose a game mode below:

1. Full Simulation (recommended)

2. Single Custom Simulation

3. Player vs Player (manual game)

4. Player vs AI (will be asked to select AI playstyle)

Ctrl + C to terminate program at any time.

Your choice (1, 2, 3, or 4): Please select the type of simulation you wish to run by entering in it's number.

1. Run Minimax-A-B algorithm

0. Run Alpha-Beta-Search algorithm

Algorithm for RED - Player 1 (1 for minimax, 0 for ab-Search):

Evaluation for RED - Player 1 (1 (David's), 2 (Randy's), 3 (Boris'), 4 (returns 1st available move)): Algorithm for BLACK - Player 2 (1 for minimax, 0 for ab-Search):

Evaluation for BLACK - Player 2 (1 (David's), 2 (Randy's), 3 (Boris'), 4 (returns 1st available move)): Enter the depth for the search tree (2 or 4 recommended; min = 2, max = 15): Running a SINGLE game, specific simulation!

Round 1 Black's Move...

Black moves from 21 to destination (in sequence): dest: 17

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | r | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | b | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 1 Red's Move...

Red moves from 9 to destination (in sequence): dest: 14

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | r | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | r | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | b | | b | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 2 Black's Move...

Black moves from 25 to destination (in sequence): dest: 21

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | r | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | r | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | b | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 2 Red's Move...

Red moves from 10 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | r | | r | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | b | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 3 Black's Move...

Black moves from 17 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | b | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | r | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | b | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

BLACK player took 1 piece(s).

Round 3 Red's Move...

Red moves from 7 to destination (in sequence): dest: 14

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | r | | r | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | b | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | b | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

RED player took 1 piece(s).

Round 4 Black's Move...

Black moves from 23 to destination (in sequence): dest: 18

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | r | | r | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | b | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | b | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 4 Red's Move...

Red moves from 14 to destination (in sequence): dest: 23

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | r | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | r | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | b | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

RED player took 1 piece(s).

Round 5 Black's Move...

Black moves from 26 to destination (in sequence): dest: 19

dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | b | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

BLACK player took 2 piece(s).

Round 5 Red's Move...

Red moves from 6 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | r | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | | | b | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

RED player took 1 piece(s).

Round 6 Black's Move...

Black moves from 24 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | r | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 6 Red's Move...

Red moves from 15 to destination (in sequence): dest: 24

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | | | r | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | b |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

RED player took 1 piece(s).

Round 7 Black's Move...

Black moves from 28 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | r | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

BLACK player took 1 piece(s).

Round 7 Red's Move...

Red moves from 1 to destination (in sequence): dest: 6

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 8 Black's Move...

Black moves from 19 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | b | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 8 Red's Move...

Red moves from 11 to destination (in sequence): dest: 18

dest: 25

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | r | | | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| b | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

RED player took 2 piece(s).

Round 9 Black's Move...

Black moves from 29 to destination (in sequence): dest: 22

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | r | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

BLACK player took 1 piece(s).

Round 9 Red's Move...

Red moves from 6 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | r | | | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| b | | b | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 10 Black's Move...

Black moves from 21 to destination (in sequence): dest: 17

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | r | | | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | b | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 10 Red's Move...

Red moves from 10 to destination (in sequence): dest: 14

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | r | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | b | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | b | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 11 Black's Move...

Black moves from 17 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | b | | | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | b | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

BLACK player took 1 piece(s).

Round 11 Red's Move...

Red moves from 12 to destination (in sequence): dest: 16

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | b | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | b | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 12 Black's Move...

Black moves from 27 to destination (in sequence): dest: 23

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | b | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | b | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 12 Red's Move...

Red moves from 8 to destination (in sequence): dest: 11

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | b | | r | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | b | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 13 Black's Move...

Black moves from 22 to destination (in sequence): dest: 18

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | b | | r | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | b | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 13 Red's Move...

Red moves from 16 to destination (in sequence): dest: 20

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | b | | r | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | b | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | b | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 14 Black's Move...

Black moves from 31 to destination (in sequence): dest: 27

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | b | | r | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | b | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 14 Red's Move...

Red moves from 11 to destination (in sequence): dest: 16

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | b | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | b | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 15 Black's Move...

Black moves from 10 to destination (in sequence): dest: 6

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | r | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | b | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | b | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 15 Red's Move...

Red moves from 2 to destination (in sequence): dest: 9

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | b | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

RED player took 1 piece(s).

Round 16 Black's Move...

Black moves from 18 to destination (in sequence): dest: 14

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 16 Red's Move...

Red moves from 9 to destination (in sequence): dest: 18

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | r | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

RED player took 1 piece(s).

Round 17 Black's Move...

Black moves from 23 to destination (in sequence): dest: 14

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| r | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

BLACK player took 1 piece(s).

Round 17 Red's Move...

Red moves from 5 to destination (in sequence): dest: 9

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | r | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 18 Black's Move...

Black moves from 14 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | r | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| b | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

BLACK player took 1 piece(s).

Round 18 Red's Move...

Red moves from 3 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| b | | | | r | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 19 Black's Move...

Black moves from 5 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | r |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | r | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 19 Red's Move...

Red moves from 4 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | b | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 20 Black's Move...

Black moves from 27 to destination (in sequence): dest: 23

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 20 Red's Move...

Red moves from 20 to destination (in sequence): dest: 24

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | b | | r | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 21 Black's Move...

Black moves from 1 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | b | | r | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 21 Red's Move...

Red moves from 24 to destination (in sequence): dest: 27

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | r | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 22 Black's Move...

Black moves from 5 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | r | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | r | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 22 Red's Move...

Red moves from 7 to destination (in sequence): dest: 11

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | r | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | r | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 23 Black's Move...

Black moves from 23 to destination (in sequence): dest: 18

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | r | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | r | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | b | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | r | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 23 Red's Move...

Red moves from 8 to destination (in sequence): dest: 12

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | b | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | r | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | b | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 24 Black's Move...

Black moves from 32 to destination (in sequence): dest: 23

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | r | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | b | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

BLACK player took 1 piece(s).

Round 24 Red's Move...

Red moves from 16 to destination (in sequence): dest: 20

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | b | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 25 Black's Move...

Black moves from 18 to destination (in sequence): dest: 14

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | r | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 25 Red's Move...

Red moves from 11 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | r | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 26 Black's Move...

Black moves from 1 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | r | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 26 Red's Move...

Red moves from 15 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | r | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | b | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 27 Black's Move...

Black moves from 23 to destination (in sequence): dest: 16

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | b | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

BLACK player took 1 piece(s).

Round 27 Red's Move...

Red moves from 12 to destination (in sequence): dest: 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | r | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

RED player took 1 piece(s).

Round 28 Black's Move...

Black moves from 5 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | r | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 28 Red's Move...

Red moves from 19 to destination (in sequence): dest: 23

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | r | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 29 Black's Move...

Black moves from 1 to destination (in sequence): dest: 6

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | B | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | r | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 29 Red's Move...

Red moves from 23 to destination (in sequence): dest: 27

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | B | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | r | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | b | | | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 30 Black's Move...

Black moves from 30 to destination (in sequence): dest: 26

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | B | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | b | | r | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 30 Red's Move...

Red moves from 20 to destination (in sequence): dest: 24

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | B | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | r | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | b | | r | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 31 Black's Move...

Black moves from 6 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | r | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | b | | r | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 31 Red's Move...

Red moves from 24 to destination (in sequence): dest: 28

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | b | | r | | r |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 32 Black's Move...

Black moves from 1 to destination (in sequence): dest: 6

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | B | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | b | | r | | r |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 32 Red's Move...

Red moves from 27 to destination (in sequence): dest: 31

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | B | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | b | | | | r |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 33 Black's Move...

Black moves from 26 to destination (in sequence): dest: 22

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | B | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | b | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | r |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | R | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 33 Red's Move...

Red moves from 31 to destination (in sequence): dest: 26

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | B | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | b | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | R | | | | r |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 34 Black's Move...

Black moves from 22 to destination (in sequence): dest: 18

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | B | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | b | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | R | | | | r |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 34 Red's Move...

Red moves from 28 to destination (in sequence): dest: 32

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | B | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | b | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | R | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 35 Black's Move...

Black moves from 18 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | B | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | b | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | R | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 35 Red's Move...

Red moves from 26 to destination (in sequence): dest: 22

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | B | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | b | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | R | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 36 Black's Move...

Black moves from 6 to destination (in sequence): dest: 9

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | b | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | R | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 36 Red's Move...

Red moves from 22 to destination (in sequence): dest: 17

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | b | | b | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | R | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 37 Black's Move...

Black moves from 14 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | b | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | b | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | R | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 37 Red's Move...

Red moves from 17 to destination (in sequence): dest: 22

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | b | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | b | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | R | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 38 Black's Move...

Black moves from 10 to destination (in sequence): dest: 6

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | b | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | b | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | R | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 38 Red's Move...

Red moves from 22 to destination (in sequence): dest: 18

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | b | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | b | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | R | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 39 Black's Move...

Black moves from 15 to destination (in sequence): dest: 11

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | b | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | b | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | R | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 39 Red's Move...

Red moves from 18 to destination (in sequence): dest: 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | b | | | | | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | b | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 40 Black's Move...

Black moves from 11 to destination (in sequence): dest: 8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | b | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | R | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 40 Red's Move...

Red moves from 15 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | b | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 41 Black's Move...

Black moves from 6 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 41 Red's Move...

Red moves from 10 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 42 Black's Move...

Black moves from 1 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 42 Red's Move...

Red moves from 7 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 43 Black's Move...

Black moves from 5 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 43 Red's Move...

Red moves from 10 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 44 Black's Move...

Black moves from 1 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 44 Red's Move...

Red moves from 7 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 45 Black's Move...

Black moves from 5 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 45 Red's Move...

Red moves from 10 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 46 Black's Move...

Black moves from 1 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 46 Red's Move...

Red moves from 7 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 47 Black's Move...

Black moves from 5 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 47 Red's Move...

Red moves from 10 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 48 Black's Move...

Black moves from 1 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 48 Red's Move...

Red moves from 7 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 49 Black's Move...

Black moves from 5 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 49 Red's Move...

Red moves from 10 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 50 Black's Move...

Black moves from 1 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 50 Red's Move...

Red moves from 7 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 51 Black's Move...

Black moves from 5 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 51 Red's Move...

Red moves from 10 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 52 Black's Move...

Black moves from 1 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 52 Red's Move...

Red moves from 7 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 53 Black's Move...

Black moves from 5 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 53 Red's Move...

Red moves from 10 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 54 Black's Move...

Black moves from 1 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 54 Red's Move...

Red moves from 7 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 55 Black's Move...

Black moves from 5 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 55 Red's Move...

Red moves from 10 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 56 Black's Move...

Black moves from 1 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 56 Red's Move...

Red moves from 7 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 57 Black's Move...

Black moves from 5 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 57 Red's Move...

Red moves from 10 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 58 Black's Move...

Black moves from 1 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 58 Red's Move...

Red moves from 7 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 59 Black's Move...

Black moves from 5 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 59 Red's Move...

Red moves from 10 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 60 Black's Move...

Black moves from 1 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 60 Red's Move...

Red moves from 7 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 61 Black's Move...

Black moves from 5 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 61 Red's Move...

Red moves from 10 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 62 Black's Move...

Black moves from 1 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 62 Red's Move...

Red moves from 7 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 63 Black's Move...

Black moves from 5 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 63 Red's Move...

Red moves from 10 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 64 Black's Move...

Black moves from 1 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 64 Red's Move...

Red moves from 7 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 65 Black's Move...

Black moves from 5 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 65 Red's Move...

Red moves from 10 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 66 Black's Move...

Black moves from 1 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 66 Red's Move...

Red moves from 7 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 67 Black's Move...

Black moves from 5 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 67 Red's Move...

Red moves from 10 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 68 Black's Move...

Black moves from 1 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 68 Red's Move...

Red moves from 7 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 69 Black's Move...

Black moves from 5 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 69 Red's Move...

Red moves from 10 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 70 Black's Move...

Black moves from 1 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 70 Red's Move...

Red moves from 7 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 71 Black's Move...

Black moves from 5 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 71 Red's Move...

Red moves from 10 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 72 Black's Move...

Black moves from 1 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 72 Red's Move...

Red moves from 7 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 73 Black's Move...

Black moves from 5 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 73 Red's Move...

Red moves from 10 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 74 Black's Move...

Black moves from 1 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 74 Red's Move...

Red moves from 7 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 75 Black's Move...

Black moves from 5 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 75 Red's Move...

Red moves from 10 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 76 Black's Move...

Black moves from 1 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 76 Red's Move...

Red moves from 7 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 77 Black's Move...

Black moves from 5 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 77 Red's Move...

Red moves from 10 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 78 Black's Move...

Black moves from 1 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 78 Red's Move...

Red moves from 7 to destination (in sequence): dest: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 79 Black's Move...

Black moves from 5 to destination (in sequence): dest: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | R | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 79 Red's Move...

Red moves from 10 to destination (in sequence): dest: 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | B | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

Round 80 Black's Move...

Black moves from 1 to destination (in sequence): dest: 5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | |

|\_\_\_|\_\_1|\_\_\_|\_\_2|\_\_\_|\_\_3|\_\_\_|\_\_4|

| B | | | | R | | b | |

|\_\_5|\_\_\_|\_\_6|\_\_\_|\_\_7|\_\_\_|\_\_8|\_\_\_|

| | B | | | | | | |

|\_\_\_|\_\_9|\_\_\_|\_10|\_\_\_|\_11|\_\_\_|\_12|

| | | | | | | | |

|\_13|\_\_\_|\_14|\_\_\_|\_15|\_\_\_|\_16|\_\_\_|

| | | | | | | | |

|\_\_\_|\_17|\_\_\_|\_18|\_\_\_|\_19|\_\_\_|\_20|

| | | | | | | | |

|\_21|\_\_\_|\_22|\_\_\_|\_23|\_\_\_|\_24|\_\_\_|

| | | | | | | | |

|\_\_\_|\_25|\_\_\_|\_26|\_\_\_|\_27|\_\_\_|\_28|

| | | | | | | R | |

|\_29|\_\_\_|\_30|\_\_\_|\_31|\_\_\_|\_32|\_\_\_|

RED Leaf Nodes: 19380

RED Expanded Nodes: 26817

RED Total Nodes: 46197

BLACK Leaf Nodes: 18947

BLACK Expanded Nodes: 27852

BLACK Total Nodes: 46799

DRAW!!!

Red - (ง •̀\_•́)ง ლ( `Д’ ლ) - Black

Mission FAILED...We'll get em next time!

Red player alg: Minimax-Alpha-Beta, eval: 1

Black player alg: Alpha-Beta-Search, eval: 1

Depth: 4

\_\_ \_\_ \_\_ \_\_ \_\_ \_\_

/ \_ / \/ \| \|\_\_)\\_/|\_ |

\\_\_)\\_\_/\\_\_/|\_\_/|\_\_) | |\_\_.

## Game 11: (Minmax-A-B, EF-2) vs (A-B-Search, EF-2) – Depth 4

Refer to files min-2-abs-2-4-full and short.log for full output.

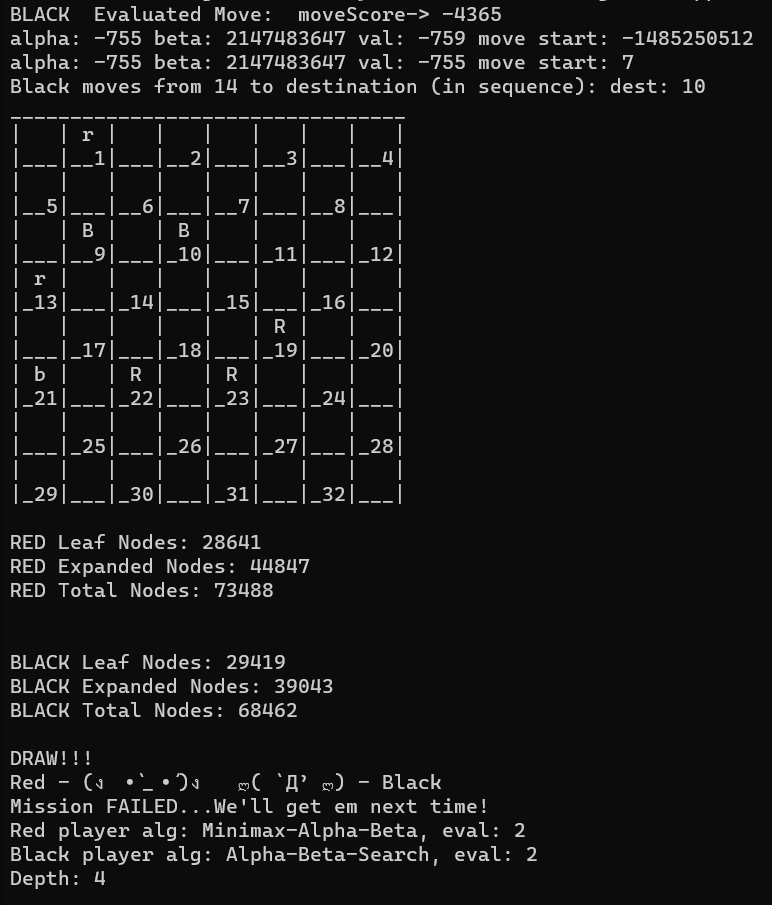


Figure : Minimax-2 vs ABS-2 Depth 4

## Game 12: (Minmax-A-B, EF-3) vs (A-B-Search, EF-3) – Depth 4

Refer to files min-3-abs-3-4-full and short.log for full output.

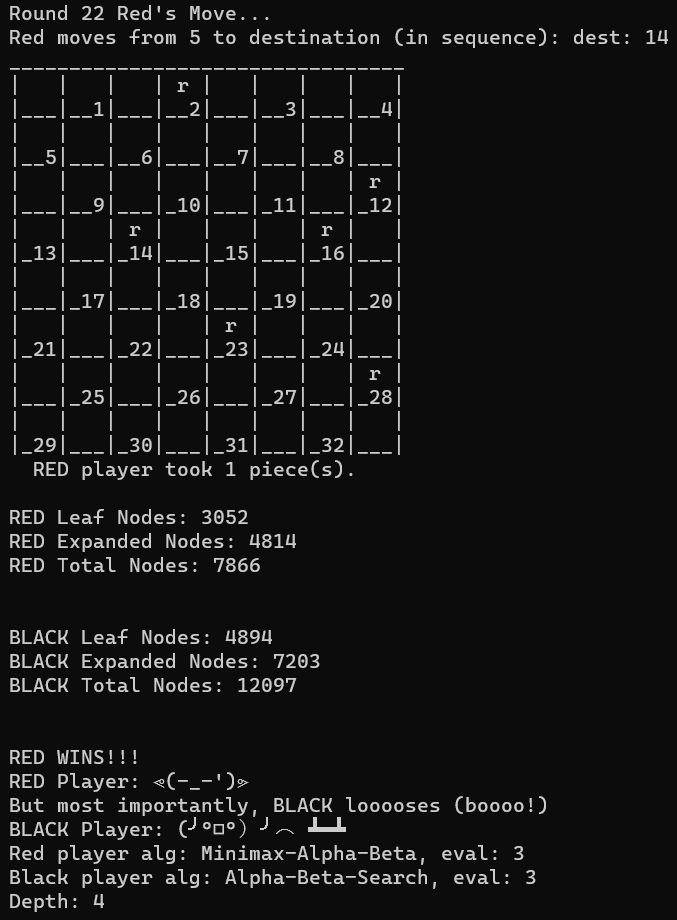


Figure : Minimax-3 vs ABS-3 Depth 4

## Game 13: (Minmax-A-B, EF-1) vs (Minimax-A-B, EF-2) – Depth 4

Refer to files min-1-min-2-4-full and short.log for full output.

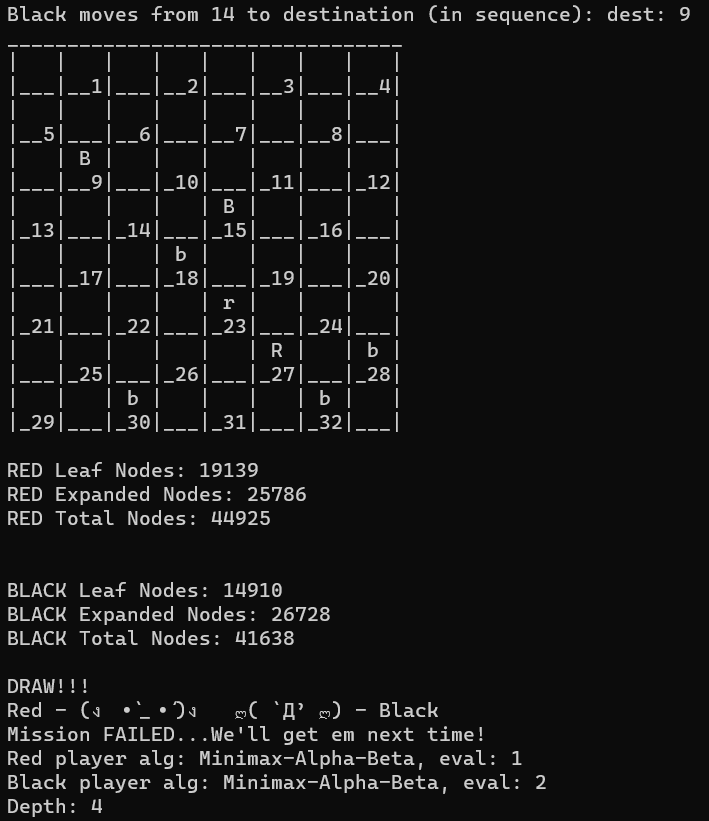


Figure : Minimax-1 vs Minimax-2 Depth 4

## Game 14: (Minmax-A-B, EF-1) vs (Minimax-A-B, EF-3) – Depth 4

Refer to files min-1-min-3-4-full and short.log for full output.

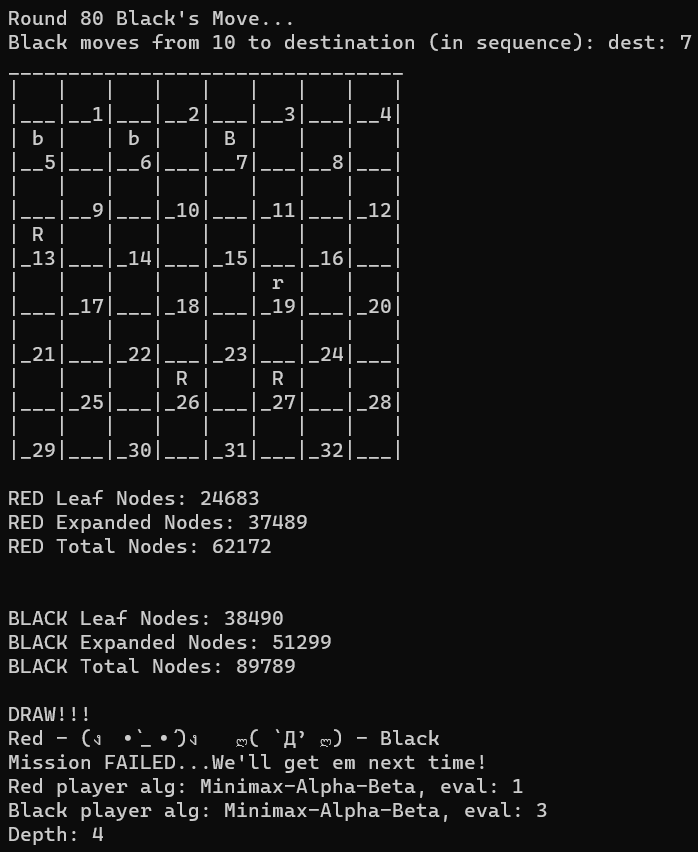


Figure : Minimax-1 vs Minimax-3 Depth 4

## Game 15: (Minmax-A-B, EF-2) vs (Minimax-A-B, EF-3) – Depth 4

Refer to files min-2-min-3-4-full and short.log for full output.

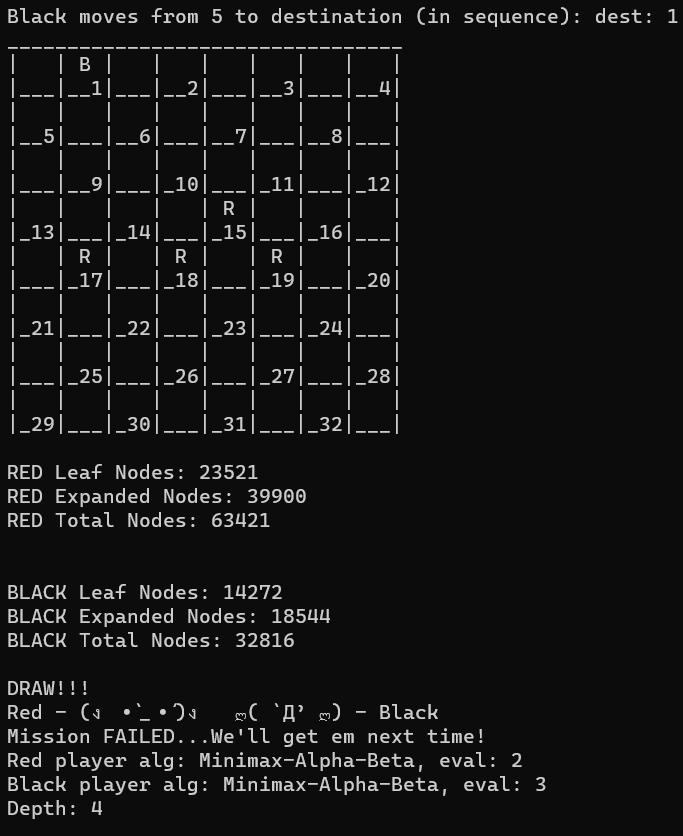


Figure : Minimax-2 vs Minimax-3 Depth 4

## Game 16: (A-B-Search, EF-1) vs (A-B-Search, EF-2) – Depth 4

Refer to files abs-1-abs-2-4-full and short.log for full output.

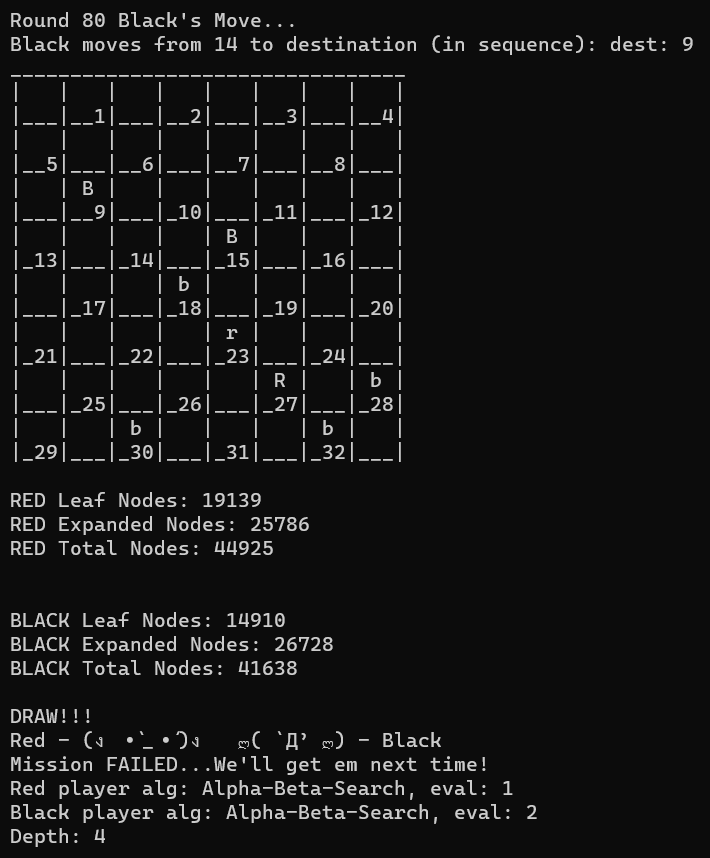


Figure : ABS-1 vs ABS-2 Depth 4

## Game 17: (A-B-Search, EF-1) vs (A-B-Search, EF-3) – Depth 4

Refer to files abs-1-abs-3-4-full and short.log for full output.

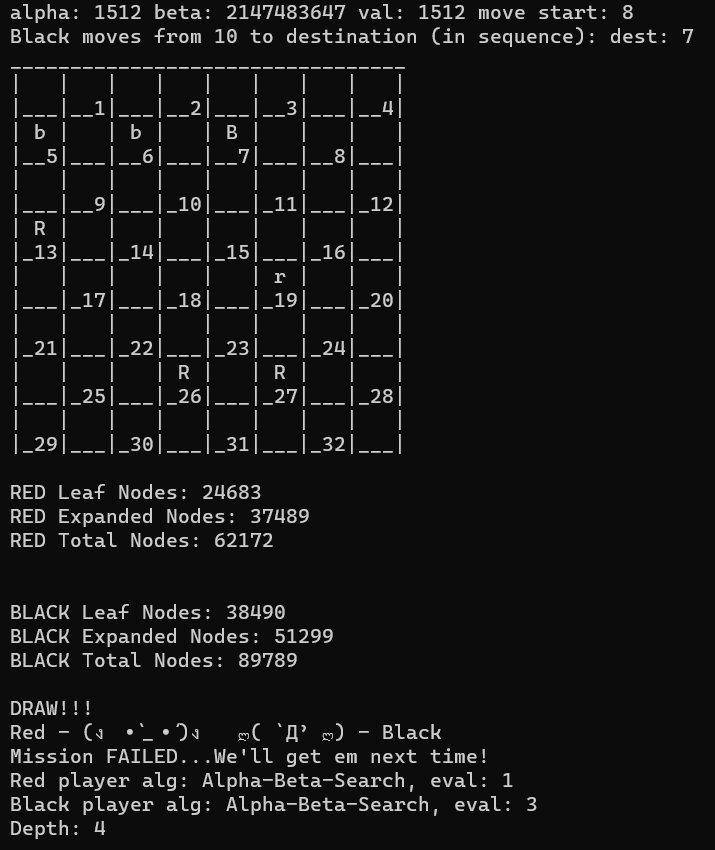


Figure : ABS-1 vs ABS-3 Depth 4

## Game 18: (A-B-Search, EF-2) vs (A-B-Search, EF-3) – Depth 4

Refer to files abs-2-abs-3-4-full and short.log for full output.

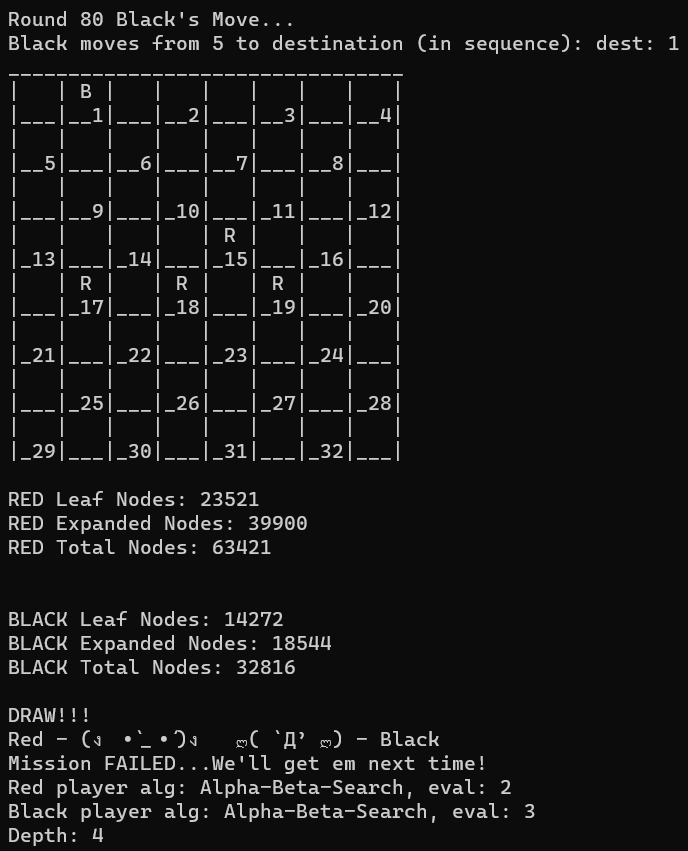


Figure : ABS-2 vs ABS-3 Depth 4

## Full Simulation – 72 Games!

Refer to Project2-fullSimulation-A04626934.log for short output of all 72 games – no text output but the full game path will be shown. Alternatively, expand the Project2-fullSimulationVerbose-A04626934.zip to see the full log for all 72 games with verbose output. Please note that this file is **very** large, over half a gigabyte, and may take some time to open.

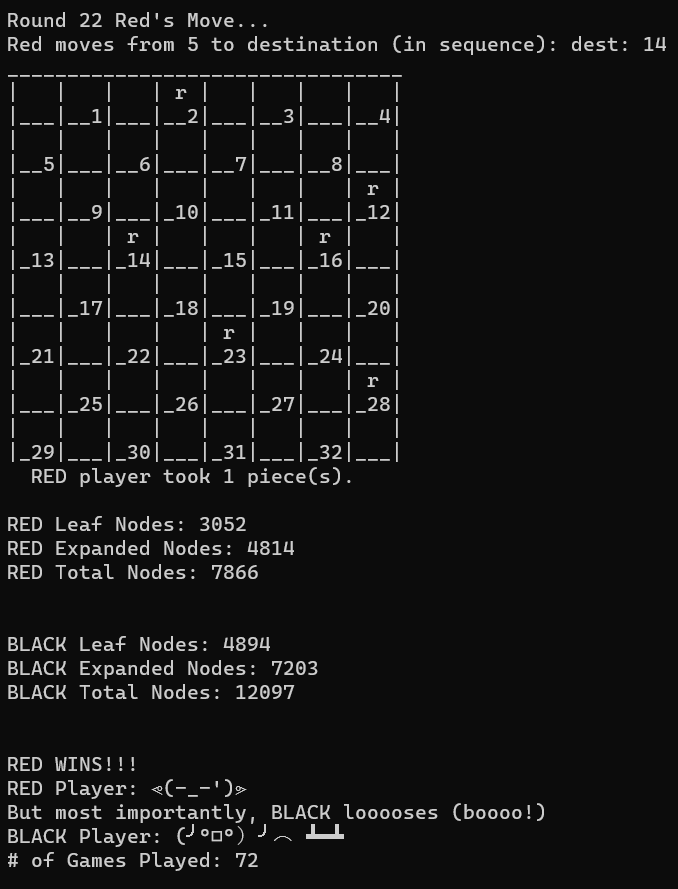


Figure : Full Simulation of 72 total games!3

--- NOTHING FOLLOWS ---