Ho Chi Minh City University of Technology

Faculty of Computer Science and Engineering
Course: Computer Architecture Lab (CO2008)

Assignment

FIVE IN A ROW

Ho Chi Minh City, April 2025

Contents

1 Outcomes

After finishing this assignment, students can proficiently use:

- MARS MIPS simulator.
- Arithmetic & data transfer instructions.
- Conditional branch and unconditional jump instructions.
- Procedures.

2 Introduction

Traditional board games have gone through many iterations throughout history. From the old game of Chess to the newer Uno or Poker, these game captivate their audience with a wide variety of fun yet skillful game play. Among them, Gomoku stood out as a simple yet strategic game known by young and old alike.

Gomoku, or Five in a Row in English, or Caro in Vietnamese, is an abstract strategy board game that has history records from the mid-1700s during the Japanese's Edo period [**?**]. The name "gomoku" is from the Japanese language, in which it is referred to as gomokunarabe. Go means five, moku is a counter word for pieces and narabe means line-up. The game is also popular in China, Korea, and other Asian countries. Ever since its creation, the game has received many changes in rules, creating variations of the game to make it feel fresh and competitive. World Gomoku Championships have been held from 1989 up to now with different host countries and champions. Needless to say, the game has aged like fine wine.

The standard Gomoku involves players alternate turns placing a stone of their color on an empty intersection. Since it is traditionally played with Go pieces (black and white stones), the black player plays first. The winner is the first player to form an unbroken line of five stones of their color horizontally, vertically, or diagonally. In some rules, this line must be exactly five stones long; six or more stones in a row does not count as a win and is called an overline. If the board is completely filled and no one has made a line of 5 stones, then the game ends in a draw. Figure 1 shows a Gomoku game where black wins.

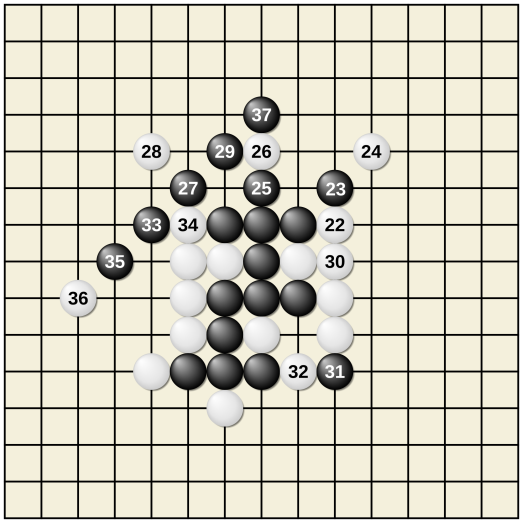

Figure 1: A standard Gomoku game

3 Requirements

As mentioned in section 2, the game has many variants ranging from different first move, to swapping, to winning conditions. For this assignment, the game will be played with a standard 15x15 board where the pieces will be "X" and "O" which are familiar to the Vietnamese people.

Your task will be to recreate the Gomoku game played be two players on a 15x15 board using the standard rule (first to five in a row wins). The five in a row can be vertically, horizontally, or diagonally.

The detailed requirements can be seen as such:

- An empty board is displayed in the terminal at the beginning to showcase the start of the game.
- A prompt for each player is needed to begin a move. The prompt should be: **Player 1, please input your coordinates** and **Player 2, please input your coordinates**.
- The program must then receive a coordinate input in the form of x,y which are the horizontal and vertical coordinates respectively. Example: **4,5** indicates row 4, column 5. Note: coordinates start from **0** and end at **14**. It is imperative that the program automatically checks for any form of incorrect input such as out of range, wrong format, etc. and provide the warning message to the player. A wrong input should be warned and a re-input is required from the player.

- The board updated with the new move must be shown right after the player input. **X** is shown in the place of player 1 moves while **O** represents player 2.
- The winning condition is the first player to get 5 pieces in a line (horizontally, vertically, or diagonally). A tie occurs when all squares have been played yet none of the players can find the winning condition. In the case of a win, a final statement **Player 1 wins** or **Player 2 wins** is shown in the terminal. A tie would instead shows **Tie**.
- The program must also write the final version of the board (either win or tie) and the final statement into a text file name "result.txt"

4 Submission

The assignment must be done in groups of 2-3 people. Students are requested to submit the MIPS program(s)/source code (.asm files) and the Assignment report to BK E-learning system (LMS) in the last lab session of your class group. A live demonstration is also required in the last lab session of your class group (all members must be present).

Students who do not submit on time will get 0 for the assignment.

The report should not contain code. Instead, students should present the algorithms as well as the idea in your implementation.

There will be NO deadline extensions.

5 Plagiarism

Similarity less than 50% in MIPS code is allowed. In other words, you will get 0 for assignment if your answers are similar to another student's more than 50%. Note that, we will use the MOSS tool developed by Stanford for checking similarity (https://theory.stanford.edu/~aiken/moss/).

6 Rubric for evaluation

6.1 Friendly interface - 1 point

Students can design and implement an amicable user interface so that users can use easily without any confusion (1 points).

Students can design and implement a friendly user interface; However, some parts of the program caused confusion (0.5 points).

Student can implement the program but with little to no effort to make the interface match the description (0.25 points).

Student can implement the program but fundamental prompts and format are not available (0 point).

6.2 Application implementation - 6 points

Students can implement an excellent application without any errors found (6 points).

Students can implement a good application with some minor errors, user doesn't need to restart the application to continue (4.0 - 5.5 points).

Students can implement the application with some errors that prevent players from using the calculator (2.0 - 4.0 point).

Students cannot implement the application or lack too many crucial functionalities (0 - 2.0 points).

6.3 Report - 3 points

Students write such an excellent report that others can understand without any difficulty (3 points).

Students write a good report but it's quite simple or lack of some information to clarify the implementation (1.5 - 2.5 points).

Students write a report with a lot of code embedded without any explanation (0.5 - 1.0 points).

Students with no report submitted (0 point).