Reversi Project Checkpoint
Computer Architecture
3340.001 #81317 - Fall 2017

Alexander L. Hayes Henry Forson Hepson Sanchen Jonathan Dubon
{alh170730, hof160030, hms160230, jdd130030}@utdallas.edu

Submitted as part of the requirements for the completion of the CS 3340.001 final project.

Instructions:

Each student must submit a Team Project Checkpoint document to show the project plan and progresses of the project. The document should be prepared as a team effort and every team member must submit the same document prepared by the team.

I: Title Page and Table of Contents

Table of Contents:

II: Instructions and Project Overview:

1. **Goal Statement:** to implement the **Reversi** game using MIPS assembly language.
2. **Submission Requirements:**

   At the end of the semester, **EACH student** must submit a ZIP file containing:

   1. A **project report** (in Word format) covering
      1. a description of the program,
      2. the challenges that you and your team had and how did you or the team overcome them,
      3. what you have learned by doing the project,
      4. a discussion of algorithms and techniques used in the program,
      5. **contributions of each team member (peer evaluation),**
      6. any suggestions you may have (optional).

   2. A **short video clip** demonstrating the program in action. (If the video is too big you can post it on a website, e.g. youTube, and submit the link). The video should have audio narration explaining the moves of players.

   3. **All MIPS assembly language modules that are needed to run your program.**

   4. A **user manual** on how to run and how to use the program.

   Obviously items 2, 3 and 4. are the same for all team members but item 1 must be prepared by each student individually (but some sections of the report, e.g. a) and d), can be shared).

   Make sure you write **what have you learned by doing this project** and **peer evaluation** part individually.

3. **Minimum Requirements:**

   The game is for a human player playing again the computer (i.e. your program).

   1. The user input (keystrokes) is minimum during each turn that the user takes.
   2. The ASCII board is the minimum requirement. Creative ways to display the board will earn the team extra credits.
   3. All moves by the users and the computer MUST be valid according to the rules of the game.
   4. An error message is displayed to explain the rule that was violated if a move by the user was not valid.

III: Capabilities List

3.1: Team Members

1. **Alexander Hayes** is a first-year Ph.D. student in Computer Science. His interests are in inductive logic programming, knowledge representation, and artificial intelligence; and he usually solves problems in Python, Racket, or Bash shell scripts. Upon completion of this project, he wants to better understand how intelligent behavior may be implemented without the abstractions that are common in higher-level languages.
2. **Henry Forson** is a sophomore student seeking double Bachelors in Computer. Science and Software Engineering. His interest are in networking, network security and cybersecurity. He seek to pursue his Masters in Business Administration. Outside of class, Henry is currently married with three kids and a Deacon at Potter's House North church in Frisco.
3. **Hepson Sanchez** is a sophomore currently seeking his Bachelors in Computer Science. His interest in the industry consists primarily of artificial intelligence and is hoping to continue his studies into graduate school with a focus in that topic. Outside of class, Hepson currently serves as the Vice-President of the Society of Hispanic Professional Engineers (SHPE) at UTD and previously has served as the Director of Technology for the Artificial Intelligence Society (AIS) at UTD.
4. **Jonathan Dubon** is a senior currently seeking his Bachelors in Computer Science. Out of his many interests, he is inclined on cybersecurity and artificial intelligence. He seeks to complete his BA and immediately pursue his Masters in CS. He is strongest in the languages of C++, Racket, and Python. Upon the completion of this product, he hopes to understand the implementation of artificial intelligence by the programmer himself.

3.2: Capabilities

Each of the authors are students in Professor Nhut Nguyen's Computer Architecture course (3340.001), with interests in computer science and software systems. In compliance with the *course objectives*, each of the authors should upon successful completion of this course be capable of: writing a fully functional, stand-alone medium size assembly program; be able to program efficiently in assembly level instruction set, including the use of addressing modes and data types.

Being students in this course, each with a sufficient background in computer science, lends the authors the desire to complete the objectives in Section II properly. And after practice on homework assignments and outside work, the authors are capable of successfully completing this project as part of the course's learning objectives.

With reliable knowledge of various high-level languages and working knowledge of MIPS assembly language, the authors will not only meet the expectations of the *course*

objectives through the project, but exceed them through strong work ethic, constant communication, and a team mindset. Progress towards completion of the project, as a team, will consist of primarily four stages, of which each and everyone of the authors is well aware of. Those stages consist of the following: Forming, Storming, Norming, and Performing.

Forming is the current stage, where the authors are focused on clarifying roles and maintaining organization. Storming will be the next stage, consisting of brainstorming and a variety of new ideas being presented. The third stage, Norming, will involve those ideas being evaluated and final objectives being understood. Performing is the final stage of a team's progress and is where the members operate with little oversight and great coordination.

IV: Methods

4.1: Code

All code for the Reversi implementation will be written in MIPS assembly. The methods for this include, but may not be limited to:

- Starting a new game. Randomly choosing which player (human or machine) goes first.
- Showing the current board state (i.e. printing the 8x8 grid and any pieces that occupy a square on the grid).
- Receiving input from the human by asking for the coordinates they want to put their piece on.
- Determining whether a given move is valid by returning true or false if a piece can be placed on a square or cannot be placed on a square.
- An error message if the user tries to make an invalid move, subsequently followed up with another prompt for an action.
- Updating the board state by placing a piece and flipping the appropriate states.
- One of the following computer strategies (possibly a combination, or possibly multiple and the human player can choose the difficult of his or her opponent):
    - Easy: Computer randomly chooses a valid square as their next action.
    - Medium: Choosing an action based on known Reversi strategies (ranking choices between corners, edges, or other openings).
    - Medium-Hard: precomputing end-game strategies (i.e. the best move for common end-game board states).
    - Hard: Calculating the optimal (or less than suboptimal) subsequent computer move using a minimax game tree and alpha-beta pruning under depth-limited tree search.

4.2: Documentation

1. Individual project reports (described in Section II).
    - Formatting will be similar to this document for common sections.
2. A short video clip (described in Section II).
    - No longer than two minutes, well-edited, and containing quick-start instructions.
3. A short user manual and an appendix for developers interested in design decisions.
    - Similar in spirit with the short video clip, but focusing on the quick-start instructions with more advanced topics for the more advanced user.
4. Code will be hosted on GitHub, including all documentation for this project.

V: Implementation Timeline

An initial version of Reversi will be implemented in a higher-level language agreed to by members of the group, likely in Python, Racket, or C++. This will give the the authors a high-level understanding of implementing the game before it is finally written in MIPS assembly. Any code, pseudocode, scratch work, drawings, or discussions will be included in the User Manual Appendix (described in Section 4.2.3).

The final project is due on Thursday, November 30, 2017. The following is the timeline the group members will follow leading up to the due date.

Date	Tasks to be completed by this date
October 22	Project Checkpoint. Common agreement on list of tasks between group members. Familiarity with playing Reversi.
October 29	Pseudocode for methods described in Section 4.1. Some methods implemented in the higher-level language.
November 5	Code written for beginning a new game.. Board state can be printed using MIPS assembly (output). User can interact by typing coordinates (input).
November 12	Player's choices update the board state. Computer player can choose actions at random. All actions chosen by either player are valid.
November 19	Computer player behaves in an intelligent manner through increasing its lookahead or using random moves guided by heuristics (strategy).
November 26	Final documentation: creating/editing video, compiling documentation, distributing documentation and code on GitHub and uploading video to YouTube.
November 30	Submission and presentation of work.

VI: Conclusion

This document serves as the project checkpoint for the authors. Any revisions will be noted in the final submission document; including challenges (and hopefully how they were overcame), assumptions that were made along the way, and compromises that needed to be made for the sake of a better experience. These will be framed in the context of the course objectives and general MIPS paradigms described in class.

VII: Acknowledgements

The authors wish to thank Professor Nhut Nguyen and teaching assistant Munawara (Munia) Saiyara for their support this semester and stimulating the collective intellect of the authors by providing this project as a challenge to learn and grow from.

> “The most valuable tools and acquisitions in scientific or technical education are the general-purpose mental tools which remain serviceable for a lifetime.”
>
> - George Forsythe (1968)

> “The road is *challenging* not because of the mountains and rivers that are in your way, but *because your mind is afraid of those rivers and mountains*.”
>
> - Vietnamese proverb