**CSCI 312 – Artificial Intelligence**

*Othello Project*

**Project Objectives:**

1. Students obtain experience representing knowledge.

2. Students will experiment with heuristic development.

3. Students will discover the problems, benefits, and various methods of adversarial search.

4. Students will have the experience of completing a fairly large scale project with a very short deadline.

**Othello -- The Game:**

*Materials (and beings) required.*

1. An 8 x 8 square board (64 squares).

2. 64 bi-color disks (one side black, the other white).

3. 2 players.

*Objective.*

To end the game with more of your disks on the board than your opponent.

*Starting Game.*

1. Players determine colors. Black moves first.

2. White pieces on d4 and e5, Black pieces on d5 and e4.

*Legal Moves.*

Place a disk in a free square. At least one of your opponent's disks must lie in a horizontal, vertical, or diagonal line between one of your existing disks on board and the square to which you wish to move. Your opponent's disks that are "caught" between one of your existing disks and your newly placed disk are changed to your color.

*Rules.*

1. Board is 8 x 8; columns labeled a-h from (left to right); rows labeled 1-8 from (top to bottom); moves and cells are defined by column and row. Examples: d5, f8.

2. If you can not make a legal move, you must pass.

3. You can not pass if you have a legal move.

4. When neither player can make a move, the game ends.

5. Each player has 10 minutes of wall-clock time for entire game. If player takes more than allotted time, he/she forfeits game.

6. If a player attempts to make a wrong move, he/she forfeits the game.

7. The player with the most disks on board wins.

**Othello -- Your implementation:**

*General Instructions.*

You are to write a C, C++, or Java program that will play Othello according to the above rules. Your program will be called by a referee program created by the instructor.

*Implementation Specifications.*

1. Program Input. Your program should accept the following as input:

I B -- Inform program to initialize and play as BLACK

I W -- Inform program to initialize and play as WHITE

B c r -- Inform program that BLACK moves to column c and row r.

Example: B a 1 means Black moves to position a1

B -- Inform program that BLACK passes

W c r -- Inform program that WHITE moves to column c and row r.

W -- Inform program that WHITE passes

C string -- A comment has been made. Referee is to ignore!!

n -- Other program claiming end game!!! Check it out!!!

n is the number of black pieces on board.

1. Program output.

R B -- Indicate program ready to play as BLACK

R W -- Indicates program is ready to play as WHITE

B c r -- BLACK moves to column c and row r. Use this format if your agent is to play as BLACK.

Example: B a 1 means Black moves to position a1

W c r -- WHITE moves to column c and row r. Use this format if your agent is to play as WHITE.

B -- BLACK passes

W -- WHITE passes

C string -- A comment has been made. Ignore!!

n -- Other program claiming end game!!! Check it out!!!

n is the number of black pieces on board.

1. Please document your modules well. Give brief statements about a module's function. Give "English" definitions of input variables. Define exactly what is to be returned by a function (if anything is to be return.) Documentation is extremely beneficial on your evaluation function. Also, document your choice of representations -- board, moves, directions, etc.
2. Write a technical report -- Discuss your choice of representations, search methods, and heuristics. Establish your test plan. Discuss problems encountered. Analyze tournament results. See section on deadlines.
3. Submissions
   * 1. Algorithms and synopsis are to be uploaded to Moodle
     2. TestRun submission
        1. Java Programs
           1. Create a jar file using Eclipse.

Right-click on the Project name

Select Export

Select Java Folder

Select Runnable JAR file

* + - * 1. Rename your jar file to accountNameOthello.jar where accountName is your email account ….for example my would be mcameronOthello.jar
        2. Upload to ~mcameron/CSCI312/Othello/Fall2019/TestRuns.
      1. C or C++ Programs
         1. Create an executable file … accountNameOthello.out
         2. Upload to ~mcameron/CSCI312/Othello/Fall2019/TestRuns
      2. Python Programs
         1. Create an executable py file… accountNameOthello.py
         2. Upload to ~mcameron/CSCI312/Othello/Fall2019/TestRuns
      3. EVERYONE -- create a script file to call your program called

accountNameOthello

* + - * 1. Don't forget to make the script file executable … for example….

chmod 755 mcameronOthello

* + - 1. Tourney submissions – similar to TestRun Submission just upload to ~mcameron/CSCI312/Othello/Fall2019/Tourney instead.
    1. Running Referee – at Linux prompt type…..

Referee3 *blackPlayer whitePlayer*

where *blackPlayer* is the script for the player that is playing as black and *whitePlayer* is the script playing as white.

to run in background:

(Referee3 *blackPlayer* *whitePlayer* >& *fileForStdErr*)&

Review your games in ~mcameron/CSCI312/Othello/Fall2019/Tourney/LogFiles

**Othello -- Deadlines:**

Sept 6 Algorithm and Representations of board and moves Draft 1.

Sept 13 Algorithm and Representations of board and moves completed.

Sept 27 Agent plays a legal game of Othello with Referee. Regional Tourney 1

October 4 Regional Tourney 2

October 11 Regional Tourney 3

October 18 Sectional Tourney

November 1 Test Tournament

November 24 Finalize entrants into Thanksgiving Tournament. Have entries in TestRuns by 11:59pm

November 23.

***November 27***  Tournament. Each region must have entries in by ***11:59pm.***

***December 3 Last Day to submit an OthelloBot that plays a legal game of Othello (11:59pm)***

***December 5*** Presentations from each Region

**Othello -- The Tournaments**

1. Tournaments will occur according to the above schedule. You may make changes to your program between the test and final rounds. You may challenge each other prior to each tournament.
2. Place your entries into your regional directory in /home/mcameron/CSCI312/Othello/Fall2019/TestRuns

3. Tournament results will be posted via Moodle.

4. You may change your entrant at anytime. Just insure that your entrant has no syntax or runtime errors.

5. Winners will be determined by most wins. In case of ties, disk count will be used.

**Othello -- Evaluation & Motivation**

This project counts 25% of your course grade. Grading will be based on the following:

|  |  |
| --- | --- |
| **Points (250 total)** | **Task** |
| **20** | Algorithm. 3% subtracted for each day late. |
| **25\*** | Plays legal Othello. Program does not make invalid moves, updates board appropriately, and provides an interface that interacts with the instructor’s Referee. If game does not play a legal game, you will earn a failing grade on this project. |
| **15** | Documentation and Style. |
| **25** | Search method and progressive deepening |
| **25** | Evaluation Function |
| **10** | Timing Implementation |
| **25** | Test Plan |
| **40** | Synopsis |
| **15** | Tournament Performance |
| **25** | Group Work |
| **25** | Regional presentation |

Revised Evaluation to encourage higher individual completion rates:

|  |  |
| --- | --- |
| **Points (250 total)** | **Task** |
| **20** | Algorithm. 3% subtracted for each day late. |
| **75\*** | Plays legal Othello. Program does not make invalid moves, updates board appropriately, and provides an interface that interacts with the instructor’s Referee. If game does not play a legal game, you will earn a failing grade on this project. Each OthelloBot will be tested for legal game according to the following process:   1. Bot must play a match against itself without making an illegal move. 2. Bot must play as Black in a match against DrMecBit90sec. 3. Bot must play as White in a match against DrMec Bit90sec. |
| **75** | AI techniques: Search method, progressive deepening, evaluation function, Timing ...If the bot does not play a legal game the grade for this component is zero. |
| **30** | Synopsis, Documentation, and Style.Tournament Presenters may elect to submit their presentation for grading in place of Synopsis. |
| **25** | Tournament Performance. Submissions into the final tournament receive a score calculated as 15 + 10 \* (1-(Place-1)/TournamentEntries). Games that play a legal game will receive a score of 10 for this component. Games that don't play a legal game will receive a score of zero for this component. |
| **25** | Final Presentation Attendance |