CPSC-315 – Programming Studio
Team Project 2: Game Playing through Search and Learning

Project Goal:
Your team is to build a system for playing the game Reversi (also known as Othello). Rather than restating the rules here, you can refer to the (relatively simple) rules at Wikipedia (http://en.wikipedia.org/wiki/Reversi) or another website. You should be able to play on an *8* x *8* board. Starting positions are given by two integers, (*a*,*b*), and the game starts with black pieces at (4, 5) and (5, 4), and white pieces at (4, 4) and (5, 5). Black moves first.

Specifically, your team should create:

- A program to "manage" a game. This program should maintain the board state, call other functions (either from interactive input or from an AI routine) to get moves, check to ensure valid moves, and determine if there's a winner.
- An AI that will play the game
- A learning agent that will (hopefully) improve its performance during gameplay. The learning agent MUST be able to print out a human readable, English language description of its reasoning.
- An interface to get user input.

For this project all teams must use the package name team<number> (e.g. "team1" if you are team 1) for all of your source files or a sub-package of that package. In addition each team will create two classes team<number>.AI and team<number>.LearningAI, such that the AI class is a (non-abstract) subclass of project2.AI and the Learning class is a (non-abstract) subclass of project2.LearningAI class (both project2.AI and project2.LearningAI are provided and should not be included when the project is turned in). Your team's AI class will implement easy, medium, and hard Othello playing strategies based on a min-max tree strategy. The LearningAI class will implement an artificial player that improves its "goodness" function for the min-max tree over time and is also capable of producing an English language description of the learning systems reasoning.

This is the basic functionality required. All of these aspects can be improved in numerous ways. Some examples include the display of the board/game state, the level to which the AI works, the way that input is taken from the user, etc. The AI in particular can be improved in many ways.

Note that any two-person team will have much lower expectations in this application program than a three-person team would.

Team Organization:

Each project team has two or three members, one of which is the "project leader" and has additional responsibilities. Each month-long team project will consist of three phases:

Week 1: Project leader develops high-level specification of Java objects to be instantiated and identifies which team member will be responsible for each object. This specification takes the form of the names of the objects, their methods (and parameters), and any externally available constants or variables and comments describing each of the above.

Weeks 2-3: Team members implement their assigned software components, communicating as needed to ensure necessary changes to the original specification are known to all team members.

Week 4: All team members work to pull together their components into the final, working system and generate all necessary documentation for the project.

After the project: Each team member will be asked to assess the work of the other team members.

For this assignment, the teams will be as follows (project leaders are listed in italics):

Team 1:
Christopher Weldon
Jacob Lillard
Jorge Cereijo-Perez

Team 2:
Jillian Graczek
Brandon Jarratt
Julio Montero

Team 3:
Matthew Moss
Andrew Johnson
Brett Hlavinka

Team 4:
Thomas Robbins
Gabriel Copley
Zachary Edens

Team 5:
Christopher Aikens
Ryan Mcauley
Mark Hill

Team 6:
Drew Havard
Andrew Reagan

Team 7:
Kourtney Kebodeaux
Robert Kern
Travis Kosarek

Team 8:
Christopher Bennett
Benjamin Unsworth
Jeffrey Deuel

Team 9:
Luke Hersman
John Laky
Patrick Robinson

Code Organization:
This project is to be completed in Java.

The project leader is required to maintain its development using a version control system. You should choose and set up an SVN repository/project site in which your team will keep current versions of source code, documentation, etc. This should be one of the first decisions your team makes. CodePlex, Google Code, and Sourceforge are suggested locations for these repositories, but you may choose another option if you would prefer. Note that your team is to make use of this repository for all development. You should not pass around files by email, saving in shared directories, etc. Significant points will be taken off of the project if code is shared via a method other than the SVN system. (You may use a version control system other than SVN, if you would prefer).

Your project leader should give access to both the Instructor and the TA to the SVN repository for download. Your project leader is required to send email to both giving the location of your team's project, and instructions for access to the SVN repository (this should be done by the time of Project Update 1, below). The instructor/TA may download SVN software at any point.

Intermediate Updates:

Organization Statement: Your project leader is to turn in a written update regarding the overall project design. Specifically, your report should have 3 main components:

- A brief summary of what system you are using for version control, and a short justification (1/2 page) for why your team chose that option. By this point, you should have emailed the professor and TA with details on your SVN repository, and any details they need regarding access.
- A statement describing how your team will be organized and any responsibilities assigned to team members. You may use any organization, but you should give a justification/reasoning behind whatever you have chosen. This summary should be ½ to 1 page in length.
- A brief (1 to at most 2 pages) summary of the approach your team will be taking for implementation of the project. It is recommended that you try to identify the major implementation issues, set intermediate deadlines for the team (building in some slack time), including for production of documentation, and assign responsibilities to members (at minimum, for the initial work)

Project Grading:

The overall project will be graded as described below. This grade will be used to determine the team grade, and individual grades will be determined by apportioning the team grade among the team members.
15% Final Application Documentation
40% Completeness/Correctness/Quality of Play of search-based AI player
35% Functionality/Extent/Correct Operation of learning-based AI program (including explanation of decision making)
10% Code style: naming/layout/commenting

The project leader will be graded as described below.
20% Source system selection & explanation of use
30% Design of Application – How complete is the design? Is it specified clearly for team members.
30% Description of Team Organization (who is responsible for what)
20% How well the team worked (from individual reports)

Dates and Deadlines:

The following are the intermediate deadlines for this project. Details of the particular requirements are described above. Note that these are final deadlines; it is likely that your team would want to complete several of these (e.g. the two Project Updates) well in advance of the "due date"

Tuesday, March 3
Project Assigned
Tuesday, March 10, 11:59 p.m.
Email Instructor and TA regarding SVN repository
Project Leader Report (source system, team organization, and API/application design/code stumps)
Thursday, April 2, 11:59 p.m.
All final project code turned in
Project documentation turned in along with code

One day after code and documentation turned in (or **Friday, April 3**, 11:59 p.m.)
Individual Reports on teamwork turned in