CSCI 561: Foundations of Artificial Intelligence
Instructor: Prof. Laurent Itti
Homework #2: Game Playing
Due on March 10 at 11:59pm, Los Angeles time, 2014

Suppose Alice is playing Connect Four against Bill. Given a current game configuration, you will help Alice to determine the next move that will maximize her chance of winning. You will use minimax algorithm with alpha-beta pruning to calculate the optimal move.

Connect Four: http://en.wikipedia.org/wiki/Connect_Four

Connect Four is played by two opponents, alternating moves on a 7x6 rectangular grid, with 7 columns and 6 rows. The two players take turns dropping one of their pieces into any of the seven columns, where the pieces will fall until it hits the bottom of the column or the topmost piece already in the column. A player wins by creating a line consisting of 4 of his/her pieces, which can be vertical, horizontal, or diagonal. If the board is filled and neither player has won, the game is a draw.

Programming Task

Alice and Bill are playing the game, and it is Alice's turn to take next move. You will write a program to determine the optimal next move for Alice. Your program will take the current board configuration as input. The input file path will be given as a command-line parameter (see grading). The input will always be valid. For example (input.txt):

```
e
abba
baa
e
abbaab
ab
b
```

This input is "rotated sideways" to make scanning it easier. In other words, each line of text corresponds to a column in the grid. The first line represents the left-most column. The interpretations are

'e' denotes an empty column.
'a' denotes Alice's piece.
'b' denotes Bill's piece.

Your program will use **minimax** algorithm with **alpha-beta pruning** to determine Alice's next move that yields the best utility for her. **For simplicity, you do not need to search all possible states until the end of the game. In this assignment, you only need to consider all possible states for the next 4 moves (2 alternating moves for each player).**

Your heuristic function will be defined as follows:

- If Alice has won, the value is 1000**.**
- If Alice has lost, the value is -1000**.**
- If the game is a draw, the value is 0.
- Otherwise, the value is (xA - xB) + 5 * (yA - yB), where
- xA and xB are the number of unterminated lines of length 2 for Alice and Bill
- yA and yB are the number of unterminated lines of length 3 for Alice and Bill
- By unterminated line, we mean a certain number of pieces of a single player are in a row, and there is a free space on at least one side of the line (i.e. it is still possible for this line to be part of a winning line).

You will want to help Alice choose a move that maximizes this heuristic value. If there are several moves that result in the same most utility, output the first one (we will parse the moves from the left column to the right). We assume Bill is rational too that he always choose a move that minimizes this heuristic value.

The output of your program will be the next optimal move for Alice, along with a printout of your minimax tree. It is achieved by traversing all possible search states for the next 4 moves using minimax algorithm with alpha-beta pruning. Below is a sample output, but is just an arbitrary example and is ***not*** necessarily correct (in fact, most of the time there will be 7 actions, one for each column):

```
A1: 2
|-B2: 2
|-|-A3: 1
|-|-|-B4: 2; h=0
|-|-|-B4: 3; h=5
|-|-A3: 3
|-|-|-B4: 4; h=0
|-|-A3: pruning 4, 5, 6, 7; alpha=1, beta=1
|-B2: 3; h = 1000
A1: 3
...

first move: 3
```

The interpretations are:

1. The output will be the traversal of possible search states, but not the optimal search path. Each move could be followed by several possible moves, we use "**|-**" to represent the traversal relationships of the two consecutive moves.
2. Each line is in the format of "**Pi: c**", where **P** could be **A** (Alice) or **B** (Bill), **i=1, 2, 3** or **4** denoting the following ***i***th move, **c** is the column number (1-7, 1 being the leftmost column).
3. If it is the ***4***th move, please output heuristic value after (e.g., **B4: 7; h=5**).
4. If the move involves alpha-beta pruning, please output all branch moves that should be pruned and the values of alpha and beta, e.g., **pruning 4, 5, 6, 7; alpha=1, beta=1**
5. After traversing all possible states, output the answer of Alice's first move, e.g., **first move: 1**

This output should be contained in a .txt file. The output file path will be given as a parameter (see grading).

Grading:

The grader will compile your program using the following command, for examples:

C++: g++ *.cpp -o hw2

Java: javac *.java

And execute your program using, for examples for each experiment:

C++: ./hw2 input.txt output.txt

Java: java hw2 input.txt output.txt

Deliverables:

You are required to hand in all code that you wrote to complete this assignment; implementation language not important. However, if you code in C++, C#, or Java, the TA will be better able to assist you ☺. **Also, we require that your code be able to run from the command line on the USC aludra.usc.edu Linux servers.** For information on accessing aludra, please see https://itservices.usc.edu/web/hosting/students/. Please include your source code, any additional files needed to compile it (makefiles, etc.), and a readme.txt to describe how to run your code, including the command(s) to run it.

The deadline for this assignment is **Monday March 10, at 11:59pm** Los Angeles time. Please turn in all materials as a .zip file via Blackboard, with the title format [firstname]_[lastname]_HW2.zip (e.g., Harry_Potter_HW2.zip).