**Artificial Intelligence**

Name and ID Osniel Quintana

U49838358

**Assignment 3**

**Due date: March 27 at 11:59pm**

You may work alone or in groups of two.

**Problem 1** (4 points)

Read Chapters 4 and 5 of the textbook and answer the following questions.

1. Explain the main differences between puzzle-solving search and game playing.

Puzzle-solving search involves the search of an optimal path to a goal state, therefore the “winning” or desired state is already known to us. In puzzle-solving search the algorithm moves through a predefined or recently constructed graph (or tree) and chooses the best next move depending on how “close” or “likely” it is to reach the goal. In game playing that is not the case, the algorithm needs to create a contingency strategy to overcome the other player’s move and win. Game playing involves the active interaction of two or more agents while puzzle-solving is usually in a one agent environment, although the environment might be stochastic. Also for game playing there is no goal state and the outcome is undetermined until reached.

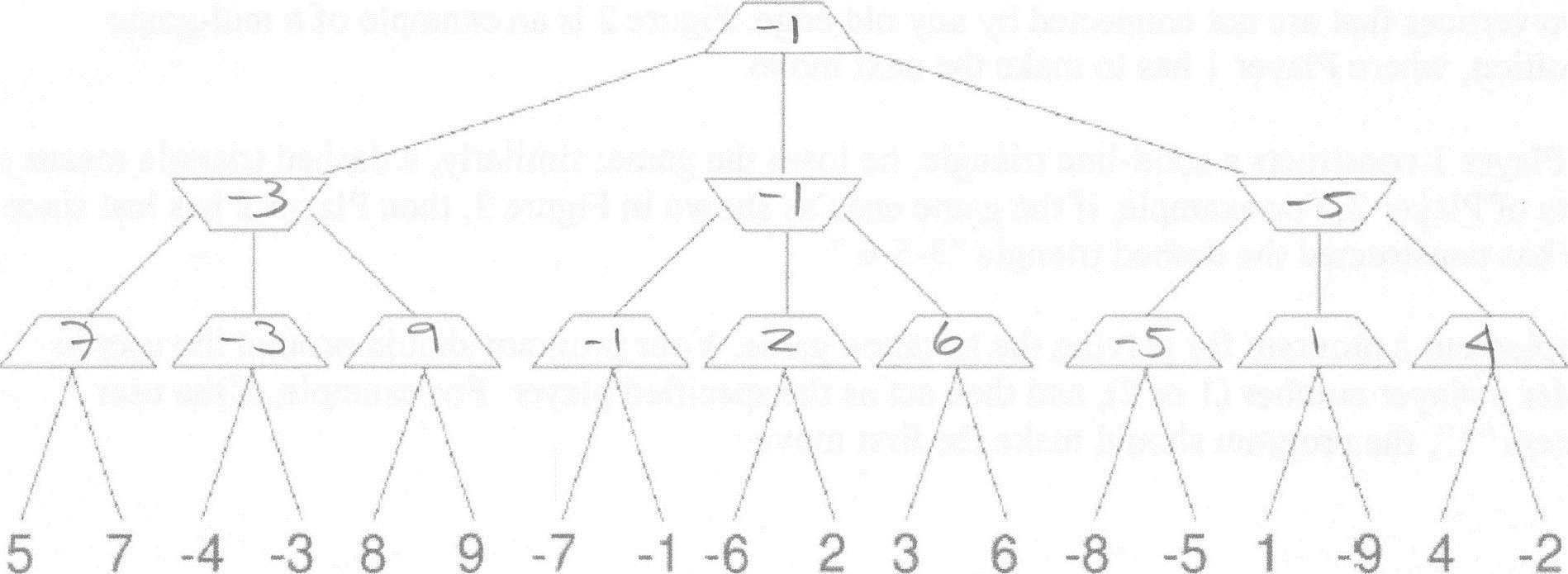
**(b)** Argue that game playing is inherently more difficult than solving one-player puzzles.

Game playing is by default more difficult than one-player puzzles since the outcome of the game is not defined. At every move, the algorithm has to come up with a better solution, therefore it has to compute another solution again. Furthermore, game playing derives from one-player puzzle, that is, it takes a similar problem but adds more agents to it. In one-player puzzle the outcome is already predefined, the goal is available and all possible paths to that goal become available at some point; with game playing the possible paths have a dynamic behavior: constantly changing depending on second’s player move.

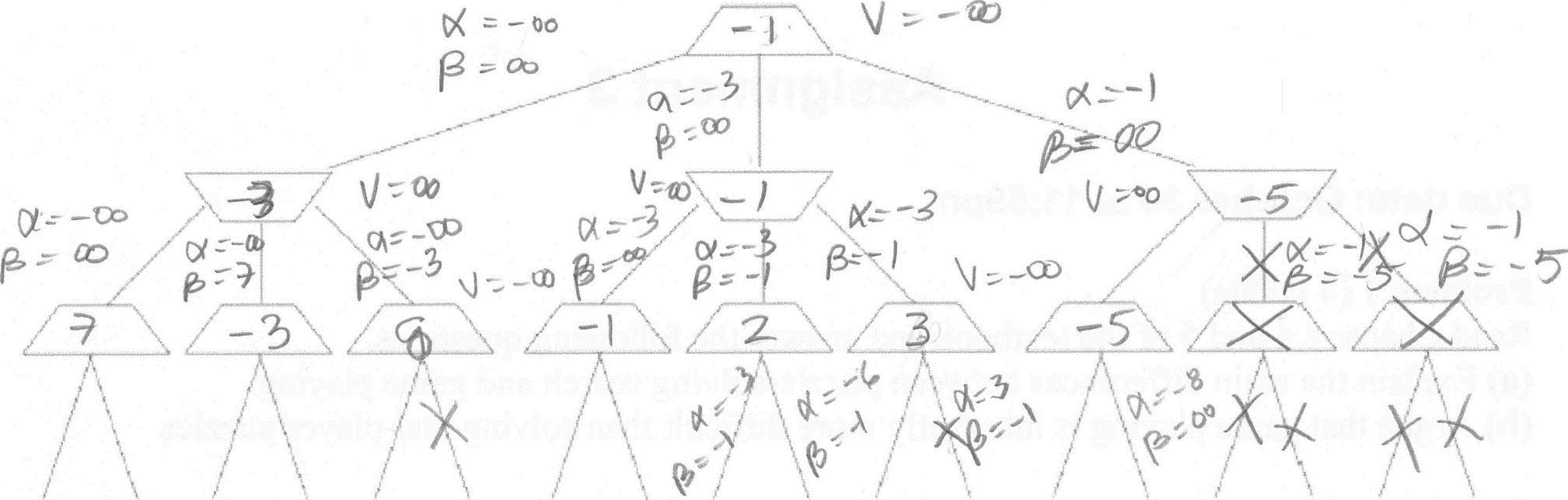
**Problem 2 (6 points)**

Consider the zero-sum game tree shown below. Trapezoids that point up, such as at the root, represent choices for the player seeking to maximize; trapezoids that point down represent choices for the minimizer.

(a) Assuming both opponents act optimally, carry out the minimax search algorithm. Write the value of each node inside the corresponding trapezoid and highlight the action the maximizer would take in the tree.



(b) Now reconsider the same game tree, but use alpha-beta pruning. Expand successors from left to right. Record the alpha-beta pair that is passed down that edge (through a call to MIN-VALUE or MAX-VALUE). Circle all leaf nodes that are visited. Put an `X' through edges that are pruned off.



(c) True / False. Minimax and alpha-beta pruning are guaranteed to find the same value of the top node.

**Answer: True**

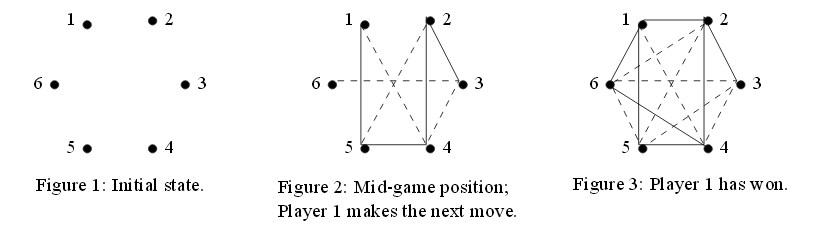
**Problem 3** (20 points)

The *hexagon game* involves two players, who gradually construct a six-vertex undirected graph with solid and dashed edges. Player 1 adds solid edges, whereas Player 2 uses dashes.

The players begin with a six-vertex graph that has no edges (Figure 1), and add new edges, one by one; Player 1 makes the first move. At each move, a player has to add a new edge between two vertices that are not connected by any old edge. Figure 2 is an example of a mid-game position, where Player 1 has to make the next move.

If Player 1 constructs a solid-line triangle, he loses the game; similarly, a dashed triangle means a loss of Player 2. For example, if the game ends as shown in Figure 3, then Player 2 has lost since he has constructed the dashed triangle “3-5-6.”

Implement a program for playing the hexagon game. Your program should prompt the user to enter a player number (1 or 2), and then act as the specified player. For example, if the user enters “1”, the program should make the first move.



**Submitting your assignment**

* Submission via Canvas Assignment.
  + It is your responsibility to submit these assignments in a timely fashion.
* All files should be zipped together.
* Name of your file must include your last name and ID number
* There should be a readme file explaining in detail the exact steps to be taken to compile and execute the code files and the title page
* In case of any code errors, partial credit may be offered based on the code and documentation.
* A report that presents the performance evaluation of your solution.

**Late Submission Policy**

* Late work will be not accepted.

**Grading Criteria:**

* Minus 20 points if code does not compile.
* Minus 15 points if it compiles but does not run.
* Further deduction will be depended on your code