Homework #2: Chapters 3 and 5

1. **Greedy [10 points]**
   - 1 pt per node (6), including 1/2 point for h() values
   - 2 pts total for correct expansion order labeling (-1 if Harrisburg is 1, instead of Philly)
   - up to -2 for additional generated nodes, extra expansions

Note that because this is a graph search, when we expand Harrisburg, we do not add Philadelphia to the frontier because it is in the explored list, and we do not add Allentown to the frontier because it is already on the frontier. It would be legal to show these nodes, as long as it was clear that they could never be expanded (perhaps by crossing them out).

The path is: Phil → Harr → Pitt → Erie
*(note this path has a cost of 438)*
A* [16 points]
- 2pts. per node (7 nodes), including 1 point for costs, and correct expansion label
- up to -2 for additional nodes

Once again since this is a graph search, some nodes are generated but not added to the frontier because they are redundant. I have chosen not to show such nodes, but it is legal to show them as long as it is clearly indicated that they can never be considered for expansion. Such nodes are generating Allentown from Harrisburg (since a cheaper Allentown is already on the frontier), Harrisburg from Allentown (since Harrisburg was already expanded), and Harrisburg from Scranton (since Harrisburg was already expanded). Also note the Scranton node with a red “X”. This node must be generated when Harrisburg is expanded (#2). However, when we go to expand Allentown (#3) we find a Scranton node with a lower f(n) (386 vs. 478). This node must replace the more expansive Scranton node on the frontier. Otherwise we would have a suboptimal solution to the problem.

The path is: Phil → Allen → Scran → Erie
(note this path is cheaper than the one discovered by greedy)
Discussion  [4 pts.]
The two algorithms picked two completely different routes. Greedy best first search is fast, finding a path to Erie after expanding just 4 nodes, but it’s not the optimal route in terms of driving distance. A* search found the optimal path, but had to expand 6 nodes. In addition, A* search needs more space to hold its frontier nodes.
2. [20 pts.]
1 pt per node (11 total), costs are worth ½ of these points
1 pt per expansion order (6)
up to -3 for incorrect expansions, additional nodes
if an error is simply due to propagation of an earlier mistake, only take of 50% for it
3. [10 pts] There are 9 places to put the first mark (an X), 8 places to put the next mark (an O), 7 to put the next mark (an X), etc. Therefore, there are up to 9! = 362,880 possible games. But, many games will never involve 9 moves because the game will terminate before the 9th move if either X or O get three in a row earlier. It is not possible to have a winner until one player has moved at least 3 times (5 total moves between the players). Therefore, there are at least 9*8*7*6*5 = 15,120 possible games. For the purpose of approximation, it suffices to say that there are between 15,120 and 362,880 possible games. Significantly more complex calculations (or brute force) would be required to get an exact count.

4. a) [15 pts]
   1 pts each node, utilities are ½ of these points
   up to -2 for additional nodes

   Board 1
   X | O | X
   O | O | X
   X | 10

   X’s Optimal move
4b) [25 pts.] 1pt for each node except the root, check all utilities (1/2 pt each)

X’s Optimal move

---

**Board 2**

**MAX**

- A
  - X
  - O
  - X
  - X
  - -1

**MIN**

- A’s children
  - B
    - X
    - O
    - X
    - X
    - -7
  - C
    - X
    - O
    - X
    - X
    - -7

**B’s children**

- D
  - X
  - O
  - X
  - X
  - -7
- E
  - O
  - X
  - X
  - -7

**A’s children**

- X
  - O
  - X
  - O
  - X
  - O
  - 3(1)+1-(1)
  - 3(1)+1-(1)
  - 3(1)+2-3(1)
  - 2-3(1)
  - = 3
  - = 3
  - = 2
  - = -1

**B’s children**

- X
  - O
  - X
  - O
  - X
  - O
  - 3+1-(10+1)
  - 3+1-(3+1)
  - 3-(3(2))
  - 3+1-(3(2))
  - = -7
  - = 0
  - = -3
  - = -2

**C’s children**

- O
  - X
  - O
  - X
  - O
  - 3+1-(10+1)
  - 3+1-(3+1)
  - -3(2)
  - 3+1-(3(2))
  - = -7
  - = 0
  - = -1
  - = -4

**D’s children**

- O
  - X
  - O
  - X
  - O
  - 3+1-(10+1)
  - 2-(3(1)+1)
  - 3+1-(3+1)
  - 3(1)-(3(1))
  - = -7
  - = -2
  - = 0
  - = 0

**E’s children**

- O
  - X
  - O
  - X
  - O
  - 3+1-(10+1)
  - 3(2)-(3(1)+1)
  - 3+1-(3+1)
  - 3(2)-3(1)
  - = -7
  - = 2
  - = 0
  - = 3