Homework #2: Chapters 4 and 6

The following exercises are due at the beginning of class on February 13.

1. *[15 points]* Use greedy best-first search to find a path from Vaslui to Sibiu, assuming the roads between cities and their lengths are given by the map in Figure 3.2 of the book (p. 63). Use the straight-line distance from each node to Sibiu as your heuristic function, assuming these distances are given by the following table:

Arad	138
Bucharest	253
Craiova	196
Dobreta	183
Eforie	433
Fagaras	96
Giurgiu	272
Hirsova	385
Iasi	298
Lugoj	97

Mehadia	146
Neamt	237
Oradea	148
Pitesti	162
Rimnicu Vilcea	78
Sibiu	0
Timisoara	141
Urziceni	302
Vaslui	345
Zerind	146

   Show your search tree, including the h(n) value for each node, and label each node with the order in which it is expanded (note, this may be different from the order it is generated). In order to reduce unnecessary search, you can ignore moves that return you to the state you just came from, however you must show any other repeated states.

2. *[15 points]* Now repeat the exercise above, but use A* instead of greedy best-first. Show your search tree, complete with f(n), g(n) and h(n) values for each node, and label each node with the order in which it is expanded. Once again, when expanding nodes, assume that you can ignore actions that return you to the previous state.

3. *[10 points]* Compare your solutions for exercises #1 and #2 above. For this search problem, how is A* better than the greedy best-first search? How (if at all) is greedy best-first better than A*?

4. *[20 points]* Use A* to solve the 8-puzzle with the initial and goal states shown below. Assume that your path cost is 1 per move and that your heuristic function is the number of tiles that are out of place (note, the blank does not count as a tile). Show your search tree, complete with f(n), g(n) and h(n) values for each node. Once again, when expanding nodes, assume that you can ignore actions that return you to the previous state.

Initial State

2	8	3
1	6	4
7		5

Goal State

1	2	3
8		4
7	6	5

5. *[40 points]* This problem looks at playing the game tic-tac-toe. Assume that X is the MAX player. Let the utility of a win for X be 10, a loss for X be -10, and a draw be 0.

   a) Given the game board **board1** below where it is X's turn to play next, show the entire game tree. Mark the utilities of each terminal state and use the minimax algorithm to calculate the optimal move.

   b) Given the game board **board2** below where it is X's turn to play next, show the game tree with a cut-off depth of two ply (i.e., stop after each player makes one move). Use the following evaluation function on all leaf nodes:
   $Eval(s) = 10X_3(s) + 3X_2(s) + X_1(s) - (10O_3(s) + 3O_2(s) + O_1(s))$
   where we define $X_n(s)$ as the number of rows, columns, or diagonals in state *s* with exactly *n* X's and no O's, and similarly define $O_n(s)$ as the number of rows, columns, or diagonals in state *s* with exactly *n* O's and no X's. Use the minimax algorithm to determine X's best move.

board1

X	O	X
O		O
	X	

board2

	X	
O		
X		O