Homework #1: Chapters 1, 2, 3

The following exercises are due at the beginning of class on February 3. Each exercise will be graded for correctness, so please start early and be sure you are confident in your answers. Also, remember that all work should be your own. Note this homework is continued on the reverse side.

1. *[15 points]* Develop a PEAS description for the following task environments:
   a) A software agent that can play a computerized version of the following traditional game of solitaire (also known as Klondike): Out of a standard 52 playing card deck, seven *tableau* piles of cards are created, ranging from one to seven cards, with the top card upturned and the other cards hidden. There are four *foundation* piles that must be built up by suit from Ace to King. The tableau piles can be built in descending order by alternate colors (e.g, a black 7 can go on a red 8). Partial or complete piles can be moved to another tableau as long as they continue to build the pile down by alternate colors. If all visible cards in a pile are moved, the next card can be turned over. Empty tableau piles can be filled by a pile with a King. In addition to moving cards and piles, the agent can turn three cards from the deck, and play the third card (by moving it to a valid foundation or tableau pile). The agent wins if it moves all cards into the foundation piles.
   b) An agent that bids on repossessed storage units, a la the A&E TV show "Storage Wars." Agents are given a chance to see what is in the locker (without being able to enter it), and then must bid on the entire contents in an auction. The auction starts at a low asking price, and at any time, a participant can choose to bid more. If after a fixed time limit, no one is willing to bid more than the current highest price, that bidder wins the good. The goal is to win a set of merchandise that is worth more than the price paid for them.
   c) A computer program that given an image of a fingerprint can find the best match in a database of criminal fingerprints.

2. *[15 points]* For each of the agents described above, categorize it with respect to the six dimensions of task environments as described on pages 41-45. Be sure that your choices accurately reflect the way you have specified your environment, especially the sensors and actuators. Give a short justification for each property.

3. *[15 points]* Consider the vacuum-cleaner world depicted in Figure 2.2 and specified on page 38.Consider the following three questions if we modify the world such that the agent is penalized one point for each movement.
   a) Can a simple reflex agent be perfectly rational for this environment? Explain.
   b) What about a model-based reflex agent? Explain.
   c) How do your answers to **a** and **b** change if the agent's percepts give it the clean/dirty status of *every* square in the environment at each time step?

4. *[20 points]* Consider the problem of coloring a two-dimensional map using only four colors, such that no two adjacent areas have the same color. Give the initial state, goal test, actions, transition model, and path cost function for this problem, making only those distinctions necessary to ensure a valid solution. Choose a formulation that is precise enough to be implemented. You may use precise English, mathematics or pseudo-code to specify the functions ACTION(*s*) and RESULT(*s*,*a*), as long as this specification is clear and unambiguous. You do not need to provide a solution (i.e., successful sequence of actions) for the problem.

5. *[25 points]* Consider the 8-puzzle with the initial and goal states shown below. Use breadth-first graph search to solve this problem. Recall, that in a graph search, any state that has already been expanded will not be added to the frontier. Show your search tree, explicitly showing the puzzle grid at each node, and labeling each node with the order in which it is expanded. To save yourself some unnecessary work, you may stop as soon as you have generated the goal state (i.e., you don't need to expand any other nodes after you have found the goal state).

Initial State

2		3
1	8	4
7	6	5

Goal State

1	2	3
8		4
7	6	5

6. *[10 points]* Now consider a basic form of depth-first tree search that does **not** avoid loopy paths by comparing new states to those along the path from the root. Also, assume that when there are multiple deepest nodes, that one is selected randomly for expansion. If a solution is "found" as soon as it is generated, then what is the minimum number of expansions before the algorithm will find the solution to the problem in exercise #5? What is the maximum number of expansions before a solution is found?