Chapter 2, Ex. 2.3

a) False. The agent maximize the performance from the percept sequence is called rational agent, even if it didn't get the whole information.
b) True. Because reflex agent do not remember history percepts.
c) True. When all actions get same rewards.
d) False. Agent function receive percept sequence while agent program only receive the current percept.
e) False. For example, the job go agent is to solve halting problem and can not be implemented by agent program.
f) True.Same situation with c.
g) True. A example can be if the two environment fit the condition as c too.
h) False. This is always a big problem we need to solve.
i) False. If the opponent has better cards, still the agent will lose.

Chapter 3, Ex. 3.3

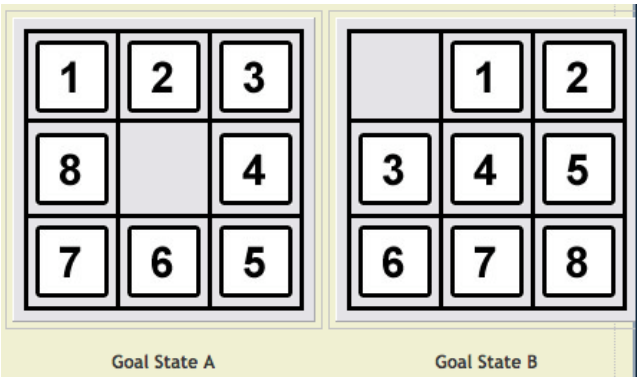
a)
1. state space: all possible city pairs (i,j).
2. successor function: all possible city pairs (x,y), which x is one step away from i and y is one step away from j.
3. goal: be at (i,i) for some i
4. step cost function: the cost begin from (i,j) to (x,y) is max(distance(i,x),distance(j,y))

b) (III) is admissible. Which means the friend head straight for each other in steps in equal size.
c) Yes. For example, two nodes with one link. And each friend is at different nodes at beginning.
d) Yes. Take c as example, if self loop is allowed, then one of the friend can loop to the same point then two friends can meet each other.

Chapter 3, Ex. 3.4

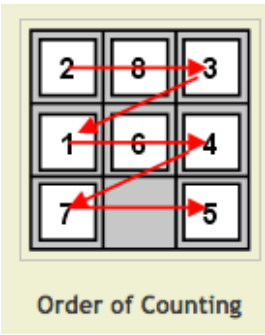
Note: in the answer, I use the thought and picture in this great blog:
http://jinchengchen.blogspot.com/2009/07/8-puzzle-algorithm.html

As we know, in the end, the 8 puzzle problem can get to two different goal:



Whether we can get into goalA or goalB is not arbitrary, it depends on our initial state. Let us first assume there are two kind of initial states, initialA and initialB. All the states from initialA to goalA are in disjoint set from all the states from initialB to goalB. These are the two disjoint set in this problem.

Then, let us get the way to devise, given a 8 puzzle states, which set should it belong to. A definition here is we count the order of 'inversion' in any state like this: The number tiles should follow the order from small number to large number in position from upper left corner to lower right corner. If not, for each number tile, we count the number of tiles smaller than it. And we sum the result.



Order of Counting

For example, in the picture left, for number 8 tile, there are 6 tiles smaller than 8, so we record 8. For tile 3, there is 1 tile smaller than it, so we record 1. There are 8 + 1 + 2 + 1 = 12 counts.

The way to decide which set a given state should belong to is like this: if the counts number is odd, it should belong to goalA set. If the counts is even, it should belong to goalB.

If we look at the goal states, we can find that goalA has counts 4+2+1 = 7(odd) and goalB has counts 0(even).

A simple proof is that no matter how we move the blank tile(which is the only way of action), the (counts mod 2) will never change.