Angelica Contero
AI Project #1
Due: 10/2/13

1) I would say that the number of possible states of the board is 9! since this is a permutation of the number of spaces on the board. However, it is known that the number of solvable states is 9!/2.

2) The average number of possible moves from a given position of the board is the average heuristic branching factor. For the 8-puzzle it's 3.81 (per the article below). This makes sense because each tile on the board can move 3 different directions, but only the one in the middle can move 4.

3) I would estimate that the number of moves required for an optimal solution to a "worst-case" problem would be exactly the number of moves it would take us to solve this puzzle which is 30 moves. I came up with this answer in the following manner, using the worst case scenario puzzle:
   First you have to move the #8 tile to the middle. (1 move)
   Then you have to shift #1-#7 over by one tile each. Since #1 starts in a position that is 4
      tiles away from it's goal position. This step will happen 4 times. (7 moves x 4)
   Finally, you can move #8 from the middle position so that it is in it's goal position (1 move)
   TOTAL MOVES: 1 + 28 + 1 = 30

4) $b^d = 3.81^{30} = 2.677622 \times 10^{17}$ nodes would have to be examined to find an answer by the brute force breadth-first search.

5) No, this is equal to 2.67762×10^{14} seconds which is 8.48505×10^{6} years! Also, it would run out of memory.

6) It's the easiest for humans to solve because we recognize easily the pattern due to the natural heuristics we have formed in our brain. Whereas if a computer is not programmed to use that heuristic, it cannot recognize the pattern.

7) A* algorithms are pretty efficient. A* using the Manhattan heuristic if faster than A* using the misplaced tile heuristic and it has to expand less nodes in the process, so it uses less memory. It is also able to figure our the worse-case scenario puzzle, while A* using the misplaced tile heuristic is unable to within a reasonable amount of time. DFBnB is much faster than IDA* and but uses a lot of memory. IDA* uses much less memory but takes a very long time.

References:

Complete Solution of the Eight-Puzzle and the Benefit of Node Ordering in IDA*
Alexander Reinefeld
http://ijcai.org/Past%20Proceedings/IJCAI-93-VOL1/PDF/035.pdf

	Puzzle Difficulty	# nodes expanded	g-value at end	total time for optimal sol'n	# steps in sol'n sequence	total time for entire program
A* w/ misplaced tile	1	14	5	1	6	
	2	55	9	6	10	
	3	157	12	19	13	
	4	n/a	n/a	n/a	n/a	
A* w/ Manhattan	1	12	5	1	6	
	2	31	9	2	10	
	3	53	12	6	13	
	4	2587	30	330	31	
DFBnB	1	21	5	2	6	2
	2	78	17	4	18	5
	3	68	18	5	19	5
	4	11485	56	530	57	1270
IDA*	1	19	5	2	6	3
	2					
	3					
	4					