Nodes vs Actions

	Nodes		Actions	
	Problem 1	Problem 2	Problem 1	Problem 2
breadth_first_search	178	30503	20	72
depth_first_graph_search	84	5602	20	72
uniform_cost_search	240	46618	20	72
greedy_best_first_graph_search with **h_unmet_goals**	29	170	20	72
greedy_best_first_graph_search with **h_pg_levelsum**	28	86	20	72
greedy_best_first_graph_search with **h_pg_maxlevel**	24	249	20	72
greedy_best_first_graph_search with **h_pg_setlevel**	28	84	20	72
astar_search with **h_unmet_goals**	206	22522	20	72
astar_search with **h_pg_levelsum**	122	3426	20	72
astar_search with **h_pg_maxlevel**	180	26594	20	72

	Nodes		Actions	
	Problem 3	Problem 4	Problem 3	Problem 4
breadth_first_search	129625	944130	88	104
uniform_cost_search	161936	1066413	88	104
greedy_best_first_graph_search with **h_unmet_goals**	230	280	88	104
greedy_best_first_graph_search with **h_pg_levelsum**	126	165	88	104
astar_search with **h_unmet_goals**	65711	328509	88	104
astar_search with **h_pg_levelsum**	3403	12210	88	104

The charts above indicate the number of nodes (or states) the algorithm has to traverse before reaching goal. Overall, **greedy_best_first_graph_search** with **h_pg_levelsum** and **greedy_best_first_graph_search** with **h_unmet_goals** is the most interesting algorithm because the node difference between problem 1 and 2 is small. This is also true with problem 3 and 4. Therefore, as the problem size increases, the algorithms mentioned can still find the goal with minimal node traversals.

Time vs Actions

	Time		Actions	
	Problem 1	Problem 2	Problem 1	Problem 2
breadth_first_search	0.024	0.348	20	72
depth_first_graph_search	0.006	0.577	20	72
uniform_cost_search	0.019	0.624	20	72
greedy_best_first_graph_search with **h_unmet_goals**	0.003	0.011	20	72
greedy_best_first_graph_search with **h_pg_levelsum**	0.492	0.755	20	72
greedy_best_first_graph_search with **h_pg_maxlevel**	0.092	0.595	20	72
greedy_best_first_graph_search with **h_pg_setlevel**	0.494	3.307	20	72
astar_search with **h_unmet_goals**	0.015	0.667	20	72
astar_search with **h_pg_levelsum**	0.292	19.354	20	72
astar_search with **h_pg_maxlevel**	0.114	54.952	20	72
astar_search with **h_pg_setlevel**	0.474	289.990	20	72

	Time		Actions	
	Problem 3	Problem 4	Problem 3	Problem 4
breadth_first_search	1.437	5.772	88	104
uniform_cost_search	1.560	9.481	88	104
greedy_best_first_graph_search with **h_unmet_goals**	0.017	0.027	88	104
greedy_best_first_graph_search with **h_pg_levelsum**	2.716	3.112	88	104
astar_search with **h_unmet_goals**	1.656	4.479	88	104
astar_search with **h_pg_levelsum**	33.803	182.459	88	104

The charts above indicate the amount of time (in seconds) it takes to run each algorithm. Overall, **greedy_best_first_graph_search** with **h_pg_levelsum** and **greedy_best_first_graph_search** with **h_unmet_goals** is the most interesting algorithm because the time difference between problem 1 and 2 is small. This is also true with problem 3 and 4. Therefore, as the problem size increases, the algorithms mentioned still finish with minimal time.

Length of Plan vs Actions

	Length		Actions	
	Problem 1	Problem 2	Problem 1	Problem 2
breadth_first_search	6	9	20	72
depth_first_graph_search	20	619	20	72
uniform_cost_search	6	9	20	72
greedy_best_first_graph_search with **h_unmet_goals**	6	9	20	72
greedy_best_first_graph_search with **h_pg_levelsum**	6	9	20	72
greedy_best_first_graph_search with **h_pg_maxlevel**	6	9	20	72
greedy_best_first_graph_search with **h_pg_setlevel**	6	9	20	72
astar_search with **h_unmet_goals**	6	9	20	72
astar_search with **h_pg_levelsum**	6	9	20	72
astar_search with **h_pg_maxlevel**	6	9	20	72
astar_search with **h_pg_setlevel**	6	9	20	72

	Length		Actions	
	Problem 3	Problem 4	Problem 3	Problem 4
breadth_first_search	12	14	88	104
uniform_cost_search	12	14	88	104
greedy_best_first_graph_search with **h_unmet_goals**	15	18	88	104
greedy_best_first_graph_search with **h_pg_levelsum**	14	17	88	104
astar_search with **h_unmet_goals**	12	14	88	104
astar_search with **h_pg_levelsum**	12	15	88	104

1) Which algorithm or algorithms would be most appropriate for planning in a very restricted domain (i.e., one that has only a few actions) and needs to operate in real time?

   greedy_best_first_graph_search *with* ***h_unmet_goals*** *would be the most appropriate because it ran the fastest on all 4 problem sets*

2) Which algorithm or algorithms would be most appropriate for planning in very large domains (e.g., planning delivery routes for all UPS drivers in the U.S. on a given day)

   Based on problem 4, which has the largest number of actions and goals, ***astar_search*** *with* ***h_unmet_goals*** *would be the best algorithm. It has a reasonable runtime and the shortest plan length*

3) Which algorithm or algorithms would be most appropriate for planning problems where it is important to find only optimal plans?

   For small problems, like problem 1 and 2, all algorithms except for depth first have the same planning length. Even for problem 3 and 4, the planning length is fairly close. ***astar_search*** *with* ***h_unmet_goals*** *has the shortest planning length and reasonable runtime*