Optimal plans:

- p1 - from `greedy_best_first_graph_search h_1`

```
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P1, SFO, JFK)
Fly(P2, JFK, SFO)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)
```

- p2 - from `greedy_best_first_graph_search h_1`

```
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Load(C3, P3, ATL)
Fly(P2, JFK, SFO)
Unload(C2, P2, SFO)
Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)
Fly(P3, ATL, SFO)
Unload(C3, P3, SFO)
```

- p3 - from `greedy_best_first_graph_search h_1`

```
Load(C2, P2, JFK)
Fly(P2, JFK, ORD)
Load(C4, P2, ORD)
Fly(P2, ORD, SFO)
Unload(C4, P2, SFO)
Load(C1, P1, SFO)
Fly(P1, SFO, ATL)
Load(C3, P1, ATL)
Fly(P1, ATL, JFK)
Unload(C3, P1, JFK)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)
```

		node expansions	# of goal tests	time elapsed (s)	new nodes	plan length	optimality**
1. breadth_first_search	p1	43	56	0.06	180	6	good
	p2	3343	4609	27.74	30,509	9	good
	p3	14663	18098	205.76	129,631	12	good
3. depth_first_graph_search	p1	21	22	0.05	84	20	suboptimal
	p2	624	625	7.34	5,602	619	bad
	p3	408	409	2.82	3,364	392	bad
7. greedy_best_first_graph_search	p1	7	9	0.01	28	6	good
	p2	998	1000	3.07	8,982	17	suboptimal
	p3	5578	5580	20.86	49,150	22	suboptimal
9. astar_search h_ignore_preconditions	p1	41	43	0.07	170	6	good
	p2	1450	1452	6.72	13,303	9	good
	p3	5040	5042	33.02	44,944	12	good
10. astar_search h_pg_levelsum	p1	18	20	20.92	77	6	good
	p2*	-	-	-	-	-	-
	p3*	-	-	-	-	-	-

*Script did not finish within 10 minutes. Test run on Macbook air (2015): 1.6 GHz Intel Core i5, 8 GB 1600 MHz DDR3
**Based on length of plan length

The best heuristic for these problems was *`astar_search h_ignore_preconditions`*, which delivered the optimal path for all 3 problems, and did it in a very efficient manner:

- Speed - time taken to complete solution was similar to or less than all the other heuristics, and for problem 3 only took 33 seconds, with `astar_search h_pg_levelsum` for instance taking well over 30 minutes (I stopped the script after not completing within this period)
- With exception to `depth_first_graph_search` (which was very suboptimal in terms of the plan length of its solution), new nodes and node expansions for this heuristic are similar to less than all the others tested

The speed advantage of *`astar_search h_ignore_preconditions`* over nearly all other heuristics can be expected, as per Russell-Norvig (Chapter 10) the relaxation of restrictions

means every action becomes allowed in each state and any single goal fluent can be achieved in one step if there is an applicable action. This suggests the number of steps needed to solve the relaxed problem is nearly just the number of unsatisfied goals.

Breadth-first search is much more effective at finding the optimal path than Depth-first search. This can be expected, per the definition of BFS and DFS (Russell-Norvig, Chapter 3). BFS explores all the nearest successors of the root first, thus giving the possibility of finding the optimal path earlier than later, while the opposite is true for DFS. DFS will spend time and computing resources searching the deepest parts of the search tree/graph before backing up closer to the root. As DFS finds solutions to the problem along the way, they will be longer/suboptimal paths rather than shorter/optimal ones that are further up the tree