Problem 1

search methods	problem	#expansion	#goal test	#new nodes	time elapsed/s	plan length/optimality
breadth_first_search	p1	43	56	180	0.047	6
depth_first_graph_search	p1	12	13	48	0.013	12
uniform_cost_search	p1	55	57	224	0.051	6
astar_search h_pg_levelsum	p1	11	13	50	1.626	6
astar_search h_ignore_preconditi	p1	41	43	170	0.046	6
astar_search h_1	p1	55	57	224	0.058	6

Optimal Path Length: 6
Optimal Plan: astar_search h_ignore_precondition
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P1, SFO, JFK)
Fly(P2, JFK, SFO)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO), JFK)
Fly(P2, JFK, SFO)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)

Problem 2

search methods	problem	#expansion	#goal test	#new nodes	time elapsed/s	plan length/optimality
breadth_first_search	p2	3343	4609	30509	20.757	9
depth_first_graph_search	p2	1669	1670	14863	19.251	1444
uniform_cost_search	p2	4852	4854	44030	16.377	9
astar_search h_pg_levelsum	p2	86	88	841	127.726	9
astar_search h_ignore_preconditi	p2	1450	1452	13303	5.71	9
astar_search h_1	p2	4852	4854	44030	17.285	9

Optimal Path Length: 9
Optimal Plan: astar_search h_ignore_precondition
Load(C2, P2, JFK)
Load(C1, P1, SFO)
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)

Problem 3

search methods	problem	#expansion	#goal test	#new nodes	time elapsed/s	plan length/optimality
breadth_first_search	p3	14663	18098	129631	149.329	12
depth_first_graph_search	p3	592	593	4927	4.718	571
uniform_cost_search	p3	18235	18237	159716	72.081	12
astar_search h_pg_levelsum	p3	318	320	2934	609.478	12
astar_search h_ignore_preconditi	p3	5040	5042	44944	21.379	12
astar_search h_1	p3	18235	18237	159716	85.985	12

```
Optimal Path Length: 12
Optimal Plan: astar_search h_ignore_precondition
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P1, SFO, ATL)
Load(C3, P1, ATL)
Fly(P2, JFK, ORD)
Load(C4, P2, ORD)
Fly(P2, ORD, SFO)
Fly(P1, ATL, JFK)
Unload(C4, P2, SFO)
Unload(C3, P1, JFK)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)
```

I will briefly evaluate all search methods here according to their performance.

1. Breadth-first-search will reach optimal solution (in the search section Video 10,11), but it's time consuming due to exploring too many nodes for shortest path.

2. Depth-first-search in contrast, won't search for optimal solution (AIMA 3.4.3) and therefore produced higher length path. So this one is always passed.

3. Uniform_cost_search and astar_h1, both expand the same most number of nodes and both used best-first-graph search method. Compared to breadth-first-search, these search methods are not satisfied with reaching a goal, but also looking for best path (Search section, Video 16). Therefore, the search will be more thorough and time-consuming than BFS, but the result is guaranteed to be optimal.

4. A* algorithm is implemented with 3 different heuristic functions. The essence of the algorithm is to expand the path with minimal function value f (AIMA 3.5.2) where:
   f = g + h
   g = path cost
   h = estimated cost to goal

The best heuristics used is the h1 ignore_precondition and it's also the best algorithm in solving all the 3 problems, in that

1) Among all the search algorithm and heuristics that produce the optimal plan, h1 ignore_precond explored the least number of nodes and reached the result with least amount of time-elapsed.

2) Compared to more advanced heuristic function h_pg_levelsum, h_ignore_precond is easier to compute and thus computationally less

expensive, since h_pg_levelsum has to go through multiple levels for goal test and h_ignore_precond only need to deal with the current state level.

3) The advantage of h_ignore_precond over breadth-first search is not that obvious when dealing with problem 1 and search space is still small enough for brutal-force method. However, as the complexity explodes exponentially, using a heuristic function saves a lot of time.