Project 3 - Report

- Based on the results of search algorithms, the optimal sequence of actions for each problem is as follows
  - Problem 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)
  - Problem 2
    - 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)
  - Problem 3
    - Load(C1, P1, SFO)
    - Load(C2, P2, JFK)
    - Fly(P2, JFK, ORD)
    - Load(C4, P2, ORD)
    - Fly(P1, SFO, ATL)
    - Load(C3, P1, ATL)
    - Fly(P1, ATL, JFK)
    - Unload(C1, P1, JFK)
    - Unload(C3, P1, JFK)
    - Fly(P2, ORD, SFO)
    - Unload(C2, P2, SFO)
    - Unload(C4, P2, SFO)
- Part 1 results
  - Part 1 involves solving the 3 air cargo planning problems using non-heuristic search algorithms. The ones I picked were breadth first search, depth first search and greedy best first search. As we can see in table 1, breadth first search was the only one that gave the optimal solution for all three problems, although its processing time, node expansion and goal tests were higher than the other two. The results are consistent with what I learnt from the lecture videos - depth first search finds results very quickly (the difference is most obvious in problem 3) as it does not need to store the frontier; breadth first search goes back to the origin every time to calculate the next shortest path to expand, therefore it takes longer to process but can find the optimal solution; greedy best first search expands the path that is closest to the goal, but may not find the optimal solution if there are obstacles on the way.
- Part 2 results
  - Part 2 involves solving the 3 air cargo planning problems using the A* algorithm with 2 different heuristics - ignore preconditions and level sum. Results in table 2 show both heuristics produced the optimal solution. For all 3 problems, the level sum heuristic required much less node expansions and goal tests, but it took around 4 times as much processing time. Therefore I would pick the ignore preconditions heuristics.
- What was the best heuristic used in these problems? Was it better than non-heuristic search planning methods for all problems? Why or why not?
  - Breadth-first search was the best heuristic used in the 3 problems. It was also better than the heuristic search planning methods, because it was able to produce the optimal solution using a much lower processing time.
- Please, explain the reason for the observed results using at least one appropriate justification from the video lessons or from outside resources (e.g., Norvig and Russell's textbook).
  - Out of the uninformed algorithms, depth first search was the fastest when it came to more complex problems like problem 3, and breadth first search was the slowest across all 3 problems. This is because of the storage requirement. Depth first search has an advantage here because it does not need to store an explored set. This means the frontier only has n nodes, compared to b^n nodes for breadth first search, where *b* is the branching factor and *n* is the depth.

Table 1 - Results for Part 1

	Algorithm	# node expansion	# goal tests	# new nodes	time elapsed	plan length
air_cargo_p1	breadth_first_search	43	56	180	0.03	6
	depth_first_graph_search	21	22	84	0.02	20
	greedy_best_first_graph_search h_1	7	9	28	0.01	6
air_cargo_p2	breadth_first_search	3343	4609	30509	7.88	9
	depth_first_graph_search	624	625	5602	3.24	619
	greedy_best_first_graph_search h_1	998	1000	8982	2.23	17
air_cargo_p3	breadth_first_search	14663	18098	129631	39.08	12
	depth_first_graph_search	408	409	3364	1.66	392
	greedy_best_first_graph_search h_1	5398	5400	47665	14.37	26

Table 2 - Results for Part 2

	Heuristic	# node expansion	# goal tests	# new nodes	time elapsed	plan length
air_cargo_p1	ignore-preconditions	55	57	224	0.05	6
	level-sum	11	13	50	0.49	6
air_cargo_p2	ignore-preconditions	4853	4855	44041	12.25	9
	level-sum	86	88	841	41.86	9
air_cargo_p3	ignore-preconditions	18151	18153	159038	54.82	12
	level-sum	314	316	2894	217.48	12