Heuristic Analysis:

Our report produces the following result analysis:

- Optimal plan for Problems 1,2, and 3.
- Contrast non-heuristic search result metrics(Optimality,time elapsed, number of nodes expansion) for Problems 1,2, and 3.
- Contrast non-heuristic search result metrics using A* with "ignore preconditions" and "level-sum" heuristics for Problems 1,2, and 3.
- Best heuristic used in our problem.

Optimal plan for Problems 1,2 and 3:

Problem 1:

	Expansions	**Goal Tests**	**New Nodes**	**Plan Length**	**Time Elapsed**
breadth_first_search	43	56	180	6	0.037
breadth_first_tree_search	1458	1459	5960	6	0.734
depth_first_graph_search	12	13	48	12	0.006
depth_limited_search	101	271	414	50	0.124
uniform_cost_search	55	57	224	6	0.031
recursive_best_first_search h_1	4229	4230	17029	6	2.102
greedy_best_first_graph_search h_1	7	9	28	6	0.004
astar_search h_1	55	57	224	6	0.035
astar_search h_ignore_preconditions	41	43	170	6	0.025
astar_search h_pg_levelsum	6	8	28	6	0.355

Table1

Optimal plan:
Load(C2,P2,JFK)
Load(C1,P1,SFO)
Fly(P2,JFK,SFO)
Unload(C2,P2,SFO)
Fly(P1,SFO,JFK)
Unload(C1,P1,JFK)

Problem 1 is fairly easy problem to solve. Additionally, each search algorithm was able to find an goal state in a reasonable amount of time. In the case of non-heuristic search, greedy best graph search outperforms the other search algorithms.

Problem 2:

	Expansions	**Goal Tests**	**New Nodes**	**Plan Length**	**Time Elapsed**
breadth_first_search	3346	4612	30534	9	14.620
breadth_first_tree_search	N/A	N/A	N/A	N/A	N/A
depth_first_graph_search	1124	1125	10017	1085	8.073
depth_limited_search	213491	1967093	1967471	50	949.335
uniform_cost_search	4853	4855	44041	9	11.630
recursive_best_first_search h_1	N/A	N/A	N/A	N/A	N/A
greedy_best_first_graph_search h_1	998	1000	8982	21	2.167
astar_search h_1	4853	4855	44041	9	10.305
astar_search h_ignore_preconditions	1450	1452	13303	9	3.304
astar_search h_pg_levelsum	9	11	86	9	10.097

Table2

Optimal plan:
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 2 introduces more complexity. Breadth First tree search, depth limited search, and recursive best first search were unable to find a plan in a reasonable matter of time. Out of the non-heuristic , greedy best first graph search outperforms the other search techniques in term of speed.

Problem 3:

	Expansions	**Goal Tests**	**New Nodes**	**Plan Length**	**Time Elapsed**
breadth_first_search	14663	18098	129631	12	38.131
breadth_first_tree_search	N/A	N/A	N/A	N/A	N/A
depth_first_graph_search	627	628	5176	596	2.770
depth_limited_search	N/A	N/A	N/A	N/A	N/A
uniform_cost_search	18235	18237	159716	12	46.543
recursive_best_first_search h_1	N/A	N/A	N/A	N/A	N/A
greedy_best_first_graph_search h_1	5614	5616	49429	22	14.367
astar_search h_1	18235	18237	159716	12	46.180
astar_search h_ignore_preconditions	5040	5042	44944	12	13.012
astar_search h_pg_levelsum	17	19	148	14	18.657

Table3

Optimal plan:
Load(C2,P2,JFK)
Load(C1,P1,SFO)
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)

Again, breadth first tree search, depth limited search, and recursive best first search were unable to find a plan in a reasonable matter of time for this problem. From a time perspective, it would appear that depth first graph search is the winner of non-heuristic search.

Uniformed Non-heuristic search

The uninformed non-heuristic planning was experimented with for **Breadth First Search** (BFS), **Depth First Search** (DFS) and **Uniform Cost Search** (UCS). The results for the same can be summarized in Table 4.

Table 4 Metrics for Uninformed Non-Heuristic Search

Problem	**Search Type**	**Plan Length**	**Time Elapsed(secs)**	**#Nodes Expanded**	**#Goal Tests**	**Is Optimal**
P1	BFS	6	0.037	43	56	Yes
P1	DFS	12	0.006	12	13	No
P1	UCS	6	0.031	55	57	Yes
P2	BFS	9	14.620	3346	4612	Yes
P2	DFS	1085	8.073	1124	1125	No
P2	UCS	9	11.630	4853	4855	Yes
P3	BFS	12	38.131	146663	18098	Yes
P3	DFS	596	2.770	627	628	No
P3	UCS	12	46.543	18235	18237	Yes

Heuristic search

Problem	A* Heuristic used for Search	Plan Length	Time Elapsed (seconds)	# Nodes Expanded	# Goal Tests	Is Optimal
P1	h_1	6	0.035	55	57	Yes
P1	h_ignore_preconditions	6	0.025	41	43	Yes
P1	h_pg_level_sum	6	0.355	6	8	Yes
P2	h_1	9	10.305	4853	4855	Yes
P2	h_ignore_preconditions	9	3.304	1450	1452	Yes
P2	h_pg_level_sum	9	10.097	9	11	Yes
P3	h_1	12	46.180	18235	18237	Yes
P3	h_ignore_preconditions	12	13.012	5040	5042	Yes
P3	h_pg_level_sum	14	18.657	7	19	Yes

Non-heuristics: We choose to compare heuristics 1 (BFS), 3 (DFTS) and 5 (UCS). DFTS is the fastest but does not find an optimal plan for all problems. BFS and UCS are guaranteed to find an optimal path but are slower. In all cases, UCS expands more nodes than BFS; however, UCS is faster since the implementation of the priority queue is more efficient. Note that we notice that in problem 1 UCS is slightly slower than BFS, but we are interested in comparing the efficiency for more complex problems. Therefore problems 2 and 3 are more appropriate benchmark problems to compare algorithm efficiency. In theory, UCS time/space complexity should be larger than BFS; however, due to implementation details, UCS has a smaller time complexity. In particular, BFS stops as soon as it finds a goal, while UCS examines all the nodes at the goal's depth to see if one has a lower cost. Therefore,
UCS does more work by expanding nodes at depth d unnecessarily. Nonetheless, I will recommend the UCS for non-heuristic planning domains since it takes less time using the implementation I was given for this project.

Heuristics: We now compare 9 (IP) and 10 (LS). It's clear that IP and LS heuristics provide significant reduction in the number of nodes expanded. In particular, the LS heuristic provides a more accurate heuristic than the IP heuristic (inferred by the difference in the number of nodes expanded); however, the time to compute the LS heuristic is much larger than the IP heuristic. For problem 3, the IP heuristic is able to expand approximately 2226 nodes per second, while the LS algorithm can only expand approximately 11 nodes per second. This huge reduction in node expansion efficiency outweighs the performance gain in reducing the number of nodes expanded. For these reasons, I recommend the IP algorithm.

Best heuristic: Overall, the ignore preconditions (IP) heuristic was the best for the reasons above. It significantly outperformed non-heuristic search planning methods for the more complex problems, namely problems 2 and 3. For problem 1, it still outperformed all non-heuristic search planning methods in terms of number of nodes expanded; however, it was 0.01 seconds slower than BFS. Overall the IP heuristic has a more substantial cost improvement for more complex problems. For these reasons, I recommend the IP search algorithm.