Problem 1:

Init(At(C1, SFO) ∧ At(C2, JFK)
∧ At(P1, SFO) ∧ At(P2, JFK)
∧ Cargo(C1) ∧ Cargo(C2)
∧ Plane(P1) ∧ Plane(P2)
∧ Airport(JFK) ∧ Airport(SFO))
Goal(At(C1, JFK) ∧ At(C2, SFO))

The Optimal Plan: Greedy Best First Graph Search
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P1, SFO, JFK)
Fly(P2, JFK, SFO)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)

Table 1 provides insight into the performance of seven search algorithms. Greedy Best First Graph Search was the best algorithm for this problem. The reason Greedy Best First Graph Search works in this case is because this problem is small and not complex, therefore Greedy Best First Graph can just select the current best value and finish the problem fast. The result is the lowest number of expansions and new nodes which lead to the second fastest overall time. Depth First Graph Search had the second lowest expansions, new nodes and the third fastest overall time, but resulted in a not optimal plan length of 20. Breadth First Search had the third lowest expansions, new nodes, and the fourth fastest overall time, but resulted in a optimal plan length of 6. The reason for the difference between runtime Breadth First Search and Depth First Graph Search could be because of the structure of the problem. Depth First Graph Search can traverse to the leaves of the tree and find the goal quickly. However, Breadth First Search has to traverse through every nodes on each level in order to find the goal. This is a tradeoff between Plan Length and runtime.

Problem 1 Non-Heuristic Search Results					
Search Algorithm	Expansions	New Nodes	Plan Length	Time (s)	Optimal
Breadth First Search	43	180	**6**	0.030	**Yes**
Breadth First Tree Search	1458	5960	**6**	0.821	**Yes**
Depth First Graph Search	21	84	**20**	0.013	**No**
Depth Limited Search	101	414	**50**	0.078	**No**

Uniform Cost Search	55	57	**6**	0.034	**Yes**
Recursive Best First Search	4229	17023	**6**	2.353	**Yes**
Greedy Best First Graph Search	7	28	**6**	0.005	**Yes**

Table 2 provides an insight into the performance of A* Search with three different heuristics. All three algorithms achieved the optimal plan length of 6. It is interesting to see that even though the level sum heuristic had the same node expansions and plan length as h_1 heuristic, the level sum heuristic was 20 times slower than the h_1 heuristic.

Problem 1 Heuristic Search Results					
Search Algorithm	Expansions	New Nodes	Plan Length	Time (s)	Optimal
A* Search with h_1	55	224	**6**	0.034	**Yes**
A* Search (ignore precondition)	41	170	**6**	0.033	**Yes**
A* Search (levelsum)	55	224	**6**	0.728	**Yes**

Problem 2:

Init(At(C1, SFO) ∧ At(C2, JFK) ∧ At(C3, ATL)
∧ At(P1, SFO) ∧ At(P2, JFK) ∧ At(P3, ATL)
∧ Cargo(C1) ∧ Cargo(C2) ∧ Cargo(C3)
∧ Plane(P1) ∧ Plane(P2) ∧ Plane(P3)
∧ Airport(JFK) ∧ Airport(SFO) ∧ Airport(ATL))
Goal(At(C1, JFK) ∧ At(C2, SFO) ∧ At(C3, SFO))

The Optimal Plan: Breadth First Search
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)

Table 3 are the results of all seven non-heuristic search algorithms. Breadth First Tree Search, Depth Limited Search, and Recursive Best First Search never finish in desired time. Greedy Best First Graph Search is the fastest algorithm for this problem. However, it has more expansions and new nodes than Depth First Graph Search. And its path length is the second worst in all the non-heuristic algorithms. Based on plan length, Breadth First Search is the optimal solution because it has Plan Length of 9. However, Greedy Best First Graph Search is the fastest method for this problem. The reason Breadth First Search is the optimal solution in this problem instead of Greedy Best First Graph is because the problem is now more complex, with the best plan not always start with the best value. The reason Breadth First Search is faster than Uniform Cost Search is because Uniform Cost Search is more similar to Depth First Search Than Breadth First Search.

Problem 2 Non-Heuristic Search Results					
Search Algorithm	Expansions	New Nodes	Plan Length	Time (s)	Optimal
Breadth First Search	3343	30509	**9**	3.573	**Yes**
Breadth First Tree Search	-	-	**-**	-	**-**
Depth First Graph Search	624	5602	**619**	3.587	**No**
Depth Limited Search	-	-	**-**	-	**-**
Uniform Cost Search	4852	44030	**9**	11.68	**Yes**
Recursive Best First Search	-	-	**-**	-	**-**
Greedy Best First Graph Search	990	8910	**17**	2.436	**No**

Table 4 results for A* search with various heuristics. All of these heuristics achieve the optimal path length. However, Ignore precondition heuristic is the fastest and it has the least amount of expansions and new nodes. Level sum heuristic is the slowest heuristic among the three. H_1 heuristic has the exact same expansions, new nodes and path length at Level sum. However, Level sum is 42 times slower.

Problem 2 Heuristic Search Results					
Search Algorithm	Expansions	New Nodes	Plan Length	Time (s)	Optimal
A* Search with h_1	4852	44030	**9**	10.78	**Yes**
A* Search (ignore precondition)	1450	13303	**9**	3.897	**Yes**
A* Search (levelsum)	4852	44030	**9**	427.58	**Yes**

Problem 3:

Init(At(C1, SFO) ∧ At(C2, JFK) ∧ At(C3, ATL) ∧ At(C4, ORD)
∧ At(P1, SFO) ∧ At(P2, JFK)
∧ Cargo(C1) ∧ Cargo(C2) ∧ Cargo(C3) ∧ Cargo(C4)
∧ Plane(P1) ∧ Plane(P2)
∧ Airport(JFK) ∧ Airport(SFO) ∧ Airport(ATL) ∧ Airport(ORD))
Goal(At(C1, JFK) ∧ At(C3, JFK) ∧ At(C2, SFO) ∧ At(C4, SFO))

The Optimal Plan: A* Search (ignore precondition)
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)

Table 5 are the results of all seven non-heuristic search algorithms. Breadth First Tree Search, Depth Limited Search, and Recursive Best First Search never finish in desired time. On this problem, Depth First Search is the fastest algorithm for this problem, and it has the minimum amount of expansion and new nodes. However, it has the longest path length.

Problem 3 Non-Heuristic Search Results					
Search Algorithm	Expansions	New Nodes	Plan Length	Time (s)	Optimal
Breadth First Search	14663	129631	**12**	45.64	**Yes**
Breadth First Tree Search	-	-	**-**	-	**-**
Depth First Graph Search	408	3364	**392**	1.538	**No**
Depth Limited Search	-	-	**-**	-	**-**
Uniform Cost Search	18235	159716	**12**	46.352	**Yes**
Recursive Best First Search	-	-	**-**	-	**-**
Greedy Best First Graph Search	5614	49429	**22**	14.282	**No**

Table 6 results for A* search with various heuristics. All of these heuristics achieve the optimal path length. However, Ignore precondition heuristic is the fastest and it has the least amount of expansions and new nodes. Level sum heuristic is the slowest heuristic among the three. H_1 heuristic has the exact same expansions, new nodes and path length at Level sum. However, Level sum is 54 times slower.

Problem 3 Heuristic Search Results					
Search Algorithm	Expansions	New Nodes	Plan Length	Time (s)	Optimal
A* Search with h_1	18235	159716	**12**	46.165	**Yes**
A* Search (ignore precondition)	5040	44944	**12**	16.199	**Yes**
A* Search (levelsum)	18235	159716	**12**	2500.302	**Yes**

Conclusion:
Overall, A* Search ignore precondtion is the best heuristic for these three problems. For the small problems, A* Search ignore precondtion might not be the fastest optimal solution. However, as the problem grows, A* Search ignore precondition becomes more and more reliable heuristic. If we do not care about Plan Length and only for runtime, Depth First Graph Search is definitely the best solution.

A* Search levelsum heuristic is the slowest algorithm. This is because of the cost of evaluating the level sum heuristic.