Problem 1
a)
Planner used A star. The state is defined by (row, col, time).

Creating the heuristic Map:
Before beginning the planner, a heuristic map of the whole grid is created. This is done by representing all the goal nodes as one, and running a dijkstra on the grid in reverse direction, but with no final goal. In this way the Dijkstra will spread throughout the grid and the G values of each cell would be saved in a hashmap as heuristic values. This map is used during Astar planning.

Astar Loop:
The F value is calculated using heuristic value of the cell (from the map created in previous step) and the summation of the cost of all the cells the robot has travelled to reach that cell, plus the cost of that cell.
With every iteration the node with lowest F value is expanded in all five directions. Further hashmap is created for both open and closed list. If the new node created is already present in the closed the list then it is not added to open list. And if the same node with lower cost is present in open list then also it is not added to open list.

Handling Multiple Goals: Since state is (row, column, time), the goal is given by its position on the grid and the number of time steps the target reaches that point.
In each iteration of A* , goal check is done, where if the robot has reached any of the goal, the A* star loop terminates, and the plan is created by backtracking from the final node.

Summary:
For problem 0
Execution time:~ 0.0011 s (~0.0005s for creating Heuristic map, ~0.0006s for finding the plan)
Cost of path: 8
Number of states expanded = 19

For Problem 1
Takes a lot of time,
Execution time: > 38 s

b)
Putting weight on the heuristic increases the speed of A* at the cost of optimality.
Weight = 20.

For problem 0
Execution time: ~ 0.0008 s (~0.0006 s for creating Heuristic map, ~0.0002 s for finding the plan)

Cost of path = 9
Number of states expanded = 13

For problem 1
Execution Time: ~ 39.6 s (~39.1 s for creating Heuristic map, ~0.5 s for finding the plan)
Cost of Path: 83690
Number of States expanded: 20043

Problem 2
2.1
a)
h(s) = min(h1(s),h2(s))
For consistency
h(s) <= c(s,s') + h(s') where s' is successor state

Since h1 and h2 consistent:
therefore:
h1(s) <= c(s,s') + h1(s') …… (1)
h2(s) <= c(s,s') + h2(s') …….(2)
Now if 's' lie in a region 1 where h2(s) < h1(s) ….. (3),
This implies from (1) and (3):
h2(s) < h1(s) <= c(s,s') + h1(s') … (4)
I.e.
min(h2(s),h1(s)) <= c(s,s') + h1(s')
Vice -versa for h1(s) < h2(s) in region 1 could also be showed,
Further manipulating region of ' s' ' (such that either h1(s') > h2(s') or vice versa) would lead to (2) or (4)
Thus,
min(h2(s),h1(s)) <= c(s,s') + min(h2(s'),h1(s'))
h(s) <= c(s,s') + h(s)
And thus h(s) is consistent

b)
using the same terminology as part a)
Since h1 and h2 consistent:
therefore:
h1(s) <= c(s,s') + h1(s') …… (1)
h2(s) <= c(s,s') + h2(s') …….(2)

Now if 's' ' lie in a region 2 where h2(s) < h1(s) ….. (3),
Thus implying from (2) and (3)

$h2(s) <= c(s,s') + h2(s') < c(s,s') + h1(s')$ …….(4)
$h2(s) <= c(s,s') + max(h1(s'),h2(s'))$
Vice -versa for $h1(s') > h2(s')$ in region 1 could also be showed,
Further manipulating region of ‘ s ‘ (such that either $h1(s) > h2(s)$ or vice versa) would lead to (1) or (4)

Thus,
$max(h2(s),h1(s)) <= c(s,s') + max(h2(s'),h1(s'))$
$h(s) <= c(s,s') + h(s)$
And thus h(s) is consistent

2.2
b) Monotonically non decreasing sequence. This follows from the consistency condition of heuristic function.
2.3
f) true
2.4
e) true