1)

a) The algorithm would try to explore all paths before finding one that is ideal, so if the path cost can be negative and arbitrarily large, any path could have an arbitrarily large negative cost.

b) It is not enough to specify a lower bound for path costs. Even a single negative past cost could be enough to cause a cycle in the graph which the algorithm would traverse indefinitely.

c) Any agent which seeks to minimize cost would be stuck in an infinite loop.

d) Humans drive on scenic routes because the utility of driving the route outweighs the time and financial costs. The reason humans do not do this indefinitely because other factors come into play, such as total amount of free time, disposable income etc. So adding factors such as these which eventually outweigh the utility of a negative cost route could help agents avoid looping.

e) Working a job on a regular schedule is an example of looping. The step cost of going to work could be considered negative since more money (hopefully) is earned than it takes to get there.

2)

a) State – The current environment which the agent is a part of. There are also other possible states.

b) State Space – All of the possible states.

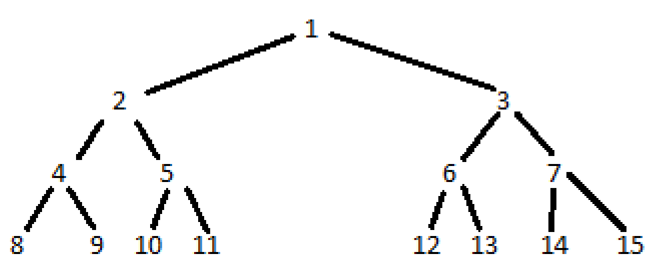
c) Search Node – A node inside of a search tree, usually contains where it was visited from.

d) Goal – The state which the agent is trying to achieve.

e) Action – A function which the agent can take.

3)

a)



b) BFS: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11

DLS: 1, 2, 4, 8, 9, 5, 10, 11

IDS: (1), (1, 2, 3), (1, 2, 4, 5, 3, 6, 7), (1, 2, 4, 8, 9, 5, 10, 11)

c) A bi-directional search would work fairly well. From the start the branching factor is 2, from the goal it is 1.

d) If you are given the goal state, you can simply traverse backwards to the start in this case.

e) Yes, consider the goal as a binary number. Using the digits of this binary number, map 0 to go left and 1 to go right.

5)

a) $x = (n^2)(n^2 - 1)(n^2 - 2) \ldots. (n^2 - n)$

b) 5^n

c) $h_i = |n - i + 1 - x_i| + |n - y_i|$

d) Only (i) and (ii) are admissible heuristics as they are always larger than the actual cost. Taking the min of the heuristics would result in several cases where the heuristic is smaller than the real cost.

Search algorithm report

Sample output:

```
maze1.txt
BFS path: [(2,2), (3,2), (4,2), (5,2), (6,2), (6,3), (6,4), (6,5), (6,6), (5,6), (4,6), (4,5), (4,4), (3,4), (2,4), (2,5)
BFS path length = 241
BFS explored states: 518
DFS path: [(2,2), (3,2), (4,2), (5,2), (6,2), (6,3), (6,4), (6,5), (6,6), (5,6), (4,6), (4,5), (4,4), (3,4), (2,4), (2,5)
DFS path length = 241
DFS explored states: 518
A* path: [(2,2), (3,2), (4,2), (5,2), (6,2), (6,3), (6,4), (6,5), (6,6), (5,6), (4,6), (4,5), (4,4), (3,4), (2,4), (2,5),
A* path length = 241
A* explored states: 517
BDS path: [(2,2), (3,2), (4,2), (5,2), (6,2), (6,3), (6,4), (6,5), (6,6), (5,6), (4,6), (4,5), (4,4), (3,4), (2,4), (2,5)
BDS path length = 241
BDS explored states: 441

maze2.txt
BFS path: [(2,2), (2,3), (2,4), (2,5), (2,6), (2,7), (2,8), (2,9), (2,10), (2,11), (2,12), (2,13), (2,14), (2,15), (2,16)
BFS path length = 69
BFS explored states: 1247
DFS path: [(2,2), (2,3), (1,3), (0,3), (0,4), (1,4), (2,4), (3,4), (4,4), (5,4), (6,4), (7,4), (8,4), (9,4), (10,4), (11,
DFS path length = 1135
DFS explored states: 1264
A* path: [(2,2), (2,3), (2,4), (2,5), (2,6), (2,7), (2,8), (2,9), (2,10), (2,11), (3,11), (4,11), (5,11), (5,12), (5,13),
A* path length = 69
A* explored states: 434
BDS path: [(2,2), (3,2), (4,2), (5,2), (6,2), (7,2), (8,2), (9,2), (10,2), (11,2), (12,2), (13,2), (14,2), (15,2), (16,2)
BDS path length = 69
BDS explored states: 1236

maze3.txt
BFS path: [(2,2), (3,2), (4,2), (5,2), (6,2), (7,2), (8,2), (9,2), (10,2), (11,2), (12,2), (13,2), (14,2), (15,2), (16,2)
BFS path length = 18
BFS explored states: 217
DFS path: [(2,2), (2,3), (3,3), (4,3), (5,3), (6,3), (7,3), (8,3), (9,3), (10,3), (11,3), (12,3), (13,3), (14,3), (15,3),
DFS path length = 26
DFS explored states: 1264
A* path: [(2,2), (3,2), (4,2), (5,2), (6,2), (7,2), (8,2), (9,2), (10,2), (11,2), (12,2), (13,2), (14,2), (15,2), (16,2),
A* path length = 18
A* explored states: 16
BDS path: [(2,2), (3,2), (4,2), (5,2), (6,2), (7,2), (8,2), (9,2), (10,2), (11,2), (12,2), (13,2), (14,2), (15,2), (16,2)
BDS path length = 18
BDS explored states: 154
```

(Paths are cut off)

All algorithms find a correct path in every maze.

Every algorithm except for Depth First Search is optimal, since the state space is finite. Additionally, you can see that every algorithm except Depth First Search produced the same length path, suggesting they are optimal.