# Report Lab 2:

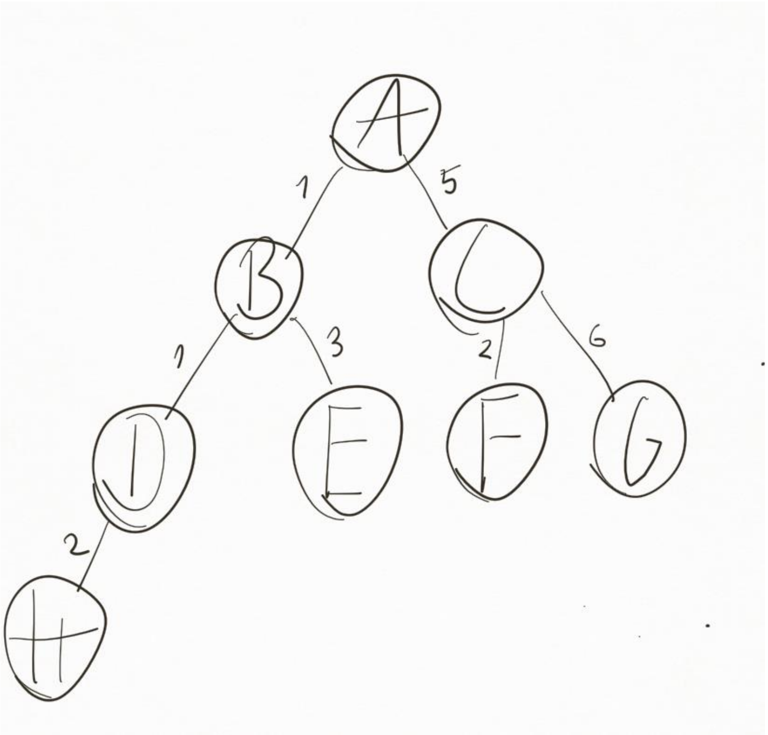
1. **In the vacuum cleaner domain in part 1, what were the states and actions? What is the branching factor?**
   The states were the agent's position (x,y), the actions of the agents were moving either west, south, east or north onto an empty position on the grid (not wall), the branching factor is 4.

2. **What is the difference between Breadth First Search and Uniform Cost Search in a domain where the cost of each action is 1?**
   If the cost of each action is 1, there is no difference between the search algorithms.

3. **Suppose that h1 and h2 are admissible heuristics (used in for example A*). Which of the following are also admissible?**
   a) **(h1+h2)/2**
      admissible – it is an average of h1 and h2, therefore it won't be bigger than h1 or h2.
   b) **2h1**
      not admissible – the cost is the double of h1
   c) **max (h1,h2)**
      admissible – it will either choose h1 or h2 and the largest number will still be smaller

4. **If one would use A* to search for a path to one specific square in the vacuum domain, what could the heuristic (h) be? The cost function (g)? Is it an admissible heuristic?**
   The heuristic could be the Manhattan distance to the specific node.
   The cost function could be the number of nodes traveled to the node.

5. **Draw and explain. Choose your three favorite search algorithms and apply them to any problem domain (it might be a good idea to use a domain where you can identify a good heuristic function). Draw the search tree for them, and explain how they proceed in the searching. Also include the memory usage. You can attach a hand-made drawing.**



**Uniform-Cost search – Dijkstra's algorithm:**

Path: A,B,D,H,E,C,F,G

Memory: queues every node on every level

**Breadth-First search:**

Traverses the tree in level-order, FIFO queue

Path: A,B,C,D,E,F,G,H

Memory: queues every node on every level

**Depth-First search:**

LIFO queue

Path: A,B,D,H,E,C,F,G

Memory: needs only one node on every level

6. **Look at all the offline search algorithms presented in chapter 3 plus A* search. Are they complete? Are they optimal? Explain why!**

   **Breadth-first search:**
   **Complete** – if the shallowest goal node is at some finite depth, because it will eventually find it
   **Optimal** – if the path cost is a nondecreasing function of the depth of the node.

   **Uniform-cost search:**
   **Complete –** for the same reasons as Breadth-first search

**Optimal** – it is optimal even if the path cost varies

**Depth-first search**

**Complete** – in graph search

**Incomplete** – in tree search, because it can end in an infinite loop

**Not optimal** - because it always expands the deepest node and might miss a solution

**Depth-limited search**

**Incomplete –** if all solutions are deeper than the limit, no solution will be found

**Not optimal** – if the limit is higher than the optimal solution

**Iterative deepening depth-first search**

**Complete** – because it will explore every level

**Optimal** - if the path cost is a nondecreasing function of the depth of the node.

**Bidirectional search**

**Complete** – if Breadth-first search is used and the branching factor is not infinite, will eventually find it

**Optimal** – if BFS is used

**A* search**

**Complete –** like Uniform-cost search

**Optimal** - like Uniform-cost search

7. **Assume that you had to go back and do Lab 1/Task 2 once more (if you did not use search already). Remember that the agent did not have perfect knowledge of the environment but had to explore it incrementally. Which of the search algorithms you have learned would be most suited in this situation to guide the agent's execution?**
   We would use Breadth-first search, as it can find the shortest path. We would search for unexplored points in the room.