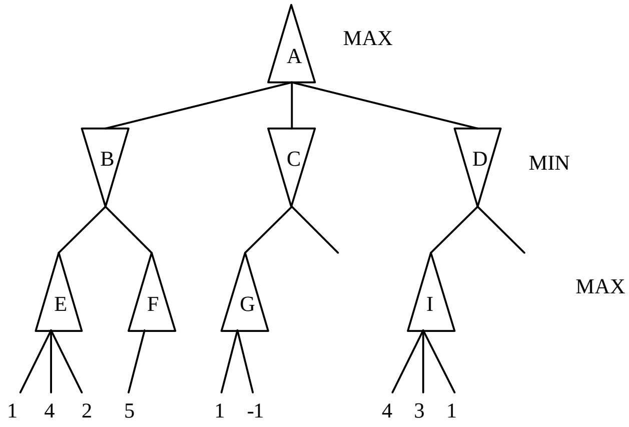
Question 1

a) Solution: **4** Path: **A→ B → E**

b) The resulting pruned tree if we apply a-ß pruning with left-to-right evaluation is:



c) This question was discussed in homework 2 (problem 4). Assuming you have the perfect static evaluation *function*, you would not need to do the search at all. You just follow the node that the evaluation function estimates as maximizing the gains at the top and that is the best you can do. Note that this function will give you the perfect value if you evaluate nodes B, C and D, so you only need to search the children of A! Therefore, the search space is reduced to be linear in the braching factor (and the number of steps to the solution if we wish to find out the path to follow!). A lot of people gave the answer that the branching factor is reduced to $O\left(\sqrt{b}\right)$ which is true if we have the *perfect order in which to search the tree* not the perfect static evaluation function!!

Question 2

a) Note that the solution is the same as the nodes visited since DFS stops once it finds a path that leads to the goal (even though it's not optimal)
Nodes visited in order = **0, 4, 7, 11, 12, 8, 9, 10, 13**

b) Note that the question asks for all the nodes of the search tree visited by the algorithm and not just the solution/winning path! If the tree (which you were not supposed to draw as part of the solution) reflected the nodes visited, then partial credit was assigned.
Nodes visited in order = **0, 4, 1, 7, 5, 2, 11, 8, 3, 12, 9, 6, 10, 13**

c) The manhattan distance is an admissible heuristic, since it never overestimates the path cost. This is true, because we need to make at least $|x_1-x_2|$ moves horizontally and at least $|y_1-y_2|$ vertically in order to reach the goal. Thus we have to at least cover manhattan distance to get to G so it never overestimates! If you did not provide a sufficient explanation of why h does not overestimate, you lost a small percentage of the points ☺
Another thing you might mention (but it did not affect the grade) is that: $h(G) = 0$

d) This is from the lecture notes. Please note that DFS is complete if we have a finite depth (as stated in the question!)

Type of Search	**Optimal?**	**Complete?**	**Time Complexity**	**Space Complexity**
BFS	**YES**	**YES**	**b^d**	**b^d**
DFS	**NO**	**YES**	**b^m**	**bm**
Iterative-Deepening DFS	**YES**	**YES**	**b^d**	**bd**

3. Coming Soon

4.

a) A proof system is sound if:
   a. Every sentence which is provable/deducible is valid/entailed OR
   b. If we start from true premises, we can only get to true/valid conclusions OR
   c. No false sentences can be deduced from the system.

   Any one of these answers would get full points. People usually got full points here.

b) A proof system is complete if:

   $\models B \Rightarrow \vdash B$ for every B.

   Some people gave the definition as $\models B \Leftrightarrow \vdash B$ which is not correct since there are proof systems which are complete but not sound! (consider a system where you can derive anything)

c) To make a proof system incomplete:

   Take out an axiom. Now you can no longer prove the same set of statements that you were able to before. Therefore from some B, $\models B \Rightarrow \vdash B$ no longer holds. Note that for this to work, your axioms should be independent, i.e. you should not be able to derive the axiom you took out using other axioms.

   The common mistake that people did here was that they confused soundness/completeness and actually gave a method to make the system unsound instead of incomplete. Another common problem was to say: “Change the axioms such that $\models B \Rightarrow \vdash B$ no longer holds”. This is correct and it is the first step to take while attacking this problem. But, obviously it is not the solution.

d) To make a proof system unsound:

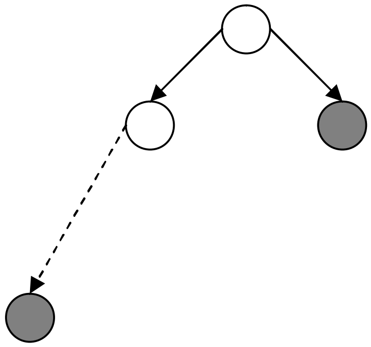
   This was trickier. The correct answer is to insert an axiom to the system such that we have an inconsistency. If we have an inconsistency, then we can derive False using M.P., and from there we can derive anything since False ⇒ X is true for all X.

   One common mistake here was to say that removing an axiom should do the trick. Well it does not. The reason is that if you remove an axiom, you are still able to prove a subset Q of what you have been able to prove before, and Q is certainly valid since it is a subset of valid sentences (the system was sound before). Even if you remove all the axioms such that you cannot prove anything, the system is still sound by definition because $\vdash B$ is not true for any B.

   Some people said “negating an axiom” and got deducted some points. Actually, the answer is correct, BUT the reason it is correct is that all the axioms we have in this setting are tautologies so if you negate them, you get FALSE by definition. This obviously does make the system unsound. Some people actually gave me an example of such a negation and therefore got full points. Others who did not explain why negation would cause an inconsistency got taken off some points.

5.

a) Most people got this part. The answer is no. BFS does not always consume more space than DFS and the counterexample is the following tree:



The gray circles are solutions and the dashed arrow shows a deep branch (depth = m > 3). In this setting, assuming the DFS starts searching from left, it will go down the deep branch requiring m nodes worth of memory, whereas BFS will find the other solution at level 2 at the expense of 3 nodes worth of memory.

b) There were many plausible answers to this part and most people got full points here as well. One acceptable answer was that common sense requires a lot of background knowledge which in itself is hard to represent. Things that we accept as common sense are actually mental shortcuts that we use to make our everyday life easier and they might contain ambiguity. Some examples such as the ones given in lecture (definition of “left” in a circular table) were also accepted as answers.

c) The answer is no. Although it would be easier for us to represent the common concepts in natural language, we only moved the complexity of the problem to another level since now we have to parse and represent natural language which is a very complex process because of the ambiguous definitions of words and the non-standard grammar.