Name:

CS 662 Sample Midterm

1 True/False, plus corrections

35 pts, 5 pts each

Each of the following statements is either true or false. If it is true, mark it true. If it is false, correct the statement so that it is true. **Note:** Adding "not" or otherwise negating the sentence is not acceptable. You must change the facts in the sentence if it is false. For example:

Question: The Turing Test is a test of whether a computer program is rational.

Bad answer, no credit: The Turing Test is *not* a test of whether a computer program is rational.

Good answer: The Turing Test is a test of whether a computer program is indistinguishable from a human.

- A stochastic environment is one in which an action may have more than one result.

  True.

- A chart parser is more efficient than a top-down parser because it never backtracks.

  False. A Chart parser is more efficient because it caches partial solutions in a chart, thereby avoiding re-expending portions of the search tree.

- A rational agent is an agent that does what a human would do in any situation.

  False. A rational agent selects actions that maximize expected performance measure.

- A toy problem is a problem used to illustrate or study the performance of an algorithm.

  True.

- A complete algorithm is one that always finds all possible solutions to a problem.

  False. A complete algorithm is guaranteed to find a solution if one exists.

- A corpus is something that a vampire eats.

  False. A corpus is a collection of documents used to create a language model.

- A Goal-based agent selects actions that are likely to maximize its utility. **False. A goal-based agent selects actions that will achieve its goal.**

2 Agents and environments

a. 24 pts, 6 pts each

For each of the following agents and environments, characterize the environment according to the six properties used in R & N (static vs dynamic, discrete vs. continuous, fully vs. partially observable, deterministic vs stochastic, episodic vs. sequential, single-agent vs. multi-agent). If necessary, you may want to give a **brief** justification of your answers.

- The game of checkers. **Static, discrete, fully observable, deterministic, sequential, multi-agent.**

- An autopilot that successfully lands a plane. **Dynamic, continuous, partially observable, stochastic, sequential, single-agent**
- An handwriting recognition agent that reads the addresses on envelopes and places each of them in the appropriate mailbox. **Static, discrete, fully observable, deterministic, episodic, single-agent**
- A music-playing accompanist agent that can play along with a human piano player. **Dynamic, continuous, partially observable, deterministic, sequential, multi-agent**

b. **6 pts**
Consider the following problem: We want to build an agent that can play word games.

For our first task, we would like to build an agent that can recognize palindromes. (A palindrome is a word or phrase that reads the same backwards as forwards excepting punctuation, such as "Anna" or "Madam, I'm Adam", or "A man, a plan, a canal - Panama!")

Assume that our agent receives its input one letter at a time. That is, at time t=0 it sees 'a', and and time t=1 it sees 'n', and so on. Once the string is seen completely, the agent must output 'yes' or 'no'.

Can a reflex agent solve this problem? If so, provide pseudocode for how such an agent might work. (You may use the back of the page if needed.) If not, explain **precisely** why it cannot solve this problem. **No - a reflex agent has no memory. Recognizing a palindrome requires memorizing past characters to ensure that a particular pattern has been seen.**

For our second task, we want to build an agent that can find anagrams. That is, rearrangements of a phrase that are also a phrase. For example, "scab" can be rearranged to "cabs", and "artificial intelligence" can be rearranged to make (among other things): "Intergalactic lie? Fie! Nil!" (punctuation added).

Let's assume that we want to build a goal-based agent that will use search to find an anagram for an arbirtary input phrase. Your task in this problem will be to formulate the problem precisely enough so that it can be implemented. In grading this, I'll ask two questions. 1) Is the answer correct? 2) Could I write a program from this answer without any additional information?

a) **4 pts** Give an initial state for this search.

An input string, and an empty output string.

b) **4 pts** Give a description of how a goal test could be implemented. **Split the phrase on space and look up each word in a dictionary. If all letters are used and all words are in the dictionary, the goal is reached.**

c) **4 pts** Describe what the successor function does. In other words, for a given input, what is produced? **For a given index i, return n-1 new strings with the letters at positions i and n-i exchanged.**

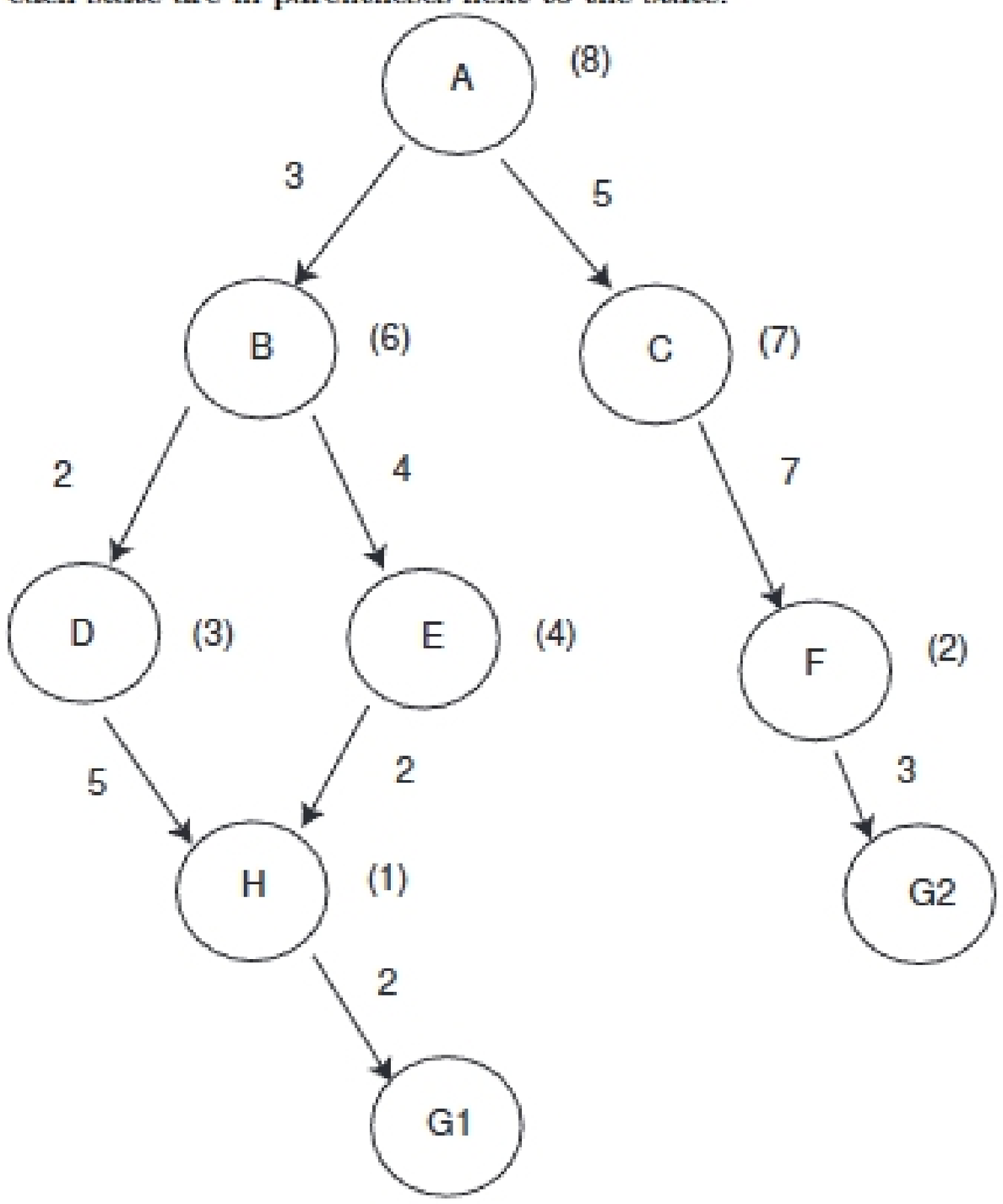
3 Search

a) **(4 pts each)** Give the **time and space** requirements in big-O terms for each of the following algorithms. Assume that the search tree has a branching factor of b, the solution is at depth d, and the tree has a maximum depth of n.

- Breadth-first search **Time:** $O(b^{d+1})$, **Space:** $O(b^{d+1})$
- Depth-first search **Time:** $O(b^n)$, **Space:** $O(bn)$
- Depth-limited search (depth $= d$) **Time:** $O(b^d)$, **Space:** $O(bd)$
- Iterative deepening search **Time:** $O(b^d)$, **Space:** $O(bd)$

b) **18 points - 6 points for each search**

Trace BFS, DFS, and A* on the following graph. Uninformed search methods should expand nodes from left-to-right. (B should be visited before C.) Both g1 and g2 are goal states; your algorithm may stop as soon as either is expanded. Show the order in which nodes are visited. For A*, also show the f,g, and h costs for each node visited. For A*, the costs for each edge are indicated next to the edge, and the heuristic costs for each state are in parentheses next to the state.



BFS: A,B,C,D,E,F,H,G2
DFS: A,B,D,H,G1
A*: A [B (g = 3, h = 6, f = 9), C (g = 5, h = 7, f = 12)]
B [D (g = 5, h = 3, f = 8), E (g = 7, h = 4, f = 11), C (g = 5, h = 7, f = 12)]
D [*H (g = 10, h = 1, f = 11), *E (g = 7, h = 4, f = 11), C (g = 5, h = 7, f = 12)]
H [E (g = 7, h = 4, f = 11), G1 (g = 12, h = 0, f = 12), C (g = 5, h = 7, f = 12)]
E [H (g = 9, h = 1, f = 10), G1 (g = 12, h = 0, f = 12), C (g = 5, h = 7, f = 12)]
H [G1 (g = 11, h = 0, f = 11), C (g = 5, h = 7, f = 12)]
G1. Goal. Optimal Path: A, B, E, H, G1
(*) Note: E and H can be enqueued in either order since they have the same f-value. Both answers are correct.