For each problem on this test, below “Perfect” gives the percentage who received full credit, “Partial” gives the percentage who received partial credit, and “Zero” gives the percentage of students who received zero credit.

(Due to rounding, values below may be only approximate estimates.)

Problem 1
Perfect: ~30% (~62 students), Partial: ~70% (~143 students), Zero: ~0% (~0 students)

Problem 2
Perfect: ~35% (~71 students), Partial: ~65% (~132 students), Zero: ~1% (~2 students)

Problem 2.b
Perfect: ~63% (~131 students), Partial: ~28% (~58 students), Zero: ~7% (~16 students)

Problem 2.c
Perfect: ~70% (~144 students), Partial: ~4% (~9 students), Zero: ~25% (~52 students)

Problem 2.d
Perfect: ~73% (~150 students), Partial: ~18% (~37 students), Zero: ~8% (~18 students)

Problem 2.e
Perfect: ~56% (~115 students), Partial: ~20% (~43 students), Zero: ~22% (~47 students)

CS-171, Intro to A.I., Fall Quarter, 2015 — Quiz # 1 — 20 minutes

NAME: _____

YOUR ID:_____ ID TO RIGHT:_____ ROW:_____ SEAT:_____

1. (52 pts total, 4 pts each) Properties of task environments. owing terms or phrases on the left, write in the letter corresponding to th he corr ample.

See Section 2.3.2 and Fig. 2.5.

For this test only, either L or N is accepted as correct. For future tests, we will insist upon the definitions as given your textbook, which for test purposes is a gold standard and a reliable study guide.

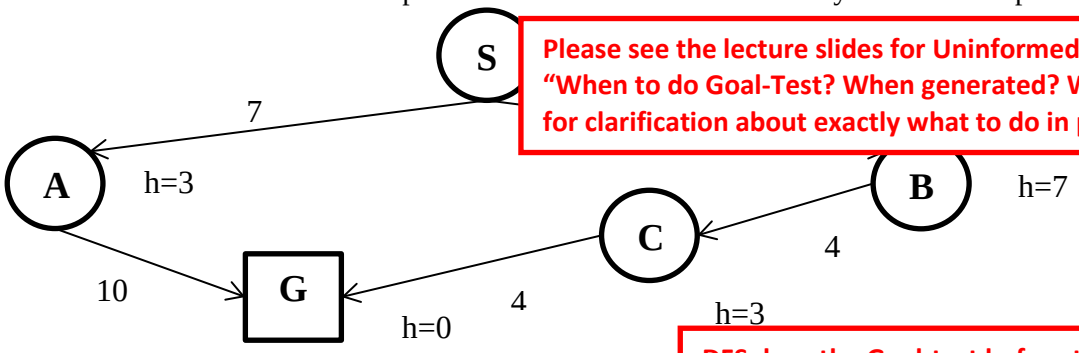
A			actuators
K	Episodic	B	Sensors give complete state of environment at each time point
G	Discrete	C	More than one agent in the task environment
I	Static	D	Next state is exactly determined by current state and agent action
N	Nondeterministic	E	The current decision could affect all future decisions
E	Sequential	F	Environment can change while the agent is deliberating
J	Semidynamic	G	Finite number of states, percepts, and actions
D	Deterministic	H	The outcomes (or probabilities) for all actions are given
B	Fully observable	I	Environment does not change while the agent is deliberating
M	Uncertain	J	Environment does not change while the agent is deliberating, but its performance measure does
H	Known	K	A series of atomic episodes, each independent of prior agent actions
C	Multiagent	L	Next state not exactly determined by current state and agent action
L	Stochastic	M	Not fully observable or not deterministic
F	Dynamic	N	Actions are characterized by their *possible* outcomes, but no probabilities are attached to them

A question arose about 'Stochastic' vs. 'Nondeterministic.' After rereading the definitions in Section 2.3.2, it appears that Nondeterministic is a subset of Stochastic. Stochastic appears to refer to the general case in which an action in a state might not exactly determine the next state, for any of a wide variety of reasons; e.g., for the taxi-driving robot, if the street is wet the braking action might unexpectedly produce a skid state instead of a halt state, if there is a mechanical error in the steering mechanism a steering wheel turn of 90 degrees might unexpectedly produce a different turn angle, if there is a moron in the next lane his sudden and unfortunate lane change into us might unexpectedly produce a crash state instead of a travel state, etc. Nondeterministic appears to refer to a special case of Stochastic, which is analogous to a nondeterministic finite automaton in that a given action in a given state results nondeterministically in one of several possible outcome states (often for reasons unknown to the agent); e.g, the action of eating an unknown mushroom results nondeterministically in either a dead state or a live state (depending on whether or not the mushroom was poisonous).

I agree that this distinction is subtle and confusing (though I believe it is useful). For purposes of this Quiz #1 only, both L and N will be accepted as correct answers to both Stochastic and Nondeterministic. For future tests, we will insist upon the definitions as given your textbook, which for test purposes is a gold standard and a reliable study guide.

2. (48 pts total, 12 pts each) Execute Tree Search through this graph (i.e., do not remember visited nodes). Step costs are given next to each arc. Heuristic values are given next to each node (as h=x). The successors of each node are indicated by the arrows out of that node. Successors are returned in left-to-right order.

For each search strategy below, show the order in which nodes are expanded (i.e., to expand a node means that its children are generated), ending with the goal node that is found. Show the path from start to goal, or write "None". Give the cost of the path found. The first one is done for you as an example.



Please see the lecture slides for Uninformed Search, topic "When to do Goal-Test? When generated? When popped?" for clarification about exactly what to do in practical cases.

2.a. DEPTH FIRST SEARCH.

See Section 3.4.3 and Fig. 3.17.

DFS does the Goal-test before the child is pushed onto the queue. The goal is found when A is expanded.

Order of node expansion: S A G

Path found: S A G Cost of path found: 17

2.b. (12 pts) UNIFORM COST SEARCH.

See Section 3.4.2 and Fig. 3.14.

UCS does goaltest when node is popped off queue.

(5 pts) Order of node expansion: S B A

(5 pts) Path found: S B C G **(2 pts)** Cost of path found: 12

2.c. (12 pts) GREEDY (BEST-FIRST) SEA

GBFS has the same behavior whether the goaltest is done before node is pushed or after node is popped, because h=0 for a goal node, so goal nodes always sort to the front of the queue anyway.

(5 pts) Order of node expansion: S A G

See Section 3.5.1 and Fig. 3.23.

(5 pts) Path found: S A **(2 pts)** Cost of path found: 17

2.d. (12 pts) ITERATED DEEPENING SEARCH.

IDS does the Goal-test before the child is pushed onto the queue. The goal is found when A is expanded.

(5 pts) Order of node expansion: S S A G

See Sections 3.4.4-5 and Figs. 3.18-19.

(5 pts) Path found: S A **(2 pts)** Cost of path found: 17

2.e. (12 pts) A* SEARCH.

A* does goaltest when node is popped off queue.

(5 pts) Order of node expansion: S A B C G

See Section 3.5.2 and Figs. 3.24-25.

(5 pts) Path found: S **(2 pts)** Cost of path found: 12

TECHNICAL NOTE: Technically, the goal node is not expanded, because no children of a goal node are generated. The goal node is listed in "Order of node expansion" for your convenience. Your answer is correct if you do not show the goal node in "Order of node expansion" — but it is a nicety to do so, because it is informative to see which goal was found when. For clarity and simplicity, answer keys always will show the goal node in "Order of node expansion," even though you are not strictly obliged to do so. Nevertheless, "Path found" *always* must show the goal node, because a path to a goal always must end in a goal.

A question arose about IDS search (question 2.b above). For clarity:

(1) Please review Fig. 3.17 & Fig. 3.18 in your textbook.

(2) Please remember that IDS begins with L=0, not L=1. This is so that you can solve trick problems like, "Starting at Arad, go to Arad." At L=0 the start node is tested to see if it is a goal, but it is not expanded (= no children are generated).

(3) Please follow along Fig. 3.17 in your textbook as we work the question in detail. (Note that I ignore their cutoff and cutoff_occurred variables for simplicity because they do not play a role below anyway.)

(3.0) Do Recursive-DLS [called RDLS below] on start node S with limit=0. Goal-test on S fails. limit=0 so return. No nodes were expanded (= no children were generated).
Nodes expanded this iteration=NIL.
Cumulative order of node expansion=NIL

(3.1) Do RDLS on S with limit=1. Goal-test on S fails. limit=1 so continue.
Expand S to yield children A, B.
Do RDLS on A with limit=0. Goal-test on A fails. limit=0 so return. (Do not expand A, i.e., do not generate A's children.)
Do RDLS on B with limit=0. Goal-test on B fails. limit=0 so return. (Do not expand B, i.e., do not generate B's children.)
Nodes expanded this iteration=S.
Cumulative order of node expansion=S.

(3.2) Do RDLS on S with limit=2. Goal-test on S fails. limit=2 so continue.
Expand S to yield children A, B.
(3.2.1) Do RDLS on A with limit=1. Goal-test on A fails. limit=1 so continue.
Expand A to yield children G1, C.
Do RDLS on G1 with limit=0. Goal-test on G1 succeeds. Return G1 as the search goal result that was found.
Nodes expanded this iteration=S, A, G1.
Cumulative order of node expansion=S, S, A, G1.