Below, for each problem on this test, “Perfect” is the percentage of students who received full credit, “Partial” is the percentage who received partial credit, and “Zero” is the percentage who received zero credit.

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

Problem 1
Perfect: ~50% (~61 students), Partial: ~50% (~61 students), Zero: ~0% (~0 students)

Problem 2
Perfect: ~5% (~6 students), Partial: ~93% (~114 students), Zero: ~2% (~2 students)

Problem 2.b
Perfect: ~11% (~13 students), Partial: ~70% (~85 students), Zero: ~20% (~24 students)

Problem 2.c
Perfect: ~61% (~74 students), Partial: ~2% (~2 students), Zero: ~38% (~46 students)

Problem 2.d
Perfect: ~30% (~37 students), Partial: ~34% (~42 students), Zero: ~35% (~43 students)

Problem 2.e
Perfect: ~54% (~66 students), Partial: ~26% (~32 students), Zero: ~20% (~24 students)

We will release these numbers as they become available.

CS-171, Intro to A.I., Winter Quarter, 2016 — Quiz # 2 — 20 minutes

NAME: _____

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

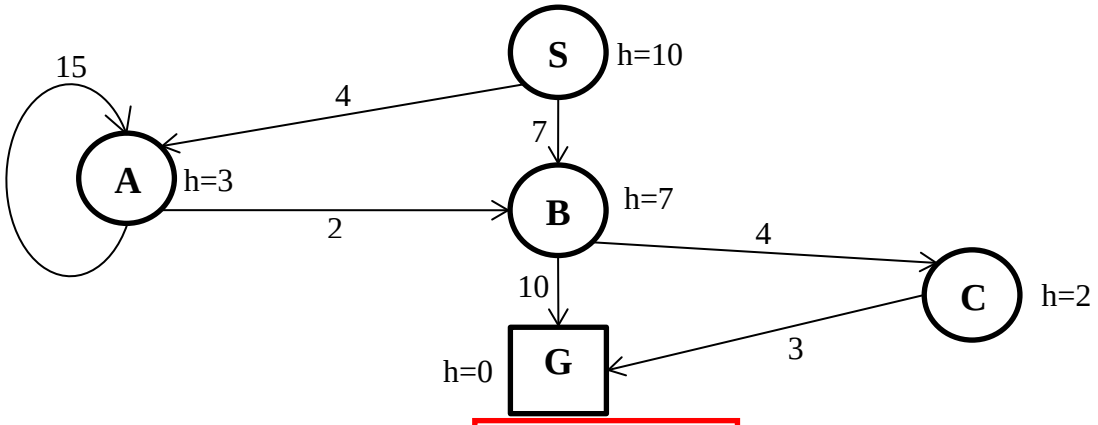
1. (60 pts total, 4 pts each) For each of the following terms on the left, write in the letter corresponding to the best answer or the correct definition on the right. The first one is done for you as an example.

A	Agent	A	Perceives environment by sensors, acts by actuators
K	Percept	B	All states reachable from the initial state by a sequence of actions
P	Performance Measure	C	Guaranteed to find a solution if one is accessible
L	Rational Agent	D	Process of removing detail from a representation
B	State Space	E	Maximum number of successors of any node
I	Search Node	F	Set of all leaf nodes available for expansion at any given time
N	Link between nodes	G	Estimates cost of cheapest path from current state to goal state
J	Path	H	Guaranteed to find lowest cost among all accessible solutions
D	Abstraction	I	Represents a state in the state space
H	Optimal Search	J	Sequence of states connected by a sequence of actions
C	Complete Search	K	Agent's perceptual inputs at any given instant
M	Expand a state	L	Agent that acts to maximize its expected performance measure
F	Frontier	M	Apply each legal action to a state, generating a new set of states
O	Search Strategy	N	Represents an action in the state space
E	Branching Factor	O	How a search algorithm chooses which node to expand next
G	Heuristic Function	P	Evaluates any given sequence of environment states for utility

****** TURN PAGE OVER AND CONTINUE ON THE OTHER SIDE ******

2. (40 pts total, 10 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, i.e., successors of S are (A, B), successors of A are (A, B), and successors of B are (G, C), in that 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, or indicate the repeating cycle if the search gets stuck in a loop. 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.



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.

Order of node expansion: S A A A A A ...

Path found: None Cost of path found: N/A

2.b. (10 pts) UNIFORM COST SE

UCS does goaltest when node is popped off queue.

See Section 3.4.2 and Fig. 3.14.

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

(4 pts) Path found: S A B C G **(2 pts)** Cost of path found: 13

2.c. (10 pts) GREEDY (BEST-FIRST) SEARC

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.

(4 pts) Order of node expansion: S A A A A A ...

See Section 3.5.1 and Fig. 3.23.

(4 pts) Path found: None **(2 pts)** Cost of path found: N/A

2.d. (10 pts) ITERATED DEEPENING SEARCH.

IDS does the Goal-test before the child is pushed onto the queue.

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

See Sections 3.4.4-5 and Figs. 3.17-19.

(4 pts) Path found: S B G **(2 pts)** Cost of path found: 17

2.e. (10 pts) A* SEARCH.

A* does goaltest when node is popped off queue.

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

See Section 3.5.2 and Figs. 3.24-25.

(4 pts) Path found: S A B C G **(2 pts)** Cost of path found: 13

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. 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.d 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). Thus, the goal-test at L=0 does NOT contribute to "Order of Nodes Expanded" (recall that to expand a node means to generate its children).

(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 A, B.
Do RDLS on A with limit=0. Goal-test on A fails. limit=0 so return. (Do not expand A again, i.e., do not generate A's children again.)
Expand B to yield children G, C.
Do RDLS on G with limit=0. Goal-test on G succeeds. Return G as the search goal result that was found.
Nodes expanded this iteration=S, A, B, (G).
Cumulative order of node expansion=S, S, A, B, (G).

(Technically, the goal node is not expanded, because no children of a goal node are generated. The goal node is listed in "Nodes expanded this iteration" and "Cumulative order of node expansion" for your convenience. Your answer is correct if you do not show the goal node in "Order of node expansion" above — but it is a nicety to do so.)