6.034 Quiz 1

30 September 2015

Name	Alissa P. Hacker
Email	

Circle your TA **(for 1 extra credit point)**, so that we can more easily enter your score in our records and return your quiz to you promptly.

Ryan Alexander	**Ben Greenberg**	**Neil Gurram**
Dylan Holmes	**Eeway Hsu**	**Brittney Johnson**
Veronica Lane	**Robert Luo**	**Jessica Noss**
Mycal Tucker	**Sarah Vente**	**Jess Wass**

Note: We only expected students to write the parts of the solutions that are written in black, or highlighted in green.

Problem number	Maximum	Score	Grader
1-Rules	40		
2-Search	30		
3-Games	30		
Total	100		

There are 14 pages in this quiz, including this one, but not including tear-off sheets. Tear-off sheets with duplicate drawings and data are located after the final page of the quiz. As always, open book, open notes, open just about everything, including a calculator, but no computers.

This page contains no quiz material.

Problem 1: Rule-Based Systems (40 points)

A newspaper journalist wants to prove that Tony Stark is Iron Man, starting from the following rules:

P0	IF '(?a) sells weapons'
	THEN '(?a) is a genius'

P1	IF AND('(?a) is a genius',
	       OR('(?a) is captured',
	          '(?a) is evil'))
	THEN '(?a) builds a suit'

P2	IF '(?a) plots against (?b)'
	THEN ('(?a) is evil',
	      '(?b) is captured')

P3	IF AND('(?a) sells weapons',
	       '(?b) is evil')
	THEN ('(?a) catches (?b) selling illegal weapons',
	      '(?a) stops selling weapons')

P4	IF AND('(?a) builds a suit',
	       '(?a) catches (?b) selling illegal weapons')
	THEN '(?a) is Iron Man'

Note: For your convenience, a copy of these rules is provided on a tear-off sheet after the last page of the quiz.

Part A: Backward Chaining (20 points)

Make the following assumptions about backwards chaining:

- The backward chainer tries to find a matching assertion in the list of assertions. If no matching assertion is found, the backward chainer tries to find a rule with a matching consequent. In case no matching consequents are found, the backward chainer concludes that the hypothesis is false.
- The backward chainer never alters the list of assertions, so it can derive the same result multiple times.
- Rules are tried in the order they appear.
- Antecedents are tried in the order they appear.
- Lazy evaluation/short circuiting is in effect

The journalist starts with four **assertions for backward chaining:**

```
A0: 'Stark is a genius'
A1: 'Stark sells weapons'
A2: 'Stark is captured'
A3: 'Obadiah is evil'
```

Using these assertions, perform backward chaining starting from the hypothesis:

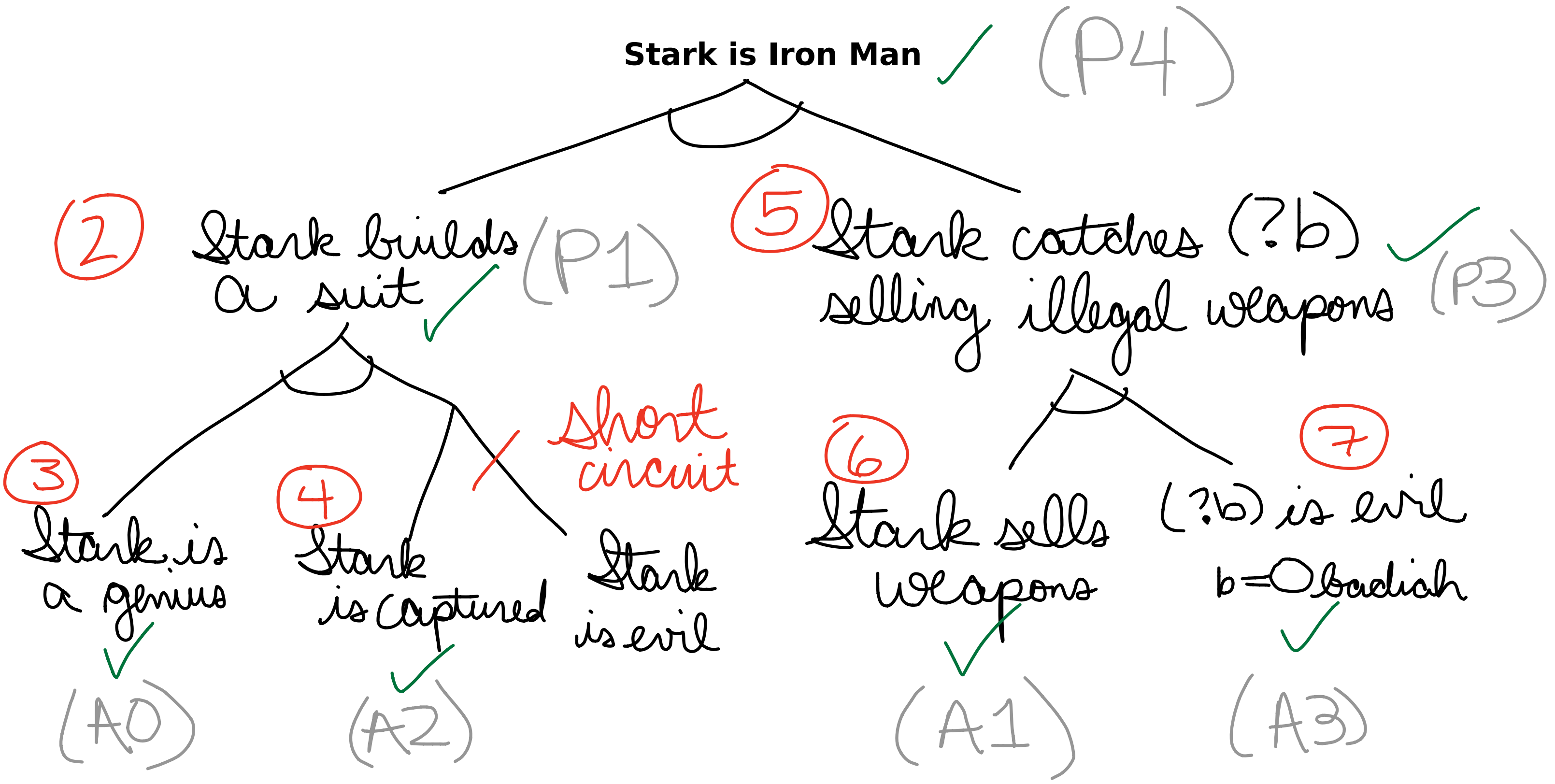
'Stark is Iron Man'

- In the table below, write all the hypotheses that the backward chainer checks, in the order they are checked. (Some lines have been filled in for you, and the table may have more lines than you need.)
- You can show your work for partial credit: Use the space on the next page to draw the goal tree that would be created by backward chaining from this hypothesis.

Hypotheses looked for in backward chaining

1. Stark is Iron Man
2. Stark builds a suit
3. Stark is a genius
4. Stark is captured
5. Stark catches (?b) selling illegal weapons
6. Stark sells weapons
7. (?b) is evil
8.
9.
10.
11.
12.
13.
14.
15.

Space provided to draw a goal tree (for partial credit).



Part B: Forward Chaining (20 points)

Another journalist uses the same rules, but starts with a **different list of assertions**:

```
A0: 'Obadiah is a genius'
A1: 'Stark sells weapons'
A2: 'Obadiah plots against Stark'
A3: 'Stark is a genius'
```

Using **this new list of assertions**, fill in the table below for the first four iterations of forward chaining.

- For the first three iterations, list the rules whose antecedents match the assertions, the rule that fires, and any new assertions that are added. (The matched rules for iterations 2-4 have been filled in for you.)
- For the fourth iteration, supply only the fired rule.
- If no rules match or fire, or no new assertions are generated, write NONE in the corresponding box.

Make the following assumptions about forward chaining:

- When multiple rules match, rule-ordering determines which rule fires.
- New assertions are added to the bottom of the list of assertions.
- If a particular rule matches in multiple ways, the matches are considered in top-to-bottom order of the matched assertions. (For example, if a particular rule has a match with A1, and another match with A2, the match with A1 is considered first.)

P0 NEVER fires

'Stark is a genius' is not new

Iter	Rules Matched	Rule Fired	New Assertions Added
1	P0, P2	P2	Obadiah is evil Stark is captured
2	P0,P1,P2,P3	P1	Obadiah builds a suit
3	P0,P1,P2,P3	P1	Stark builds a suit
4	P0,P1,P2,P3	P3	

multiple assertions may be added when a rule fires

Each Obadiah assertion precedes each Stark assertion, so Obadiah matches 1st

Only ONE rule fires at a time

Problem 2: Search (30 points)

Part A: Shortest Path (5 points)

Consider the following list of search methods. In general, for any graph for which you have access to an **admissible** (not necessarily consistent) heuristic, which of these search methods are **guaranteed to find the shortest path to the goal**? (Assume that a path to the goal exists.)

A. A* search (with an extended set and with the admissible heuristic)

B. British Museum search

C. Branch and bound with no extended set and without any heuristic

D. Branch and bound **with the admissible heuristic** (and no extended set)

E. Branch and bound **with an extended set** (without any heuristic)

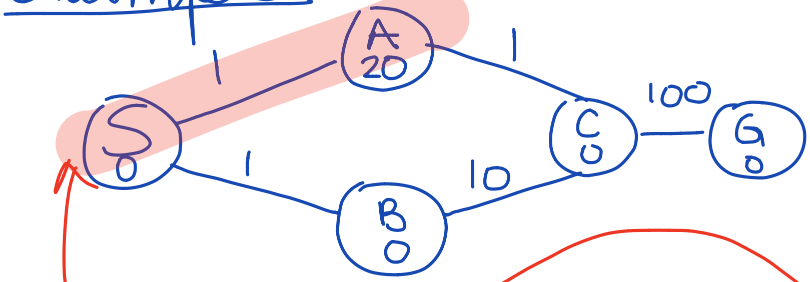
List **all methods (A-E) that apply**, or NONE if none apply

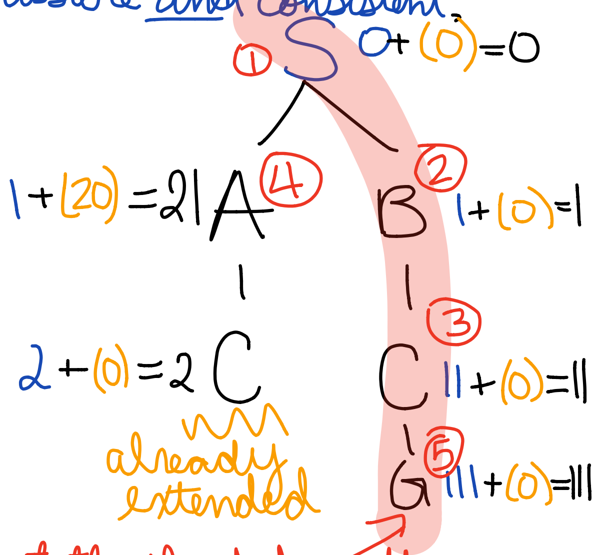
BCDE

Explanation

A. A* is not guaranteed to find the shortest path because the heuristic is not necessarily consistent. It must be both admissible and consistent.

Example





inconsistent:

$$|h(S,G)-h(A,G)| \overset{?}{\leq} c(S,A)$$

$$|0-20| \cancel{\leq} 1$$

not the shortest path

B. British Museum searches every possible path, therefore, it will find the shortest one (although inefficiently).

C) Branch & Bound is an optimal search, so it finds the shortest path.

D) With an admissible heuristic, it finds the shortest path (more efficiently than it would with no heuristic). Because it does not use an extended set, the heuristic does not have to be consistent.

E) With an extended set, it still finds the shortest path (more efficiently than with no extended set), because it always extends the shortest path so far. So even if the search algorithm gets to a node twice, the first time it extends the node will be the shortest path to the node.

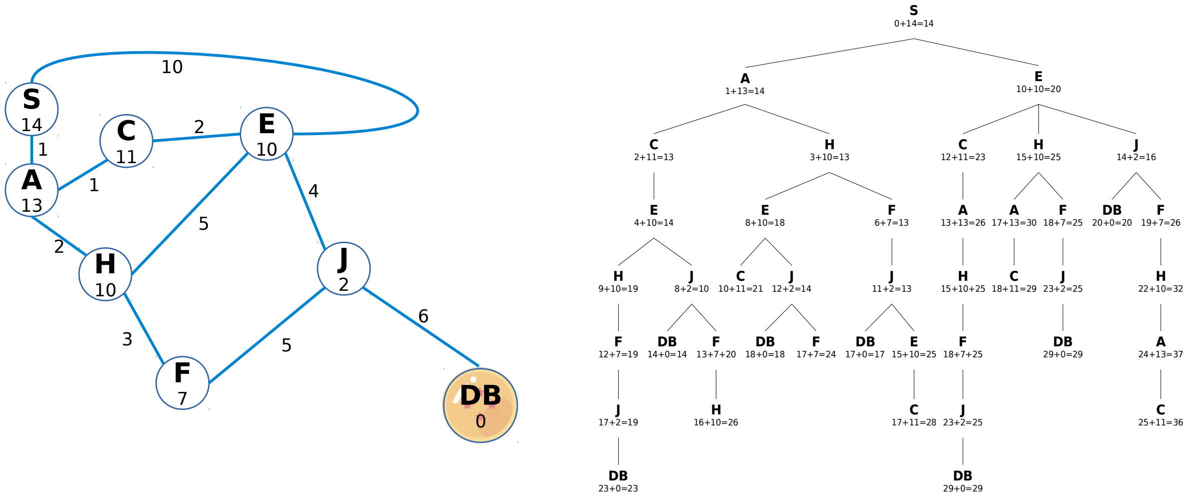
Part B: Instant Transmission (10 points)

Goku and Bulma are looking for the 4-star dragon ball in the British Museum of Natural History. They want to find a path from the entrance at “S” to the dragon ball at “DB” using the map shown below. The map has hallway lengths marked, and numbers inside each node indicate heuristic estimates of the distance to the goal. Note that the heuristic is **admissible but not consistent**.

To help visitors with their calculations, the museum has also provided the complete British Museum search tree, shown below. Each node is marked with three numbers in the following form:

(path length) + (heuristic to goal) = (path length + heuristic)

You can use this tree as a reference when drawing trees in this problem, Parts B and C.

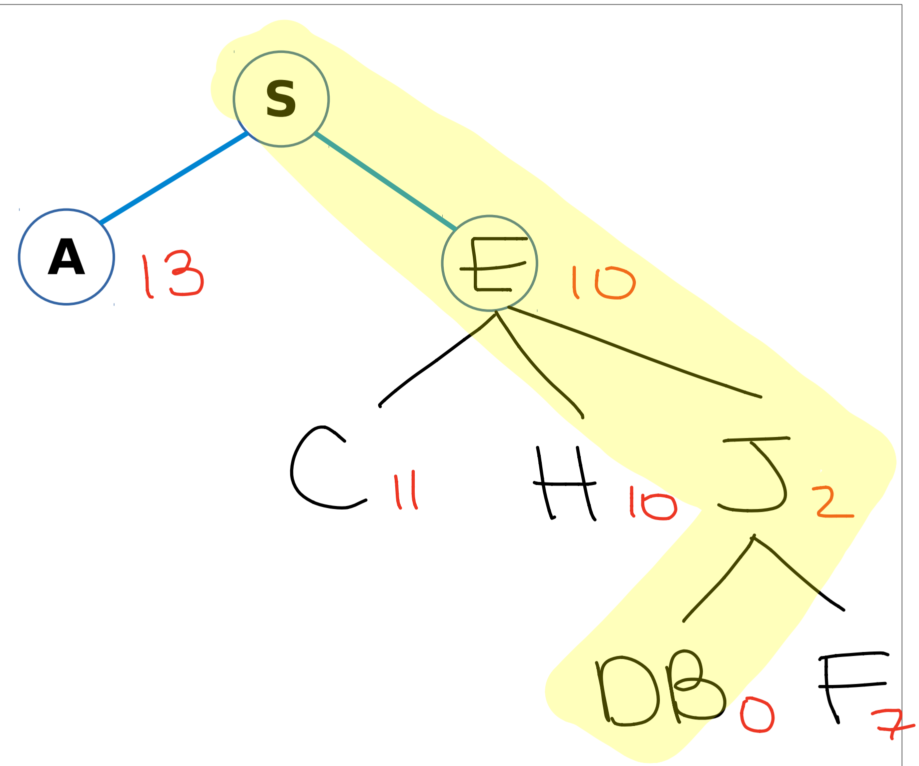


See tear-off sheet for a larger copy of this map and tree

B1 (8 points) Goku, who can teleport between rooms, volunteers to try hill-climbing search. Use **hill-climbing search** to help Goku find a path from “S” to “DB”. Draw your search tree in the space provided on the next page.

- Draw the children of each node in alphabetical order. (A < B < C)
- Break any ties using lexicographic/dictionary order. (S-P-Y < S-Q-X) (If two paths S-P-Y and S-Q-X are tied, extend path S-P-Y before path S-Q-X because 'SPY' would come before 'SQX' in the English dictionary.)
- As in Lab 2, the search should exit when a path to goal is **first removed from the agenda**.
- Do **<u>NOT</u>** use an extended set.
- Use backtracking.

For credit, **finish drawing the search tree** for **hill-climbing search** in the space below. Some of the labels are missing, and you may need to add additional nodes to the tree.



Hill Climbing uses only the heuristic (not path length) to sort nodes at each level.

Hill Climbing is like DFS but it extends the node with the best heuristic value at a given level.

B2 (2 points) What is the path that you found using hill-climbing search in part B1? (Write the path found, including nodes S and DB, or write NONE if hill-climbing search found no path.)

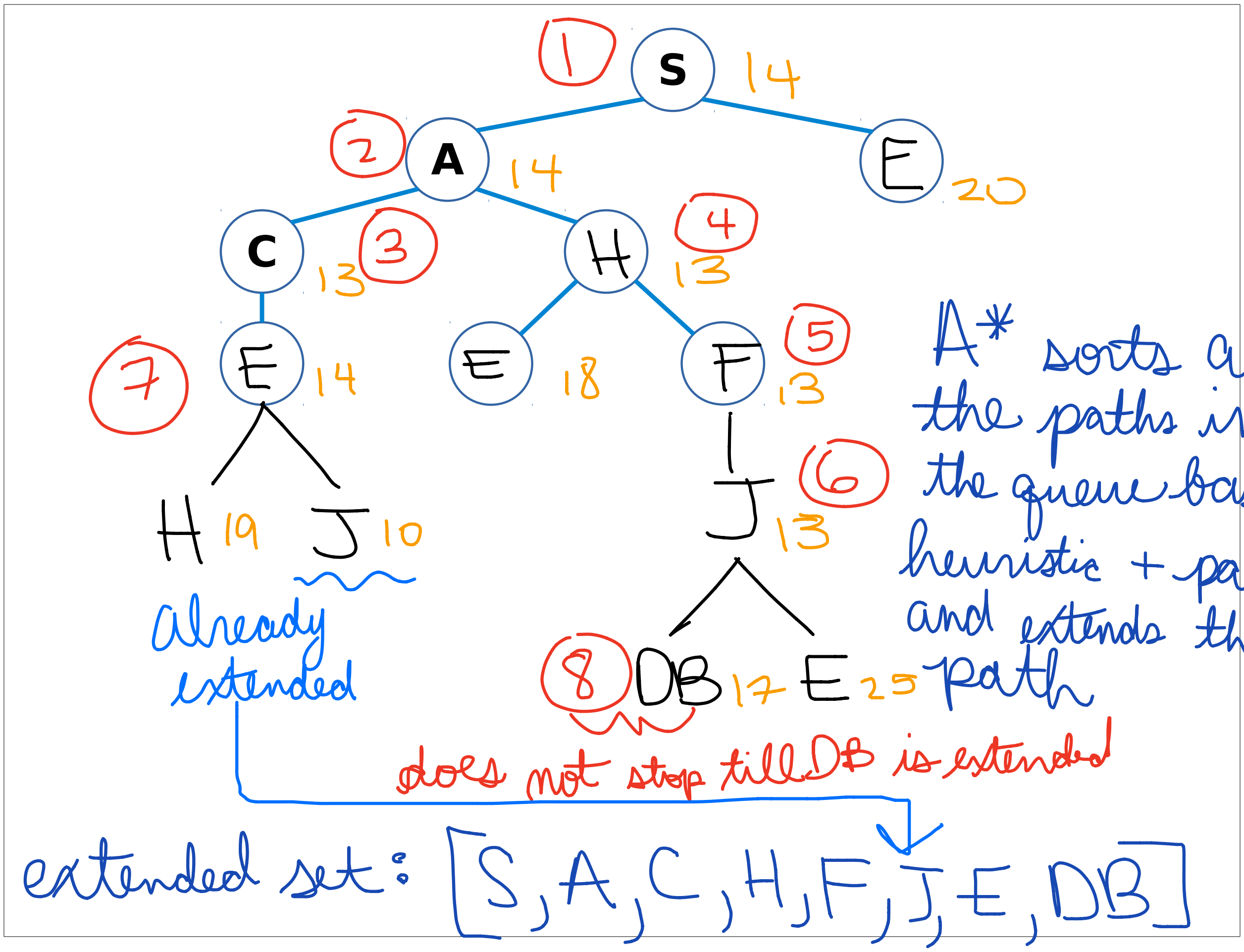
S, E, J, DB

Part C: Puny Humans (15 points)

C1 (13 points) Bulma, a common human, is forced to walk between the rooms. She wants to use A* search to find the shortest path to the goal. Use **A* search** to help Bulma find a path from “S” to “DB”.

- Draw the children of each node in alphabetical order. (A < B < C)
- Break any ties using lexicographic/dictionary order. (S-P-Y < S-Q-X)
- Use the naive implementation of an extended set: If a path's last node is already in the extended set, don't extend that path.
- For your convenience, a **duplicate copy** of the partial search tree for A* is provided on the next page. If you write on both copies, indicate clearly which one you want us to grade.

For credit, **finish drawing the search tree** for **<u>A* search</u>** in the space below. Some of the labels are missing, and you may need to add additional nodes to the tree. For reference, you can use the complete British Museum search tree on the tear-off sheet.



A* sorts all the paths in the queue based on heuristic + path length and extends the best path

already extended

does not stop till DB is extended

extended set: [S, A, C, H, F, J, E, DB]

C2 (2 points) What is the path that you found using A* search in part C1? (Write the path found, including nodes S and DB, or write NONE if A* search found no path.)

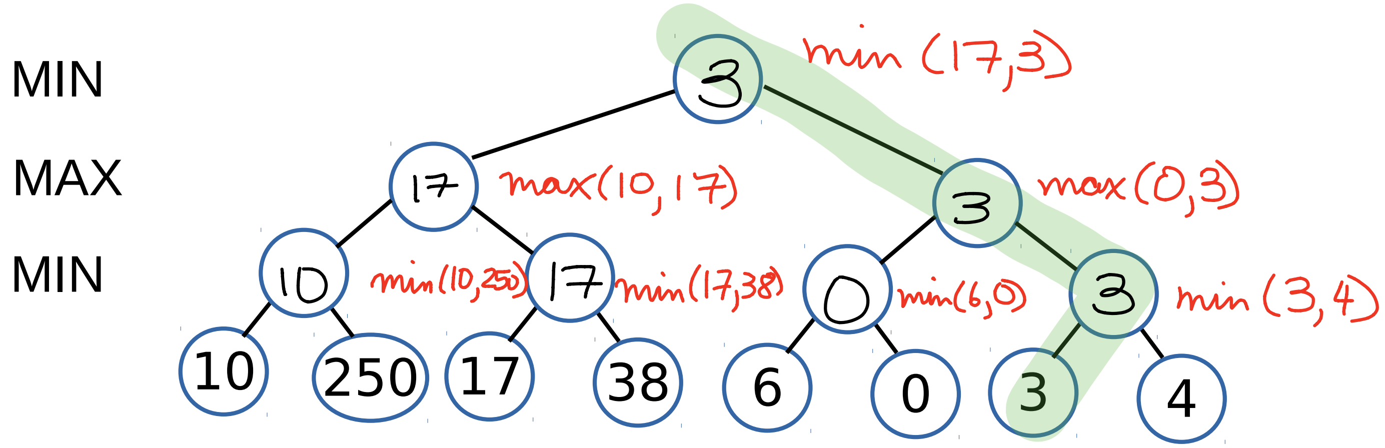
S, A, H, F, J, DB } note that because the heuristic is inconsistent this is not the shortest path

Problem 3: Games (30 points)

Part A: Minnie-max (10 points)

Minnie Mouse is celebrating her birthday by playing a board game with Mickey. Minnie uses a strategy she calls minnie-max (minimax). Minnie wants to minimize the score, while Mickey wants to maximize it. Because it is her birthday, **Minnie (MIN) plays first**.

The game tree below shows the static values (heuristic scores) three moves ahead.



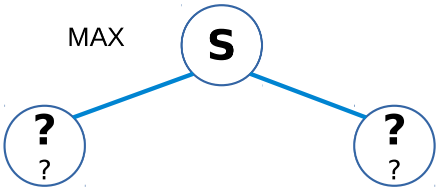
Complete the game tree above using minimax with NO pruning:

1. Write the minimax score in each node.
2. **<u>**Circle the minimax path**</u>** (the optimal sequence of moves that Minnie and Mickey will play).

Part B: Progressive Deepening (20 points)

Donald Duck wants to play a new game, but is short on time due to his presidential campaign. He challenges Minnie using **minimax with alpha-beta pruning and progressive deepening**. Donald also uses the **reordering trick** to increase his chances of pruning.
It is Donald's (**MAX**'s) turn.

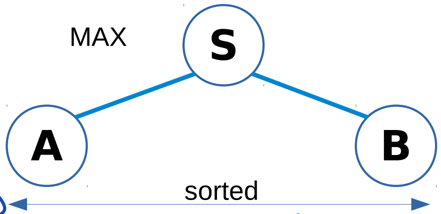
B1 (3 points) First, Donald looks one move ahead, evaluating both of the nodes at depth 1 (values not shown here). If Donald runs out of time now, before reordering, which move will he make? (Circle one)



Left **Right** [✓ **Can't tell**]

Before reordering, it's impossible to know if the right or left choice is better. Reordering sorts left to right best to worst.

B2 (3 points) Next, Donald sorts the nodes at depth 1 to optimize for alpha-beta pruning at depth 2. Based on the resulting tree (shown at right), what move will Donald make if he runs out of time before moving to depth 2? (Circle one)

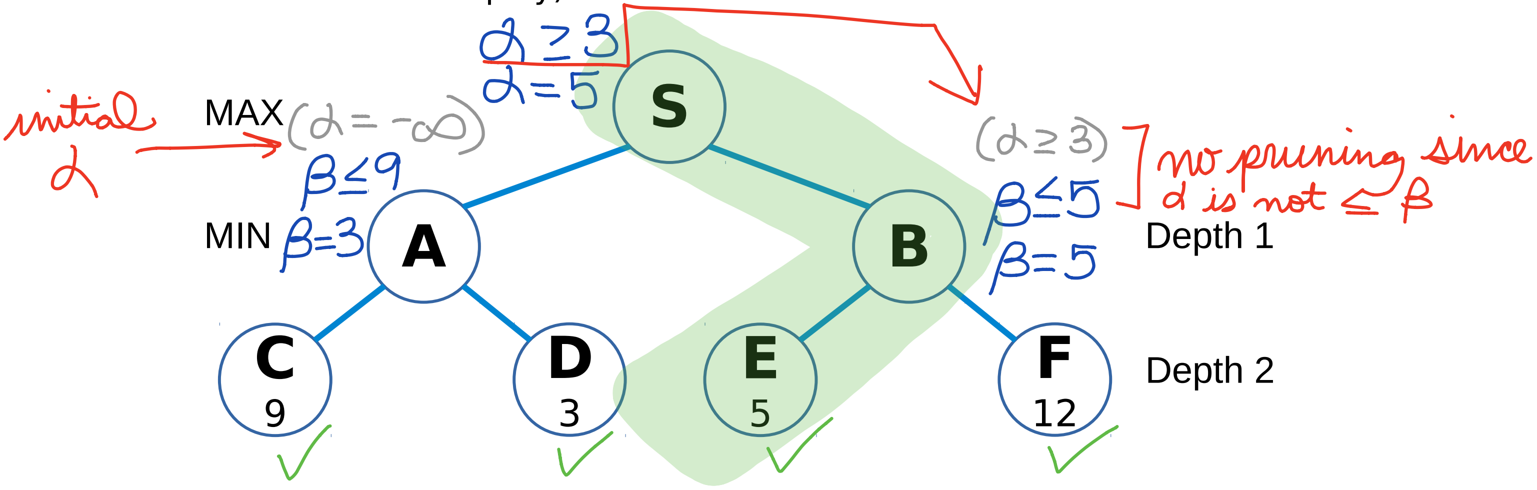


α-β always sorts (left) best to worst (right), based on which player is choosing for max, this is greatest to least, for min it's least to greatest.

[✓ Move A] Move B Can't tell

Therefore, after sorting the best choice is always on the left

B3 (3 points) Now Donald is ready to search to depth 2 using minimax with alpha-beta pruning. In the game tree below, using the static values shown, find and **circle the minimax path** (the optimal sequence of moves that Donald and Minnie will play).



B4 (3 points) In the game tree above (searching to depth 2), which leaf nodes would be pruned by minimax with alpha-beta? Write **all leaf nodes that were pruned** (C, D, E, F), or NONE if no nodes were pruned.

None

No nodes are pruned because α is never greater than β. If F were 2 for example, then max would have picked A.

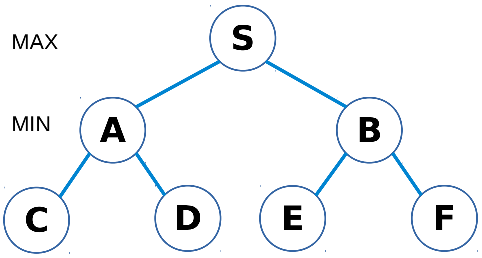
B5 (3 points) If Donald runs out of time after searching to depth 2, what move will he make? (Circle one)

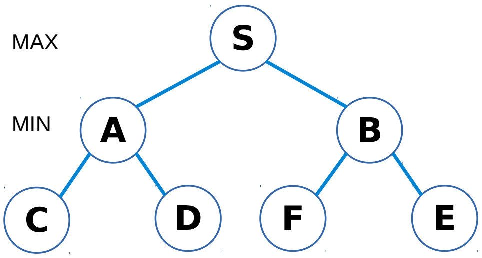
Move A [✓ Move B] Can't tell

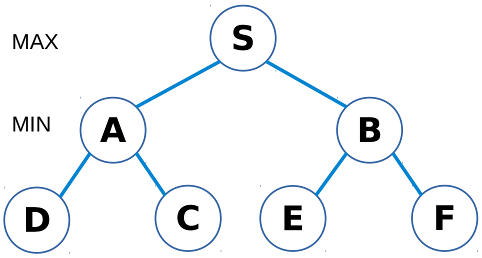
Move B gives him the highest possible score since min will choose E. If max chose A, min would have chosen D & max would get a score of 3.

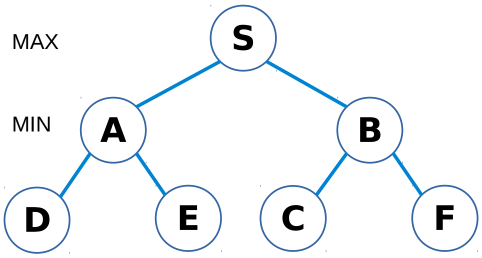
B6 (5 points) Finally, Donald will sort the tree in preparation for doing progressive deepening to depth 3. Which of the following trees represents the correctly sorted tree? **(Circle one)**

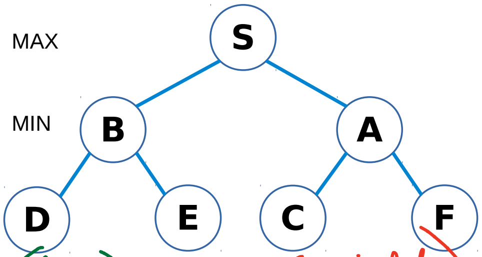
(For reference, a copy of the depth-2 tree from Part B3 is provided on a tear-off sheet.)

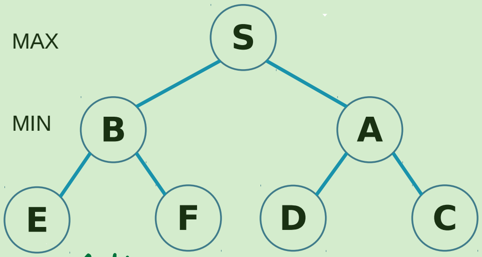












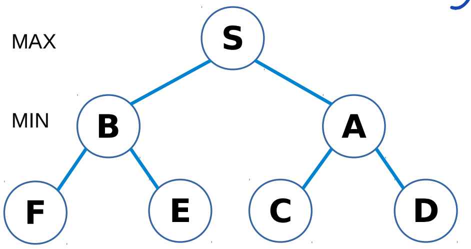
(left) (right)

Sort best to worst. For max: B > A

left right

For min: E < F

D < C



NONE OF THESE

(Tear-off sheet)

Rules for Problem 1, Parts A & B

P0	IF '(?a) sells weapons'
	THEN '(?a) is a genius'

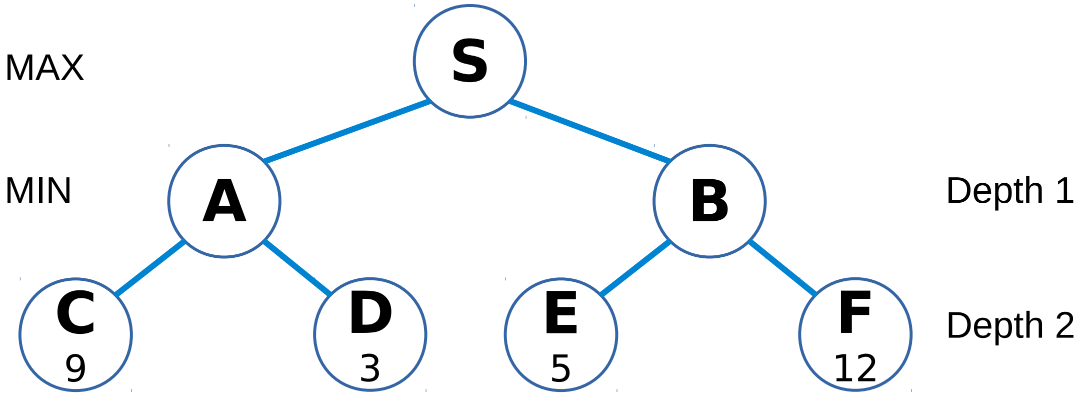
P1	IF AND('(?a) is a genius',
	OR('(?a) is captured',
	'(?a) is evil'))
	THEN '(?a) builds a suit'

P2	IF '(?a) plots against (?b)'
	THEN ('(?a) is evil',
	'(?b) is captured')

P3	IF AND('(?a) sells weapons',
	'(?b) is evil')
	THEN ('(?a) catches (?b) selling illegal weapons',
	'(?a) stops selling weapons')

P4	IF AND('(?a) builds a suit',
	'(?a) catches (?b) selling illegal weapons')
	THEN '(?a) is Iron Man'

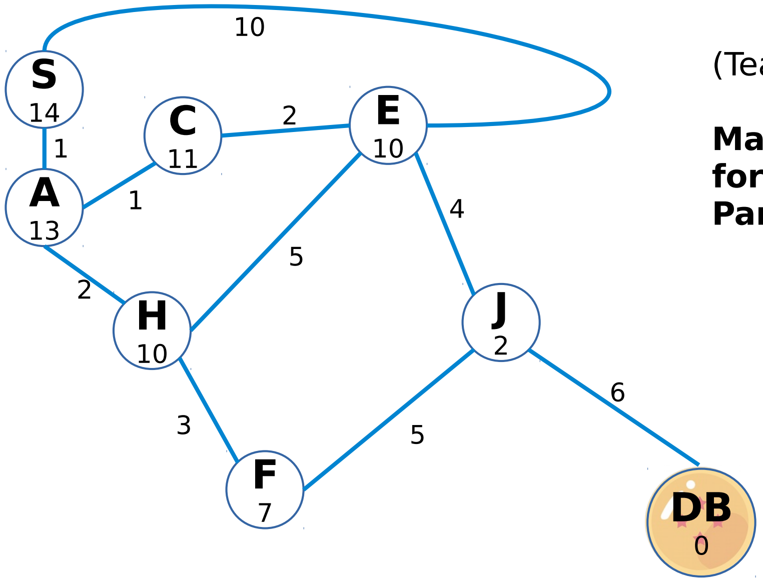
Tree for Problem 3, Part B6

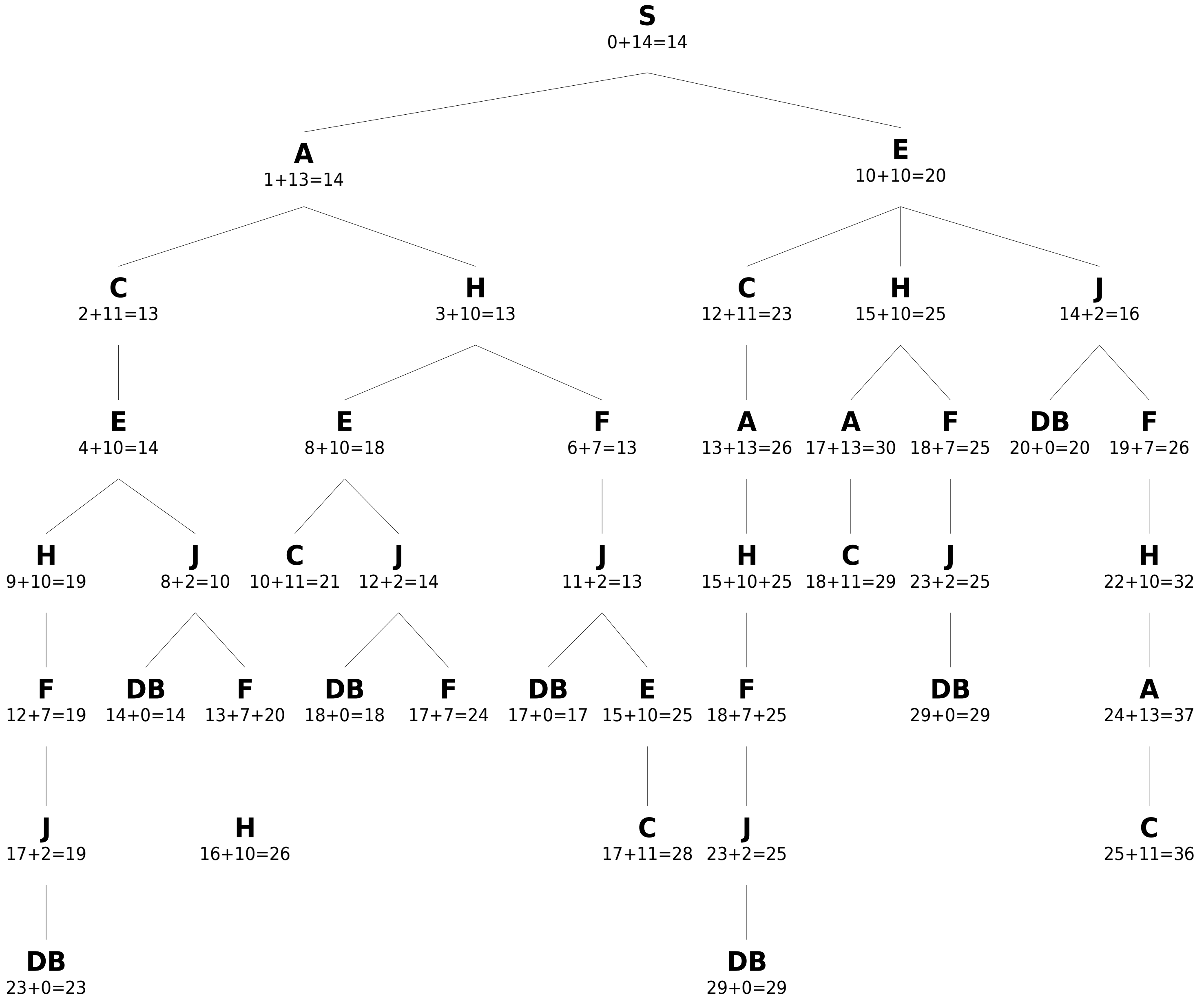


(see reverse for Problem 2)

(Tear-off sheet)

Map and Tree for Problem 2, Parts B & C


S
14
10
C
11
2
E
10
1
A
13
1
4
5
2
H
10
J
2
6
3
5
F
7
DB
0


S
0+14=14
A
1+13=14
E
10+10=20
C
2+11=13
H
3+10=13
C
12+11=23
H
15+10=25
J
14+2=16
E
4+10=14
E
8+10=18
F
6+7=13
A
13+13=26
A
17+13=30
F
18+7=25
DB
20+0=20
F
19+7=26
H
9+10=19
J
8+2=10
C
10+11=21
J
12+2=14
J
11+2=13
H
15+10+25
C
18+11=29
J
23+2=25
H
22+10=32
F
12+7=19
DB
14+0=14
F
13+7+20
DB
18+0=18
F
17+7=24
DB
17+0=17
E
15+10=25
F
18+7+25
DB
29+0=29
A
24+13=37
J
17+2=19
H
16+10=26
C
17+11=28
J
23+2=25
C
25+11=36
DB
23+0=23
DB
29+0=29