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: ~47% (~28 students), Partial: ~53% (~32 students), Zero: ~0% (0 students)

Problem 2
Perfect: ~53% (~32 students), Partial: ~45% (~27 students), Zero: ~2% (~1 student)

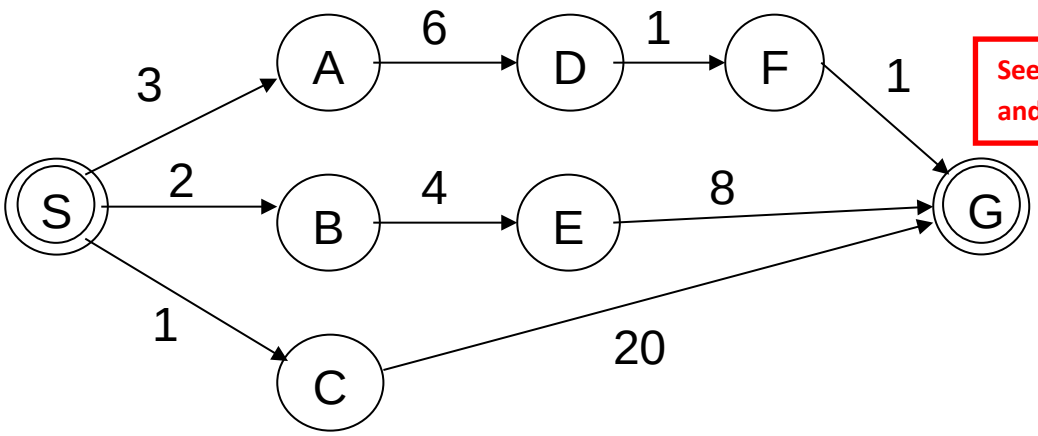
Problem 3
Perfect: ~92% (~55 students), Partial: ~7% (~4 students), Zero: ~2% (~1 students)

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

NAME: _____

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

1. (48 pts total, 3 pts each) Execute Uniform Cost Search using Tree Search (i.e., do not remember visited nodes). S is the Start node, and G is the only Goal node. Step costs are given next to each arc. The successors of each node are indicated by arrows.



See Section 3.4.2 and Figs. 3.14-15.

At each step, indicate (a) the current queue (order is important!), (b) the node expanded (= the node first on the queue), and (c) its children. Label each node as [X, g(X)] where X is the node name and g(X) is the path cost so far to X. Name the first goal node G1, the second G2, and the third G3.

The first two are done for you, as an example. (This problem is lecture slide "Exercise for at home.")

1. Queue = [S, 0]

Expanded Node = [S, 0] Children = [A, 3], [B, 2], [C, 1]

2. Queue = [C, 1], [B, 2], [A, 3] (order is important!)

Expanded Node = [C, 1] Children = [G1, 21]

3. Queue = [B, 2], [A, 3], [G1, 21]

Expanded Node = [B, 2] Children = [E, 6]

4. Queue = [A, 3], [E, 6], [G1, 21]

Expanded Node = [A, 3] Children = [D, 9]

5. Queue = [E, 6], [D, 9], [G1, 21]

Expanded Node = [E, 6] Children = [G2, 14]

6. Queue = [D, 9], [G2, 14], [G1, 21]

Expanded Node = [D, 9] Children = [F, 10]

7. Queue = [F, 10], [G2, 14], [G1, 21]

Expanded Node = [F, 10] Children = [G3, 11]

8. Queue = [G3, 11], [G2, 14], [G1, 21]

Expanded Node = [G3, 11] Children = none, success

Some students lost points because they removed an old expensive goal from the queue when a new cheaper goal was found. No! Instead, just leave that old expensive goal undisturbed on the queue. It will sort behind the new cheap goal, which will be found first. Here, note that all three goals remained on the queue, and that the cheapest goal sorted to the front and was found first.

If you made this mistake, then you lost points on the step at which you made the error — but if your work thereafter was correct (given that earlier error), then you received full credit for all correct steps in that subsequent work.

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

2. (32 pts total, 8 pts each) English and FOL: Fun in the kinship domain. For each English sentence, write the letter of the best or closest FOL sentence (wff, or well-formed formula). **ParentOf(x, y)** means x is a parent of y. **MarriedTo(x, y)** means x is married to y. **Female(x)** means x is female. *Assume that all objects are persons, i.e., there is no need for Person(x) guard predicates.* Once a predicate has been defined in a problem, it may be used freely in subsequent problems. English definitions are "Your ... is/has ... of/ x,y) ⇔...); y = you. To help you, the intended variable bindings you, as an example.

Problem 2.a originally omitted the condition "(x ≠ y)" which means you are not a sibling of yourself(!).

That omission has been repaired in this corrected answer key.

See Section 8.3.2.

2.example B **"Your child (x) is some**
A. ∀x ∀y ChildOf(x, y) ⇔ ParentOf(x, y)
B. ∀x ∀y ChildOf(x, y) ⇔ ParentOf(y, x)

2.a (8 pts) C **"Your sibling (x) is someone not you, with a common parent (z) of you (y)."**
A. ∀x ∀y SiblingOf(x, y) ⇔ ((x ≠ y) ∧ ∀ z ParentOf(z, x) ∧ ParentOf(z, y))
B. ∀x ∀y SiblingOf(x, y) ⇔ ((x ≠ y) ∧ ∀ z ParentOf(z, x) ⇒ ParentOf(z, y))
C. ∀x ∀y SiblingOf(x, y) ⇔ ((x ≠ y) ∧ ∃ z ParentOf(z, x) ∧ ParentOf(z, y))
D. ∀x ∀y SiblingOf(x, y) ⇔ ((x ≠ y) ∧ ∃ z ParentOf(z, x) ⇒ ParentOf(z, y))

2.b (8 pts) D **"Your Steppare**
A. ∀x ∀y StepparentOf(x, y) ⇔ (∀
B. ∀x ∀y StepparentOf(x, y) ⇔ (∃
C. ∀x ∀y StepparentOf(x, y) ⇔ (∀
D. ∀x ∀y StepparentOf(x, y) ⇔ (∃

For problem 2.a, a student question arose as to why 2.a(B) was not correct? The answer is that it is too strong of a condition. "Sibling" is true if you share only one parent, i.e., your half-sister is your sibling. However, 2.a(B) requires siblings to share both parents:

∀ z ParentOf(z, x) ⇒ ParentOf(z, y))

This universally quantified statement says that if z is a parent of x (your sibling) then z is a parent of y (you). However, this requires **BOTH** parents of your sibling to be parents of you; but the definition of sibling requires **ONLY ONE** common parent.

2.c (8 pts) B **"Your first co**
A. ∀x ∀y FirstcousinOf(x, y) ⇔ (∀
B. ∀x ∀y FirstcousinOf(x, y) ⇔ (∃
C. ∀x ∀y FirstcousinOf(x, y) ⇔ (∃
D. ∀x ∀y FirstcousinOf(x, y) ⇔ (∀

2.d (8 pts) A **"Your grandchild (x) has a parent (z) of whom you (y) are a parent."**
A. ∀x ∀y GrandchildOf(x, y) ⇔ (∃ z ParentOf(z, x) ∧ ParentOf(y, z))
B. ∀x ∀y GrandchildOf(x, y) ⇔ (∃ z ParentOf(z, x) ⇒ ParentOf(y, z))
C. ∀x ∀y GrandchildOf(x, y) ⇔ (∀ z ParentOf(z, x) ∧ ParentOf(y, z))
D. ∀x ∀y GrandchildOf(x, y) ⇔ (∀ z ParentOf(z, x) ⇒ ParentOf(y, z))

3. (20 pts total, 4 pts each) Logic-To-English. For each of the following FOPC sentences on the left, write the letter corresponding to the best English sentence on the right. Use these intended interpretations: (1) "Person(x)" is intended to mean "x is a person." (2) "Flavor(x)" is intended to mean "x is a flavor." (3) "Likes(x, y)" is intended to mean "x likes y." The first one is done for you.

D	∀p ∃f Person(p) ⇒ [Flavor(f) ∧ Likes(p, f)]	A	Every person likes every flavor.
F	∃f ∀p Flavor(f) ∧ [Person(p) ⇒ Likes(p, f)]	B	For every flavor, there is some person who likes that flavor.
B	∀f ∃p Flavor(f) ⇒ [Person(p) ∧ Likes(p, f)]	C	There is some person who likes some flavor.
E	∃p ∀f Person(p) ∧ [Flavor(f) ⇒ Likes(p, f)]	D	For every person, there is some flavor that the person likes.
A	∀p ∀f [Person(p) ∧ Flavor(f)] ⇒ Likes(p, f)	E	There is some person who likes every flavor.
C	∃p ∃f Person(p) ∧ Flavor(f) ∧ Likes(p, f)	F	There is some flavor that every person likes.

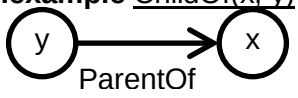
See Section 8.2.6

Note that ⇒ is the natural connective to use with ∀.

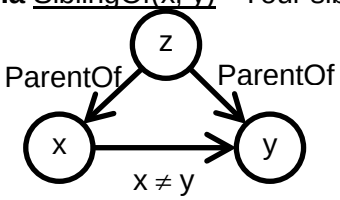
Note that ∧ is the natural connective to use with ∃.

For the benefit of non-native English speakers, the diagrams below illustrate the intended relationships described by the English statements above. The arc tail is the predicate first argument, the arc head is the second argument, and additional predicates are given as text. For example, the diagrem in 2.example expresses ParentOf(y, x), i.e., "y is a parent of x."

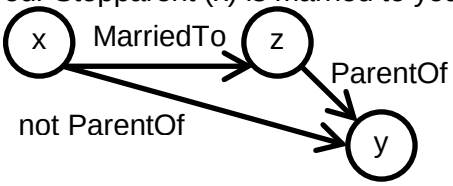
2.example ChildOf(x, y) "Your child (x) is someone of whom you (y) are a parent."



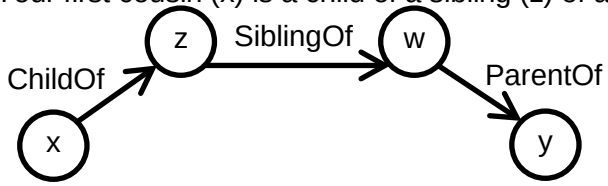
2.a SiblingOf(x, y) "Your sibling (x) is someone not you, with a common parent (z) of you (y)."



2.b StepparentOf(x, y)
"Your Stepparent (x) is married to your parent (z) and is not a parent of you (y)."



2.c FirstcousinOf(x, y)
"Your first cousin (x) is a child of a sibling (z) of a parent (w) of you (y)."



2.d GrandchildOf(x, y) "Your grandchild (x) has a parent (z) of whom you (y) are a parent."