93

1. The "if and only if" logical operator, "↔", is used to indicate that two predicates are consequences of each other. That is, **P ↔ Q** is the same as **(P → Q) → (Q → P)**. Using a truth table, show that **P ↔ Q** is logically equivalent to **(P ∨ Q) → (P ∧ Q)**.

P	Q	$P \vee Q$	$P \wedge Q$	$(P \vee Q) \to (P \wedge Q)$
T	T	T	T	T
T	F	T	F	F
F	T	T	F	F
F	F	F	F	T

$\parallel$

$P \longleftrightarrow Q$

$\parallel$

P	Q	$P \to Q$	$Q \to P$	$(P \to Q) \to (Q \to P)$
T	T	T	T	T
T	F	F	F	F
F	T	F	F	F
F	F	T	T	T

2. Trace the entire execution of the following call of the **unify** algorithm, showing each recursive call and the resulting substitutions.

unify((parents X (father X) (mother bill)),
(patents bill (father bill) Y))

/4

① parents X (father X) (mother bill)
parents bill (father bill) Y

Substitutions

{X/bill} X is substituted for X

② parents bill (father X) mother bill
Parents bill (father bill) Y

③ using {X/Bill} substitution
parents bill (father bill) mother bill
parents bill (father bill) Y

{Y/bill}

?

④ parents bill (father bill) mother bill
parents bill (father bill) mother bill

Substitutions made total: {X/bill, ~~Y/bill~~}

1) unify parents / parents {}

2) unify (X (father X) (mother bill))
(bill (father bill) Y))

3) unify X / bill {X/bill}

4) unify ((father bill) (mother bill))
((father bill) Y)

5) unify (father bill)
(father bill)

6) unify father / father {}

7) unify bill
(bill)

8) unify bill / bill {}

9) unify {}

10 unify (mother bill)
Y

11 {mother bill / Y}

12 {bill/X, (mother bill / Y}

3. Briefly, and concisely, explain the difference between a data-driven *(forward chaining)* search and a goal-driven *(backward chaining)* search. Indicate when each would be an appropriate strategy to employ.

- Choose a goal driven if the goal state is a computable known state and choose a data driven search for if the goal state is an unknown.

- The backward chaining search tries to work its way to the goal state since that is a known and forward chaining keeps progressing forward until it thinks it has found itself a goal state.

- Example of when to use one:

  8-Puzzle - Goal Driven because you know the configuration of the goal state so you work to it.

  8-Queens - Data Driven because you don't know what the goal state looks like.

4. An evaluation function is defined as **f(n) = g(n) + h(n)**. Explain what the functions **g** and **h** compute. When is an evaluation function considered to be admissible?

/3

Best first + h

• g(n) - computes the number of moves you have made on the way to a solution

Can be simple, better, or best → • h(n) - computes the distance needed to travel to a goal state from the current state.

• f(n) - is the sum of these two.

• f(n) is admissible when a Best heuristic function + best first search is used. like in the a Star algorithm

5. Briefly, and concisely, explain **minmax** and **alpha-beta pruning**.

minmax is used for 2-player games where one player will be minimizing the chances for the other player to win while the second player will maximize their own chances of winning.

You divide up the tree into levels by alternating between min and max labels respectively. After labeling the levels, start with the bottom leafs. If min will win at that state then tag a Ø to the state, else place a tag of 1. As you move up the trees you see what label you are at, if its a min level then take the minimum value of the children's tags for each node on that level. If its a max level then take the maximum value of the children's tags for each node on that max level. Then work your way up to the top of the tree.

Alpha-beta pruning will go through and if it leads no where it will cut that part out of the Solution. (Hint the pruning)

6. Explain the difference between **prior,** or **unconditional,** probability and **posterior**, or **conditional** probability.

(prior)
Unconditional probability has the values in it independent of each other where you know the odds of winning before hand like a casino table game

(posterior)
Conditional probability you have Best guess at the probability that you will win because you

- simply don't know what the variables will do depending on the conditions. Like traffic slowing down for a reck or construction or both or none.