CPSC 121 Some Sample Questions for the Final Exam

[0] 1. **Tautologies and Contradictions:**

Determine whether the following statements are tautologies (definitely true), contradictions (definitely false), or contingencies (true or false depending on the values/definitions of the variables, domains, and predicates in the expressions). Indicate your answer by circling *one* of TAUTOLOGY, CONTRADICTION, and CONTINGENCY for each. (Use any technique you wish, truth tables, equivalences, proofs, etc.)

Assume all domains have at least one element (are non-empty).

[0] a. $(q \wedge r) \rightarrow F$

TAUTOLOGY CONTRADICTION CONTINGENCY

[0] b. $(a \vee b) \wedge ((a \wedge b) \rightarrow \sim c)$

TAUTOLOGY CONTRADICTION CONTINGENCY

[0] c. $\sim((q \wedge r) \rightarrow (\sim p \vee q))$

TAUTOLOGY CONTRADICTION CONTINGENCY

[0] d. $(p \vee q) \vee \sim(p \wedge q)$

TAUTOLOGY CONTRADICTION CONTINGENCY

[0] e. $\forall x \in \mathbf{S}, P(x) \rightarrow \exists y \in \mathbf{S}, P(y) \vee x \neq y$

TAUTOLOGY CONTRADICTION CONTINGENCY

[0] f. $\exists y \in \mathbf{S}, Q(y) \vee P(y)$

TAUTOLOGY CONTRADICTION CONTINGENCY

[0] g. $(\exists x \in \mathbf{S}, \exists y \in \mathbf{S}, x \neq y) \wedge (\exists y \in \mathbf{S}, \forall x \in \mathbf{S}, x = y)$

TAUTOLOGY CONTRADICTION CONTINGENCY

[0] h. $\forall x \in \mathbf{T}, \exists y \in \mathbf{S}, x = y$

TAUTOLOGY CONTRADICTION CONTINGENCY

[0] 2. **Quantifier Order and the Challenge Method:**

Consider the following theorem:

$$\exists x \in \mathbb{Z}, \forall y \in \mathbb{N}, ((\forall z \in \mathbb{Z}, P(x,y,z)) \wedge (\exists w \in \mathbb{N}, Q(x,y,w)))$$
$$\rightarrow \exists a \in \mathbf{A}, \forall b \in \mathbf{A}, R(x,y,a,b)$$

Imagine using a direct proof approach to prove this theorem (i.e., no contradiction, contrapositive, or other use of logical equivalences to alter the form of the theorem). For each quantifier, indicate whether you will **prove** the quantifier or **assume** it. If you are **proving** it, indicate whether you get to choose the variable's value and, if so, on the basis of which other variables. If you are **assuming** it, give an example of how you might make use of the assumption.

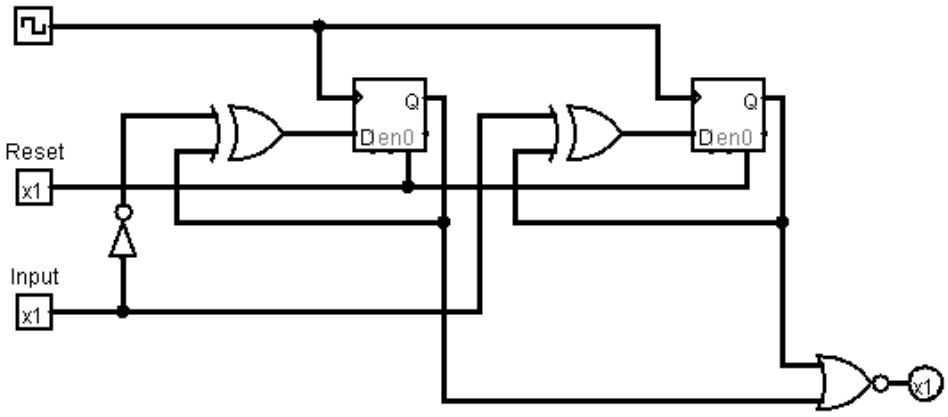
[0] 3. **Direct Proof Approaches:**

For the following theorem, give a "direct" proof approach—no proof by contradiction or contrapositive, no use of logical equivalences—that is as complete as possible. (If there are choices that the person using your approach will need to make, be sure to give as much detail as possible about how they can make those choices.)

$$\forall x_1 \in \mathbf{A},\, \exists y_1 \in \mathbf{B}, P(x_1, y_1, 0) \wedge$$
$$\forall x_2 \in \mathbf{A},\, \exists y_2 \in \mathbf{B},\, \forall z_1 \in \mathbf{A}, P(x_2, y_2, z_1) \rightarrow \exists z_2 \in \mathbf{A}, Q(z_2, z_1) \wedge P(x_2, y_2, z_2)$$

[0] 4. **Lab Circuits (This is much harder than circuits you expect to see in the exam) :**

Consider the following circuit:



Assume that all of the flip-flops start with the value 0. Assume that "Input" is a sequence of bits provided by the user and that the clock ticks each time a new bit is available. Ignore the Reset button. This resets the flip-flops to start with 0.

[0] a. Give three different input sequences of different lengths that all lead to outputs of 1.

[0] b. Give an input and a state for the circuit (i.e., a bit stored by the left flip-flop, a bit stored by the right flip-flop) such that when the clock ticks **both** flip-flops change their stored values **or** explain why no such input and state exists.

[0] c. **Briefly** explain in English the meaning of the left flip-flop's value.

[0] d. **Briefly** explain in English the meaning of the right flip-flop's value.

[0] e. **Briefly** explain in English what the circuit does.

[0] 5. **Predicate Logic:**

Much of Facebook's data is described as a "graph" (not the kind with x- and y-axes), which is made up of nodes and edges that connect nodes together. The edges express relationships between the nodes.

We'll reason about a similar graph using the following definitions:

- N is the set of all nodes
- $T = \{$"friend", "located in", "likes", "owns"$\}$ is the set of relationship types
- Edge(a, b, t) means that node a has the relationship t with node b. (Note that a may have the "likes" relationship with b while b does not have that relationship with a; so, order matters.)
- Person(a) means node a represents a person
- Business(a) means node a represents a business
- Location(a) means node a represents a location

For convenience, you may use E for Edge, P for Person, B for Business, and L for Location.

Defining Predicates:

Define the following predicates in predicate logic in terms of the predicates and sets given above.

[0] a. Locale(l) means l is a location and is not itself located in any other location.

Locale(l) ≡

[0] b. Neighbors(a, b) means nodes a and b are at the same locale. (You may assume every node is in at most one locale, but you should **not** assume that every node is in *exactly one* locale.)

Neighbors(a, b) ≡

Making Statements:

State the following facts in predicate logic. **YOU MAY** (but do not have to) **USE THE EXTRA PREDICATES YOU WERE ASKED TO DEFINE ABOVE.** (Assume they are correct.)

[0] a. Friendship is "symmetric"; that is, two people either have the friend relationship with each other or neither has the friend relationship with the other.

[0] b. Every node is a person, business, or location, but not more than one of these.

[0] c. A business owner's friends all like the business.

[0] 6. **Indirect Proof:**

Consider the theorem: Every DFA either accepts an infinite number of different inputs or rejects an infinite number of different inputs (or both).

Prove the theorem using proof by *contradiction* (and no other steps preceding the contradiction step).

[0] 7. **Designing DFAs:**

[0] a. Design a Deterministic Finite-State Automaton (DFA) that takes as input a sequence of bits—`0` or `1`—and accepts exactly those sequences that include at least three bits of which at least two are `0`s. (So, for example, `000`, `001`, `010`, `100`, `11010`, and `1111011001011` should all be accepted. `00`, `110`, `1110111`, `1`, `0`, and the empty string should all be rejected.)

[0] b. Now, write a regular expression that matches the same set of strings.

[0] 8. **Direct Proof: Prove**

$$\forall a \in \mathsf{Z}^+, \forall b \in \mathsf{Z}^+, \exists n \in \mathsf{N}, n > b \wedge n^2 > an + 1.$$

[0] 9. **Working Computer:**

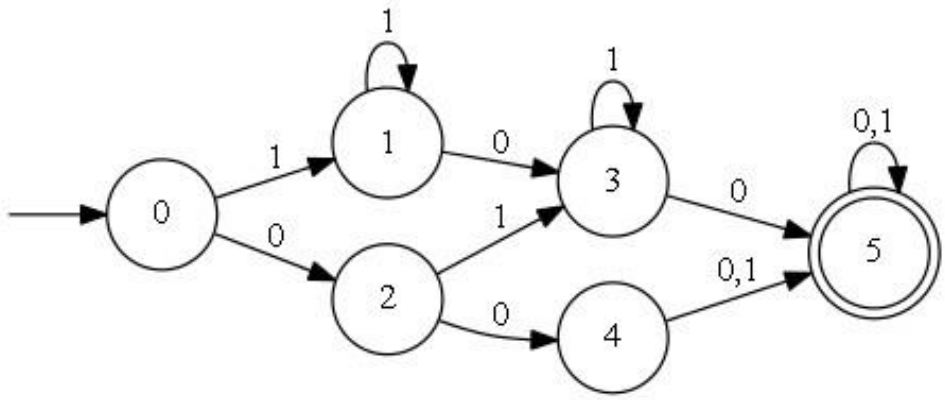
Our working computer's instructions can come in various lengths.

[0] a. Why are some instructions longer than others? Briefly explain, ideally with an example of what might cause an instruction to require more space.

[0] b. What part of an instruction indicates how much memory the instruction takes up?

[0] 10. **DFA Implementation:**

Consider the following DFA:



[0] a. What state will this be in after processing the string 11011?

[0] b. Would this DFA still be legal if we changed state 0 to be an accepting state?

[0] c. Would this DFA still be legal if we eliminated the arc from state 5 to itself?

[0] d. Implement this DFA as a circuit. [Try some parts of it. At least find out how many flip-flops we need for the states and implement one of the multiplexors for the next state. You don't need to complete the whole circuit.

[0] 11. **Induction:**

For any positive integer n, the n^{th} Fibonacci number F_n is defined to be:

$$F_n = \begin{cases} 1 & if\ n = 1 \\ 1 & if\ n = 2 \\ F_{n-1} + F_{n-2} & if\ n \geq 3 \end{cases}$$

[0] a. Prove that the sum of two odd numbers is an even number. (For this and the next two problems: probably not by induction!)

[0] b. Prove that the sum of two even numbers is an even number.

[0] c. Prove that the sum of an odd number and an even number is an odd number.

[0] d. (**Good practice question, but much harder than exam questions)** Prove **by induction** that every third Fibonacci number (and no others) is even. That is: F_n is even when n is divisible by 3 and odd otherwise. (*Hint*: draw your recursive structure from the definition of the Fibonacci numbers, divide into three cases, and use the facts you just proved above.)