
*On the cover page of your assignment, you must list everyone with whom you discussed this assignment, and which problems you discussed with each person. You must also write **and sign** the following statement: “I have read and understood the policy on collaboration on homework assignments stated in the Course Information handout.” Without these, your homework will not be marked.*

1. Formulate each of the following English queries as a formula in the logical language presented in class for the database about students, instructors, courses, and grades. (For your information, the expected answer to each query is included.) [10]
 - (a) Are Ann and May taking a course together? (Yes, CS238.)
 - (b) Are all courses without prerequisites taught by Pat? (No, M100, for one.)
 - (c) Is CS230 the only course that Bill failed? (Yes.)
 - (d) Does someone other than Sue teach a course and all of its prerequisites? (No.)
 - (e) Does every course with a prerequisite have a student who is only taking courses with prerequisites? (Yes. For CS230, Jill; for CS238, May; for M200, Jill.)

2. Formulate each of the following logical formulas as an English query, and state the yes-no answer for the database presented in class about students, instructors, courses, and grades. If it helps, you may pose your query in the form “Is it true that ...?” [15]
 - (a) $\neg\exists x(Prereq(cs230, x) \wedge Prereq(cs148, x))$
 - (b) $\forall x\forall y(Grade(tom, x, y) \supset y = b)$
 - (c) $\exists x\forall z(\exists wPrereq(w, z) \supset Instructor(x, z))$
 - (d) $\forall x(\exists y_1[Grade(jack, x, y_1) \wedge PassingGrade(y_1)] \equiv \exists y_2[Grade(bill, x, y_2) \wedge PassingGrade(y_2)])$
 - (e) $\exists x\forall y\{Enrolled(x, y) \equiv \exists z[(y = z \vee Prereq(y, z)) \wedge \exists u\exists v(Grade(u, z, v) \wedge \neg PassingGrade(v))]\}$

3. Consider a logical language with one 3-place predicate symbol, R , and two sentences in that language, $\forall x\exists y\forall zR(x, y, z)$ and $\exists y\forall x\exists zR(x, y, z)$. Give a structure whose domain is $\{0, 1\}$ where the first sentence is true and the second one is false, and another structure with the same domain where the second sentence is true and the first one is false. [10]

4. Consider a logical language with no constant symbols and two predicate symbols, L and P , both binary. Let \mathcal{N} be a structure for that language whose domain is \mathbb{N} , and where $L^{\mathcal{N}} = \{(n, m) : n < m\}$.

(a) Prove that the sentence $\exists x \exists y \exists z \forall u (L(u, x) \equiv u = y \vee u = z)$ is true in \mathcal{N} or prove that it is false in \mathcal{N} . No marks will be given for choosing badly! [10]

(b) Give a value for $P^{\mathcal{N}}$ that makes the following sentence true in \mathcal{N} : [5]

$$\exists x P(x) \wedge \exists y \neg P(y) \wedge \forall z [P(z) \supset \exists w (L(z, w) \wedge P(w))].$$

(c) Prove that the following sentence is true in \mathcal{N} (no matter the value of $P^{\mathcal{N}}$): [15]

$$\{\forall y (P(y) \supset \exists x (L(x, y) \wedge P(x)))\} \supset \neg \exists z P(z)$$

Hint: Use mathematical induction.

5. In this final question, we will consider a logical specification for *sorting*. Informally, we start with an array A , and we wish to produce an array B whose elements are those of A rearranged to be in (say) ascending order. To formalize parts of this, we will use a first-order logical language with equality \mathcal{L}_S having no constant symbols and three predicate symbols, L , A , and B , all binary. Intuitively, $L(x, y)$ will be used to say that x is less than y , $A(x, y)$ will be used to say the element of A at index x is y , and similarly for $B(x, y)$.

The first thing we need to do is ensure that A and B are indeed arrays, that they have the same dimension, and that B is ordered.

(a) (Single-valued) Write a sentence of \mathcal{L}_S saying that there is at most one element of A at each index, and similarly for B . [5]

(b) (Same dimension) Write a sentence of \mathcal{L}_S saying that there is a number (the dimension) such that A has an element at some index iff the index is less than that number, and similarly for B . (So taking the natural numbers as the domain, we are imagining arrays indexed from 0 to $n - 1$ for some n .) [5]

(c) (Ordered) Write a sentence of \mathcal{L}_S saying that the elements of B are in ascending order. [5]

Next, comes the tricky part. We would like to say that the elements of B are a rearrangement of those of A . We will have trouble doing in this in general.

(d) (Permutation version 1) Write a sentence of \mathcal{L}_S without using L or $=$ saying that the elements of B include all the elements of A . [5]

(e) If the elements of A are all different, then we're done. However, present a structure for \mathcal{L}_S whose domain is the natural numbers, where L is interpreted as $<$ (as in Question 4 above), and where the four sentences (a), (b), (c), and (d), are true, but B is not a sorted version of A . [5]

(f) (Permutation version 2) Consider the following sentence: [10]

$$\forall x_1 \forall x_2 \forall r_1 \forall r_2 [x_1 \neq x_2 \wedge A(x_1, r_1) \wedge A(x_2, r_2) \supset \exists y_1 \exists y_2 (y_1 \neq y_2 \wedge B(y_1, r_1) \wedge B(y_2, r_2))].$$

State a general condition on the input array A such that the first three sentences (a), (b), and (c) together with the one above are sufficient to guarantee that B is a sorted version of A . Write this condition as a sentence of \mathcal{L}_S .

This last question on sorting is a bonus question. *Do not attempt any bonus work until the regular part of your assignment is complete.*

The easiest way to deal with the problem of repetitions in A is to take a different tack. Instead of saying that the output B is permutation of the *elements* of A , we can insist that B be a permutation of the *indices* of A . In other words, B is an array of numbers from 0 to $n - 1$, where intuitively, $B[i]$ tells us where element i of A will end up in the ordering after sorting.

(g) Write a sentence(s) of \mathcal{L}_S saying that B defines an appropriate mapping over the indices of A that puts the elements into ascending order.