Predicate Logic

The propositional-logic lecture introduced several Boolean operations, including AND, OR, and NOT. Each is an example of a **Boolean function**, i.e. a function whose domain and codomain sets are all Boolean. On the other hand, if we allow for a function of one or more variables to have domain sets of any type, yet still require a Boolean codomain, then we have what is called a **predicate function**. Predicate functions are often used to determine whether or not a member of some set has a given property. For example, let n be a positive integer, and consider the predicate function $p(n)$ that assigns n the value `true` iff n is a prime number. Then for any positive integer n we may evaluate $p(n)$ to determine if n has the property of being prime.

Example 1. Consider the set membership operation $\in$. We may view this as a binary predicate function whose first input is a member u of some universe $\mathcal{U}$, whose second input is a subset A of $\mathcal{U}$, and whose output is `true` iff $u \in A$. For example, if $\mathcal{U} = \mathcal{N}$, then $3 \in \emptyset$ evaluates to `false`, while $5 \in \{2, 3, 5, 7\}$ evaluates to `true`. In general, $\in$ has the signature $\in (\mathcal{U}, \mathcal{P}(\mathcal{U}), \mathcal{B})$. □

Given a predicate function $f : A \to \mathcal{B}$, in order to provide a rule for f, we must have access to operations that take members from A and assign to them a Boolean value. Although most of these operations will depend on the type of domain that A represents, there are two binary operations that are universal to all sets, namely $=$ and $\neq$. Indeed, if x and y are variables having domain A, then $x = y$ evaluates to `true` iff x and y have been assigned the same domain member of A. On the other hand $x \neq y$ evaluates to `true` iff x and y have been assigned different domain members. Thus, evaluating the operations $=$ and $\neq$ amounts to being able to distinguish members of A.

Example 2. Let A denote the power set of $\{1, 2, 3\}$. Then $=$ and $\neq$ are two predicate functions defined on A, For example, $\{1, 2\} = \{1, 3\}$ evaluates to `false`, while $\{1, 2\} \neq \{1, 3\}$ evaluates to `true`. □

Example 3. Define an equality function for the set of all manufactured automobiles.

Example Solution.

Relational operations

The above two equality operations are examples of what is called a **relational operation**. As the name suggests, a relational operation is a binary operation that tests whether or not two domain members are related in some way. For example, `equals` tests if two members are identical. Although most other relational operations apply only to a particular domain, the relational operations $<$, $\leq$, $>$, and $\geq$ apply to any domain whose members are ordered in some way. Of course, the most common ordered domains are the natural, integer, and real numbers. Indeed, statements like $3 < 5$ and $-2.754 \geq -4.13$ are quite common in these domains. For this reason each of these operations is called an **ordering relation**. Generally speaking, any finite or countably infinite set can be ordered so as to allow the use of ordering relations.

Example 4. Let $A = \{a, b, c, d, \ldots, z\}$ denote the set of alphabet letters. This set has a familiar ordering taught in elementary school, and that statements like $t < v$ and $s \geq s$ true, and statements like $c > z$ and $m \leq b$ false. □

Example 5. The binary set operation $\subseteq$ is an example of a relational operation over any collection of sets $\mathcal{S}$ since, for any $A, B \in \mathcal{S}$, $A \subseteq B$ evaluates to either `true` or `false`. In this case $\subseteq$ has the signature $\subseteq (\mathcal{S}, 2, \mathcal{B})$. □

Boolean operations on predicate functions

Recall from the functions lecture that if $f, g : A \to B$ are functions, and B is closed under some binary operation, then we may apply this binary operation to functions f and g. In particular, if f and g are predicate functions, then the binary operations AND, OR, NOT, etc. may be applied to f and g.

Definition 1. Let $f, g : A \to \mathcal{B}$ be predicate functions, Then the following functions are well-defined for all $a \in A$.

Not $(\neg f)(a) = \neg f(a)$

And $(f \wedge g)(a) = f(a) \wedge g(a)$

Or $(f \vee g)(a) = f(a) \vee g(a)$

Exor $(f \oplus g)(a) = f(a) \oplus g(a)$

IfThen $(f \to g)(a) = f(a) \to g(a)$

Equivalence $(f \leftrightarrow g)(a) = f(a) \leftrightarrow g(a)$

Example 6. Let $m(s, d)$ be a predicate function that evaluates to `true` iff student s in some discipline d. Also, let $p(s, l)$ be a predicate function that evaluates to `true` iff student s has programmed in computer language l. Write a predicate logic formula that models the statement "student s has programmed in C++ provided s is majoring in either computer science or computer engineering". Do the same for the statement "Tom is majoring in discipline d and has not programmed in Python".

Example Solution.

Example 7. Suppose we have a definition for the relational operations $x < y$ and $x = y$. Then the remaining operations $x \neq y$, $x \leq y$, $x > y$, and $x \geq y$ may all be defined by using $x < y$, $x = y$, and the Boolean operations on predicate functions. Indeed, for all x and y, we have

$$(x \neq y) = \neg(x = y),$$

$$(x \leq y) = ((x < y) \vee (x = y)),$$

$$(x > y) = \neg(x \leq y) = (\neg((x < y) \vee (x = y))),$$

and

$$(x \geq y) = ((x > y) \vee (x = y)) = ((\neg((x < y) \vee (x = y))) \vee (x = y)) =$$

$$\neg(x < y) \vee (x = y).$$

□

Quantifiers

The predicate functions together with the Boolean operations on predicate functions constitute two of the three fundamental building blocks for constructing a statement in **predicate logic**. In this section we study the third and final ingredient: quantifiers. To motivate the need for quantifiers, recall that logical statements provide a formal means for representing truth about a given system or theory. Moreover, such a statement should i) be able to be evaluated as either being `true` or `false`, and ii) be as succinct as possible so as not to obfuscate its meaning. Now consider the predicate function $f : P \to \mathcal{B}$, where P is a set of people, and $f(p)$ asserts that person p had fun at the amusement park. Furthermore, suppose we want to use f to make the statement "everyone had fun at the amusement park". As a first attempt, suppose we model this statement as "$f(p)$", where $\text{dom}(p) = P$. The problem with this is that this statement can only evaluate to `true` or `false` when p has been assigned a specific person. For example, if John had fun at the amusement park, then $f(\text{John}) =$ `true`. Thus requirement i) is not met. As a second attempt, we could make a conjunctive statement that uses every member in P. For example, suppose $P = \{p_1, \ldots, p_n\}$ consists of n people, then to express that everyone had fun at the amusement park, we could write

$$f(p_1) \wedge f(p_2) \wedge \cdots \wedge f(p_n).$$

This statement is well-constructed, and evaluates to `true` iff everyone had fun, and so has the correct meaning. However, this statement may not be succinct for large values of n. Furthermore, we may want to make the statement about everyone having fun, but perhaps prefer to keep the attendee's names private. The **universial quantifier** $\forall$ solves both these issues. Indeed, the statement

$$\forall p f(p)$$

reads as "for all members p of P, $f(p)$ is true".

The universal quantifier $\forall$ is an example of what is called an **iterator function** because it works in iterative steps, where each iteration involves assigning a variable v, called the **index variable**, one of its domain values, followed by evaluating an expression that depends on v, and combining the answer with the answers obtained in previous iterations. An iterator only stops until the desired final answer has been computed, which may take at most n iterations, where $n = |\text{dom}(v)|$. For example, in the statement $\forall p f(p)$, p serves as the index variable, while $f(p)$ is the expression that gets repeatedly evaluated, once for each member of p's domain. In this case the $\forall$ iterator is looking for a person p in domain set P for which $f(p) = 0$. If such a person is found, then the final answer is `false`, since it is not true that everyone had fun at the amusement park. On the other hand, after iterating through all the members of P and discovering that $f(p) = 1$ for each member, then the final answer is `true`, since everyone had fun.

Example 8. Let variable n have domain $\{1, \ldots, 10\}$, and let prime(n) denote the predicate function that evaluates to `true` iff n is prime. Then the statement $\forall n$prime(n) evaluates to `false` since are members (e.g. 1,4, 6, 8, 9,10) of n's domain that are not prime numbers. Also, $\forall n \neg$prime(n) also evaluates to `false` since it is saying that every member is not prime, which is also not true, since 2,3,5, and 7 are prime. □

Example 9. Let $m(s, d)$ be a predicate function that evaluates to `true` iff student s in some discipline d. Also, let $p(s, l)$ be a predicate function that evaluates to `true` iff student s has programmed in computer language l. Write a predicate logic formula that models the statement "Any student who has majored in computer science has programmed in either C++ or Python". Also, provide a layperson translation of the predicate-logic formula

$$\neg\forall l p(\text{Ton}, l) \wedge m(\text{Ton}, \text{CE}).$$

Returning to the quantifier iterators used in predicate logic, the **existential quantifier**, $\exists$, works in the same manner as the universal quantifer, but now the statement $\exists p f(p)$ reads "there exists a p for which $f(p)$ is true". For example, letting f denote the "had fun at the amusement park" predicate, in layperson terms $\exists p f(p)$ states that at least one person (may be more) had fun at the amusement park. Moreover, if the domain set of p is $P = \{p_1, \ldots, p_n\}$, then $\exists p f(p)$ has the same logical meaning as

$$f(p_1) \vee f(p_2) \vee \cdots \vee f(p_n).$$

Thus, $\exists$ represents the dual of $\forall$, meaning that, for any tautology there is another tautology that can be formed by replacing $\forall$ with $\exists$ (and vice versa), AND with OR, and 1 with 0.

Example 10. Write a predicate logic formula that models the statement, there is a rational number that lies between 3.14 and 3.141. Also, provide a layperson translation of the predicate-logic formula

$$\exists y(y^2 = 2),$$

where the domain of y is all rational numbers.

Example 11. Consider the two predicate-logic statments $\forall x(p(x) \wedge q(x))$ and $\forall x p(x) \wedge \forall x q(x)$. These statements are logically equivalent, since their evaluations will always result in the same truth value. To see this, suppose the domain of x is a group of people, $p(x)$ stands for "x likes pizza" and $q(x)$ stands for "x likes quiche". Then the first statement is saying "everyone likes both pizza and quiche", while the second is saying "everyone likes pizza and everyone likes quiche". Both are two ways of saying that everyone in the domain likes both these kinds of foods. Therefore we have the identity

$$\forall x(p(x) \wedge q(x)) = \forall x p(x) \wedge \forall x q(x),$$

whose corresponding dual identity is

$$\exists x(p(x) \vee q(x)) = \exists x p(x) \vee \exists x q(x).$$

□

Example 12. We now show that the predicate-logic statments

$$\forall x(p(x) \vee q(x))$$

and $\forall x p(x) \vee \forall x q(x)$ are *not* logically equivalent. To do this we must provide an example of two predicate functions p and q, and a domain for x for which the two statements have different truth evaluations. Using the same predicate functions p and q from the previous example, suppose the domain of x is $\{\text{Alice}, \text{Bob}\}$. Moreover, suppose Alice likes pizza but not quiche, and Bob likes quiche but not pizza. Then $\forall x(p(x) \vee q(x))$ evaluates to `true` since both Alice and Bob like either pizza or quiche (Alice likes pizza, Bob likes quiche). However, $\forall x p(x) \vee \forall x q(x)$ evaluates to `false` since neither everyone likes pizza (Bob does not like it) nor everyone one like quiche (Alice does not like it). We leave it as an exercise to show that

$$\exists x(p(x) \wedge q(x))$$

and $\exists x p(x) \wedge \exists x q(x)$ are *not* logically equivalent. □

The previous two examples address the ability (and inability) for the universal and existential quantifiers to distribute over a binary operation (such as AND and OR). The reader is encouraged to continue this study using the other logic operations, such as IF-THEN, EXOR, and EQUIVALENCE.

It's important to note that the truth or falsehood of a predicate-logic statement will often depend on the choice of domains for each of it's index variables. For example, the statement $\exists x(x < 0)$ is true if $\text{dom}(x) = \mathcal{I}$, but is false if $\text{dom}(x) = \mathcal{N}$. Also, the choice of domain can often affect the structure of the statement itself.

Example 13. Let $p(x)$ be the predicate function that evaluates to `true` iff person x has programmed with the Python programming language, where the domain of x is the set of all people with computer science degrees. Let $\text{cs}(y)$ be the predicate function that evaluates to `true` iff person y has a computer science degree, where the domain of y is the set of all living people. Now consider the statement "Every person with a CS degree has programmed with Python." Using variable x, this statement may be modeled as

$$\forall x p(x). \tag{1}$$

However, using variable y, $\forall y p(y)$ implies that every living person has programmed with Python, which is too strong a statement. To weaken the statement we must place a condition on when someone has programmed with Python, namely the condition that the person has a CS degree. In other words, for any living person, if that person has a CS degree, then they have programmed with Python". This is modeled as

$$\forall y(\text{cs}(y) \rightarrow p(y)). \tag{2}$$

To see that this statement has the same truth value as 1, first assume that the former evaluates to `true`. Now consider an arbitrary living person y. If this person has no CS degree, then $\text{cs}(y)$ is false which makes the statement $\text{cs}(y) \rightarrow p(y)$ vacuously true. On the other hand, if y does have a CS degree, then, since $\text{cs}(y)$ is true, y is a member of the domain of x, in which case $p(y)$ is true. Hence, $\text{cs}(y) \rightarrow p(y)$ evaluates to $(\texttt{true} \rightarrow \texttt{true}) = \texttt{true}$. We leave it as an exercise to show that the truth of 2 implies the truth of 1. □

The quantifiers introduced thus far allow us to state that all members of some domain possess a property, or that at least one member does. But what if we want to say that at least one half of the members possess a property, or that no more than 20 possess the property? The following extensions of the existential quantifier allow us to make such statements.

$\exists! x p(x)$ There exists a unique domain member a of x for which $p(a)$ evaluates to `true`.

$\exists^{=n} x p(x)$ There exist exactly n domain members a of x for which $p(a)$ evaluates to `true`.

$\exists^{<n} x p(x)$ There exist fewer than n domain members a of x for which $p(a)$ evaluates to `true`.

$\exists^{>n} x p(x)$ There exist more than n domain members a of x for which $p(a)$ evaluates to `true`.

$\exists^{\leq n} x p(x)$ There exist no more than n domain members a of x for which $p(a)$ evaluates to `true`.

$\exists^{\geq n} x p(x)$ There exist at least n domain members a of x for which $p(a)$ evaluates to `true`.

Example 14. Suppose $f(p)$ is the predicate function that evaluates to `true` iff person p had fun at the amusement park, where the domain of p is a set of 1,000 people. Then the statement "anywhere between 80 to 90 percent of the people had fun at the amusement park" can be modeled with the statement

$$\exists^{\geq 800} p f(p) \wedge \exists^{\leq 900} p f(p).$$

□

Before leaving this section it is worth to note in passing that there are De Morgan-like identities for quantifier statements, namely

$$\neg\forall x p(x) = \exists x \neg p(x) \tag{3}$$

and

$$\neg\exists x p(x) = \forall x \neg p(x). \tag{4}$$

For example, it's not true that everyone likes pizza if and only if it is true that at least one person does not like pizza. Similarly, it's not true that someone likes pizza if and only if it is true everyone does not like pizza.

Set-builder notation

The list notation introduced in the sets lecture represents the most direct way to describe the members of a set S. However, this notation has its limitations, especially for the case when a set has very large cardinality, and there is no starting sequence of members (such as $\{1, 2, 3, \ldots\}$) that captures the entire set. On the other hand, suppose we have a predicate function $p(x)$, where i) $S \subseteq \text{dom}(x)$, and ii) $p(x) = 1$ iff $x \in S$. Then **set-builder notation** allows us to describe the set as

$$\{x | p(x)\},$$

which reads "the set of domain members a of x for which $p(a)$ evaluates to `true`". When using set-builder notation, we must specify the domain of x in case there is a chance that it may not be understood.

Example 15. Let $\text{div}(x, y) = 1$ iff integer x divides evenly into integer y. Then we may use div and set-builder notation to describe the set of integers between 1 and 1000 that are divisible by either 2 or 3. Indeed, letting x have domain equal to $\mathcal{I}$, this set is equal to

$$\{x | x \geq 1 \wedge x \leq 1000 \wedge (\text{div}(2, x) \vee \text{div}(3, x))\}.$$

Note that, had we stated the domain of x as all integers between 1 and 1000, then the above simplifies to

$$\{x | \text{div}(2, x) \vee \text{div}(3, x)\},$$

since the domain of x is already understood. □

Nested Quantifiers

In the previous section we studied the use of quantifiers for a predicate-logic statement that has a single variable. In section we extend the study to statments involving two or more variables.

To begin, consider two disjoint sets of people A and B, where the members of A and B live on Alameda Island and Balboa Island, respectively. Next, let predicate function $c(x, y)$ evaluate to `true` iff person x has ever phoned person y, where x and y have domains A and B, respectively. Since there are two variables, there are 8 different ways to quantify $c(x, y)$. For example, $\exists x \forall y c(x, y)$, $\forall y \exists x c(x, y)$, and $\exists x \exists y c(x, y)$ are three such ways. In what follows we provide a definition for how to interpret the meaning of each one, and interpret this meaning using $c(x, y)$.

$\exists x \exists y c(x, y)$ There is some member $a \in A$, and some member $b \in B$ for which $c(a, b) = 1$. Example: someone on Alameda (and belonging to set A) has phoned someone on Balboa (and belonging to set B).

$\exists y \exists x c(x, y)$ There is some member $b \in B$, and some member $a \in A$ for which $c(a, b) = 1$. Example: someone on Balboa has received a call from someone on Alameda.

$\exists x \forall y c(x, y)$ There is some member $a \in A$ such that, for every member $b \in B$, $c(a, b) = 1$. Example: someone on Alameda has phoned everyone on Balboa.

$\exists y \forall x c(x, y)$ There is some member $b \in B$ such that, for every member $a \in A$, $c(a, b) = 1$. Example: someone on Balboa has received a call from everyone on Alameda.

$\forall x \exists y c(x, y)$ For every member $a \in A$ there is a member $b \in B$ for which $c(a, b) = 1$. Example: everyone on Alameda has phoned someone on Balboa.

$\forall y \exists x c(x, y)$ For every member $b \in B$ there is a member $a \in A$ for which $c(a, b) = 1$. Example: everyone on Balboa has received a call from someone on Alameda.

$\forall x \forall y c(x, y)$ For every member $a \in A$ and every member $b \in B$, $c(a, b) = 1$. Example: everyone on Alameda has phoned everyone on Balboa.

$\forall y \forall x c(x, y)$ For every member $b \in B$ and every member $a \in A$, $c(a, b) = 1$. Example: everyone on Balboa has received a call from everyone on Alameda.

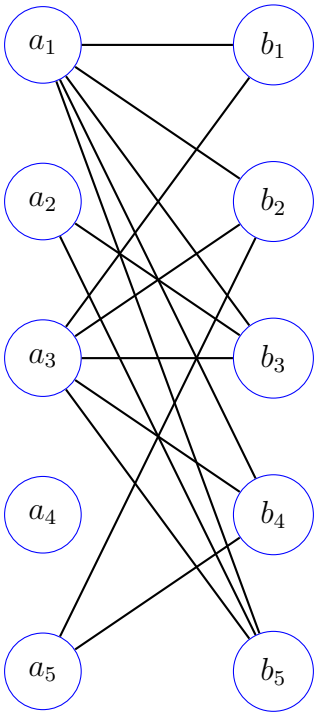

Figure 1: A binary predicate graph that defines $c(x, y)$ so that $\exists x \forall y c(x, y)$ evaluates to `true`

One way to visualize each of the above statements is to draw a **binary predicate graph** whose left and right nodes represent members of A and B, respectively. Furthermore, a line segment joins a left node labeled with $a \in A$ with a right node labeled with $b \in B$ iff $c(a, b) = 1$. Figure 1 shows a binary predicate graph for the case when $A = \{a_1, a_2, a_3, a_4, a_5\}$ and $B = \{b_1, b_2, b_3, b_4, b_5\}$. Notice that, for the statement $\exists x \forall y c(x, y)$ to be true, there needs to be at least one member of A that is joined with every member of B. Looking at the graph, we see that there are actually two such members: a_1 and a_3. Therefore, $\exists x \forall y c(x, y)$ is a true statement.

As another example, Figure 2 shows another binary predicate graph that defines a predicate $c(x, y)$ for which $\forall y \exists x c(x, y)$ evaluates to `true`. To see this, notice that every member of B is joined with a least one member of A.

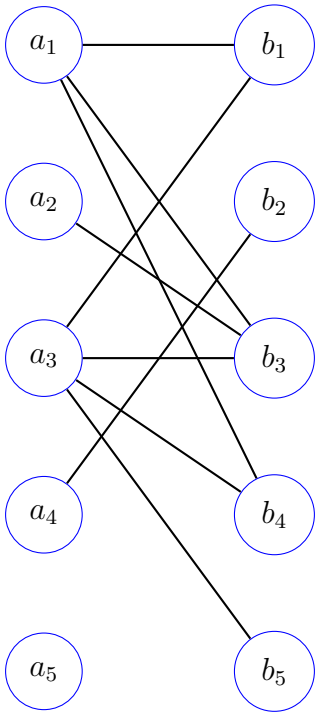

Figure 2: A binary predicate graph that defines $c(x, y)$ so that $\forall y \exists x c(x, y)$ evaluates to `true`

Example 16. Suppose predicate function depart(f, a) evaluates to `true` iff flight f departs from airport a, and carrier(f, c) evaluates to `true` iff flight f is owned by carrier c. The problem is to use these predicate functions to model the statement "every carrier owns a flight that departs from Los Angelest International Airport (LAX)". Translating a natural-language statement into predicate logic is often an iterative process, where we write an answer, analyze it for correctness, and make changes where necessary. In this case we are making a statement about *every* carrier, namely that it owns a flight that departs from LAX. So we might start off and write

$$\forall c(\text{carrier}(f, c) \wedge \text{depart}(f, \text{LAX})).$$

Here we use $\wedge$ since there are two things that are both true about flight f, namely that it's owned by c and departs LAX. This is almost what we want, but our answer treats f as a **free variable**, meaning that it is not indexed by some quantifier. Moreover, when a variable is free, it means that the statement cannot be evaluated until f has been assigned a particular flight in its domain. But the problem does not mention any particular flights that pertain to f's domain. Thus, it seems more appropriate to index f with a quantifier. So as a second attempt we may write

$$\forall c \forall f(\text{carrier}(f, c) \wedge \text{depart}(f, \text{LAX})).$$

Staring at this answer, we notice that it seems too strong, in that it is saying that, for any carrier c, and for any flight f, the carrier owns f and f departs LAX. This means that every carrier owns every flight, which seems nonsensical, since a flight is usually owned by a single carrier. Finally, if we change $\forall f$ to $\exists f$, then we arrive at

$$\forall c \exists f(\text{carrier}(f, c) \wedge \text{depart}(f, \text{LAX})),$$

which now seems consistent with the original statement. Now, for any carrier c, we are just asking for one flight f that is both owned by c and departs LAX. □

The following are some guidelines for modeling a natural-language statement with a predicate logic formula.

1. **Identify all relevant domains.** Modeling a statement with predicate logic is typically done in relation to modeling some problem of interest, which itself has a set of relevant domains. For example, if the problem is to develop a schedule of courses to offer at a university, then the relevant domains may include course subjects, rooms, instructors, students, and times. Then a statement such as "an instructor cannot teach two different courses at the same time", refers to the instructor, course, and time domains.

2. **Identify all constants.** Constants arise when a statment makes reference to a particular member of some domain. For example, the statement "Dr. Gold is only able to teach courses that start at or after 5:00 pm" makes reference to two constants: Dr. Gold and 5:00 pm. These constants will appear as inputs to one or more predicate functions, such as $\text{start}(c, \text{5:00 pm})$ and $\text{ins}(c, \text{Gold})$, where the domain of c is the set of all courses.

3. **Identify all variables.** The use of variables arises when a statement makes references to generic members of some domain. For example, the statement "an instructor cannot teach two different courses at the same time" makes generic references to instructors, courses, and times. Thus, a model of this statement should make use of variables, such as i, c, and t, whose domains correspond with the domains mentioned in the statement.

4. **Identify the predicate functions needed to support the statement.** Since the statement being made is asserting a truth about the problem being modeled, there must be one or more predicate functions that together (with the help of logical operations) help formally represent this truth. For example, the statement "an instructor cannot teach two different courees at the same time", we can support this statement using three predicate functions $\text{teach}(c, i)$, which is true iff instructor i teaches course c, $\text{time}(c, t)$ which is true iff course c is taught during time (interval) t, and $\text{overlap}(t_1, t_2)$ which is true iff time intervals t_1 and t_2 overlap.

5. **Provide an initial predicate-logic formula without quantifiers.** This step allows one to provide the basic logical structure of the statement without getting bogged down by the types of quantifiers needed for each variable, and the ordering of the quantifiers. Using our running example, we want to imagine an instructor who is teaching two or more courses, and then assert that the time intervals of these courses do not overlap. Since we are thinking of two different courses, it seems appropriate to have two course variables c_1 and c_2. Moreover, we do not want these courses to overlap in time *provided* the instructor is teaching both of them. This suggests the if-then statement

$$(\text{teach}(c_1, i) \wedge \text{teach}(c_2, i) \wedge \text{time}(c_1, t_1) \wedge \text{time}(c_2, t_2)) \rightarrow \neg\text{overlap}(t_1, t_2).$$

6. **Assign quantifiers to each variable and assign an order to the quantifiers.** The predicate-logic formula provided in the previous step cannot evaluate to `true` or `false` until all variables are indexed by a quantifier. A universal quantifier is appropriate when the statement is referring to all generic members. For example, we know that *all* instructors have a real problem when it comes to teaching two different courses at the same time. On the other hand, an existential quantifier seems appropriate when the generic member represents some particular member, but not too particular to where it where it can be identified with a constant. In other

words, existential is appropriate for a generic member that possesses some particular property that is not necessarily possessed by every member. For example, recall the statement "every carrier has some flight that departs LAX". This statement suggests two variables c and f, with c being assigned a universal quantifier, and f being assigned an existential one. This is because the statement refers to all carriers, and to a particular (albeit generic) flight owned by that carrier. Finally, the order of $\forall c \exists f$ means "for every carrier this a flight owned by that carrier", where as $\exists f \forall c$ means "there is a flight such that every carrier owns that flight" which is not what we want. Indeed, to verify the statement we would first go through a list of all carriers, and, for each carrier in the list, we would have to examine all the flights owned by the carrier, looking for one that departs LAX. On the other hand, to verify the statement "there is a course for which every student received an A grade", we would first look through a list of courses, searching for at least one course whose gradebook lists final grades consisting of all A's.

Returning to our running example, it seems appropriate to assign a universal quantifier to each of the variables i, c_1, c_2, t_1, and t_2. This is because the statement applies to all instructors, courses, and times. In other words, if you are given any instructor, any two courses, and any two times, and are told that the instructor is teaching the two courses, and that the two times correspond with the times of the courses, then you must conclude that the two times do not overlap. Therefore we have the statement

$$\forall i \forall c_1 \forall c_2 \forall t_1 \forall t_2 (\text{teach}(c_1, i) \wedge \text{teach}(c_2, i) \wedge \text{time}(c_1, t_1) \wedge \text{time}(c_2, t_2)) \tag{5}$$

$$\rightarrow \neg \text{overlap}(t_1, t_2).$$

Notice that, since the quantifiers are all universal, the order in which they are written does not matter.

7. **Check for exceptional cases that may invalidate the formula.** At times a formula that seems correct will actually fail for one or more exceptional cases. And usually such exceptions are of a technical nature, rather than involving the actual problem being modeled. For example, in formula 5 above, since c_1 and c_2 have the same domain set of all courses, there will be times when c_1 and c_2 simultaneously assume the same course, in which case the times t_1 and t_2 should coincide as well, rather than not overlap. Thus, we need the added conditional hypothesis that $c_1 \neq c_2$, which gives the revised formula

$$\forall i \forall c_1 \forall c_2 \forall t_1 \forall t_2 (\text{teach}(c_1, i) \wedge \text{teach}(c_2, i) \wedge \text{time}(c_1, t_1) \wedge \text{time}(c_2, t_2) \wedge$$

$$c_1 \neq c_2) \rightarrow \neg \text{overlap}(t_1, t_2).$$

Example 17. Let's see how the above guidelines help to model the statement "with the exception of zero, every natural number is preceded by some other natural number".

1. The statement is about the domain of natural numbers.
2. The number 0 is the only particular natural number mentioned in the statement.
3. The statement speaks of two generic numbers, where the first number, call it x, is some arbitrary natural number, while the second number, call it y, is a natural number that precedes it.
4. To say that y comes before x requires the binary relation $<$, while $\neq$ is needed to assert that $x \neq 0$.
5. A plausible first attempt at a predicate logic formula is
$$x \neq 0 \wedge y < x.$$
6. The statement implies that x can be any natural number (except 0) while y is a particular number that depends on x. For example, if $x = 7$, then y should be a number less than 7, such as 6. Therefore, the quantifier sequence $\forall x \exists y$ seems appropriate.
7. The formula is now
$$\forall x \exists y (x \neq 0 \wedge y < x).$$
Unfortunately, this does not work for $x = 0$, since $0 \neq 0 \wedge y < 0$ is a false statement since $0 \neq 0$ is false. Rather than using $x \neq 0$ as a statement that must be true, instead it may serve as a condition that must be true in order for $y < x$ to be true for some y. Thus we have
$$\forall x \exists y (x \neq 0 \rightarrow y < x).$$
Equivalently, since y does not appear in the hypothesis, $\exists y$ may be moved inward to attain the statement
$$\forall x (x \neq 0 \rightarrow \exists y (y < x)).$$

□

Example 18. We translate the statement "between any two real numbers there is another real number that lies between them" into predicate logic. This statement makes reference to the domain of real numbers. It does not mention any particular real number, but does mention three generic real numbers, call them x, y, and z, where x and y are the "any two numbers", and z is the third number that "lies between them". Using the less-than relation, we may form the initial quantifier-free predicate formula

$$x < z \wedge z < y.$$

Adding quantifiers, we assign universals to x and y, since x and y are any two real number, and assign an existential to z, since z is depends on x and y (i.e. choosing any z will not always work). This yields

$$\forall x \forall y \exists z (x < z \wedge z < y).$$

Finally, this formula works for almost all cases, except for when $x \geq y$. So we must add the condition $x < y$ to obtain the final formula

$$\forall x \forall y (x < y \rightarrow \exists z (x < z \wedge z < y)).$$

□

Nested quantifiers and nested loops

Example 19. Let `p(int i, int j)` denote a Boolean function that has been implemented in some language (such as C++ or Java). Show how to make calls to `p` within a program that determines the truth value of the statement $\forall x \exists y p(x, y)$. Assume that both i and j have domain $\{0, 1, 2, \ldots, n-1\}$, for some integer n.

Example Solution.

```
//n is the domain size of the variable inputs to p
bool forall_x_exists_y_p_x_y(int n)
{
    int x,y;

    for(x=0; x < n; x++)
    {
        for(y=0; y < n; y++)
            if(p(x,y))
                break;

        if(y == n) return false;
    }

    return true;
}
```

Example 20. Repeat the previous example, but now use $\forall x \forall y p(x, y)$.

Example Solution.

Exercises

1. Let C denote the set of automobiles that registered in your state of residence. Let c be a variable whose domain is C. Define functions $f(c)$ and $g(c)$ for which f is a predicate function and g is not a predicate function. Provide the codomain set for each function. Hint: answers may vary.

2. Repeat the previous exercise, but now let C denote the set of major courses that you must complete to earn your degree.

3. Define an equality function for the set P of people living in the world.

4. Let A and B be sets, and let $\mathcal{F}$ denote the set of all functions that can be defined having domain A and codomain B. For which of these sets, A or B, must equality be defined in order to define an equality function for $\mathcal{F}$? Explain.

5. Which of the following C code snippets represent predicate-logic formulas? Explain.

```
a. x = y+7;
b. else if(x > 3)
c. (x <= 3)
d. while(y != 0)
```

6. Consider the following relational operation $<$ on finite sets of integers. Given finite integer sets A and B, then $A < B$ iff either $|A| < |B|$ or $|A| = |B|$ and $\min(A - B) < \min(B - A)$, where, e.g. $\min(A - B)$ equals the least the least member of $A - B$. For example, $\{10, 5\} < \{4, 7, 12\}$ and $\{10, 3, 34\} < \{5, 18, 36\}$ are both true statements. Determine the truth or falsehood of the following.

   a. $\{8, 3, 10, 12\} < \emptyset$
   b. $\{8, 3, 10, 12\} < \{2, 7, 4, 11\}$
   c. $\{7, 5, 3, 1\} < \{3, 5, 4, 2\}$

7. Consider the following relational operation $<$ on the set of tuples whose components are integers. Given finite tuples t and u, then $t < u$ iff either $|t| < |u|$ or $|t| = |u|$ and there is an $1 \leq i \leq |t|$ for which $t_1 = u_1, \ldots, t_{i-1} = u_{i-1}$ and $t_i < u_i$. Note: here $|t|$ denotes the length of tuple t. For example, $(10, 5) < (4, 7, 12)$ and $(10, 13, 37) < (10, 18, 36)$ are both true statements (the latter one being true since $10 = 10$, but $13 < 18$). Determine the truth or falsehood of the following.

   a. $(8, 3, 10, 12) < (1000)$
   b. $(8, 3, 10, 12) < (18, 16, 14, 13)$
   c. $(7, 5, 3, 3) < (7, 5, 3, 1)$

8. Let predicate function $S(f, c)$ evaluate to `true` iff flight f serves city c, and $L(f, a)$ evaluate to `true` iff flight f lands at airport a. Provide a predicate logic formula has the meaning "For Flight 286 to land in Buffalo, it is necessary that it also land in Boston".

9. Let predicate function $S(f, c)$ evaluate to `true` iff flight f serves city c, and $L(f, a)$ evaluate to `true` iff flight f lands at airport a. Provide a predicate logic formula has the meaning "Flight f serves NYC provided it lands at Kennedy Airport or LaGuardia Airport".

10. Suppose variables x, y each have an integer domain. Provide a predicate-logic formula that models the statement "if x or y is a member of set A, then their sum cannot be a member of A, but their product is a member of A".

11. Suppose variables x, y each have an integer domain. Provide a predicate-logic formula that models the statement "(x, y) is a member of set A if and only if x is even or y is even, but not both". Hint: make use of the `mod` function, where a mod b evaluates to the remainder when dividing a by b.

12. Provide a statement that is logically equivalent to $\neg(x < 7 \wedge x = 5)$, but does not use negation. Assume the domain of x to be a subset of the integers.

13. Let variable s have domain equal to the set of CSULB students. Let predicate function cecs228(s) evaluate to `true` iff student s has completed CECS 228, and predicate function pl(s) evaluate to `true` iff student s has studied predicate logic.

    a. Translate the statement "Every student who has completed CECS 228 has studied predicate logic" into predicate logic.

    b. Provide a layperson translation of the following predicate-logic formula.

    $$\exists s(\text{pl}(s) \wedge \neg\text{cecs228}(s).$$

14. Let $b(p, c)$ evaluate to `true` iff person p was born in city c, and $r(p, c)$ evaluate to `true` iff person p resides in city c.

    a. Translate the statement "There is someone who resides in LA and was born in either Tuscon or Phoenix".

    b. Provide a layperson translation of the following predicate-logic formula.

    $$\exists c(b(\text{Garrett}, c) \wedge r(\text{Garrett}, c).$$

15. Let $P(x)$ be the statment $x = x^2$, where x has domain the set of integers. Evaluate the following:

    a. $P(0)$

    b. $P(1)$

    c. $P(2)$

    d. $\exists x P(x)$

    e. $\forall x P(x)$

16. Let $C(x)$ represent the statement "x is a comedian", while $F(x)$ represents the statement "x is funny". Provide layperson translations of the following predicate-logic expressions. Assume the domain of x is the set of all living people.

    a. $\forall x(C(x) \rightarrow F(x))$

    b. $\exists x(C(x) \rightarrow F(x))$

    c. $\exists x(C(x) \wedge F(x))$

d. $\forall x(C(x) \vee F(x))$

17. Find the largest non-empty domain for x that is a subset of the integers and which makes the following statements true.

    a. $\forall x(x > 0)$

    b. $\neg\exists x(x \leq 5 \vee x > 20)$

    c. $\forall x(x \geq 5 \wedge x \leq 5)$

    d. $\neg\exists x(x \neq 7)$

18. Repeat the previous problem, but now find a single smallest domain that makes each of the statements false.

19. Let $L(x, y)$ denote the statement that "x likes y", where x and y have domain equal to some group of people who include Al, Ben, and Cathy. Provide predicate logic statements that model the following English statements.

    a. Either Ben likes Al or Cathy, but not both.

    b. Someone likes Cathy.

    c. Every person likes himself or herself.

    d. Someone liking Al is necessary for Al to like himself.

20. Let $F(x, y)$ stand for "flight x lands in city y" where x has the domain equal to the set of all flights for some airline company, while x is the set of all cities that have airports. Provide predicate logic statements that model the following English statements.

    a. Flight 3245 lands in Boston.

    b. No flight lands in Fullerton.

    c. If some flight lands in Fullerton, then some flight lands in Merced.

    d. If a flight lands in Nashville, then it cannot land in any other city. Hint: introduce another variable z that has domain equal to the set of all cities.

21. A positive integer $n > 1$ is considered prime if the only positive integers that divide evenly into n is 1 and n. Write a predicate-logic formula that evaluates to true if and only if n is prime. Hint: to do so you will need to define an index variable x, and use the infix predicate $|$, where, e.g., $a|b$ is true provided a divides evenly into b.

22. Let y have domain equal to the set of all html pages located at a csulb.edu address, while z has the domain the set of all webpages found at a .edu address. Let $C(x)$ evaluate to true iff html page x is located at csulb.edu, while $LB(x)$ evaluates to true iff the content of x includes the phrase "Long Beach". Translate the following statements into predicate logic.

    a. Using variable y: all html pages located at csulb.edu include the phrase "Long Beach".

    b. Using variable z: all html pages located at csulb.edu include the phrase "Long Beach".

    c. Using variable y: more than 10,000 html pages at csulb.edu do not include the phrase "Long Beach". Hint: use a different existential operator.

d. Using variable z: there is an html page at csulb.edu that does not include the phrase "Long Beach".

23. Let $D(x)$ represent the statement "x is dating Linda". Let $A(x)$ represent the statement "x is attending the party". Translate the following statements into predicate logic formulas.

    a. Any one who is dating Linda will attend the party if and only Linda attends the party.

    b. If more than 10 people attend the party, then Sam is not attending.

    c. Ron is neither attending the party nor dating Linda.

    d. Joel is dating Linda provided both attend the party.

24. Two predicate-logic formulas p and q are called logically equivalent iff the formula $p \leftrightarrow q$ always evaluates to true; regardless of how one defines the domains of the index variables, and/or the predicate functions. Provide a domain for x and define $P(x)$ and $Q(x)$ in such a way that it becomes clear that $\forall x(P(x) \rightarrow Q(x))$ is not logically equivalent to $\forall x P(x) \rightarrow \forall x Q(x)$.

25. Repeat the previous problem, but now assume the formulas are $\exists x(P(x) \rightarrow Q(x))$ and $\exists x P(x) \rightarrow \exists x Q(x)$.

26. Which of the following pairs of formulas are logically equivalent? Explain.

    a. $\forall x(P(x) \wedge Q(x))$, $\forall x P(x) \wedge \forall x Q(x)$

    b. $\exists x(P(x) \wedge Q(x))$, $\exists x P(x) \wedge \exists x Q(x)$

    c. $\forall x(P(x) \vee Q(x))$, $\forall x P(x) \vee \forall x Q(x)$

    d. $\exists x(P(x) \vee Q(x))$, $\exists x P(x) \vee \exists x Q(x)$

    e. $\forall x(P(x) \oplus Q(x))$, $\forall x P(x) \oplus \forall x Q(x)$

    f. $\exists x(P(x) \oplus Q(x))$, $\exists x P(x) \oplus \exists x Q(x)$

27. Use set-builder notation to give a description of the following sets.

    a. $\{0, 3, 6, 9, 12\}$

    b. $\{-3, -2, -1, 0, 1, 2, 3\}$

    c. $\{1, 8, 27, 64, 125, \ldots\}$

28. Let A be the set of nonnegative integers divisible by 2, while B is the set of nonnegative integers divisible by 3. Use set-builder notation to express $A \cup B$, $A \cap B$, $A \oplus B$, $A - B$, and $B - A$.

29. Let x have domain a set of people, y have domain a set of sports. Provide a layperson translation for each of the following. Let $P(x, y)$ represent the statement "x plays sport y".

    a. $\forall x \exists y P(x, y)$

    b. $\forall x \forall y P(x, y)$

    c. $\exists x \forall y P(x, y)$

    d. $\exists x \exists y P(x, y)$

    e. $\forall y \exists x P(x, y)$

f. $\exists y \forall x P(x, y)$

30. Let x and y both have domain $\{0, 1, \ldots 10\}$. Which of the following statements are true? Explain

    a. $\forall x \exists y (x \geq y)$
    b. $\forall x$
    c. $\exists x \forall y (x \geq y)$
    d. $\exists x \exists y (x \neq y)$

31. Let x have domain a set of people, y have domain a set of sports. Let $P(x, y)$ represent the statement "x plays sport y". Use these to translate the following statements into predicate logic.

    a. Tom plays exactly 4 different sports.
    b. More than 10 million people play tennis.
    c. All people who play hockey also play either lacrosse or baseball.
    d. All people who play tennis also play raquetball, and vice versa.
    e. Everyone plays tennis except for Joanne.
    f. There are some people who play lacrosse, but not hockey.

32. Let x have domain a set of people, y have domain a set of sports. Let $P(x, y)$ represent the statement "x plays sport y", while $W(x, y)$ represents "x watches sport y".

    a. Tom watches every sport he plays.
    b. More than 1 million people play hackeysack, but no one watches hackeysack.
    c. There is a sport that everyone either watches or plays.
    d. Alex does not watch the sports he plays, except tennis.
    e. Every sport is either watched by Richard, or played by Richard, but not both.
    f. Every person watches or plays some sport.

33. Let i and j have domains $0, 1 \ldots (n-1)$. For each of the following predicate-logic formulas, write C/Java code that is capable of evaluating the statment.

    a. $\forall i \exists j p(i, j)$
    b. $\forall i \forall j p(i, j)$
    c. $\exists i \forall j p(i, j)$
    d. $\exists i \exists j p(i, j)$

    In all cases, assume that you have calling access to Boolean function $p(i, j)$.

34. A real-valued function $f(x)$ is said to be **monotone** iff $f(x) \leq f(y)$ whenever $x \leq y$. Write a predicate-logic formula that models the statement "$f(x) = x^3$ is a montone function".

35. Provide a statement that is logically equivalent to $\neg \forall x \exists y (x > y)$, but does not use negation. Assume the domain of x to be a subset of the integers.

Exercise Answers and Hints

1. Let $f(c)$ equal `true` iff auto c has been smog tested within the past year. Codomain: $\mathcal{B}$. Let $g(c)$ equal the license plate number of automobile c. Codomain: the set of all alphanumeric sequences having a length that does not exceed 10.

2. Let $f(c)$ equal `true` iff course c is listed as an elective. Codomain: $\mathcal{B}$. Let $g(c)$ equal the number of units earned from completing course c. Codomain: $\{1, 2, 3, 4\}$.

3. Let p_1 and p_2 both have domain P. Then $p_1 = p_2$ iff p_1 and p_2 have matching fingerprints.

4. Let f and g be to functions, both having domain A and codomain B. Then $f = g$ iff $f(a) = g(a)$ for every $a \in A$. Notice that this requires an equality function for set B. Since $f(b)$ and $g(b)$ must be compared.

5. Which of the following C code snippets represent predicate-logic formulas? Explain.

```
a. x = y+7; //No: command to assign y+7 to x
b. else if(x > 3) //x > 3 is a predicate-logic formula
c. (x <= 3)  //Yes
d. while(y != 0) //y != 0 is a predicate-logic formula
```

6. We have the following.

   a. $\{8, 3, 10, 12\} < \emptyset$ is false since $|\emptyset| < |\{8, 3, 10, 12\}|$.

   b. $\{8, 3, 10, 12\} < \{2, 7, 4, 11\}$ is false since $2 < 3$.

   c. $\{7, 5, 3, 1\} < \{3, 5, 4, 2\}$ is true since $1 < 2$.

7. We have the following.

   a. $(8, 3, 10, 12) < (1000)$ is false since $|(1000)| = 1 < |(8, 3, 10, 12)| = 4$.

   b. $(8, 3, 10, 12) < (18, 16, 14, 13)$ is true since $8 < 18$.

   c. $(7, 5, 3, 3) < (7, 5, 3, 1)$ is false since $7 = 7, 5 = 5, 3 = 3$, but $3 > 1$.

8. We have
$$L(286, \text{Buffalo}) \rightarrow L(286, \text{Boston}).$$

9. We have
$$L(f, \text{Kennedy}) \vee L(f, \text{LaGuardia}) \rightarrow S(f, \text{NYC}).$$

10. We have
$$(x \in A \vee y \in A) \rightarrow (x + y \notin A \wedge xy \in A).$$

11. We have
$$(x, y) \in A \leftrightarrow (x \bmod 2 = 0 \oplus y \bmod 2 = 0).$$

12. $x \geq 7 \vee x \neq 5$

13. a. $\forall s(\text{cecs228}(s) \rightarrow \text{pl}(s))$.

b. There is a CSULB student who has not completed CECS 228, but has studied predicate logic.

14. a.
$$\exists p(r(p, \text{LA}) \wedge (b(p, \text{Tuscon}) \oplus b(p, \text{Phoenix})).$$
b. Garrett was born in the same city for which he resides.

15. a,b, and d evaluate to true, while c and e evaluate to false.

16. a. All comedians seem funny.
b. There is some person such that, if he or she is a comedian, then he or she seems funny.
c. There is some person who is a funny comedian.
d. Everyone is either a comedian, or seems funny.

17. a. The set of positive integers
b. $\{6, 7 \ldots, 20\}$
c. $\{5\}$
d. $\{7\}$

18. A domain of $\{0\}$ makes each statement false.

19. For example, we let constant a stand for "Al".

a. $L(b, a) \oplus L(b, c)$
b. $\exists x L(x, c)$
c. $\forall x L(x, x)$
d. $L(a, a) \rightarrow \exists x L(x, a)$

20. For example, let b be a constant that stands for "Boston".

a. $F(3245, b)$
b. $\neg\exists x F(x, f)$
c. $\exists x F(x, f) \rightarrow \exists x F(x, m)$
d. $\forall x(F(x, n) \rightarrow \forall z(F(x, z) \rightarrow z = n))$

21. Hint: to do so you will need to define an index variable x, and use the infix predicate $|$, where, e.g., $a|b$ is true provided a divides evenly into b.

22. a. $\forall y LB(y)$
b. $\forall z(C(z) \rightarrow LB(z))$
c. $\exists^{>10000} y LB(y)$
d. $\exists z(C(z) \wedge \neg LB(z))$

23. Lowercase letters will be used for the names of people.

a. $\forall x(D(x) \rightarrow (A(x) \leftrightarrow A(l)))$

b. $\exists^{>10} x A(x) \rightarrow \neg A(s)$

c. $\neg A(r) \wedge \neg D(r)$

d. $(A(l) \wedge A(j)) \rightarrow D(j))$

24. Let x have domain the set of all people. $P(x)$ is true if x likes pizza. $Q(x)$ is true if x likes quiche.

25. Hint: define P so that it will be sometimes false and sometimes true. Define Q so that it is always false.

26. Only a. and d. are logically equivalent.

27. In the following we assume that domain(x) = domain(y) = Integers.

a. $\{x | x \geq 0 \wedge x \leq 12 \wedge x \bmod 3 = 0\}$

b. $\{x | x \geq -3 \wedge x \leq 3\}$

c. $\{x | x > 0 \wedge \exists y(y^3 = x)\}$

28. $A \cup B = \{x | x \bmod 2 = 0 \vee x \bmod 3 = 0\}$, $A \cap B = \{x | x \bmod 2 = 0 \wedge x \bmod 3 = 0\}$, $A \oplus B = \{x | x \bmod 2 = 0 \oplus x \bmod 3 = 0\}$, $A - B = \{x | x \bmod 2 = 0 \wedge x \bmod 3 \neq 0\}$, $B - A = \{x | x \bmod 2 \neq 0 \wedge x \bmod 3 = 0\}$.

29. a. Everyone plays a sport.

b. Everyone plays every sport.

c. Somone plays every sport.

d. Someone plays a sport.

e. Every sport is played by someone.

f. There is a sport that everyone plays.

30. All but b evaluate to true.

31. a. $\exists^{=4} y P(t, y)$

b. $\exists^{>10000000} x P(x, t)$

c. $\forall x(P(x, h) \rightarrow (P(x, l) \vee P(x, b)))$

d. $\forall x(P(x, t) \leftrightarrow P(x, r))$

e. $\forall x(x \neq j \rightarrow P(x, t)) \wedge \neg P(j, t)$

f. $\exists x(P(x, l) \wedge \neg P(x, h))$

32. a. $\forall y(P(t, y) \rightarrow W(t, y))$

b. $\exists^{>1000000} x P(x, h) \wedge \forall x \neg W(x, h)$

c. $\exists y \forall x(W(x, y) \vee P(x, y))$

d. We interpret the statement to mean that Alex does play and watch tennis, but does not watch any other of the sports that he may play.

$$\forall y((y \neq t \wedge P(a, y)) \rightarrow \neg W(a, y)) \wedge P(a, t) \wedge W(a, t)$$

e. $\forall y(W(r, y) \oplus P(r, y))$

f. $\forall x \exists y(W(x, y) \vee P(x, y))$

33. See Example 19 for a hint on how to code each of the functions.

34. We have

$$\forall x \forall y(x \leq y \rightarrow x^3 \leq y^3).$$

35. $\exists x \forall y(x \leq y)$