Practice Midterm Exam 4

This exam is closed-book and closed-computer. You may have a double-sided, 8.5” × 11” sheet of notes with you when you take this exam. You may not have any other notes with you during the exam. You may not use any electronic devices (laptops, cell phones, etc.) during the course of this exam. Please write all of your solutions on this physical copy of the exam.

You are welcome to cite results from the problem sets or lectures on this exam. Just tell us what you're citing and where you're citing it from. However, please do not cite results that are beyond the scope of what we've covered in CS103.

On the actual exam, there'd be space here for you to write your name and sign a statement saying you abide by the Honor Code. We're not collecting or grading this exam (though you're welcome to step outside and chat with us about it when you're done!) and this exam doesn't provide any extra credit, so we've opted to skip that boilerplate.

You have three hours to complete this practice midterm. There are 32 total points. This practice midterm is purely optional and will not directly impact your grade in CS103, but we hope that you find it to be a useful way to prepare for the exam. You may find it useful to read through all the questions to get a sense of what this practice midterm contains before you begin.

You have three hours to complete this exam. There are 48 total points.

<table>
<thead>
<tr>
<th>Question</th>
<th>Points</th>
<th>Graders</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) Mathematical Logic</td>
<td>/ 12</td>
<td></td>
</tr>
<tr>
<td>(2) Set Theory</td>
<td>/ 12</td>
<td></td>
</tr>
<tr>
<td>(3) Proofwriting I</td>
<td>/ 12</td>
<td></td>
</tr>
<tr>
<td>(4) Proofwriting II</td>
<td>/ 12</td>
<td></td>
</tr>
<tr>
<td></td>
<td>/ 48</td>
<td></td>
</tr>
</tbody>
</table>

You can do this. Best of luck on the exam!
When we discussed first-order logic, we spent a decent amount of time talking about how to translate statements from English into first-order logic. You practiced this skill on Problem Set Two. In this problem, we’d like you to show us what you’ve learned about the art of first-order translation.

Leo Tolstoy’s classic novel *Anna Karenina* begins with this famous line:

“All happy families are alike; each unhappy family is unhappy in its own way.”

This question explores the first-order structure of this sentence.

i. **(3 Points)** Given the predicates

   - Happy\( (x) \), which states that \( x \) is happy;
   - Family\( (x) \), which states that \( x \) is a family; and
   - Alike\( (x, y) \), which states that \( x \) and \( y \) are alike,

   translate the statement “all happy families are alike” into first-order logic. (You’ll translate the second half of the opening line of *Anna Karenina* in the next part of this problem.)

ii. **(5 Points)** Given the predicates

   - UnhappyFamily\( (x) \), which states that \( x \) is an unhappy family;
   - Reason\( (x) \), which states that \( x \) is a reason; and
   - UnhappyBecause\( (x, y) \), which states that \( x \) is unhappy because of \( y \),

   translate the statement “each unhappy family is unhappy in its own way” into first-order logic. You should interpret the statement to mean “each unhappy family is unhappy for a reason, and no other unhappy families are unhappy for that same reason.”
We've talked a lot about negating and simplifying statements in first-order logic, a useful skill with applications to disproofs, proofs by contradiction, and proofs by contrapositive. You practiced this on Problem Set Two, and in this question we'd like you to demonstrate what you've learned.

iii. **(4 Points)** Consider the following statement in first-order logic:

\[
(\forall x. (K(x) \leftrightarrow \forall z. \neg A(x, z))) \land (\forall x. (H(x) \rightarrow \exists y. (B(x, y) \land D(x, y))))
\]

Give a statement in first-order logic that is the negation of this statement. As in Problem Set Two, your final formula must not have any negations in it, except for direct negations of predicates. Do not introduce any new predicates, functions, or constants.

While you're not required to show every step of your work, it does make it a bit easier for us to give you partial credit in the event that you accidentally make a mistake. If you do show each step, please clearly indicate what your final answer is.

As a courtesy to your TAs, please try to make the parenthesization of your final statement as clear as possible.
Problem Two: Set Theory  (12 Points)

(CS103 Midterm, Fall 2017)

On Problem Set One, you wrote C++ code that took in a pair of objects \( S \) and \( T \), then returned whether those objects had various properties. For example, you wrote code to test whether \( S \in T \), whether \( S \subseteq \wp(\wp(T)) \), whether \( S = \{T\} \), etc.

In this problem, we'd like you run this process in reverse. Below are five C++ functions that take as input some number of objects, then determine whether some property holds of them. For each of those functions, write a statement using set theory notation indicating what that function checks for. No justification is necessary.

To receive full credit, express your answers purely using set theory notation (\( \in \), \( \subseteq \), \( \cup \), \( = \), \( \wp \), etc.) and without using any quantifiers or propositional connectives. For example, if a piece of code checked whether \( S \) was a subset of \( T \), you should write \( S \subseteq T \) rather than \( \forall x. (x \in S \rightarrow x \in T) \). Answers using quantifiers or propositional connectives will only receive partial credit.

A syntax refresher:

- The function \( \text{isSet}(S) \) returns whether \( S \) is a set.
- The function \( \text{asSet}(S) \) returns a view of object \( S \) as a set.
- The notation \( S.\text{count}(x) \) returns whether \( x \in S \).
- The notation \( S.\text{size()} \) returns |\( S \)|.
- The notation \( \text{for} \ (\text{Object } x: S) \{ \ldots \} \) iterates over all elements of \( S \), one at a time.

```cpp
bool mysteryPredicateOne(Object S) { // 1 Point
    return isSet(S) && asSet(S).size() == 0;
}
```

What this code tests for: ___________________________________________

```cpp
bool mysteryPredicateTwo(Object S, Object T) { // 2 Points
    if (!isSet(S) || !isSet(T)) return false;

    for (Object x: asSet(S)) {
        if (asSet(T).count(x)) return true;
    }

    for (Object x: asSet(T)) {
        if (asSet(S).count(x)) return true;
    }

    return false;
}
```

What this code tests for: ___________________________________________

(Continued on the next page)
```cpp
bool mysteryPredicateThree(Object S, Object T) { // 3 Points
    if (!isSet(T)) return false;
    for (Object X: asSet(T)) {
        if (isSet(X) && asSet(X).size() == 1) {
            if (asSet(X).count(S)) return true;
        }
    }
    return false;
}

What this code tests for: ____________________________________________

bool mysteryPredicateFour(Object S, Object T, Object U) { // 3 Points
    if (!isSet(S) || !isSet(T) || !isSet(U)) return false;
    for (Object x: asSet(S)) {
        if (!asSet(T).count(x)) return false; // Note the ! sign here.
        if (asSet(U).count(x)) return false; // There is no ! sign here.
    }
    return true;
}

What this code tests for: ____________________________________________

bool mysteryPredicateFive(Object S) { // 3 Points
    if (!isSet(S)) return false;
    for (Object T: asSet(S)) {
        if (!isSet(T)) return false;
        for (Object x: asSet(T)) {
            if (!asSet(S).count(x)) return false;
        }
    }
    return true;
}

What this code tests for: ____________________________________________
On Problem Set Two, you explored tournaments, contests between groups of \( n \) players. This problem explores another property of tournaments.

Let’s begin by refreshing some definitions. A tournament is a contest among \( n \) players. Each player plays a game against each other player, and either wins or loses the game (let’s assume that there are no draws). We can visually represent a tournament by drawing a circle for each player and drawing arrows between pairs of players to indicate who won each game. For example, in the tournament to the left, player \( A \) beat player \( E \), but lost to players \( B \), \( C \), and \( D \).

A tournament champion is a player in a tournament who, for each other player, either won her game against that player, or won a game against a player who in turn won his game against that player (or both). For example, in the graph on the left, players \( B \), \( C \), and \( E \) are tournament champions. However, player \( D \) is not a tournament champion, because he neither beat player \( C \), nor beat anyone who in turn beat player \( C \). Although player \( D \) won against player \( E \), who in turn won against player \( B \), who then won against player \( C \), under our definition player \( D \) is not a tournament champion. (Make sure you understand why!)

i. **(4 Points)** Let \( T \) be an arbitrary tournament. What is the contrapositive of the following statement? (You don’t need to expand out the definition of a tournament champion when writing out your answer.)

If \( p \) and \( q \) are tournament champions in \( T \), then at least one of \( p \) and \( q \) won more than one game in \( T \).
As a refresher from the previous page, a tournament champion is a player in a tournament who, for each other player, either won her game against that player, or won a game against a player who in turn won his game against that player (or both).

ii. (8 Points) Let $T$ be an arbitrary tournament with at least four players, and let $p$ and $q$ be two different players in $T$. Prove the following statement using a proof by contrapositive:

If $p$ and $q$ are tournament champions in $T$, then at least one of $p$ and $q$ won more than one game in $T$.

Some hints on this problem:

- You may want to draw some pictures to help build your intuition for what’s going on, though your final proof should be purely written and not involve any diagrams.
- Remember that $p$ and $q$ played a game against one another. You can assume, without loss of generality, that $p$ won that game.
- The fact that $T$ has at least four players in it is significant. This result isn’t true if the tournament has only three players.
- You may want to introduce some variables to refer to other players in the tournament.

Feel free to use the space below for scratch work. There’s room to write your answer to this problem on the next page of this exam.
(Extra space for your answer to Problem Three, Part (ii), if you need it.)
Problem Four: Proofwriting II  
(CS103 Midterm, Fall 2017)  

You now have quite a bit of experience writing proofs about sets. On Problem Set One, you practiced writing proofs and disproofs about properties of sets. On Problem Set Two, you explored hereditary sets and proved some results about them. This problem is designed to let you show us what you’ve learned about writing proofs pertaining to sets.

This particular problem concerns a special kind of a set called a transitive set. Formally speaking, we’ll say that a set $S$ is **transitive** if $S \subseteq \wp(S)$. These sets have all sorts of fun properties and show up in the study of the mathematical nature of infinity.

i. **(4 Points)** Below are a list of sets. For each set, determine whether or not it’s a transitive set. No justification is required, and there’s no penalty for an incorrect guess.

<table>
<thead>
<tr>
<th>Set</th>
<th>Transitive</th>
<th>Not Transitive</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathbb{N}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\emptyset$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>${\emptyset, {\emptyset}}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>${{\emptyset}}$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
As a refresher from the previous page, we’ll say that a set $S$ is a **transitive set** if $S \subseteq \mathcal{P}(S)$.

ii. **(8 Points) Prove the following result using a direct proof:**

If $S$ is a transitive set, then $\mathcal{P}(S)$ is also a transitive set.

As you write up your proof of this result, please make sure to call back to the formal definitions of the subset relation and the power set. Setting the proof up properly – clearly articulating your start and end points, introducing new variables as appropriate, etc. – will make things easier.

Feel free to use this page as scratch space if you’d like. There’s more space to write your answer on the next page.
(Extra space for your answer to Problem Four, Part (ii), if you need it.)