CS103 Cynthia Bailey Lee
Autumn 2020

Problem Set 3

Checkpoint due Monday, January 27th at 2:30PM.

Remaining problems due Friday, January 31st at 2:30PM.

*These **two** checkpoint problems on this problem set are due on Monday at 2:30PM.*

Checkpoint Problem One: Binary Relations

The first part of this problem revolves around a mathematical construct called ***homogenous coordinates*** that shows up in computer graphics. If you take CS148, you'll get to see how they're used to quickly determine where to display three-dimensional objects on screen.

Let $\mathbb{R}^2$ denote the set of all ordered pairs of real numbers. For example $(137, 42) \in \mathbb{R}^2$, $(\pi, e) \in \mathbb{R}^2$, and
$(-2.71, 103) \in \mathbb{R}^2$. Two ordered pairs are equal if and only if each of their components are equal. That is, we have $(a, b) = (c, d)$ if and only if $a = c$ and $b = d$.

Consider the relation E defined over $\mathbb{R}^2$ as follows:

$$(x_1, y_1)\ E\ (x_2, y_2) \quad \text{if} \quad \exists k \in \mathbb{R}.\ (k \neq 0\ \land\ (kx_1, ky_1) = (x_2, y_2)).$$

For example, $(3, 4)\ E\ (6, 8)$ because $(2 \cdot 3, 2 \cdot 4) = (6, 8)$.

For this problem, we want to see how you would structure a proof that E is an equivalence relation over $\mathbb{R}^2$. The structure has 6 important parts, an "assume" and a "want to show" for each of the three properties: reflexive, symmetric, and transitive. We want you to separately state what these six parts of the proof would be. It would also be a good idea for you to just go ahead and complete the proof from there, but *you won't turn that in.* (If you want feedback on the fully-written proof, you're welcome to compare it to the solutions, which will include that, or come to office hours for consultation.)

i. On its own line, clearly marked "ASSUME:", write down the assumption for the **reflexive** part of the proof. As we learned in class, this includes properly introducing any variables you need to state the assumption.
ii. On its own line, clearly marked "W.T.S.:", write down the "want to show" for the **reflexive** part of the proof.
iii. On its own line, clearly marked "ASSUME:", write down the assumption for the **symmetric** part of the proof. As we learned in class, this includes properly introducing any variables you need to state the assumption.
iv. On its own line, clearly marked "W.T.S.:", write down the "want to show" for the **symmetric** part of the proof.
v. On its own line, clearly marked "ASSUME:", write down the assumption for the **transitive** part of the proof. As we learned in class, this includes properly introducing any

variables you need to state the assumption.

vi. On its own line, clearly marked "W.T.S.:", write down the "want to show" for the **transitive** part of the proof.

Remember that the "if" in the definition of the relation E means "is defined as" and isn't an implication. Follow the advice of the Guide to Proofs on Discrete Structures and the Discrete Structures Proofwriting Checklist: don't use quantifiers or connectives in your written proof. The in-class questions include predicting the beginnings of proofs, so those would be a great guide for what this means.

You may want to start off by taking the first-order statement in the definition here, and the first-order logic definitions of reflexive, symmetric, and transitive, to help guide your determination of what the assumption and want to show are. Here is an (unrelated) example to remind you. For theorem "If n is even, then n^2 is even," we would write: "ASSUME: Pick an arbitrary even integer n," and "W.T.S.: We want to show that n^2 is even."

Problem One: Computational Equivalence Relations

Equivalence relations are powerful tools for reasoning about groups of objects. We'll make extensive use of them later in the quarter when talking about the limits of finite-memory computers. In the meantime, we'd like you to get warmed up with how equivalence relations work and with some of their properties. Download the starter files for this problem from the course website. As usual, feel free to submit as many times as you'd like; we'll only look at the score for the last version you submit.

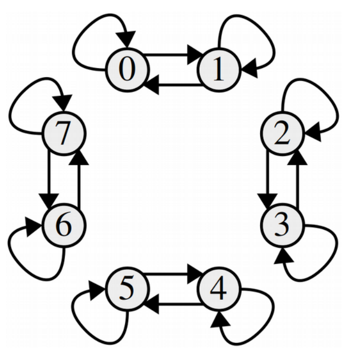
i. In the file `BinaryRelations.cpp`, implement a function

```
bool isEquivalenceRelation(Relation R, std::set<Object> A);
```

that takes as input a binary relation R and its underlying set A, then returns whether R is an equivalence relation. The `Relation` type represents a binary relation. Syntactically, to check if xRy holds, use the syntax `R(x, y)`, as in

```
if (R(x, y)) {
    /* … xRy holds … */
}
```

ii. What are the equivalence classes of the equivalence relation drawn here? Edit the constant `kTheEquivalenceClasses` in `BinaryRelations.cpp` with your answer.



iii. Implement a function

```
std::set<Object> equivalenceClassOf(Relation R, std::set<Object> A,Object x);
```

that takes as input an equivalence relation R, its underlying set A, and some element $x \in A$, then returns $[x]_R$. You can assume that $x \in A$ and that R is indeed an equivalence relation, and your function can behave however you'd like if this isn't the case. As a note, you can add elements to a `std::set` by using the syntax `mySet.insert(myObject);`

iv. Give two systems of representatives for the equivalence relation from part (ii) of this problem. Edit the constants `kSystem1` and `kSystem2` in `BinaryRelations.cpp` with your answer. A system of representatives is a set containing exactly one element from each equivalence class of an equivalence relation.

v. Implement a function

```
std::set<Object> systemOfRepresentativesFor(Relation R, std::set<Object> A);
```

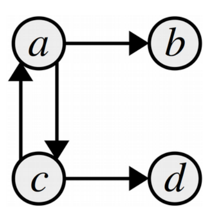
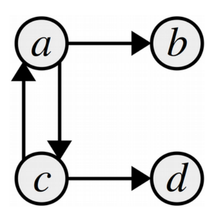
that takes as input a binary relation R over a set A and returns some system of representatives for the relation R. (There may be many different systems of representatives, and you can pick whichever one you want.) You can assume that R is an equivalence relation and your function can behave however you'd like if this isn't the case.

Our starter files are designed to call your `isEquivalenceRelation` function to determine whether the currently-displayed relation is an equivalence relation. If so, it'll invoke `systemOfRepresentativesFor` to get a system of representatives, and then use `equivalenceClassOf` on each representative to find and then color-code the equivalence classes. You can see the output of each of these function calls at the bottom of the window.

Problem Two: Building Binary Relations

Let's begin with a new definition. A ***preorder*** is a binary relation that's both reflexive and transitive.

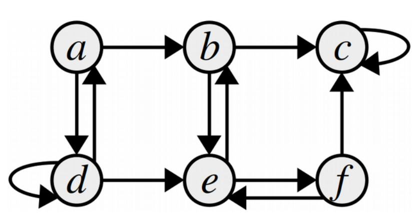
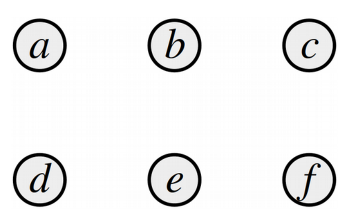
i. Below is a drawing of a relation that is not a preorder. Add the smallest number of arrows possible to this diagram to make it a preorder. No justification is required.

Original Relation	***That Relation, Expanded to a Preorder***
	

This problem is about building new relations from old ones. Here's one way to do this. If R is a binary relation over A, the ***symmetric component of R***, denoted $\boldsymbol{E_R}$, is a binary relation over A defined as follows:

$$aE_Rb \quad \text{if} \quad aRb \land bRa.$$

ii. Draw the symmetric component of the relation on the left by adding arrows on the right.

Original Relation	***Its Symmetric Component***
	

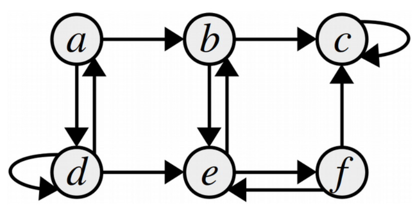
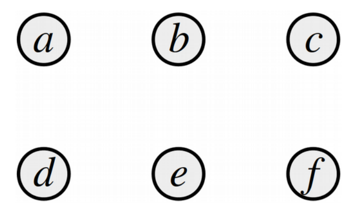
iii. Prove that if R is a preorder over a set A, then E_R is an equivalence relation.

E_R is defined in terms of R, but you can't assume E_R reflexive and transitive just because R is. You need to prove that. Go slowly. To prove E_R is transitive, what would you assume, and what do you need to show?

Here's another way to form new relations from old ones. If R is a binary relation over A, the ***asymmetric component of R***, denoted $\boldsymbol{S_R}$, is a binary relation over R over A defined as follows:

$$aS_Rb \quad \text{if} \quad aRb \land b\not R a$$

iv. Draw the asymmetric component of the relation on the left by adding arrows on the right.

Original Relation	***Its Asymmetric Component***
	

UPDATE: *you can skip this problem.*

~~v. Prove that if R is a preorder over a set A, then S_R is a strict order.~~

~~*Careful – transitivity says that if xRy and yRz, then you can conclude xRz. It does **not** say that if $x\not R y$ and $y\not R z$, then $x\not R z$. This is generally not true: for example, what if R is the $=$ relation, x is 1, y is 2, and z is 1? Also, remember that you only need to prove that S_R is irreflexive and transitive.*~~

Problem Three: Composition, Identity, Existence, and Uniqueness

Although we often speak about how relations are applied (for example, that $1 < 4$ or that $\emptyset \subseteq \mathbb{N}$), binary relations are mathematical objects in their own right, and we can talk about manipulating relations algebraically. This question explores how.

Given two relations R_1 and R_2 over the same set A, we say that $\boldsymbol{R_1 \subseteq R_2}$ if the following statement is true:

$$\forall a \in A.\ \forall b \in A.\ (aR_1b \ \rightarrow\ aR_2b).$$

Just as we say that two sets are equal if they're each subsets of the other, if R_1 and R_2 are binary relations over the same set A, we say that $\boldsymbol{R_1 = R_2}$ if $R_1 \subseteq R_2$ and $R_2 \subseteq R_1$.

i. Consider the following relations over the set $\mathbb{Z}$. First, there's the $\equiv_2$ relation that you saw on Problem Set One. Second, there's the $\sim$ relation where $x \sim y$ holds when $x+y$ is even. Prove that $\equiv_2 = \sim$.

This is a great spot to write out two columns, one for what to assume and one for what you need to show.

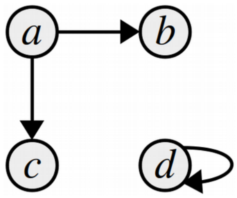
Next, let's introduce a way of combining relations together. Given two binary relations R_1 and R_2 over some set A, the ***composition*** of R_1 and R_2, denoted $\boldsymbol{R_1 \circ R_2}$, is the binary relation over A defined as

$$x(R_1 \circ R_2)y \qquad \text{if} \qquad \exists z \in A.\ (xR_1z\ \land\ zR_2y).$$

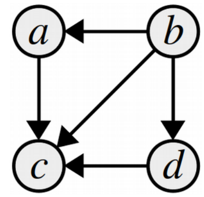
Let's do a quick check-in to make sure these definitions make sense.

ii. Below are pictures of binary relations R_1 and R_2 over some set A. Add arrows to the diagrams on the bottom to show the relation $R_1 \circ R_2$ and the relation $R_2 \circ R_1$, respectively.

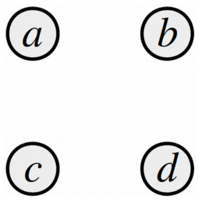
Relation R_1



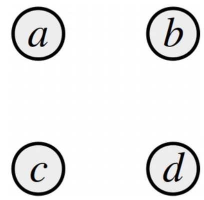
Relation R_2



Add arrows to show $R_1 \circ R_2$



Add arrows to show $R_2 \circ R_1$



Now, a new definition. Given a set A, a binary relation I_A is called an ***identity relation over A*** if the following statement is true for *every* binary relation R over A:

$$I_A \circ R = R \qquad \land \qquad R \circ I_A = R$$

In other words, composing *any* binary relation R over A with I_A, either on the left or the right, has the same effect as not doing any composition at all.

iii. Let A be an arbitrary set. Fill in the following blank in the simplest way possible to define an identity relation over A. No justification is necessary.

$$xI_Ay \qquad \text{if} \qquad \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_$$

*While you don't need to prove this, you should **definitely** triple-check your answer. You'll need to know what this relation is for another problem on this problem set.*

In the above problem we spoke of *an* identity relation over a set A, but we just as easily could have spoken of *the* identity relation over A because, for each set A, there's exactly one identity relation.

The question, then, is how you show that there's only one object with some property. From your answer to part (iii) of this problem, you know that there is *at least one* identity relation. Your answer to part (iii) is therefore sometimes called an ***existence proof***, since it shows that something exists. (We didn't ask you to formally prove that your answer was correct, but if you were to do so, that would be an existence proof.)

You now need to prove that there is *at most one* identity relation, which is often called a ***uniqueness proof***. To prove that there is only one object of some type, the standard technique is to assume you have two different objects that each have the given property, then to prove that they're actually the same object. In pseudo-first-order logic notation you'd want to prove this:

$$\forall I_1.\ \forall I_2.\ (IdentityRelation(I_1) \land IdentityRelation(I_2) \rightarrow I_1 = I_2).$$

In other words, if you think you've found two different identity relations, you've really just found two different names for the same thing.

iv. Based off the discussion in the above paragraphs, prove that there is exactly one identity relation I_A for each set A.

As a hint, you can write a fully rigorous proof of this result without referencing what it means for one *relation to be a subset of another. Look at the definition of an identity relation, which gives a series of equalities involving an identity relation. Can you make use of those equalities here?*

Problem Four: Preorders, Composition, and Identities

There's an surprising connection between the concepts from Problem Three and Problem Four:

Theorem: If R is a binary relation over a set A, then
R is a preorder if and only if $I_A \subseteq R$ and $R \circ R \subseteq R$.

This might look like a lot of symbols, so let's back up a bit and talk about what this means. Intuitively, you can think of this result as letting us switch back and forth between two different views of binary relations. The left-hand side of this biconditional talks about binary relations in the way we've mostly seen them discussed so far: we have some rules (reflexivity, transitivity) that talk about how binary relations behave with regards to individual elements. The right-hand side talks about binary relations more as whole objects in their own right, exploring more "algebraic" properties of those relations. And the biconditional linking them means that we can switch between these representations whenever we find it useful to do so!

The rest of this problem asks you to prove this.

i. Prove the reverse direction: if R is a binary relation over A where $I_A \subseteq R$ and $R \circ R \subseteq R$, then R is a preorder.

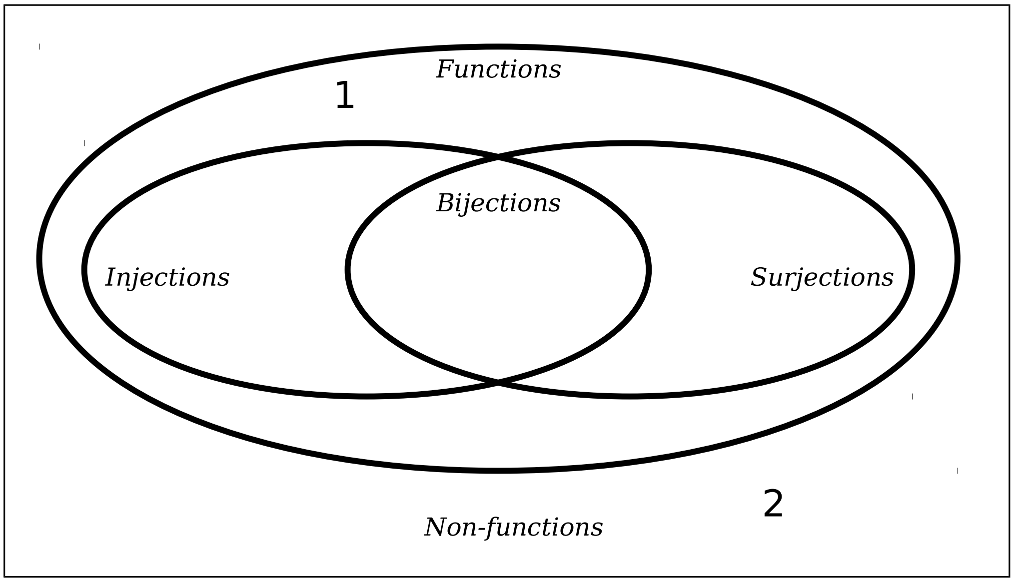
This one is all about setting things up properly and working through the definitions slowly and methodically. Use the strategy we discussed in class when working with cyclic relations: write out two columns, one containing everything you're going to assume and one containing everything that you're going to prove. Specifically think about the following: you will be assuming that $I_A \subseteq R$; what exactly does that mean? Plug I_A into the definition of what it means for one relation to be a subset of another and see what that tells you. Similarly, if you're assuming $R \circ R \subseteq R$, what exactly does that tell you? Then think about what you need to prove. What do you have to show to prove that R is a preorder? As always, drawing pictures never hurts!

ii. Prove the forward direction: if R is a binary relation over A and R is a preorder, then $I_A \subseteq R$ and $R \circ R \subseteq R$.

The same advice from part (i) applies here. Go slowly and methodically, writing out what it is that you're assuming and what you need to prove in two separate columns and drawing pictures as needed.

Problem Five: Properties of Functions

Consider the following Venn diagram:



Below is a list of purported functions. For each of those purported functions, determine where in this Venn diagram that object goes. No justification is necessary.

To submit your answers, edit the file `FunctionsVennDiagram.h` in the `src/` directory of the starter files for this problem set. To get you started, we've shown you where functions 1 and 2 go

1. $f : \mathbb{N} \to \mathbb{N}$ defined as $f(n) = 137$
2. $f : \mathbb{N} \to \mathbb{N}$ defined as $f(n) = -137$

Make sure you can explain why these first two items go where they do in this diagram!

3. $f : \mathbb{N} \to \mathbb{N}$ defined as $f(n) = n^2$
4. $f : \mathbb{Z} \to \mathbb{N}$ defined as $f(n) = n^2$
5. $f : \mathbb{N} \to \mathbb{Z}$ defined as $f(n) = n^2$
6. $f : \mathbb{Z} \to \mathbb{Z}$ defined as $f(n) = n^2$
7. $f : \mathbb{R} \to \mathbb{N}$ defined as $f(n) = n^2$
8. $f : \mathbb{N} \to \mathbb{R}$ defined as $f(n) = n^2$
9. $f : \mathbb{N} \to \mathbb{N}$ defined as $f(n) = \sqrt{n}$.

*The notation $\sqrt{n}$ denotes the **principal square root** of n, the nonnegative one.*

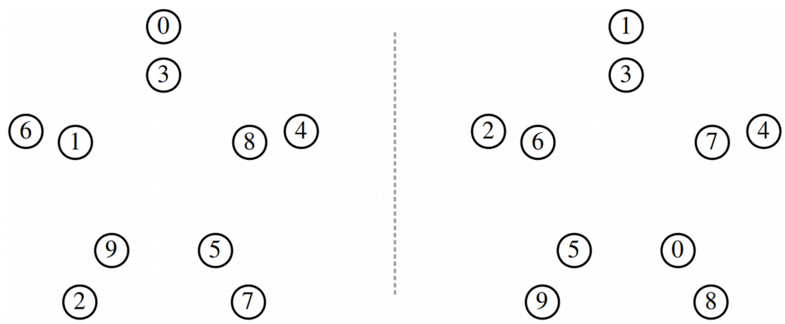
10. $f : \mathbb{R} \to \mathbb{R}$ defined as $f(n) = \sqrt{n}$.
11. $f : \mathbb{R} \to \{ x \in \mathbb{R} \mid x \geq 0 \}$ defined as $f(n) = \sqrt{n}$.
12. $f : \{ x \in \mathbb{R} \mid x \geq 0 \} \to \{ x \in \mathbb{R} \mid x \geq 0 \}$ defined as $f(n) = \sqrt{n}$.
13. $f : \{ x \in \mathbb{R} \mid x \geq 0 \} \to \mathbb{R}$ defined as $f(n) = \sqrt{n}$.
14. $f : \mathbb{N} \to \wp(\mathbb{N})$, where f is some injective function.

In other words, we tell you that we have some injective function $f : \mathbb{N} \to \wp(\mathbb{N})$, we don't tell you what it is, and yet there is enough information for you to place it in the above diagram.

15. $f : \{0, 1, 2\} \to \{3, 4\}$, where f is some surjective function.
16. $f : \{\textit{breakfast}, \textit{lunch}, \textit{dinner}\} \to \{\textit{shakshuka}, \textit{soondubu}, \textit{maafe}\}$, where f is some injection.

Problem Six: Permutation Dances

There's a dance in which each dancer has an assigned position. In the first dance, the dancers begin in the positions indicated on the left (it's a top-down view, and we've numbered the dancers 0, 1, …, 9). In the second dance, some dancers have moved to new starting positions, and the overall arrangement is what's shown on the right.



How can we model how the dancers' positions changed from the first dance to the second? For now, focus on Dancer 0. Notice that, in the second dance, Dancer 0 has moved to the inner position in the bottom-right pair. That's the spot that was occupied by Dancer 5 in the first dance. In that sense, from Dancer 0's perspective, she starts off the second dance at "the spot that Dancer 5 used to occupy."

We can do this for other people as well. Look at Dancer 8, for example. Dancer 8 ended up in the outer position in the bottom-right pair, which is where Dancer 7 used to be. So we could instruct Dancer 8 to get to his new location by saying "go to where Dancer 7 was in the previous dance."

How about Dancer 3? Notice that Dancer 3 started in the inner pair of the top-center pair, and that's where she ended up as well. If we wanted to instruct Dancer 3 how to prepare for the second dance, we could tell her "go to the spot that Dancer 3 was in in the first dance." It's a little verbose, but it works!

More generally, we can move from the first dance to the second by telling each dancer whose spot they should take. This method of rearranging a group of things (here, people, but in principle they could be anything) by indicating how each item takes on a position previously held by some item is called a ***permutation***, which is at the heart of this problem.

To make this rigorous, let's introduce some notation. First, for any natural number k, let's have

$$[\![k]\!] = \{ \, n \in \mathbb{N} \mid n < k \, \}.$$

In other words, $[\![k]\!]$ is the set of all natural numbers less than k. The people in our dance can be represented as $[\![10]\!]$. Next, let's think of how everyone swaps around. This is something we can model as a function that takes as input a person, then outputs which person's position they should move to. In our case, we could represent this as a function $f : [\![10]\!] \to [\![10]\!]$. For example, we'd have $f(0) = 5$.

i. Just to make sure everything makes sense at this point, what is $f(6)$?

Formally speaking, a ***permutation*** of a collection of items A is a bijection $\sigma : A \to A$ from A to itself. (That's a lower-case Greek letter sigma, by the way, in case you haven't encountered it before.) There's a good reason this definition says a permutation is a *bijection*, rather than just a plain old *function*.

ii. Let's go back to our example of dancers changing places. Imagine that there's a dance where the dancers start off in some initial configuration. To set up for the next dance, each dancer moves to the spot previously occupied by one of the dancers, and after everyone has set up all positions are filled. If we model that change as a function as we did here, explain why that function must be a bijection. No formal proof is necessary, but you should address the rigorous definition of a bijection in your answer.

It might help to think of things this way: what happens if that function ***isn't*** *a bijection?*

There's a nice notation that's often used to describe permutations called ***two-line notation***. In the top line, we list the objects being permuted in some nice, human-readable order. Then, below each object, we write the object whose position it ends up in after the objects move. For example, the two-line notation

$$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 1 & 3 & 5 & 7 & 9 & 2 & 4 & 6 & 8 & 0 \end{pmatrix}$$

could be read as "Dancer 0 moves to the position previously held by Dancer 1, Dancer 1 moves to the position previously held by Dancer 3, Dancer 2 moves to the position previously held by Dancer 5, etc." This isn't the permutation described in the previous picture; it's just an example of the notation.

iii. Look back at the dances from the previous page. The function f we described earlier tells each dancer whose position to take when setting up for the second dance. Express the function f using two-line notation by filling in the following blanks:

$$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ \_ & \_ & \_ & \_ & \_ & \_ & \_ & \_ & \_ & \_ \end{pmatrix}$$

Now, let's imagine that there's a third dance scheduled and the dancers yet again need to change places. At the end of the first dance, Dancer 0 moved to the spot that Dancer 5 started off in. To make things easier for Dancer 0, imagine that she adopts the following strategy: starting with the second dance, she *always* lines up for the next dance in by moving to the position Dancer 5 held in the prior dance.

iv. Where will Dancer 0 be at the start of the third dance? Provide your answer in the following way: determine which position Dancer 0 will be in, then look back at the original configuration of the dancers and tell us whose position Dancer 0 would be standing in. For example, if Dancer 0 would end up in the outer position in the bottom-right pair, you'd say that Dancer 0 ends in the position originally held by Dancer 7.

Now, imagine that *every* dancer adopts a strategy similar to Dancer 0. Each dancer n is assigned some dancer $f(n)$ that they're tasked with following. For each dance after the first, Dancer n then sets up at the spot where Dancer $f(n)$ was standing at the start of the previous dance.

v. If all the dancers adopt this strategy, where will Dancer 0 end up at the start of the *fourth* dance? Again, express your answer by looking back at the original configuration of dancers from the first dance and telling us whose position Dancer 0 will be occupying.

In parts (iv) and (v) of this problem, you've explored an important idea. We can use the original positions of the dancers as a way of identifying each location. That is, rather than saying "the dancer in the top center position," we can say "the position that Dancer 0 occupies in the first dance."

Let's imagine we want to know where some dancer is going to be in the third dance. We have a permutation f that explains how all the dancers change positions from the first dance to the second. Can we somehow manipulate f to see where everyone ends up for the third dance?

vi. Suppose you pick Dancer n and want to figure out where he starts in the third dance. Explain why he will be in the spot originally occupied by person $(f \circ f)(n)$ in the first dance. No formal proof is necessary.

vii. Starting with your answer from part (iii) of this problem, write out the permutation $f \circ f$ using two-line notation by filling in the following blanks:

$$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ \_ & \_ & \_ & \_ & \_ & \_ & \_ & \_ & \_ & \_ \end{pmatrix}$$

Permutations are important objects in mathematics. We'll see them later this quarter when exploring how they pin down abstract concepts like "symmetry" in a geometric sense.

Problem Seven: Left, Right, and True Inverses

In lecture, we briefly touched on the idea of inverse functions. It turns out that the notion of what an inverse function is is a bit more nuanced than it appears. Specifically, there are several different notions of what an inverse can be, each of which behaves in a slightly different way. This question explores three different notions of inverse functions, along with their properties.

Let $f : A \to B$ be a function. A function $g : B \to A$ is called a ***left inverse*** of f if the following is true:

$$\forall a \in A.\ g(f(a)) = a.$$

i. Find examples of a function f and two *different* functions g and h such that both g and h are left inverses of f. This shows that left inverses don't have to be unique. (Two functions g and h are different if there is some x where $g(x) \neq h(x)$.) Express your answer by drawing pictures along the lines of what we did in class: draw ovals representing the sets A and B, add dots to those ovals to denote their elements, then express f, g, and h by drawing arrows between those dots.

If you draw A and B as sets, then arrows from A to B represent applying the function f, and arrows from B back to A represent applying the function g. So look back at what you found when you expanded out the definition. Can you express that in terms of arrows going left and right between these sets?

ii. Prove that if f is a function that has a left inverse, then f is injective.

As a hint on this problem, look back at the proofs we did with injections in lecture. To prove that a function is an injection, what should you assume about that function, and what will you end up proving about it?

Let $f : A \to B$ be a function. A function $g : B \to A$ is called a ***right inverse*** of f if the following is true:

$$\forall b \in B.\ f(g(b)) = b.$$

iii. Find examples of a function f and two different functions g and h such that both g and h are right inverses of f. This shows that right inverses don't have to be unique. As in part (i), express your answer by drawing pictures of f, g, and h along the lines of what we did in lecture.

iv. Prove that if f is a function that has a right inverse, then f is surjective.

If $f : A \to B$ is a function, then a ***true inverse*** (often just called an ***inverse***) of f is a function g that's simultaneously a left and right inverse of f. In parts (i) and (iii) of this problem you saw that functions can have several different left inverses or right inverses. However, a function can only have a single true inverse.

v. Prove that if $f : A \to B$ is a function and both $g_1 : B \to A$ and $g_2 : B \to A$ are inverses of f, then $g_1(b) = g_2(b)$ for all $b \in B$.

vi. Explain why your proof from part (v) doesn't work if g_1 and g_2 are just *left* inverses of f, not full inverses. Be specific – you should point at a claim in your proof that is no longer true in this case.

vii. Explain why your proof from part (v) doesn't work if g_1 and g_2 are just *right* inverses of f, not full inverses. Be specific – you should point at a claim in your proof that is no longer true in this case.

Left and right inverses have some surprising applications. We'll see one of them next week!

We've included two optional fun problems for this problem set. Feel free to work through all of them, but please submit at most one of them for credit. If you submit answers to more than one, we won't have the bandwidth to grade all your answers and will just pick one arbitrarily. (Here, by "arbitrarily," we mean "based on a whim and without any deeper reason," as in "the CEO made her decisions arbitrarily and capriciously, much to the chagrin of her underlings.")

Optional Fun Problem One: Semilattices (Extra Credit)

A preorder R over a set A is called a ***semilattice*** if it satisfies the following property:

$$\forall a \in A.\ \forall b \in A.\ \exists c \in A.\ (cRa \land cRb \land \forall d \in A.\ (dRa \land dRb \to dRc)).$$

A semilattice R over a set A is called a ***full lattice*** if it also satisfies this property:

$$\forall a \in A.\ \forall b \in A.\ \exists c \in A.\ (aRc \land bRc \land \forall d \in A.\ (aRd \land bRd \to cRd)).$$

Give an example of a semilattice that is not a full lattice, then prove that your relation has the required properties. (That is, you'll need to prove it's a preorder, prove that it does have the first property, and prove that it doesn't have the second property.)

Optional Fun Problem Two: Infinity Minus Two (Extra Credit)

Let $[0, 1]$ denote the set $\{ x \in \mathbb{R} \mid 0 \leq x \leq 1 \}$ and $(0, 1)$ denote the set $\{ x \in \mathbb{R} \mid 0 < x < 1 \}$. That is, the set $[0, 1]$ is the set of all real numbers between 0 and 1, *inclusive*, and the set $(0, 1)$ is the set of all real numbers between 0 and 1, *exclusive*. These sets differ only in that the set $[0, 1]$ includes 0 and 1 and the set $(0, 1)$ excludes 0 and 1.

Give the definition of bijection $f : [0, 1] \to (0, 1)$ via an explicit rule (i.e. writing out $f(x) =$ ___________ or defining f via a piecewise function), then prove that your function is a bijection.