Proof Tips

Writing proofs is hard. Sooner or later, you will be trying to write a proof, and just feel stuck, not seeing what to do next.

The *least* effective way to get unstuck is to stare at your half-written proof and await inspiration. A better idea is to ask yourself questions — narrow down why you are stuck or try to simplify or reframe the problem so you can get unstuck more easily.

Think of getting unstuck on a proof as like debugging your malfunctioning code. The first time you had a bug, you probably stared at your code hoping to find the mistake (and then maybe tried changing < to <= at random – at least that's what I did...). But as you ran into more complicated bugs, that eventually stopped working, so you learned to add `println`s and use a debugger to make the bug-finding process easier.

These techniques are some of the `println`s and debuggers of proof-writing. They may not work perfectly right away, but they can help speed up the process. If you feel stuck on a problem, try these questions yourself; the TAs will use many of these same questions in office hours, so trying yourself first will be more convenient.

1. Remember: You're a Person

Writing a proof can be a difficult process, sometimes requiring creativity. It's hard to be creative if you're hungry, tired, or frustrated. Don't be a afraid to take a break. Put 311 away and do something else for an hour. You might find that the back of your brain figured it out while you were taking a break. And even if you still feel stuck, you'll be much less frustrated.

2. Understand the Claim

Think of this as going back and rereading the spec of your programming assignments. Do you have a sense of why the claim might be true? Are you trying to prove what you're supposed to be proving? Can you put the claim in your own words? It's not impossible to write a proof if the answer to these questions is "no," but it is a lot harder. Pause and think just about the claim (not trying the prove it) and see if you can get more concrete answers.
Another good technique is to translate the claim into symbolic logic (or from symbolic logic into English sentences) – it may be that after doing that translation you realize you've been trying to prove something harder than what we're actually asking!

3. Where Are We?

What have you figured out so far? Have you used all of our assumptions – if not, try to use one of those unused assumptions, see how it might connect to what you have. In fact, you should often start your proof draft by writing all of your assumptions so you don't forget them. Try to concretely state what you know so far, what your target is, and see if there's a step you can take to get closer.

4. Stay on Target

Your proof probably has a target: if you are trying to prove $p \to q$ then your target is q. If you are showing $x \equiv y$, then your target is y. Usually we start with p or x and try to transform it until we get our target. It's easy to "forget" about what your target is and just apply random algebra rules to your current expression – try to minimize the number of times you blindly apply rules. It's a bit like changing < to <= in your code until more tests pass. It might work eventually, but it'll take a while and won't be a fun trip.

Whenever you're doing a proof, keep your target toward the front of your mind – might this new step get us closer to our target? Sometimes you might need to take a step "back" to take "two forward", so you definitely do not need to immediately see why a new expression is better, but if you are just trying random rule after random rule you won't get anywhere fast.

You can also try working backwards start at the end (from y or q) and ask "what could get me here in a single step."

Be careful about the direction. If you have a theorem $q \to q'$ then you should not make q' your new target (showing q' doesn't get you q); you can make q' your target if $q' \to q$.

5. Try an Example

Examples are **not** proofs [1], but they can help you figure out how to prove something. If your claim is true for all even integers, try verifying the claim for 2 and 86 and 2002. If you are proving an algorithm is correct, run it on a few sample inputs. You should be trying a variety of possible values. Go through the examples slowly – is there a common piece of algebra you are doing every time in your calculations? Does some local variable in your algorithm only store even numbers? Maybe that can become a piece of your final proof.

6. Try a Different Proof Technique

Some implications are **much** easier to prove by contrapositive than with a direct proof for the original statement. Some statements are much easier to prove by contradiction than directly. If you have been banging your head on a problem for a while it may help to put your work away and start from scratch with a different technique.

7. Check Your Notes!

It might be that you don't know how to do the proof because the one we're expecting builds off of something from lecture or section – was there a proof on a similar topic that we did in lecture? Did you understand it fully? If not, ask for help! If you did understand it, maybe see if you can use it as a model; does where you're stuck look like where we were in another proof? What did we do in that proof?

[1] Well, except if your claim is "there exists"