My Tips for being an Effective CS TA

Naomi Long, `15

Fall 2013

Job basics

- Dress decently – no sweats, no workout clothes
- Be on time – people who come early will leave if you're late
- Write your name, class your TAing for and hours on board when you come in
- If you're going to miss a session, email your professor as soon as possible so someone can cover your shift
- Don't talk badly about the teacher you're TAing for – okay to say that you struggled with assignment X, not okay to talk about how incompetent they were at explaining assignment X
- Work in the same room students are in – they'll be too afraid to bother you and won't ask you questions
- Do the week's assignment before coming in – good coding practice, different professors have different assignments even if it's the same course you took
- Don't stay too late – if your hours are up, even if there are students left, go home
- It's okay to not know all the answers, just refer the student to the professor

Being welcoming

- Say hi to students as they enter
- Make sure everyone knows you're the TA – if a student can't find you in the chaos, they might give up and leave
- Never make a student feel afraid to ask you a question – even if they've interrupted you, don't grimace; always be ready to devote your full attention to the student
- Don't bring hard homework or something that involves focus for the slow times, because you don't want to be frustrated if you get interrupted
- Try not to work too many hours in a row if you're going to be exhausted and unwelcoming by the end of it – pick different shifts

Question answering and honor code

- Honor code violation to give large chunks of code or problem set answers outright – if student asks if they're right, can ask them for their reasoning and tell them if it seems sound instead
- If you ever have a grey area with the honor code, email the professor – they'll be glad you asked

Question answering

- Make sure you know what the student's actually asking – very easy to answer a question the student didn't ask, and oftentimes, the student isn't totally sure what they're asking either
- Factual questions deserve short answers – "Does Python 3 require parenthesis with print statements?" doesn't need a lecture, just a "Yes"
- Try to lead students to answers rather than just giving them – ask them questions as you go along, allowing them to reach the answer on your own (give an example here)
- When asking students a question, give them at least TEN WHOLE SECONDS to think of a response – don't rush them! And if they need more time, ask if they want you to come back around later once they've had time to think about it
- Don't lecture – keep explanations relatively short or refer them to a professor
- When explaining something, check in every few sentences to make sure the student's still with you – you'd be surprised how often they aren't
- Avoid grabbing the keyboard – tell the student what you want them to type instead, keeps them more engaged

My Tips for being an Effective CS TA (long version)

Naomi Long, `15
(Edited by D. Thiebaut)
Fall 2013

PART 1 – JOB FUNDAMENTALS

1a. *The basics*

Dress decently. You don't need to pull out the office wear but showing up in your workout clothes covered in sweat is disrespectful and kind of gross.

Write your name, class you're TAing for and hours you'll be working on the board when you come in. It helps students know they got the right room.

Be on time, preferably 5 minutes early. It's a matter of respect, and students who show up early to see you will be frustrated if you weren't there; they might even leave without asking any questions! The extra five minutes gives you time to turn on the lights, write your name on the board, get some water and pull up the assignment on your computer.

If you're going to miss a session, ask people to cover your shift as far in advance as possible. A hastily sent email written 30 minutes before your shift

probably won't reach anyone in time for them to cover you. Try and ask at least a few days in advance if you know you're going to have to miss work. If something happens suddenly, try to get in contact with your professor, perhaps by calling them, so they can email students and let them know that TA hours for that night are cancelled.

Don't talk badly about the teacher you're TAing for. Even if you don't like him or her or think they're a bad teacher, now's not the time to talk about it. Feel free to agree with the student if you struggled with similar things when you took the course, but you should never say things like "Isn't Professor X such a bad lecturer?" or "The instructions for this homework assignment are horribly written. No wonder you're so confused!" Remember that this is a job; you're there in a semi-professional capacity.

Work in the same room that the students are in. You may need to use the computer in the next room to finish an assignment that's due tomorrow, but while you're TAing is not the time to do this. Students will feel uncomfortable bothering you if you're in the other room clearly working on another assignment. It's disrespectful and essentially amounts to avoiding doing your job.

At the very least, look over the week's assignment *before* coming in. Even if you've taken the class before, different professors often use their own set of assignments. If this is your first time TAing for a professor, I'd recommend doing the whole assignment over as quickly as you can. You'll be a much better TA for it, and it's good coding practice, too.

Ask the professor questions if you don't get the instructions *beforehand*. This goes hand in hand with looking over the assignment before you come in. If you're confused, students will probably be too, and you'll need to give the professor time to respond to your questions.

Know how to say "no." Don't stay an hour or two after your hours are up just because it's the night before the assignment – you have homework too! Point the students towards a professor's office hours instead. Even if you don't have homework and want to stay afterward, make sure you make it clear you're doing this on your own time and not as a TA, or you'll set a precedent that students might expect other TAs to follow. Don't assume you'll get paid for overtime; the department sometimes has a weekly TA budget limit they won't be able to go over (but ask anyway, just in case).

Don't be afraid to ask for help from the other TAs or the professor. Being a TA means you're part of a community; you don't have to go at this alone. If you're not sure how to answer a question, no problem: call another TA over or shoot the professor an email. If you're afraid to admit your ignorance and try to bungle an answer together, it'll show, and you'll set the example that it's wrong to admit you don't know something – possibly the worst example to set for a student to at TA

hours! And plus, not knowing the answer gives you an excellent opportunity to try and figure it out together with the student. You're not the all-seeing oracle of computer science; you won't always know the answers and that's perfectly okay.

1b. *Your goals as a TA*

To create a safe and comfortable atmosphere for learning and admitting ignorance. This means not making people feel stupid for asking "dumb" questions. This means not alienating students based on their gender, sexual orientation, race, religion, or intelligence. This means creating a welcoming atmosphere and making it clear you're there to help them.

To act as a liaison between the professor and the students. Every so often, let the professor know how the students are doing. Maybe one night the students all have the same kind of questions. Email the professor as soon as possible so they can go over rough spots again in class.

To teach the students how to get the answers on their own. The "teach a person to fish…" proverb applies here. It's good to answer a student's question. It's better to help them figure it out for themselves. Show students your thinking process when you're helping them solve hard problems; ideally, you never want to get the same question from the same person more than once.

To bolster student's confidence in the material. Many times I've seen students ask tough questions they could totally answer on their own - they just didn't think they could. As a TA, you get to act as a cheerleader for these students. Guide them on their way to figuring out the answer while encouraging them. Keep it subtle – don't bring out the pom-poms or anything – but do congratulate them when they've finished a tough problem.

1c. *How often do students come in?*

You might not get any students some nights. This happens usually in the beginning of the semester or if you have a shift the day after a homework is just assigned. Sit back, relax, and get some reading done. If this happens for more than a few weeks in a row, let the professor know – maybe you can change the shift's time so more students will come.

Some nights, there will be so many questions you won't get to sit down at all. This is usually if you have the last shift before the homework is due, especially if it's the night of. If you need to, don't be afraid to tell students you're going to take a 2-minute water break to clear your head. And don't feel obligated to stay after your shift's done, even if there are still a dozen questions in the room – you've got your own homework to do, too.

You'll get about 5-10 or so students regularly. Most others will only come in for a one-time question and that's perfectly okay. Some people like working on the homework in a semi-noisy room with a lot of people, and others need quiet. Some like to set up a routine and come regularly, others don't. So you'll generally get a set of regulars, a few one-timers, and a whole flock when there's a particularly difficult assignment.

1d. *How should I act when students come in?*

Welcome students as they enter. Even if you're answering another question, look up and give them a little wave. A smile and an introduction can go a long way to making a student feel at ease if they're stressed and anxious.

Make it clear you're the TA. Sometimes, on a busy night, students will come in and you'll miss greeting them. They'll look around the room searching for a TA, and if they can't find you in all the chaos, they might give up and leave (I know people who have actually done this). Every so often look around the room and ask if anyone has questions. Introduce yourself to newcomers as the TA and make sure everyone knows who you are.

Occasionally walk around and ask if people have questions. Don't pester someone if they seem to be concentrating hard, though.

Never make a student feel afraid to ask you a question. Look up and smile, even if you've been interrupted from an especially funny tumblr post. Make it clear that you're happy you've been asked a question and you're excited to answer it for them. You're there to be a TA, not surf the internet. If students feel like they're interrupting you if they ask you questions, they'll just stop asking.

Related – don't work on hard homework while TAing. On most nights, you're going to be constantly interrupted with questions. It's nice to bring a book to read when things are slow, but don't bring anything that's going to require a lot of focus. If you do, you'll end up irritated when you're interrupted, and it'll show. You've got to be ready to devote all of your focus to students when they want you.

PART 2 – GENERAL QUESTION-ANSWERING

2a. *TAing and the honor code*

It's an honor code violation to give out answers directly. You are not allowed to tell students the answers to quizzes or problem sets or to give them large chunks of code for any assignment. If a student pesters you for answers, tell them this. If

there's ever a gray area, email the professor before doing anything. They'll appreciate you asking.

Sample grey area:

A student is doing a multiple-choice question from the textbook and is confused about whether the answer is a or b. I know the answer is a, but I don't tell her this. Instead, I ask her to tell me which answer she thinks it is and why. She explains to me why she thinks it's b. I point out a few flaws in her reasoning. She reconsiders, and then explains to me a new chain of reasoning why the answer is probably a. I tell her that her arguments seem sound to me.

Another grey area:

A student doesn't know where the bug is in her program, but it won't even run. I look over at her code and notice that the indentation is messed up. Instead of grabbing the keyboard and fixing it for her, I'll tell her to check her program's indentation. Then, after she's done, I'll tell her how I knew from the error message that the indentation was the problem.

2b. *The general template for question answering*

One of the most important things to remember is that **TAing is ultimately a conversation, not a lecture.** Below, I'll walk through the typical flow of a question-and-answer session between a TA and a student and show how I like to lead students to the answers. This is for questions that aren't just simple fact questions that only need one word answers.

1) Student asks a question.

2) TA rephrases the question back to the student to confirm that that's what the student wants to know, or asks questions about what the student is confused about.

3) Student and TA work together to figure out what the student's asking if there's confusion (which there often is). Here, you'll sometimes end up with an entirely different question than the one the student asked. That's totally okay.

4) TA starts answering the question and stops a few sentences in to ask if the student understands. Don't go into lecture mode. You want to constantly check in when explaining to make sure the student is still with you.

5) If the student doesn't understand, the TA explains it again *in a different way* or works through an example. Don't just try and beat the same explanation into their head! If the student does understand, then the above step is repeated, with maybe

the TA throwing in a few leading questions for good measure. Remember that you want to lead them so that they find the answer for themselves.

6) When the problem has been fixed or the question has been answered, the TA asks if the student's confusion is resolved and asks if the student has any more questions.

7) If the question hasn't been answered satisfactorily (and hopefully, the student will feel comfortable enough to admit if this is true!) then the TA can either go back to step 2, leave the student some time alone to think about the explanation (sometimes, you just need to think about it for a little while), or point them towards another resource (another TA, a good website, the professor's office hours). Otherwise, the TA moves on to another student.

2c. *Question answering tips*

Don't be condescending when answering "stupid" questions. Imagine you're stressed, frustrated, tired, and insecure about your knowledge while working on an assignment. Then imagine asking a TA a question and being made to feel stupid for asking it. Are you going to go back for more TA sessions? No. So don't do this!

If a student asks a question you feel they should definitely know (like, on the intro to programming final, they're still wondering how an if statement works), point them to a professor to get their misconceptions cleared up. Or take some time to sit down and let them ask you questions. **Just don't say things like "Oh, that's easy!" or "Shouldn't you already know that?"** If the student struggles with what you've just said is trivial, you've now made it so they can't ask you questions without admitting that they struggle with something you find easy. And if a student feels like that, usually they stop asking you any questions at all.

Distribute time between students as evenly as possible. If you're taking awhile answering one student and others are waiting, tell the person you'll come back to them after you've answered a few other questions. You have to know how to prioritize or else a lot of students will just give up and leave, frustrated that their questions aren't getting answered.

Tell the student you'll come back and go around the room, answering simple fact questions (like "What version of Python are we using?" or "What's the URL for the homework assignment?") first. After that, tackle some of the harder questions of the other students. If an answer is going to take a *really* long time, tell them to go ask the professor about it.

Remember that you don't have to sit with them through the entire answering process. As I cover in part three, it's perfectly fine to ask them a question ("Okay, so we know the bug's in this function and it has something to do with a *NameError*. Where could it be?") they haven't thought about yet and leave them to think about it.

As a TA, you don't have to and shouldn't give out all the answers up front. It's usually better to point the student in the right direction at this point and come back to them later, when they've had time to tackle it on their own.

Make sure you're actually answering the student's question. Don't you hate it when you ask a professor a question and they end up answering something you didn't ask? We want to avoid doing this ourselves, and it's surprisingly easy to fall into the trap. The trick to making sure you're answering the right question is to 1) **make the student feel comfortable** and 2) **ask them for feedback frequently**.

Make them feel comfortable. You know when you ask a professor a question and they end up answering something you didn't ask, but you feel too uncomfortable to interrupt them? Or when they give you an answer you don't understand but you don't want to look stupid so you don't ask them to repeat it? We want to avoid that. Reassure your students that you're there to help them and they can take as long as they need (and if it's too long, you'll come back to them, or point them to a professor's office hours for help). If a student feels like they're wasting your time or that you're going into lecture mode, they'll stay quiet and stay confused.

Ask for feedback often. If the student seems a little confused or their question is kind of vauge, rephrase it back to them to make sure you know what you're asking. Oftentimes, the student isn't even sure what they're asking and the two of you will need to figure it out together. Once you start explaining, check in to make sure what you're saying is relevant. You'd be surprised at how easy it is to answer a question the student never asked.

PART 3 – LEADING STUDENTS TO ANSWERS

3a. *Why we lead instead of give*

Even with honor code issues aside, **leading a student to an answer teaches them much more than just giving it to them**. The whole "teach a person to fish…" proverb seriously applies here. Finding a tough bug for a student is okay. Teaching the student how you work to find tough bugs is much, much better.

Teachers have kind of a paradoxical role – we want to make ourselves unneeded. We want to train our students to be able to do the things that we do so that we become obsolete. If you simply give students the answers, you're not teaching them how to find them on their own. They'll have to continually rely on you to finish projects, and you end up being a crutch for the student to avoid having to actually tackle the root of their misunderstandings.

When a student discovers the answer with your help, you've taught them not just what the answer is but how to find it for themselves in similar situations. You've helped them get a little bit smarter and a little bit more self-reliant. The opposite is true if you just straight-out tell them the answer.

3b. *How to lead students to answers*

First, know when not to lead. If a student asks you a simple fact question, don't try and bamboozle them into figuring out the answer for themselves. Just tell them. Questions like "Which Python version are we using for this assignment?" or "What's the command for moving a file again?" deserve a single sentence as an answer. Anything else is a waste of their time.

Ask good leading questions. A good leading question gives the student a new direction to think in without directly putting them right where the answer is. You want them to find their way to the answer, and if they can't do that with you standing over their shoulder, to move on and give them the time necessary to do it without you.

If a student comes to me and doesn't know where the bug in her program is, instead of just pointing it out, I'll tell them that I noticed something was weird with their variable names in function X. Or I'll ask them if they tried printing out the values of the variables before this point, to see if they implemented the algorithm right. I really like to encourage students to add a lot of print statements to their programs to see where things could be going wrong.

Notice how this *isn't* in the example of good leading questions: "Well, where do *you* think the bug is?" This isn't a helpful leading question because I didn't give them a new direction to go in. They called me over because they were stuck. If I don't give them a new way to go, they'll just end up frustrated.

Give them time to think about the answers. After you ask a student a leading question, wait at least TEN WHOLE SECONDS for them to answer. Fifteen is even better. Seriously. Count it out in your head. Let the student know you're comfortable with them taking their time and figuring it out.

If they're taking awhile and it's a big question, maybe ask them if they want you to come back once they've had more time to think. This is really important. If you want to help students learn to do the things you do, they need to practice thinking it through on their own – just like you had to. And if you rush them and make them feel uncomfortable, you'll end up giving them the answers instead of letting them find the answers for themselves. Take a deep breath and be patient.

Talk them through your thinking process. If you notice where the bug is, don't just point it out and say "It's over there." Explain *how* you know it's over there. Walk

them through the reasoning you used to figure it out. If I'm not sure where the bug is and I need to add print statements to their program to understand what's happening, I won't just grab the keyboard and do it. I'll tell them what to add and *why* I want them to add it. I'll tell them what I'm looking for.

Don't lecture. TAing is a dialogue, not a monologue. You're not the professor and you're not supposed to be. Any time you're giving a long explanation, every few sentences check in to see if the student understands. Go slowly and repeat what you just said at multiple points unless it seems obvious the student gets what you're saying. Ask them leading questions so you know that they know where to go from here and aren't just following you blindly. A little alarm bell should ring in your head any time you've talked for more than 15 seconds without a break.

Avoid grabbing the keyboard. Whenever you want to demonstrate something, try to tell the student what you want demonstrated and let them do it themselves as much as possible. This gives the student less chance to zone out and just let you figure out the problem without learning how to do it on their own. Sometimes it's okay to grab the keyboard and show them something complicated, but it's better to tell them what to type and let them type it.