Being An Effective TA

Department of Computer Science
Williams College
{these slides available at csci.williams.edu}
Assistants Wanted

for hazardous journey, small wages, bitter cold, long months of complete darkness, constant danger, safe return doubtful, honor and recognition in case of success.

http://csci.williams.edu/tatutor-application/
Your Job (Remember: you’re being paid)

- Answer student questions about concepts and assignments
- Help students learn how to solve problems
- Provide feedback to instruction staff
- Be responsible and responsive
What to Expect

- Significant contact with a group of regular attendees
- One-off questions from occasional students
- Hard assignments lead to busy TA sessions
- Shifts will be especially busy just before homework is due
- Could be non-stop or...nothing
  - Let your instructor(s) know if no one is coming
The Basics (of any job)

• Be well-rested

• Dress decently
  - Not like you came in straight from the gym

• Be on time (or a little early)

• Write name, class, hours on board

• Walk through assignment, yourself, beforehand

• Don’t feel obligated to stay after hours
  • Inform your instructor(s) if you’re regularly overrun
Professionalism

• Always be respectful:
  - Make effort to learn names; be friendly
  - Don’t judge student abilities; learning is hard
  - Effective help requires observing learning habits
  - Avoid micro-aggressions

• Act professionally:
  - Don’t belittle the faculty, TA’s, students
    - It reflects on you
  - Don’t disparage assignments or approaches
    - Instructors would rather you provide helpful feedback

• You represent the Department and the College
Setting the Right Tone

• Acknowledges students as they enter and leave
• Make sure everyone knows you’re the TA
• Be approachable:
  - Avoid your own work when the lab gets busy
  - Don't work with others
  - Don't chat it up (in person or on phone...)
  - Don't take a nap
  - Treat all students equally (including friends)
• Your job during TA hours is to be a TA.
Answering Questions

• Listen first:
  - What do they need?
  - Can you understand the real problem?
  - Do you understand their viewpoint?
  - Be reassuring.

• Do not give out direct answers. Don’t touch keyboards. Instead:
  - Think about milestones or mini-accomplishments
  - Develop a debugging approach / help them organize their thought process
  - Let students do the reasoning, writing, discovery

• Applaud:
  - Successes, when they occur
  - Strong efforts, even when solutions are elusive

• You don’t have to have all the answers:
  - Hand off to another TA or instructor
  - Do not answer questions for other classes
Honor Code (Really Important!)

• It applies to TA’s too:
  • Essentially: All student work must be their own
  • You are simply a catalyst

• Discuss course/assignment particulars with instructor

• Do not answer questions for other classes; you may not know the rules.

• Any gray areas, ask!
Three Kinds of Questions

• Simple:
  - “Which version of Python are we using?”

• Specific:
  - “How do I print out the elements in this list?”

• Vague:
  - “Why doesn't my program work?”
Simple Questions

• “Which version of Python are we using?”
• “What does this compiler error mean?”
• “How do you do search-and-replace in emacs?”
• “What parameters does the constructor for FilledRect take?”

• Give short answers AND...
  - show them how to find answer themselves
  - tell instructor if same questions come up a lot
Specific Questions

• “How do I print out the elements in this list?”
• Don’t lecture - keep things brief
• Check in every few sentences as you talk
• Let them do the hard work
  - Keep hands off mouse and keyboard
• It’s ok to refer them to a book or lecture example if appropriate
• Walk away as necessary, but come back later and check in
Vague Questions

• “Why isn’t my program working?”

• Guide them through the process of making a solution:
  - “What did you expect it to do?”
  - “What do you believe is actually happening?”
  - “How can you test that hypothesis?”
  - “What may be causing this difference in behavior?”
  - “How do you change it?”
Other Guiding Questions

• What does this error say about the program?
• Where are the mostly likely places for a bug?
• What’s happening in this code block? Walk me through it.
• What do you think will happen if we change this?
• What value will this variable hold here?
• Explore the bug together while verbalizing your problem-solving process
• Reassure students on their success.
If a Student is Still Having Trouble…

• Ask a specific, leading question
• Wait at least 5 – 10 seconds for response! (Yes, it seems an awfully long time while standing there...)
• Try another explanation
• Ask what they’re confused about
• Give them time to work through it on their own
  • walk away, assure them you’ll return soon
• Refer them to another TA or the professor
• Teaching is a challenge: Reflect on what worked / didn’t work for you
Time For a Break?

• Disorganized debugging effort?
• Randomized coding/theorem proving
• Glassy-Eyed Vacant Look
• Encourage students to:
  - take a break, have a snack, go for a walk, ...
  - print their work & work away
  - leave and return with "fresh eyes"
  - think about intuition/ideas behind problem, not details
Teach Good Coding (Reasoning) Practices

• Comment on style
• Plan out code (proof) before writing it
• Name variables (motivate claims) well
• Organize into functions (lemmas)
• Suggest incremental approaches & milestones

• I often make a mental note of these items and only suggest they fix them *after* they have solved their immediate problem.
  - If it's incoherent, suggest a fresh approach...
Some of you are Graders:

- Be legible (use pencil?)
- Be consistent, reasonable and fair
- Ask instructor if rubric is unclear
- No snide, sarcastic, joking, flip comments. Ever.
- **Complete on schedule**
- Include positive feedback, too.
- Bottom line: students spent worked hard on their solution. Give their work the attention it deserves.
Treat Students Respectfully

• Be careful with assumptions about ability
• Never say “that’s easy”
• Cheerleading

If given enough time, they’ll get it. Everyone here is smart enough. Your efforts will likely make someone a computer scientist!
TA Effectiveness Feedback

• Mid-semester we’ll evaluate TA effectiveness:
  • Students will fill out a quick survey in labs about their experience in TA sessions
  • Questions seek anonymous feedback
  • Faculty will review results with TA teams to improve, if necessary, TA effectiveness
• Generally: Students seem quite appreciative of help from TAs!
Today

- TAs: Meet with Instructor, organize TA hours, etc.

- Faculty: Take pictures for TA posters and email to Lauren today or tomorrow.