# Providing Feedback on Student Work

There are 56 students this year, meaning you can spend about 15 minutes on each submission without going over your working hours limit. For programming assignments, focus on:

* Fundamentals of good code style. There is no need to picky, the goal is to write code that is easy for you as the grader to read.
* Did they make the task harder than necessary? For example, did they reimplement a feature provided by the libraries?
* If they did something nice beyond what we ask for, leave a kind comment. For example, some students might use higher-order functions to make code cleaner when it is not required.
* If they are way off (for example, the assignment asks to write a recursive function over syntax trees, but they did not use recursion), try to provide a specific suggestion about what they should study or review (specific sections of the book or slides)

For written assignments (which include students leaving comments on each other’s work), focus on:

* Are they on track to be able to run a small user study on their classmates toward the end of the course?
  + Lectures on study design start halfway through the term, so their early work can be high-level and this is okay
  + At the same time, as course staff, you can help ask questions that will lead them toward working out the details of their user study
* Are they taking the peer review / comment task seriously? In any class with required discussion, it is common for students to make boring comments just to get the credit. If you see this happening, leave a comment encouraging them to provide more feedback, so they know we do care. They are expected to ask at least one specific question to each other or say at least one specific thing to focus on.
* Encourage them to have fun with the written assignment, show their own creativity, and embrace their own interests. This is why we use completion grading for written assignments.

If there are major red flags for cheating, privately contact me about the red flags. Continue grading the homework normally. Don’t spend too much time worrying about cheating and don’t make accusations toward a student since that’s the instructor’s job. Here are examples of potential red flags, though context matters a lot:

* Submitting code that is in the wrong language or relies heavily on strange libraries or syntax that are not discussed in class. This sometimes happens with ChatGPT-generated homework submissions as well as answers looked up online. For example, one of our assignments is to write a parser. If they use a completely different parser library without permission, it’s suspicious.
* Submitting solutions to a previous year’s homework if the assignment changed a lot. I only plan on making significant changes to one or two assignments, though.
* Two students submitting code that makes the same very-rare mistake that appears in no other submissions.
* Some instructors insert invisible text in assignment handouts to poison ChatGPT prompts. I have not decided whether I will do this, but if I do, I will let you know what outputs to look for.
* Two students submitting written assignments that are suspiciously similar to each other, but they have not talked to course staff to let us know that they are openly working on related projects and talking to each other.
* For written submissions, ChatGPT is known to write in a specific authoritative tone using needlessly complex language. It’s hard to be sure, but you can flag this for me to check.

The following are not red flags:

* Submitting code that does not compile due to a compilation error that is common among students (like struggling with the type system)
* Two students structuring their code in a similar way to each other or our staff solution. For most of the programming assignments in this course, there are only one or two good approaches, and correct student submissions will be similar.
* Two students making the same mistake, but it’s a common mistake.
* Seeing students working together, as long as they are not taking each other’s solutions.