General Information

Getting in Touch With Me

Office hours: I will have office hours by appointment. If you want to make an appointment, email me 4 times in the next 48 hours and I try to accommodate one of them.

TAs also hold scheduled office hours that you can use if you have any questions.

Piazza: You can post your question on Piazza. You should spend at least 10 or 15 minutes crafting a question there and check if somebody has already asked your question. See "Online Communication" for information on how to post a good question.

Email: This is only for personal communication. If you need an extension, have a question about *your* grade. Any email *title* should be prefaced with CS340. If you posted something on Piazza that has not been answered in 24 hours, you may email me a link to the Piazza post and I will make sure to get to it.

Work Outside of Class Lectures

You will need to research material and work extensively with documentation outside of lectures. I always code with one window open to a language reference or API reference. You should too. You will not be able to complete assignments just by watching the lectures. As much as possible you should try to use official documentation and forums geared towards professional developers.

When Things Go Wrong

When you are having an issue. Spend around an hour trying to solve it. If you have not made progress, then it is time to seek help. By the end of the hour, you should have read relevant documentation and done several searches on sites like StackOverflow. At this point, post to Piazza following the guidelines here. Hopefully you have other things you can do while you wait on a response. Try another part of the assignment, work on another class or do your laundry.

Assignment Expectations

Every programming assignment you turn in needs to be free of errors, warnings, notifications or anything else that indicates there may be an issue. Typically it will be your responsibility to prove this is the case. This means you need to include output from a compiler, validator, console and/or output from a run of the program. The only exceptions are if I specifically say a

warning or notification is OK. For example, this may be the case if a library we are using causes a warning to be thrown. Additionally there needs to be no issues with the code that are not caught by the above mentioned tools.

Do not assume a warning is OK. Ask me before you submit an assignment if I have not already clarified it.

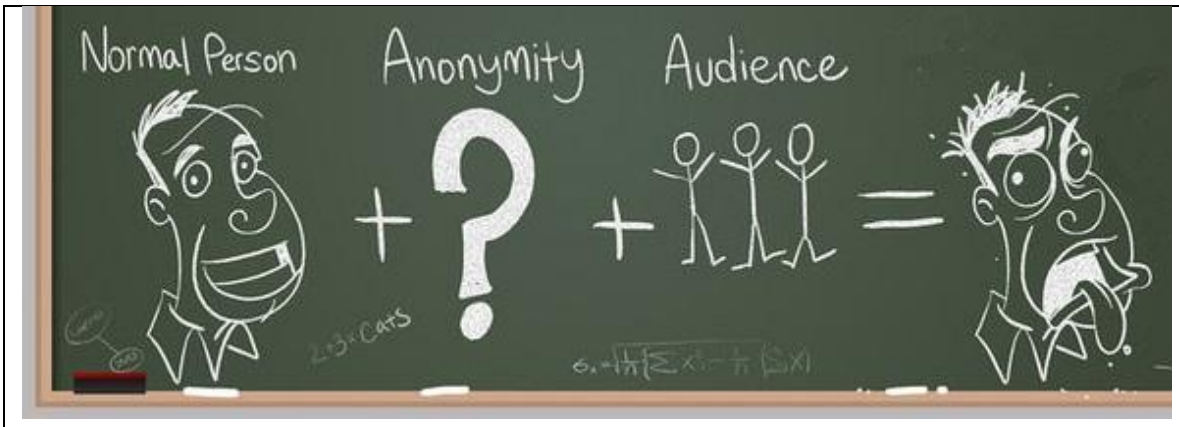
Grading

If you have an issue with a grade, first contact the TA and CC me. They did the actual grading and if there is an error with math or a file was missed they will know better than me how the scoring was worked out.

If you disagree with the number of points taken off, you can ask me to personally regrade the assignment. This means I will be going over your assignment more carefully and I may find more issues than the TA's did. This means your regraded score can be lower than your original score. So ask me to regrade with caution.

Online Communication

Communicating online is significantly different that doing so in real life so there are a few guidelines I want to go over. A popular web-comic, PennyArcade has a good summary of what can happen in online communities, edited to be more appropriate for this context.



We are striving to *avoid* making this equation true.

The basic idea is that if you take a typical, normal, well adjusted person and give them an audience and make them anonymous they can become *different* people who are much less kind and reasonable.

We have *normal people*, we have chat rooms and message boards that give us *an audience* and we are

to some extent anonymous or may at least feel that way. This is a precarious situation.

How to be a good person

- What would you do in real life?
    - If you were having this discussion with someone in real life, what would you say to them? Say that instead of what you were about to say. If you can follow this rule, you will be fine in 95% of the cases.
- What if everything I say is preserved for the rest of time?
    - There are sites like WayBackMachine which archive the internet. Your conversation may be captured there. The chat-rooms are logged, Piazza discussions are archived. When you ask me for a recommendation, I am going to search old classes for your discussions and that is going to go a long way in informing my recommendation. I value the ability to communicate about technical topics as highly if not more highly than your ability to actually implement them as a student. Follow the first rule and you should be fine here.
- Can this be misunderstood?
    - Are you making a joke? Are you being sarcastic? We don't know. But wait! You can help us! Explicitly add a /sarcasm tag in your text and that makes it clear. Could something be conveyed either postivily or negatively? Add a smiley :) or a angry person >:(to let us know what you mean. (But refer to the first two rules before you use the angry person) Really, emoticons help add context. Use them if they are appropriate.

How to ask good questions*

- Use a descriptive title
    - [C++] Segmentation fault while writing to array in for-loop As the community grows there will be more and more posts. People may not have time to read every post. A title that lets them know if they are having the same issue or can help with your problem can save a lot of time.
    - Sometimes the content of a post may change because you realize the problem you thought you were having is not the actual problem at all. Rename your thread or ask a TA or instructor to rename it for you if you can't.
- Don't spawn quasi related sub-questions in the same thread
    - If you are reading a post and realize you have a sort of related issue, make a new post about it and post a link to it in the original thread. This helps keeps things organized and titles relevant to the content in the post.
- Please put in genuine and significant effort into solving a problem for yourself first before posting it

    - This accomplishes two things. First it gives you experience researching and solving problems on your own. This is a valuable skill and is faster than having to wait for someone to answer your question. Second it gives you the information to actually post a well thought out question.

- Wikipedia has an excellent summary of how to ask a good software question

    - In brief you should have
        - A goal - What it is you are actually trying to accomplish in the slightly bigger picture
        - The problem - What specifically is going wrong, what is happening right before? When was the last time it worked? StackOverflow has a good description of what happens when you confuse the problem with the goal.
        - An example - Show us the relevant code. We don't need to see your whole project but we need some context to understand the possible causes

* much of this content was pulled from the LearnProgramming subreddit

How to give good answers

- Follow the rules of being a good person up above

    - Remember you are dealing with someone who may be stressed because they have spent all day trying to fix something and their project is due in an hour. They may not be polite but you still should be.

- Make sure you understand the question

    - Don't just guess at what is going on. If it is not clear, ask clarifying questions

- If you post code to help, make sure it works

    - Don't post bad code as a solution. If you only want to provide pseudo code, that is fine, but make it clear that it is pseudo code. Don't post a code that wont run, it is just going to make them more frustrated if your help just causes them more errors when they expect it to fix the problem.

- Don't just post code

    - You may be a coding genius, but if someone is asking for help, they want to understand how to fix the problem. Just giving them code that makes it work won't help them learn. Provide an explication and ideally links to places where they can learn more about the problem or solution.