CS2124 OOP

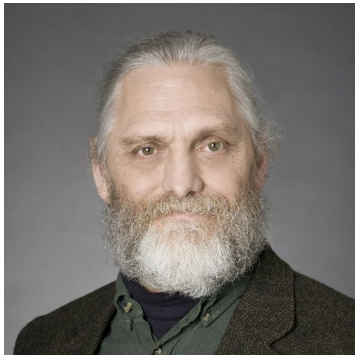
Course Overview
Spring 2024

What is the Course About?

- Writing good, readable code
- Static typing
- OOP: Encapsulation, data hiding, delegation, inheritance, polymorphism...
- Addresses and pointers
- Memory management
- Operator Overloading
- Generic classes and functions
- Functors and lambda expressions
- Recursion! (Again? Yes!)
- STL

Instructors

- John Sterling
  - Office: 845, 370 Jay
  - john.sterling@nyu.edu
- Omar Mansour
  - Office: 844, 370 Jay
  - omansour@nyu.edu



Slides?

- Slides are boring
- When using slides, instructors often just read from them
- We will rarely use them
- However they do allow more to get covered in less time
- And *this* week we have a lot that needs to be covered quickly

Administrivia

- Grade is based on:
  - Labs and Homework: 25%
  - Exams: 75%
    - Midterm: 30%
    - Final exam: 45%
  - Class participation!

Labs / Recitations

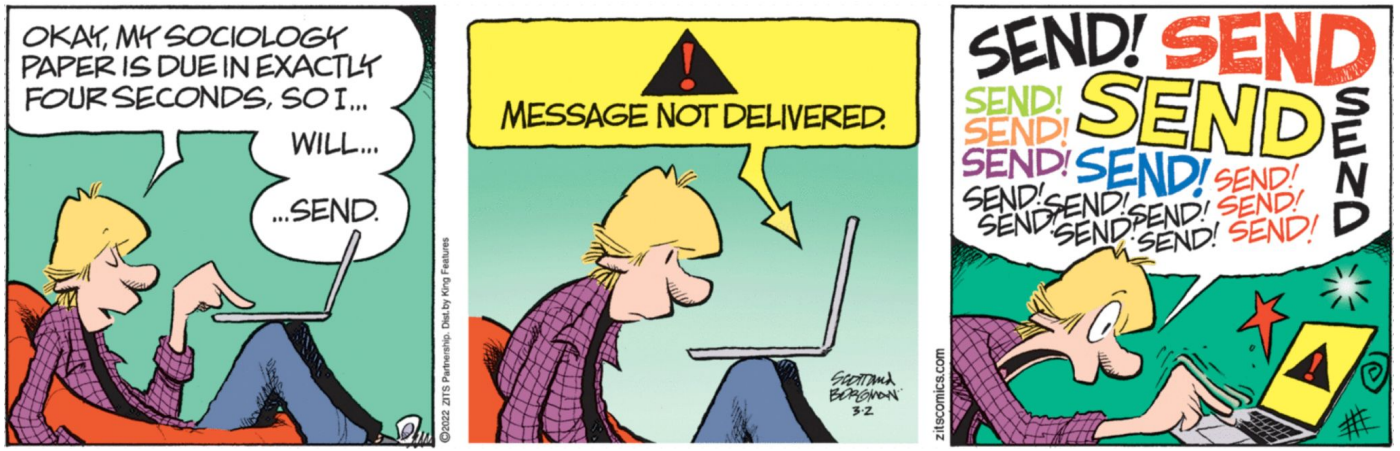
- **Install C++ *before* the first lab!!!**
  - We use the terms “lab” and “recitation” interchangeably
- **All** lab solutions **must be submitted** on Brightspace
- Can be checked off in class for full credit (still *must* submit!)
  - Attendance is mandatory. If you do not attend **or if you leave early without being checked out**, you will not receive credit for the lab.
  - If not checked off in lab, then will be graded when turned in… (scary!)
- Some labs are in the form of a “tutorial”
- Others are a single programming task
- Either way, it is expected that you *should* be able to complete it during lab.
- Labs are expected to be done *in lab*, not ahead of time, but are accepted till the end of the weekend (unless otherwise stated, e.g. rec14)
- Comments are not required in labs, but **good readable code** is.

Homework

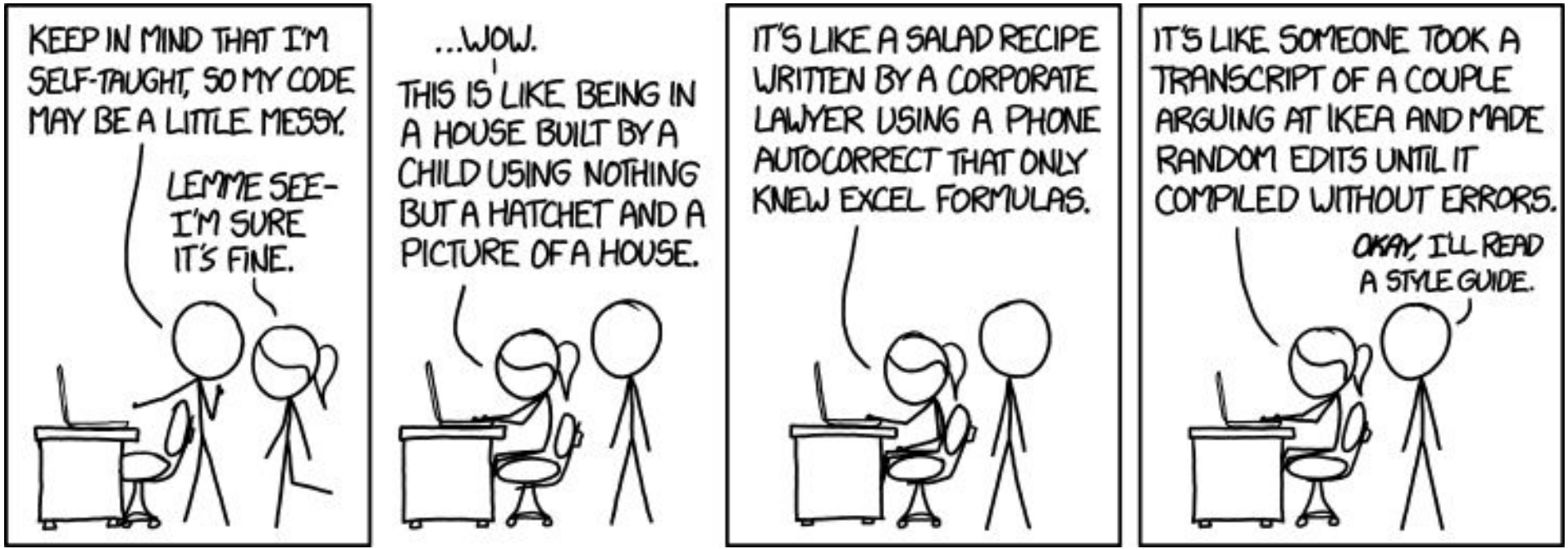
- ~8 assignments for the semester
- Early ones will be typically be due in ~week.
- Later ones my be up to two weeks
- Late assignments (not labs) **are accepted** but with severe **penalty**.
- Code is to be well-written
  - **Well commented**
  - **Good naming** for variables, types and functions
  - **Good *use* of functions**
  - Avoid long functions
  - No long lines. **80 characters max / line**, even with comments
- Do not use features / libraries that have not yet been covered in class
- **All code must be your own**.
  - Don't copy. Don't look at other's code. Don't share yours with others.

Late submissions

- Homework is accepted late but with a penalty. See the syllabus for details.
- Labs are expected to be done during class
  - But are accepted without penalty till the end of the weekend.
  - Later than that? They get a zero.



Effects of “messy” code



Exams

- Consist of approximately 50% / 50% short and long questions
  - May be 60 / 40 or 40 / 60…
- Short answer questions
  - Most commonly, “what happens if we compile and run this code?”
    - Did it build / compile? Did it run without error? If so, what was its output?
    - Some students get confused by the difference between build errors and runtime errors
  - Some “short” answer questions may require a *few* lines of code.
  - I don’t ask for definitions, but if you don’t know the terminology, you may not understand the questions.
    - This is one reason why we correct your choice of words when you ask / answer. In class.
- Long answer questions
  - Programming problems similar to (but shorter than) lab or homework questions
  - Comments are *not* required
  - Nor are “include” or “using” statements. (you’ll know what those are shortly)
  - Write clearly. If I can’t read it, …
- BTW, do not provide two answers for one question.

Class participation

- *Constructive* class participation *may* result in a boost to your grade
  - E.g. from B+ to A-
- Whether or not there is a boost will depend on how far you are from the next cut-off and ***our*** perception of the value of your participation.
  - No, don’t ask us “How far away was I from the cut-off?”

Questions?

- *Please* ask questions in class! Don't be embarrassed!
  - If you have a question, likely others have the same one.
  - We try to keep the lecture size down so that all students should be able to ask questions.
- And please try to answer questions!
  - Again don't be embarrassed if your answer is not correct! We often learn by making mistakes!
- Feel free to come to the office to ask questions!
  - Don't wait till it is too late for us to be able to help
- TAs will have office hours, too.
- Some students want to have their homework "looked at" *before* it is graded.
  - No, we won't do that.
  - We will give guidance on debugging or answer questions about what needs to be done
  - We just won't tell you if your code is good or bad or what sort of grade to expect.
    You should know.

Course information, where is it?

- Brightspace
  - Syllabus
  - Homework assignments
  - Labs
  - Your grades
    - Our goal is to have all submissions graded within a week
  - In-class code
  - Discussion forum
- Course lecture notes: cis.poly.edu/jsterling/cs2124/Notes/Syllabus.html
  - Note that there is **no textbook** (aren't we nice?)
- Good c++ language reference: http://www.cplusplus.com

Development environment

- Install a version of C++ on your machine *before* coming to lab!
- We don't force you to use one environment or another
- We recommend:
  - On Windows: https://visualstudio.microsoft.com/
  - On Mac: https://developer.apple.com/xcode/
  - Linux: g++. I expect you already have it.
  - CLion
  - For emergencies: an online compiler, onlinegdb.com. Looks ok so far…
- Personally, I (Sterling) like to use a good editor and command line compiler, e.g. g++ / clang. I find IDE's get in the way, especially for small projects. Just my opinion.