Debugging Code

- **ART** of diagnosing errors in programs and determining how to correct them.
- Must want to be a detective
- "Bugs" come from: coding errors, design errors, complex interactions, poor user interface designs, and system failures
- Debugging a program effectively requires that you learn how to identify which the cause of the problem(s) and apply appropriate techniques to eliminate them
- Most common bugs arise from:
  - Programming without thinking
  - Writing code in an unstructured manner

How to reduce the time spent debugging?

- Get in the right frame of mind. Assume the worst- something is wrong!!
  - At the outset, you don’t know what is wrong
  - It worked before …, it seems to work, except for…, I don’t see any errors… ARGH
- Work on it right away- hard to come back to
- Change one thing at a time
  - When making a lot of changes, apply them incrementally
  - Add one change, and then test that change thoroughly before starting on the next change
  - This will reduce the number of possible sources of new bugs
  - If several different changes are applied at the same time, then it is much more difficult to identify the source of the problem
  - Minor errors in different parts of the code can interact to produce errors that never would have happened if those changes had been applied one at a time

Writing a Good Program

- Does it do what it is supposed to do? "correct output for each possible input"?
- Is it written in a reasonable amount of time?
- How adaptable is it to cope with changing requirements?
- Is it efficient enough for the computing environment in which it is intended to be used?

Signs of a Good Computer Code

- Clearly defined purpose
- Simple to use
- Rugged (difficult to misuse, handles errors well)
- Delivered on time
- Reliable
- Extensibile (can be applied to other situations without huge modification)
- Efficient and not bloated
- Clear and concise comments

Signs of Good Code

- Using functions and features of programming language appropriately
  - The best tool for the job
  - “When all you have is a hammer, everything looks like a nail”
- Using vectors and arrays to minimize the complexity of the code and extensibility
  - Don’t want to have a bunch of separate vector variables that could all be in an array
- Use do loops and while statements appropriately
  - Could you use vector and array math to eliminate?
- Avoid cutting and pasting large chunks of code and then making minor edits
  - What seemed fast to do at the outset is a pain if your code has errors

Why Can't Programmers.. Program?

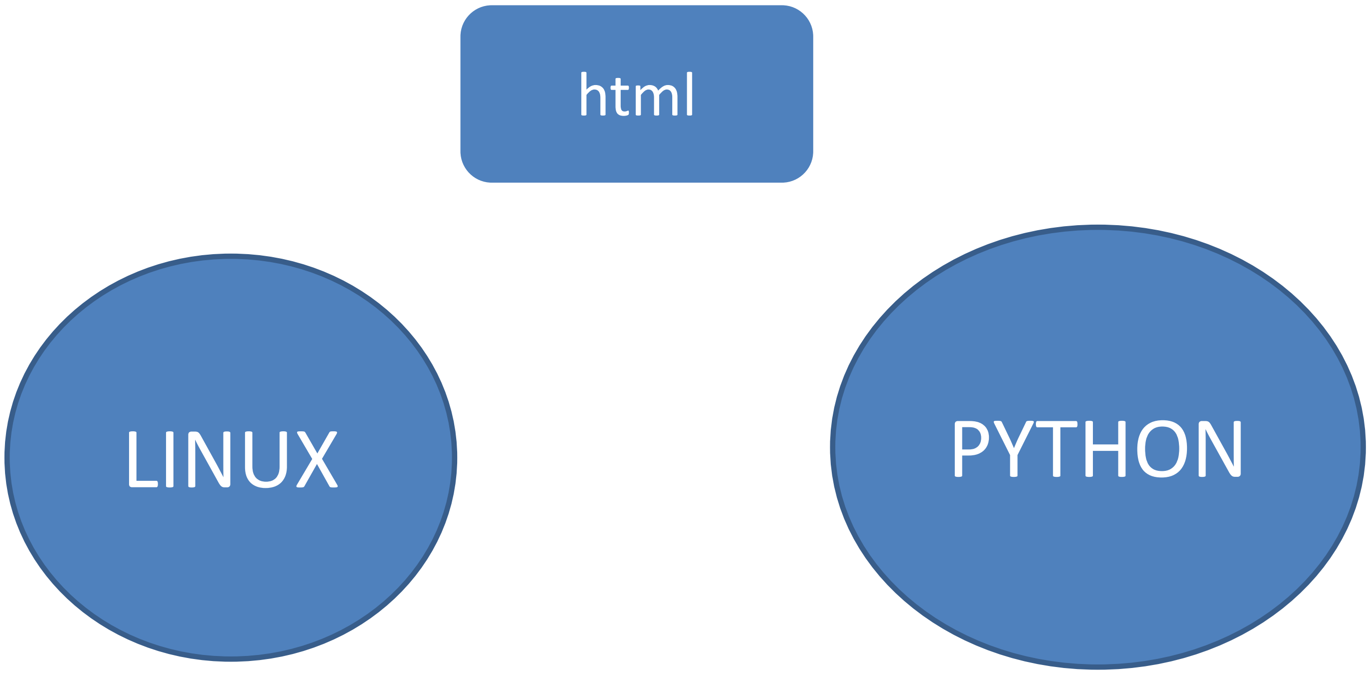
http://blog.codinghorror.com/why-cant-programmers-program/

- “Like me, the author is having trouble with the fact that 199 out of 200 applicants for every programming job can't write code at all. I repeat: they can't write any code whatsoever.”
- After a fair bit of trial and error I've discovered that people who struggle to code don't just struggle on big problems, or even smallish problems (i.e. write a implementation of a linked list). *They struggle with tiny problems*.
- A surprisingly large fraction of applicants, even those with masters' degrees and PhDs in computer science, fail during interviews when asked to carry out basic programming tasks. For example, I've personally interviewed graduates who can't answer "Write a loop that counts from 1 to 10"

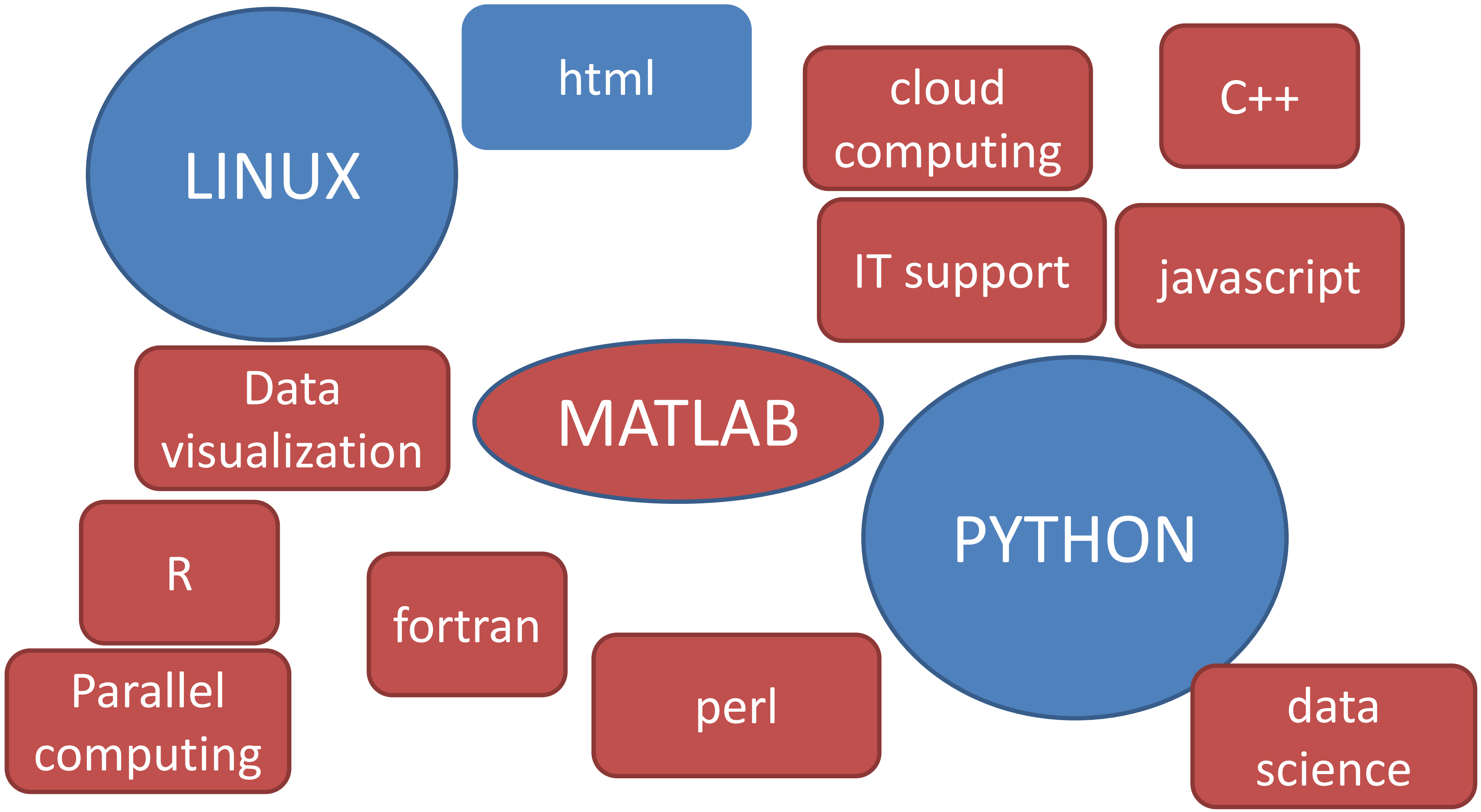
Signs that you shouldn't be a programmer

4. Dysfunctional sense of causality

- **Symptoms**
- You seriously consider malice to be a reason why the compiler rejects your program
- Your debugging repertoire includes rituals like shining your lucky golf ball, twisting your wedding ring, and tapping the nodding-dog toy on your monitor. And when the debugging doesn't work, you think it might be because you missed one or didn't do them in the right order
- **Alternative careers**
- Playing the slot machines in Vegas



Software evolves


LINUX
html
cloud computing
C++
IT support
javascript
Data visualization
MATLAB
PYTHON
R
fortran
perl
Parallel computing
data science

Finding the right computing environment for you

- What do you need to do?
  - Are you a consumer or a developer?
- How are you going to interface with the computing environment? PC vs. Mac vs. Web based
- Can you complete the task in your local environment or do you need more horsepower?
- Is cost a factor?
  - No? use proprietary resources: Matlab, ArcGIS, IDL
  - Yes? Use open source: python