- Today:
    - debugging, testing, style, little bit of quiz prep
    - Alluded to many of these techniques in lecture + OH, but today we will be explicitly talking about them
    - Debugging tips will definitely be useful for homework!
- Quiz clarification
    - If you are sitting in on the course, you may take the quiz, not guaranteed to be graded
- Why is this important
    - Smart ship USS Yorktown was left dead in the water in 1997 for nearly 3 hours after a divide by zero error.
    - Northeastern major blackout
    - Vancouver stock Exchange did something similar to integer division where they chopped off any dollar value past the third decimal place
        - Lost almost 25 dollars a month on stocks!
- Testing
    - Pull up hw1 testfile, walk through of how the bottom of the test file really works
    - Assert review
    - Test function for isPrime demo!
        - Edge cases => turning points in code (ex. First and last chars in a string, in the case of isPrime going from 2 -> 3)
        - Special cases => cases that don't fall into our general category well (1)
        - Large cases => large inputs to make sure function doesn't fail after many iterations
        - Varying outputs -> make sure your test cases test for different outputs (i.e. not just True, not just False, not just 7)
    - Console and Graphics
        - Not going to be tested, so let's just skip right along
    - Testing and 112
        - Graded for test functions starting week 2
    - Try/except syntax
    - Raising exceptions
- Debugging
    - Remember the three types of errors!
        - Syntax, Runtime, Logical
    - Syntax
        - Easiest to debug
        - Find the line, try to make sense of the error (sometimes they use big words), but even if you don't understand the error, the location usually gives enough clues
        - **Important**: make sure you look around the line as well, Python sometimes gives the wrong line
        - Go over pass
    - Runtime
        - Little bit harder
        - Read the error message first!

    - If you know which test case fails, trace through it manually
      - Most effective debugging tool
    - If you don’t, try to find the smallest test case that will fail
    - Like syntax errors, want to look at the surrounding lines
  - Logical Error
    - Hardest to debug!
    - Code runs without crashing or errors, BUT doesn’t give the right answer
    - Example: absolute value
    - Solution: print statements
      - You can use print statements at various points of your code to see what values are
    - Tip: extract test case from assert statement!
- Style
  - However, good style can help with getting your homeworks done faster!
  - Most important areas of the style guide:
    - Comments – code should generally be what’s called self-documenting. However, there are times when you have to write something that may not make sense to a reader. A comment is important in these cases to describe what is going on.
    - Test Functions – mentioned before
    - Magic Numbers – if you have a number that you might need to change over and over, it should be stored in a variable. Any number besides [-2, 2] + 10, is considered a magic number
    - Variable Names – they should describe what the variable is, exceptions are the looping variable
    - Duplicate Code – don’t do it!
    - Read the rest of them on your own time, you are responsible for them!
- Quiz Info + Mini Review
  - Mentioned this last lecture but worth reiterating
  - Tomorrow, 50 mins, during recitation
  - Will cover all topics from this week
    - Clarification: chr, ord, ASCII is an encoding
      - Maps characters to numbers
      - Just because ord(‘p’) = 112 doesn’t mean that ‘p’ == 112
      - Letters are near to each other, i.e. ‘p’ => 112, ‘q’ => 113
      - Upper case and lower case have different ascii values
    - Don’t worry about memorizing the style guide or anything like that – you should be able to identify very egregious style violations though
    - Focus will be on loops and strings, as they are the main topics to master this week
    - Mentioned on Piazza: responsible for knowing how to do integer problems without converting to string
      - I would recommend reviewing a problem like longestDigitRun, and doing that without strings

  - Syllabus question will come up
    - Won’t be something like “what is the 85th word on the syllabus spelled backwards”
    - More related to course resources: Ex. “Name one of your recitation Tas”, or “Where are TA office hours typically held?”
  - Typically involve short answer (related to material), code tracing, reasoning over code, and free response
    - Gauge how much people are familiar with each
- String CT (8 mins)
  - Work with the person next to you on the CT, with a pencil and paper
  - Submit your paper with both your Andrew IDs as attendance credit today.
  - Go over solution
  - Easy way to test CT is to just run it!
- isPalindrome (time permitting)
- String ROC (will do in recitation today!)