Computer Science 134C
Introduction to Computer Science, in Python
Lecture #6 (Iteration)
September 19

Keywords
`break`, `continue`, `for`, `input`, iterable, random numbers, `range`, `while`

We focus on controlling repetition.

1. Reminder: Monday labs are due on Thursday at noon. Tuesday labs are due on Friday at noon. Office and TA hours are posted.
2. Reminder: How to pick up grades: git pull. Numeric grade eventually posted.
3. Questions?
4. Using `bool` types.
   (a) Comparisons.
   (b) Interesting case: In Python *everything* is an *object*. Identity *vs.* equivalence and `None`.
   (c) Containment (check with `in`).
   (d) Combining bools.
5. Controlled and efficient repetition.
   (a) The `for` loop:
      i. Has the general form:
      ```
      for <variable> in <iterable>:
          <suite>
      ```
      The `for` executes the suite once for each element of an *iterable* (an object whose values can be encountered in sequence: lists, sets, strings, *etc.*).
      ii. You can immediately leave the loop if you execute a `break` statement. This should probably be guarded by an `if` statement. This may be useful when the iterable is infinite (more on this later).
      iii. Executing a `continue` finishes the current interation and begins the next (with the condition test). Again, typically this is guarded by an `if`.
   (b) The `while` loop:
      i. Has general form:
      ```
      while <condition>:
          <suite>
      ```
      Repeats suite as long as the condition is true. Typically, the suite will do something to potentially modify the condition.
      ii. The use of `break` and `continue` are possible. If condition is the tautology `True`, the `break` statement allows us to construct loops that test at the bottom or middle.

(c) Guiding principles:

   i. Use a `for` loop when you know how many iterations you want to perform (exactly 10? iterate over a `range`; one for each `x` in `S`? iterate over `S`).

   ii. Use a `while` loop when you're not sure how many times you'll execute the loop's suite (asking for input that may require re-prompting? waiting for a computation to stablize? use a `while`)

6. Recall Example: Computing the orbit of a function, revised.

7. Example: Removing spaces from a string.

8. Example: Reading lines from a file.

9. Random numbers with `random.randint(a,b)`. Typically:

   ```
   from random import randint
     ...
   x = randint(1,6)
   ```

   Picks a random integer between 1 and 6 *inclusive*. Importing and calling `seed(n)` will *seed* or restart the random sequence from a predictable point; helpful to call `seed` while debugging.

10. Example: Telling fortunes, fairly.

⋆