FUNCTIONS, PARAMETERS, AND SUBVERSION

CSSE 120 – Rose-Hulman Institute of Technology

Review: += and related operators (-=, *=, …)

- a += b is equivalent to a = a + b

```
>>> x = 5
>>> x += 6; print x
11
>>> x *= 2; print x
22
>>> x %= 7; print x
5
>>> s = "abc"
>>> s += "d"; print s
abcd
>>> nums = [1,2,3]        # Note: list concatenation!
>>> nums += [4,5]; print nums
[1,2,3,4,5]
```

Tidbit: random numbers

from random import randrange, random

randrange(start, end, step) returns a random integer from the list generated by the corresponding range statement

random() returns a random float in the range [0,1)

Includes 0, but not 1.

Outline

- Review of topics for Exam #1
- Tools: Version Control
- Basic Functions - Math, Maple, Python
- Function definition and invocation mechanics
- Exercise: writing **`distance()`**
- Nested function calls and execution order
- Code-reading exercise

Exam 1

- When? Where?: See schedule
  - Please get in the habit of checking the schedule regularly. Time management is a problem solving process too!
- Format:
  - Paper part: open book, 1 double-sided page of notes, *closed computer*
  - Programming part: open book, notes, and computer
    - Any resources you can reach from Angel by clicking only.

Possible Topics for Exam 1

- algorithm
- comment
- variable, assignment
- identifier, expression
- for loops
- phases of software development
- input, print
- import, math functions
- using functions
- int, float, long, conversion
- strings (basic operations)
- character codes (chr, ord)
- lists (concatenation, slices)
- reading, writing files
- formatted output
- using objects, graphics
- method *vs* function
- event-driven program

Software Engineering Tools

- The computer is a powerful tool
- So use it to make software development easier and less error prone!
- Some software engineering tools:
  - IDEs, like Eclipse
  - Version Control Systems—like Subversion
  - Diagramming applications—like Violet or Visio
  - Modeling languages—like Alloy, Z, or JML

Version Control Systems

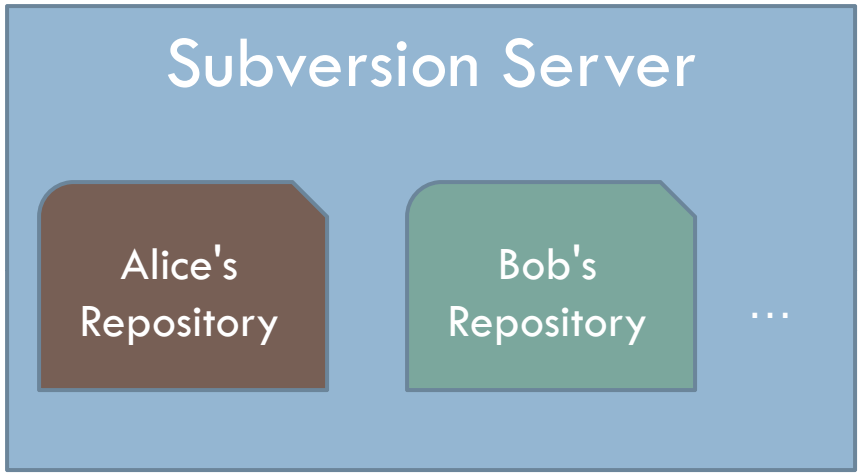
- Store "snapshots" of all the changes to a project over time
- Benefits:
  - Allow multiple users to share work on a project
  - Act as a "global undo"
  - Record who made what changes to a project
  - Maintain a log of the changes made
  - Can simplify debugging
  - Allow engineers to maintain multiple different versions of a project simultaneously

Our Version Control System

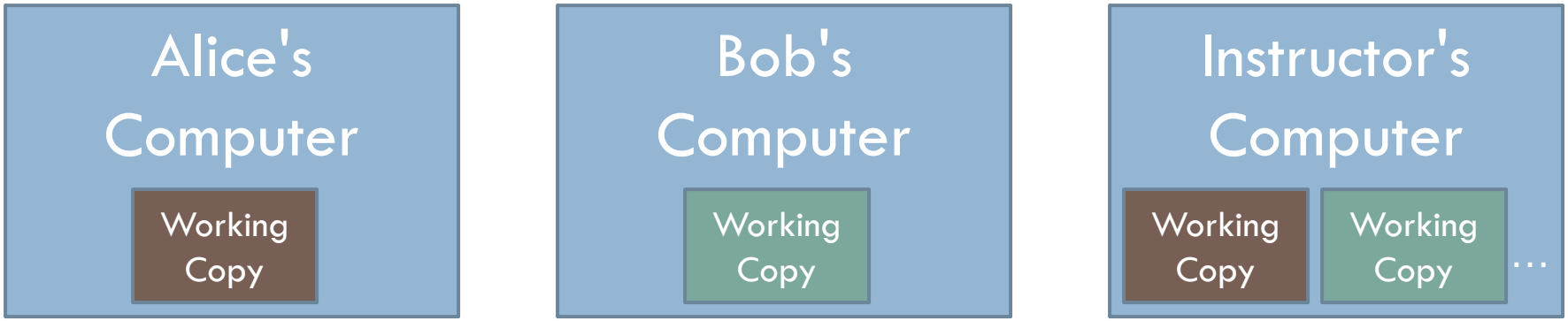
- Subversion, sometimes called SVN
- A free, open-source application
- Lots of tool support available
  - Works on all major computing platforms
  - **TortoiseSVN** for version control in Windows Explorer
  - **Subclipse** for version control inside Eclipse

Version Control Terms

Repository: the copy of your data on the server, includes ***all*** past versions

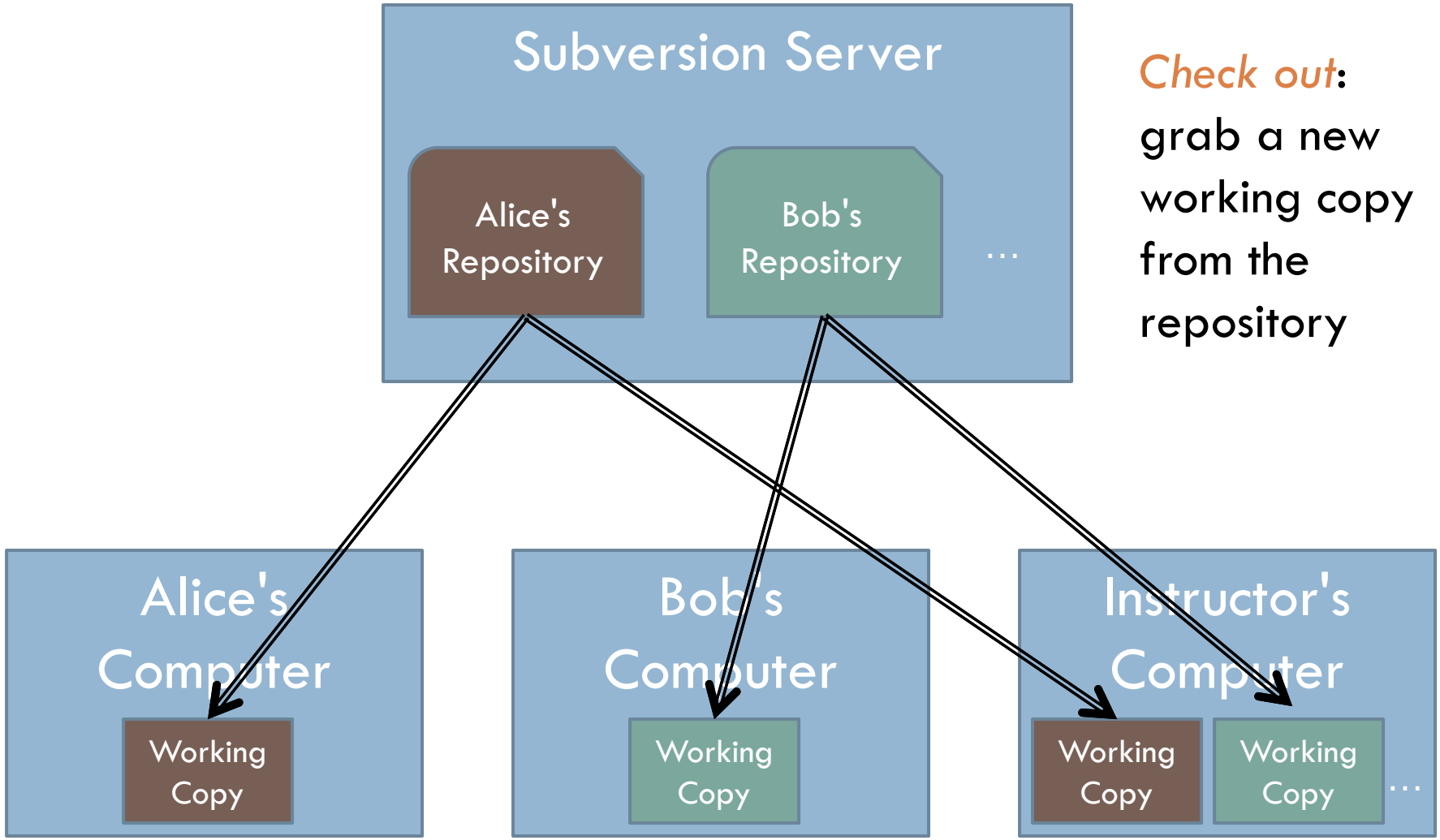


Working copy: the ***current*** version of your data on your computer



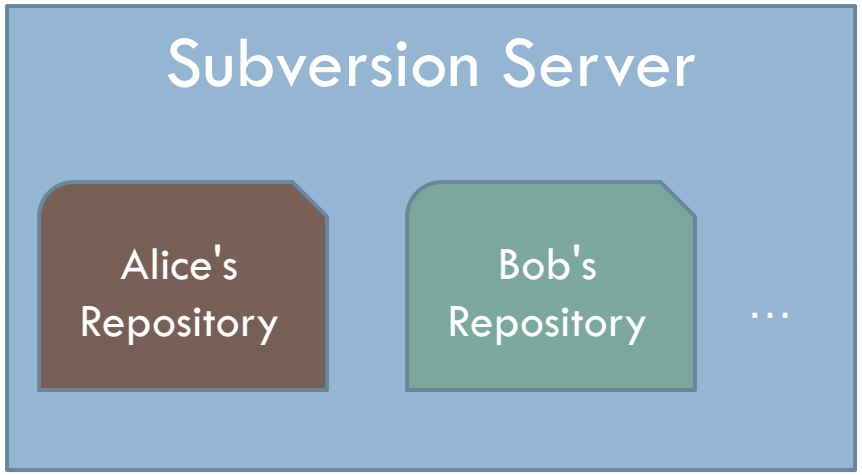
Q3b

Version Control Steps—Check Out

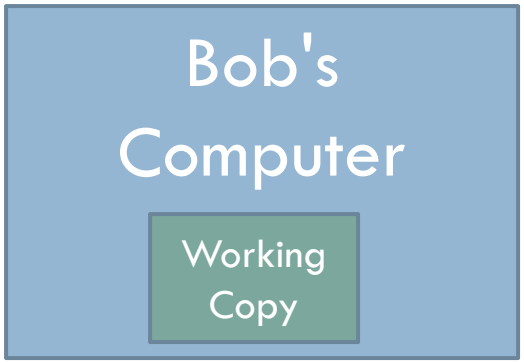


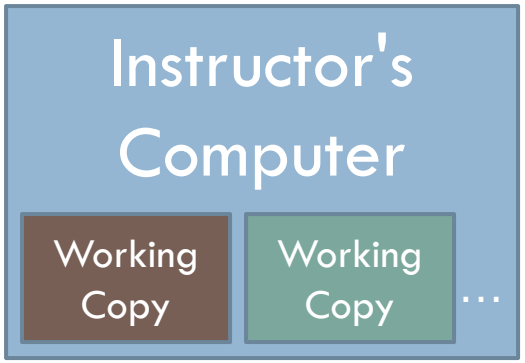
Check out: grab a new working copy from the repository

Version Control Steps—Edit

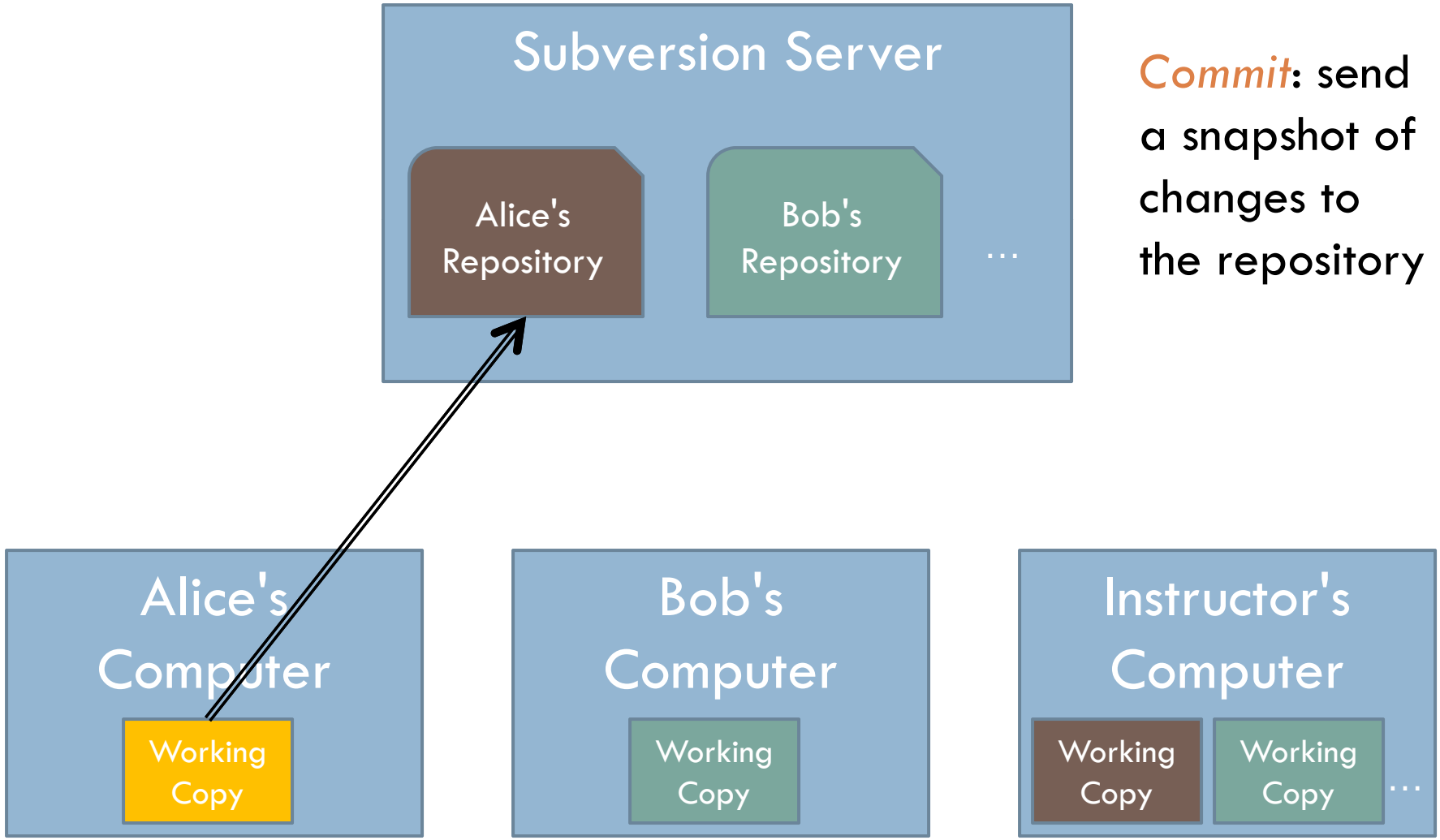


Edit: make ***independent*** changes to a working copy



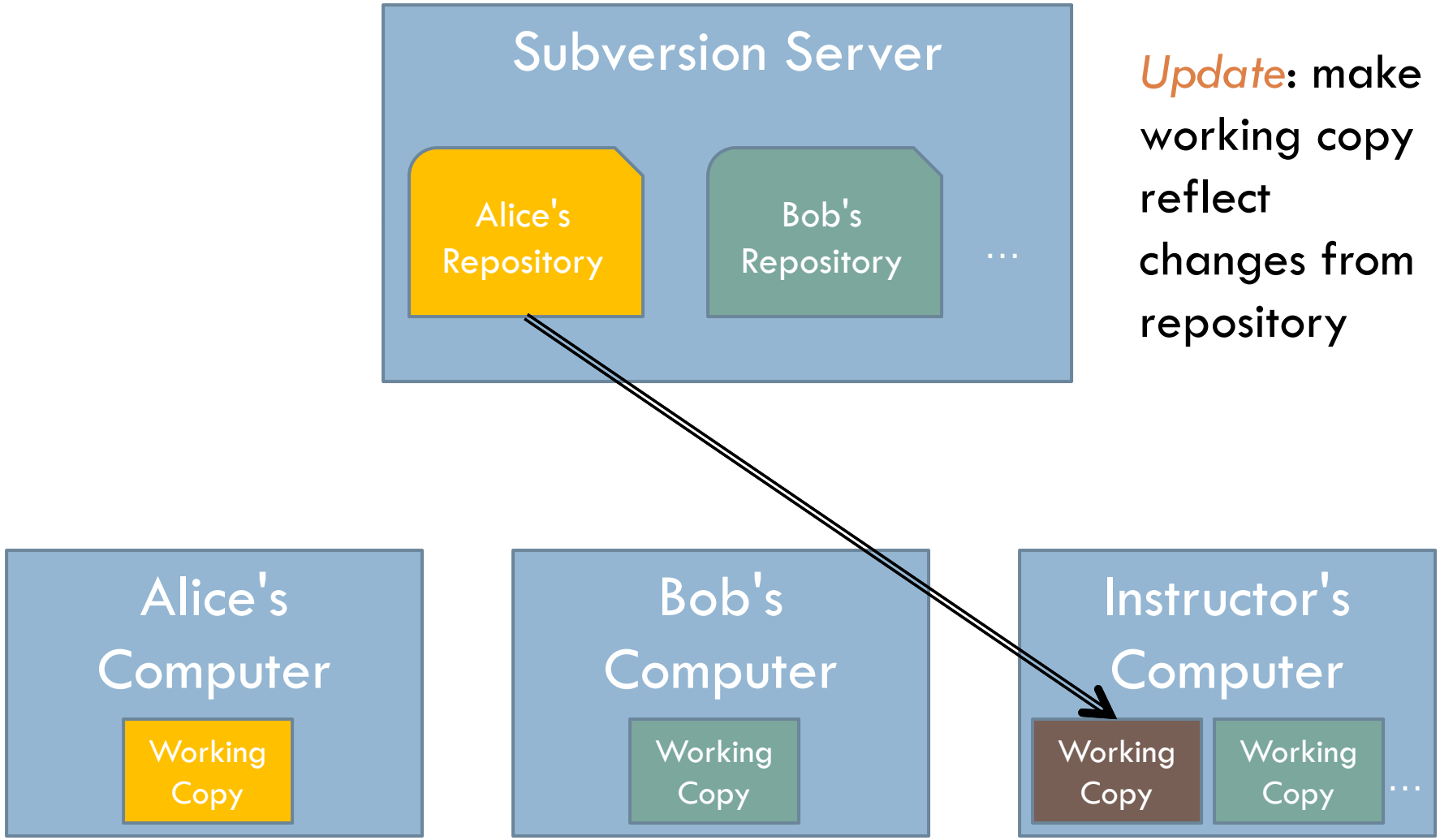


Version Control Steps—Commit



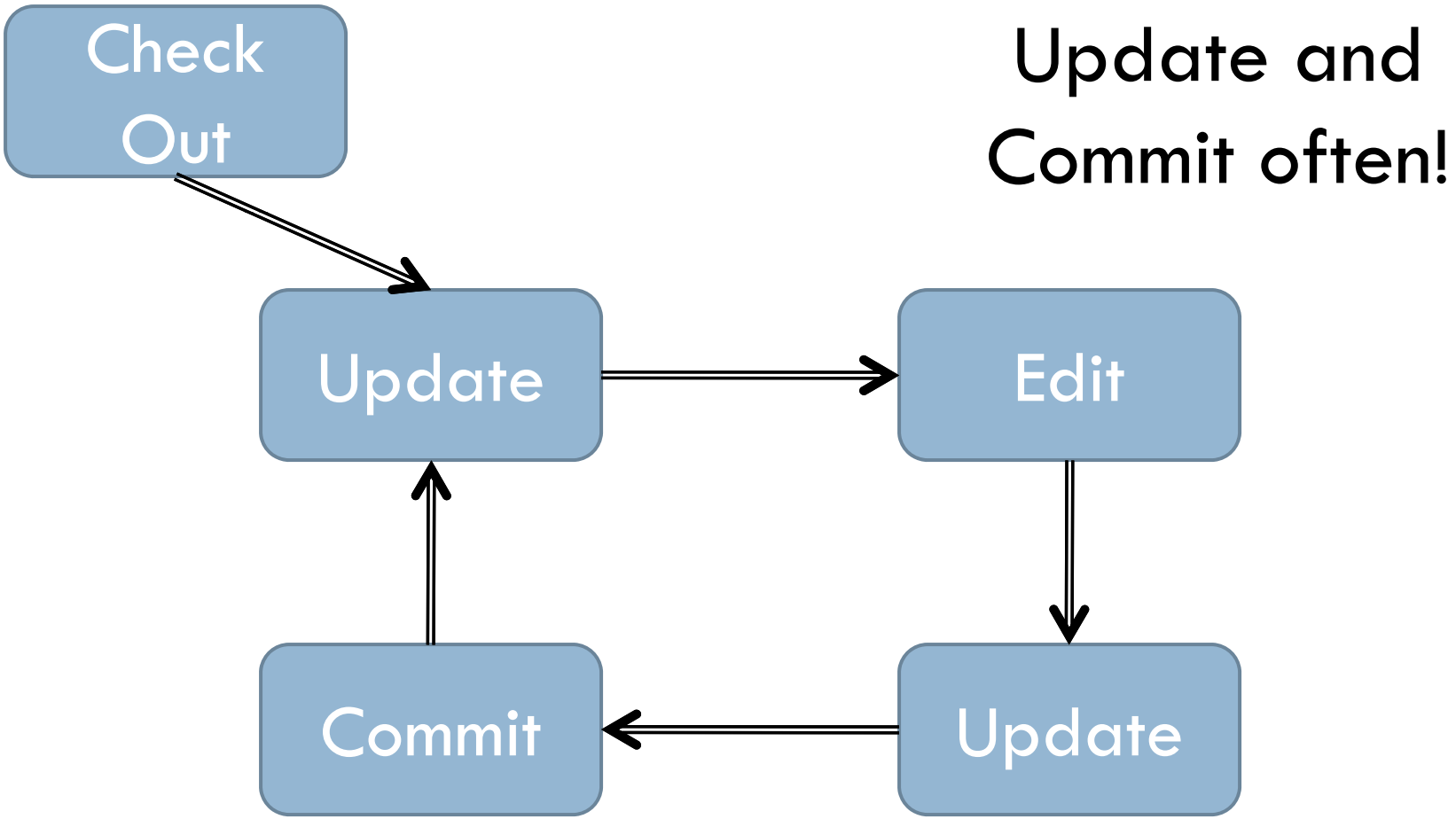
Commit: send a snapshot of changes to the repository

Version Control Steps—Update

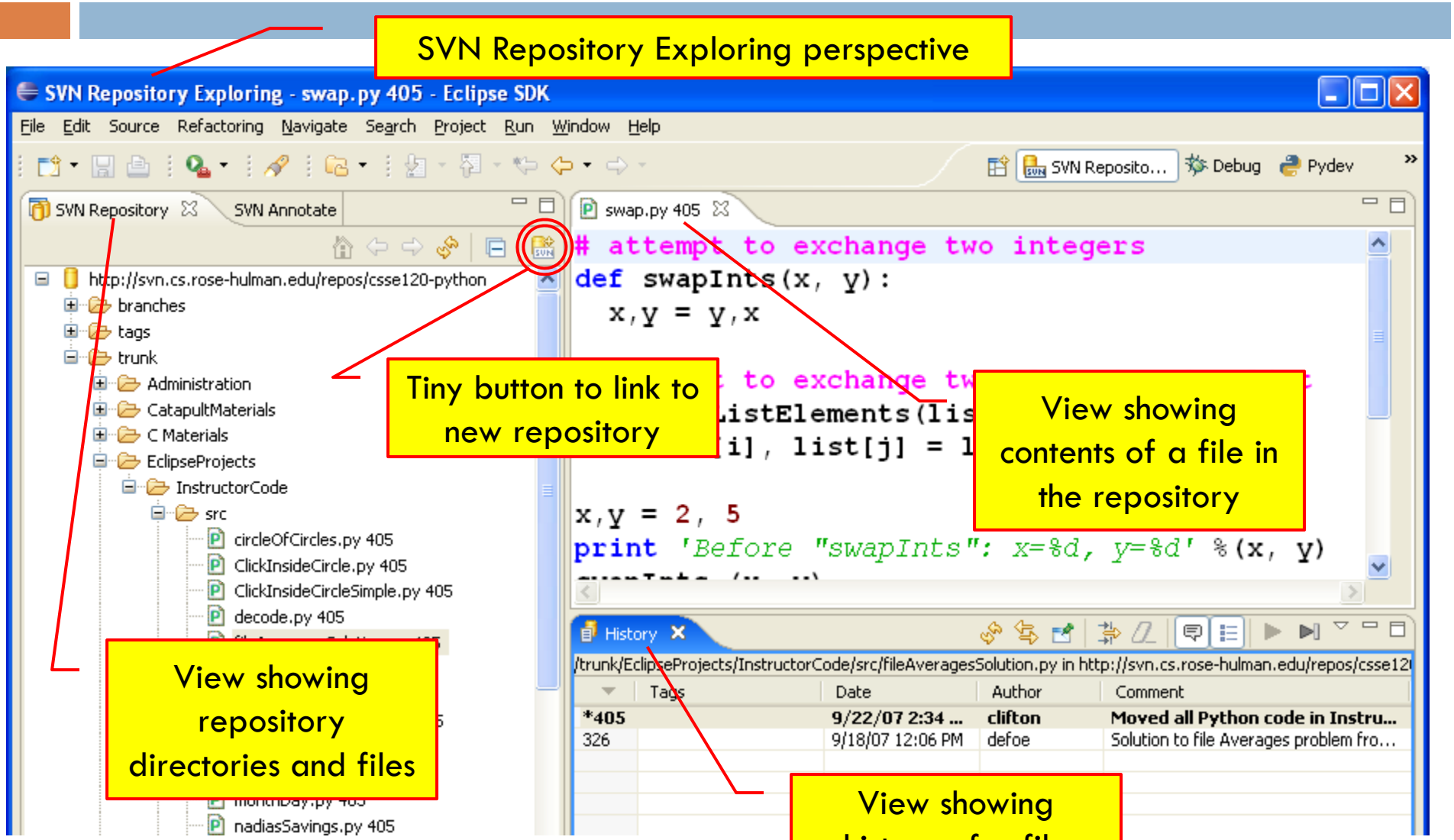


Update: make working copy reflect changes from repository

The Version Control Cycle



Subversion in Eclipse—Subclipse



Get help if you're stuck!

Check out today's exercise

- Add an SVN Repositories View to your PyDev perspective.
  - Window → Show View→ Other…
  - Expand SVN > SVN ***Repositories***
  - Click OK
- Note the new SVN Repositories view appears as a tab in the same window as the console
- Browse the view for the Session07 project
- Right-click it, and choose Check Out.
- Confirm all of the options presented
- In Package Explorer, find distance.py inside your Session07 project
- Add your name only to the comments, and commit your changes.

Why functions?

- A function allows us to group together several statements and give them a name by which they may be invoked.
  - Abstraction (easier to remember the name than the code)
  - Compactness (avoid duplicate code)
  - Flexibility (parameters allow variation)
- Example:

```
def complain(complaint):
    print "Customer:", complaint
```

Functions in different realms

We compare the mechanisms for defining and invoking functions in three different settings:

- Standard mathematical notation
- Maple
- Python

Functions in Mathematics

- Define a function:
  - $f(x) = x^2 - 5$

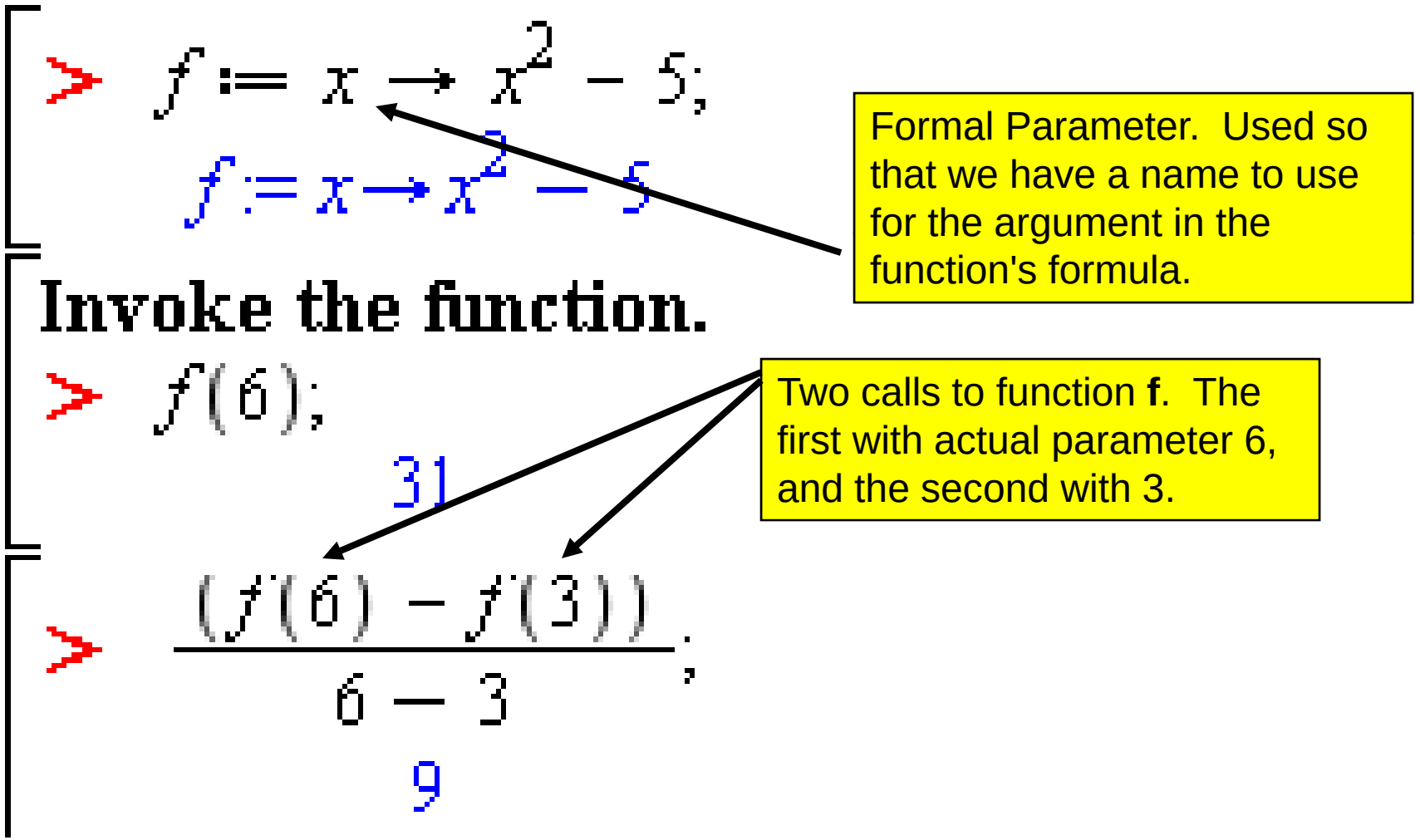
Formal Parameter. Used so that we have a name to use for the argument in the function's formula.

- Invoke (call) the function:
  - $\frac{f(6) - f(3)}{6 - 3}$

Two calls to function **f**. The first with actual parameter 6, and the second with 3.

- When the call f(6) is made, the actual parameter 6 is substituted for the formal parameter x, so that the value is $6^2 - 5$.

Functions in Maple



Functions in Python

```
>>> def f(x):
        return x*x - 5

>>> f(6)
31
>>> (f(6) - f(3)) / (6 - 3)
9
>>>
```

Formal Parameter. Used so that we have a name to use for the argument in the function's formula.

Two calls to function **f**. The first with actual parameter 6, and the second with 3.

- What would **f(f(2))** evaluate to?

- In Mathematics, functions calculate a value.
- In Python we can also define functions that instead *do something*, such as print some values.

Review: Parts of a Function Definition

```
>>> def hello():
      print "Hello"
      print "I'd like to complain about this parrot"
```

Defining a function called "hello"

Indenting tells interpreter that these lines are part of the hello function

Blank line tells interpreter that we're done defining the hello function

Review: Defining vs. Invoking

- Defining a function says what the function should do
- Invoking a function makes that happen
  - **Parentheses** tell interpreter to invoke the function

```
>>> hello()
Hello
I'd like to complain about this parrot
```

Review: Function with a Parameter

- def complain(complaint):

```
    print "Customer: I purchased this parrot not half " +
          "an hour ago from this very boutique"
    print "Owner: Oh yes, the Norwegian Blue. " +
         " What's wrong with it?"
    print "Customer:", complaint
```

- invocation:
  - complain("It's dead!")

Functions Can Return Values

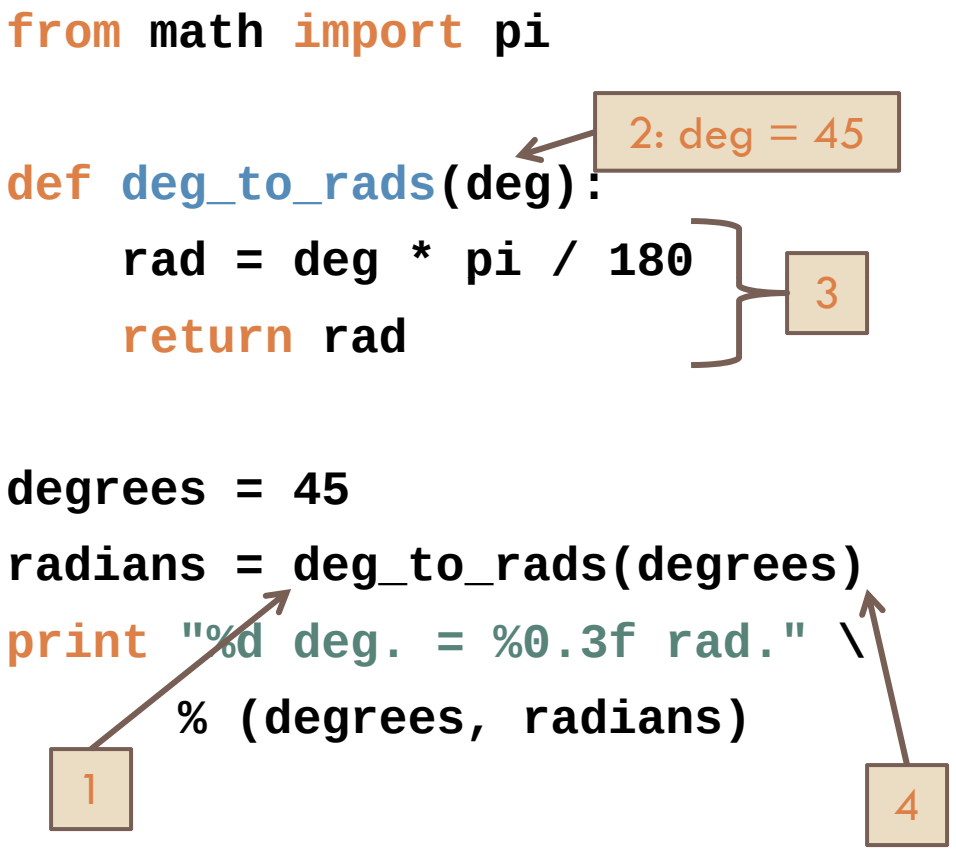
- We've **written** functions that just do things
  - hello()
  - complain(complaint)
- We've **used** functions that *return* values
  - abs(-1)
  - fn_root_1 = math.sqrt(b*b – 4*a*c)
- Define a function that returns a value

```
def square(x):
    """ Returns the square of x """
    return x * x
```

return statement

When a function is invoked Python follows a four-step process:

1. Calling program pauses at the point of the call
2. Formal parameters get assigned the values supplied by the actual parameters
3. Body of the function is executed
4. Control returns to the point just after where the function was called and the value of the function is what was returned



Exercise – writing a `distance()` function

- Go to the Session07 project you checked out in Eclipse
- Add a comment at the top of the file to say what the program does

a. Write and test a `distance` function:

```
def distance(p1, p2):
    """Parameters are Points, returns distance between them."""
```

b. Generate two random points using randrange and compute and print the distance between them:

- The distance between (5,7) and (4,2) is 5.099

- When you have it working, commit your code back to your repository

If a Function Calls a Function …

```
def g(a,b):
    print a+b, a-b

def f(x, y):
    g(x, y)
    g(x+1, y-1)

f(10, 6)
```

- Trace what happens when the last line of this code executes

An exercise in code reading

- With a partner, read and try to understand the code that is on the handout.
- You can probably guess what the output will be. But how does it work?
- Figure that out, discuss it with your partner and answer quiz question 9.
- Optional Challenge Problem for later: try to write "There's a Hole in the Bottom of the Sea" or "The Green Grass Grew All Around" in a similar style.
- When you are done, turn in your quiz and start HW