Terms: Test (Case) vs. Test Suite

- **Test (case)**: one execution of the program, that may expose a bug

- **Test suite**: a set of executions of a program, grouped together
  - A test suite is made of test cases

```
suite::tests    flock:sheep
```
Kinds of Testing

**Manual testing:**
- A human being sits at a program and enters inputs into a program to test it
- Often employees non CS people
- Beta testing, user testing obviously

- Sometimes means a human writes out tests that run automatically, but usually means a person does the testing
Kinds of Testing

**Automated testing:**

- One meaning is that humans design tests, but use scripts or other methods to have the computer execute the tests without help.
- Another meaning is that humans write programs that produce tests.
- Essential for speed, automatic checks when a program is compiled, etc.
Kinds of Testing

● Which is better?
  - Some academics will say that manual testing is ineffective and slow, often poorly thought out
  - Some folks in industry will say that automated testing is often a waste of time/money and humans find more bugs more cheaply

  - Both are right, both are wrong
  - Neither kind of testing is superior, and a good test engineer knows when to use each
Kinds of Testing

- For more information on good manual testing:
  - Books by James Whittaker
    - Originally academic, then at MS, then google, then back to MS
  - Whittaker’s ideas bring us to next distinction, two kinds of manual testing
Kinds of Testing

- **Scripted testing vs. exploratory testing**
  - In scripted manual testing, a tester follows a fixed script: “first open this file using this dialog, then click here, then resize the font…”
  - In exploratory testing, the tester may have some general guidelines “try all the choices in the File dialog” or an area to focus on but is free to explore
  - Whittaker’s book covers the value of guided exploratory manual testing
Unit, Integration, System Testing

Stages of testing

- **Unit testing** is the first phase, done by developers of modules
- **Integration testing** combines unit tested modules and tests how they interact
- **System testing** tests a whole program to make sure it meets requirements

- “**Design testing**” is testing prototypes or very abstract models *before implementation*
**Terms: Black Box Testing**

- **Black box testing**
  - Treats a program or system as opaque
  - That is, testing that does *not* look at source code or internal structure of the system
  - Send a program a stream of inputs, observe the outputs, decide if the system passed or failed the test
  - Abstracts away the internals – a useful perspective for integration and system testing
  - Sometimes you don’t have access to source code, and can make little use of object code
    - True black box? Access only over a network
Terms: White Box Testing

**White box testing**

- Opens up the box!
  - (also known as glass box, clear box, or structural testing)

- Use source code (or other structure beyond the input/output spec.) to design test cases

- Brings us to the idea of *coverage*