Black-box testing

- Test suite designed without looking at code
  - Can be done by someone other than implementer
  - Will avoid inherent biases of implementer, exposing potential bugs more easily
  - Testing designed without knowledge of implementation, thus can be reused even if implementation changed

Paths through a specification

```
def sqrt(x, eps):
    """Assumes x, eps floats
            x >= 0
            eps > 0
       returns res such that
            x-eps <= res*res <= x+eps"""
```

- Paths through specification:
  - x = 0
  - x > 0
- But clearly not enough

Paths through a specification

- Also good to consider boundary cases
  - For lists: empty list, singleton list, many element list
  - For numbers, very small, very large, “typical”

Example

- For our sqrt case, try these:
  - First four are typical
    - Perfect square
    - Irrational square root
    - Example less than 1
  - Last five test extremes
    - If bug, might be code, or might be spec (e.g. don't try to find root if eps tiny)

Partition

x	eps
0.0	0.0001
25.0	0.0001
.05	0.0001
2.0	0.0001
2.0	1.0/2.0**64.0
1.0/2.0**64.0	1.0/2.0**64.0
2.0**64.0	1.0/2.0**64.0
1.0/2.0**64.0	2.0**64.0
2.0**64.0	2.0**64.0

Small