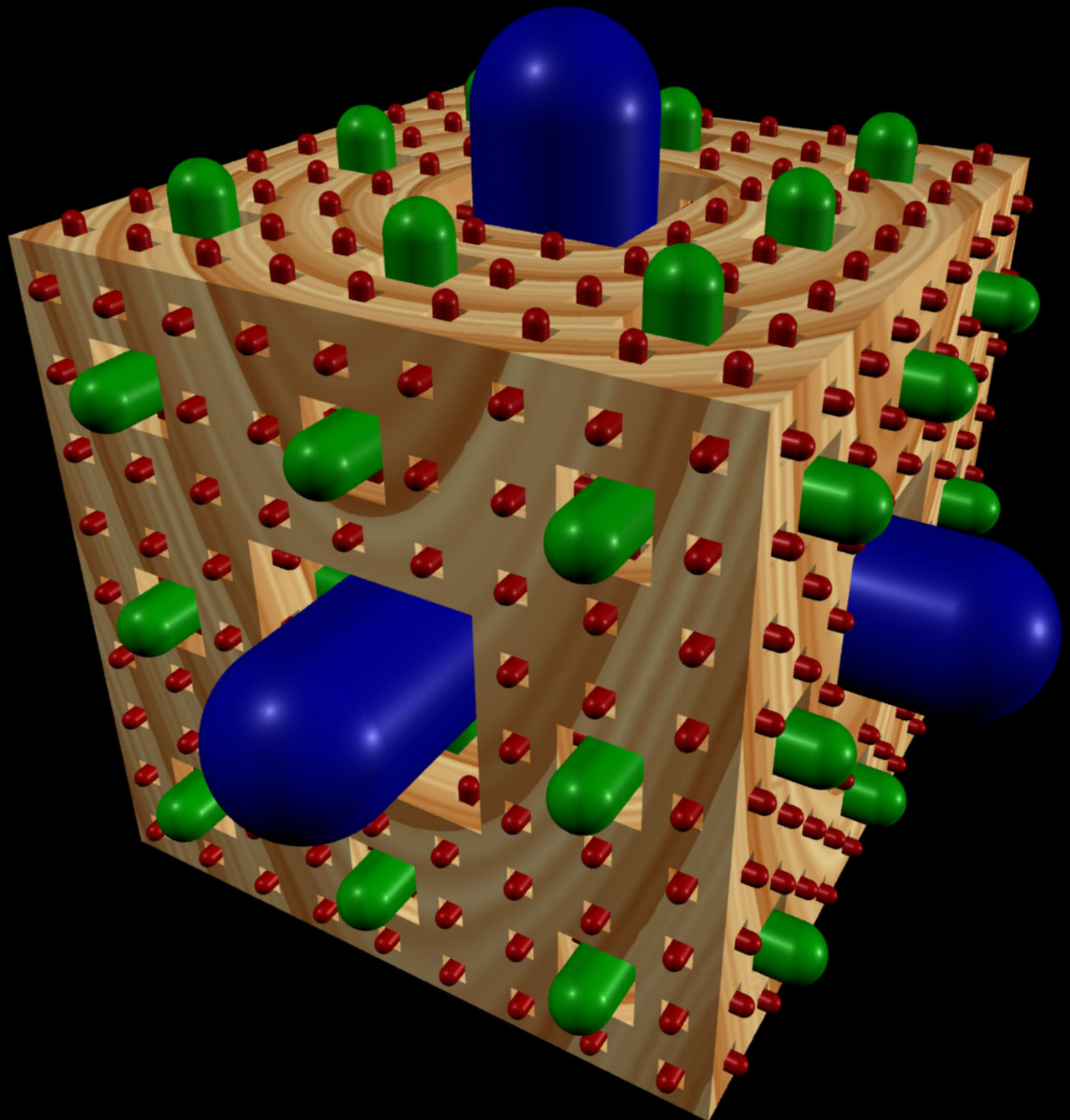
Functional Abstraction

Announcements

Did you watch the lecture videos for today and follow along by typing the Python?



Functional Abstraction (review)

- A function has inputs & outputs
  - Possibly no inputs
  - Possibly no outputs (if block is not pure)
    - In this case, it would have a "side effect", i.e., what it does (e.g., move a robot, print)
- The contract describing what block does is called a specification or spec

What is IN a spec? (review)

- Typically they all have
  - NAME
  - INPUT(s)
    - (and types, if appropriate)
    - Requirements
  - OUTPUT
    - Can write "none"
  - (SIDE-EFFECTS)
  - EXAMPLE CALLS
- Example
  - NAME : **Double**
  - INPUT : **n** (a number)
  - OUTPUT: **Twice input**
  - SAMPLE: `>>> double(10)`
    `20`

What is NOT in a spec?

- How!
  - That's the beauty of a functional abstraction!
  - It doesn't say how it will do its job.
- Example: `double(n)`
  - Could be `n * 2`
  - Could be `n + n`
  - Could be `n + 1` (`n` times)
    - if `n` is a positive integer
- This gives great freedom to author!
  - You choose Algorithm(s)!

Ten-to-0-by-1-or-2 Rules

Two players alternate turns, on which they can remove 1 or 2 from the current total

The total starts at 10

The game end whenever the total is 0

The last player to move **wins** (i.e., ”if you can’t move, you lose”)

(Demo)