Lecture 18: Recursion

CS 51G

Spring 2018

Kim Bruce

Test Programs

- Test programs 2 and 3 available now!
- No coverage of GUI components
    - Dragging and interacting w/objects
    - Designing classes
    - Animations

Midterm

- Friday in class: 50 minutes
- Coverage: Chapters 1-9, 20.

Recursion

- Explain things naturally
- How to draw a target
  - If small enough, just draw bullseye
  - Otherwise draw outer ring and then draw smaller target inside
- Can write programs like that!
  - http://www.cs.pomona.edu/classes/cs051G/demos/TargetApp/TargetApp.grace

Creating Recursive Objects

1. Create a type with all methods necessary
2. Define an object(s) representing the simplest (base) cases.
3. Define the recursive case
    1. has an instance variable/def of same type, but simpler.
    2. Write initialization assuming initialization of simpler part is correct.
4. Write methods under assumption it works correctly for all simpler objects.

Examples

- Scribble that can be moved
    - http://www.cs.pomona.edu/classes/cs051G/demos/SingleScribble/
- Chain reaction
    - http://www.cs.pomona.edu/classes/cs051G/demos/ChainReaction/ChainReaction.grace
- Broccoli
    - http://www.cs.pomona.edu/classes/cs051G/demos/Broccoli/

Recursive Methods

- Can have recursion on methods where it is just parameters that get simpler. Assume exponent is integer (or won't stop!!)

```
method simpleRecPower (base: Number, exponent: Number)
                  -> Number {
   if (exponent == 0) then {
       1
    } else {
       base * simpleRecPower (base, exponent - 1)
    }
}
```

Call with simpler (smaller) exponent!

More Power

- Can find even faster if use divide-and-conquer technique based on:
  - $b^0 = 1$
  - $b^{n+1} = b * b^n$
  - $(b^n)^m = b^{n*m}$
  - http://www.cs.pomona.edu/classes/cs051G/demos/Powers/

Questions?