Class 33: Numeric Recursion

Held: Monday, April 5, 2010

Summary: We visit a slightly different kind of recursion, numeric recursion. In this technique, we once again have procedures call themselves. However, the parameter that we "simplify" at every step is a number, rather than a list.

Related Pages:

* EBoard.
* Lab: Numeric Recursion.
* Reading: Numeric Recursion.

Notes:

* Welcome back! I hope you had a great break.
* Reading for tomorrow: Geometric Art.
* Exams returned. We'll spend a good deal of class going over them. Hence, the numeric recursion lab is an assignment due on Wedesnday.
* EC for today's Disability Studies talk on making a movie (4:15 in JRC101).
* EC for tonight's Disability Studies movie (7:00 in ARH 302).
* EC for tomorrow's CS and Disability talk (4:30 in Science 3821).
* EC for Wednesday's CS and Disability talk (4:15 in JRC101).
* EC for Thursday's "Teaching Millenials" CS Extra (4:30 in Science 3821).
* EC for Friday's "Computational Games" CS Extra (noon in Science 3821; Free Pizza; I'll need a count on Wednesday).

Overview:

* Recursion, Generalized.
* Thinking About Natural Numbers.
* Numeric Recursion.

Patterns of Recursion

* While we've seen and written a variety of examples of direct recursion, they typically have the following form:

```
(define recursive-proc
  (lambda (params)
    (if (base-case-test)
        (base-case params)
        (combine (partof params)
                 (recursive-proc (simplify params))))))
```

- In many cases, the combination ends up being a choice between two activities. In those cases, we might write:

```
(define recursive-proc
  (lambda (params)
    (cond
      ((base-case-test)
       (base-case params))
      ((special-case-test)
        (combine (partof params)
                 (recursive-proc (simplify params))))
      (else
        (recursive-proc (simplify params)))))))
```

- For lists, the simplification was almost always "take the cdr" and the "part-of" was almost always "take the car".

Recursion with Numbers

- While most of the recursion we've been doing has used lists as the structure to recurse over, you can recurse with many different kinds of values.
- It is fairly common to recurse using numbers.
- The natural base cases for integers are when you hit 0 or when you hit 1.
- The natural simplification step for recursive procedure using numbers calls typically involves subtracting 1 from the argument.
  - Other simplifications, such as dividing in half, are also possible.

Lab

- Do the lab.