

Class 29: Numeric Recursion

Held: Tuesday, 11 March 2008

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:

- Sara Bebeau from the Fund for Public Interest Research plans to give an announcement at the start of class.
- I will reserve our normal start-of-class time for comments on campus events.
- Are there final questions on the exam?
- I head off to my conference after class today and am unlikely to have email access until Wednesday afternoon. Please send questions to Dr. Davis.
- Reading for tomorrow: Geometric Art.

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.

Copyright © 2007-8 Janet Davis, Matthew Kluber, and Samuel A. Rebelsky. (Selected materials copyright by John David Stone and Henry Walker and used by permission.) This material is based upon work partially supported by the National Science Foundation under Grant No. CCLI-0633090. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. This work is licensed under a Creative Commons Attribution-NonCommercial 2.5 License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc/2.5/> or send a letter to Creative Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, USA.