University of Virginia 30 September 2011
cs1120: Introduction of Computing
Explorations in Language, Logic, and Machines

Class 17: Golden Sneezewort

Upcoming Schedule

- **Monday, 3 October:** Problem Set 4
- **Wednesday, 12 October:** Exam 1 Due (will be take-home and open book). Exam 1 will be handed out on **Friday, 7 October**. Exam 1 covers:
  - Problem Sets 1-4 including PS Comments 1-4
  - Course Book Chapters 1-6
  - Classes 1-18 (but not the new material on cost and running time)

  Information on the Assistant Coaches' review session for Exam 1 and office hours next week will be posted on the course site later today.

Generalizing Loops

Define a general loop procedure that takes 5 inputs:
- **index** (a value that is the current object for looping),
- **result** (a value that accumulates the result),
- **test** (a procedure that takes the current index as input and outputs true if the loop should continue, false otherwise),
- **update** (a procedure that takes the current index as input and outputs the next index), and
- **proc** (a procedure that takes two inputs, the current index and current result, and outputs the next result).

```
(define (loop index result test update proc)
```

```
 (define (____________ n)
   (loop 1 0 (lambda (i) (<= i n)) (lambda (i) (+ i 1)) (lambda (i res) (+ res i))))
```

```
(define (factorial n)
  (loop _____ ______ ___________________________
        __________________________________________))
```

```
(define (list-length p)
   (loop
```

```
(define (list-accumulate f base p)
   (loop
```

Measuring Cost

Hofstadter's Definition: (modified to use 0 base)

"These numbers are best defined recursively by the pair of formulas

FIBO (n) = FIBO (n – 1) + FIBO (n – 2) for n > 1

FIBO (1) = 1, FIBO (0) = 0"

Define a (simple recursive) procedure that computes the nth Fibonacci number:

Define a procedure that computes the nth Fibonacci number using **loop**:

How should computer scientists measure the *cost* of evaluating a procedure application?

Why is it more useful to understand how the cost scales with the size of the input, than to know the absolute running time for some particular inputs?

If $f(0) = c$ and $f(n) = k \cdot f(n\text{-}1)$ how can we directly compute $f(n)$? (Try $c = 1$ and $k = 2$ as an example.)