Outline for May 27, 2009

Reading: §13.2

---

1. Recursion
    a. Express problem in terms of itself
    b. Recursive definitions
        i. Base case (not recursive)
        ii. Recursive part (must eventually reduce to base case)
    c. Relationship to mathematical induction
2. Example: n! (see fact.py)
    a. Definition
        i. Base case: 0! = 1
        ii. Recursive part: n! = n(n–1)!
    b. Show stack for n = 3
3. Example: reverse string (see reverse.py)
    a. Definition
        i. Base case: empty string
        ii. Recursive part: reverse(s) = reverse(s[1:]) + s[0]
    b. Show stack for str = “yes”
4. Example: binary search (see binsearch.py)
    a. Definition
        i. Base case: high < low, return failure; word is list[mid], return mid
        ii. Recursive part: if word < list[mid], search word[0..mid-1]; if word > list[mid], search word[mid+1..high]
    b. Show for list.txt from last time
5. Example: list of permutations of string (see perm.py)
    a. Definition
        i. Base case: empty string gives list of empty string
        ii. Recursive part: for each permutation of (string without first char), put first letter of this string in each position
    b. Show stack for s = “012”
6. Example: Fibonacci numbers (see rfib.py)
    a. Definition
        i. Base case: fib(0) = 1, fib(1) = 1
        ii. Recursive part: fib(n) = fib(n–1) + fib(n–2)
    b. Show stack for n = 3