MASSACHVSETTS INSTITVTE OF TECHNOLOGY
Department of Electrical Engineering and Computer Science
6.001—Structure and Interpretation of Computer Programs
Spring Semester, 1998

Recitation – Friday, February 18

1. Student Presentation

Sometimes we have procedures that are supposed to be exact inverses of each other. For example, there might be a program that squares its arguments and another that takes the square root. One test of these programs is to see if the square root of the square of a number is the original number.

Write, and explain to the class, a program named `make-inverse-test`. This program should take in two functions, some procedure and its inverse, and it should return a function that, when used, applies the procedure, applies the inverse, and returns the result.

2. Diophantine Equation Problem

Given integers a and b, we need to find integers x and y such that $ax + by = 1$.

For this to be possible, a and b must not have any common factors (i.e. $\gcd(a, b) = 1$).

By **wishful thinking**, suppose we could solve this problem for a different pair of integers, namely, the pair of integers b and r, where $r = a \bmod b$.

- This means we would have found two integers $\overline{x}$ and $\overline{y}$ such that $b\overline{x} + r\overline{y} = 1$
- If $r = a \bmod b$ then for some q, $a = bq + r$
- We could then solve our original problem:

$$b\overline{x} + r\overline{y} = 1 \tag{1}$$
$$b\overline{x} + (a - bq)\overline{y} = 1 \tag{2}$$
$$a\overline{y} + b(\overline{x} - q\overline{y}) = 1 \tag{3}$$

Have we **reduced** our problem, or just changed it? Why?
Can we be sure that this process will always end eventually? How?
Now, how would you write this in scheme?

```
(define (solve-ax+by=1 a b)
  (if (= b 0)                          ; simple base case
      (if (= a 1)
          (list 1 0)                   ; X=1, Y is any integer
          (error "GCD wasn't 1"))
      (let ((q (quotient a b))
            (r (remainder a b)))
                                       ; find solution to simpler problem
        (let ((simpler (solve-ax+by=1 b r)))
          (let ((xbar (car simpler))    ; extract the pieces
                (ybar (cadr simpler)))
                                       ; construct and return solution
            (list ybar (- xbar (* ybar q))))))))
```

3. More Higher Order Procedures

Consider the following two functions that we've seen before. They have a lot in common.

```
(define (mult m n)
  (if (= n 0)
      0
      (+ m (mul1 m (- n 1)))))
```

```
(define (expt b e)
  (if (= e 0)
      1
      (* b (expt b (- e 1)))))
```

Can we generalize these functions using higher-order procedures? Consider the generic accumulate function below.

```
(define (accumulate combine test next ident)
  (define (recurse x y)
    (if (test y)
        ident
        (combine x (recurse x (next y)))))
  recurse)
```

How would you write `mult` using `accumulate`?

```
(define mult (accumulate + zero? decr 0))
```

How would you write `expt` using `accumulate`?

```
(define expt (accumulate * zero? decr 1))
```

How would you write `repeated` using `accumulate`?

Recall that `(repeated f n)` returns a function that takes a value `x` and returns $f^n(x)$ or $\overbrace{f(f(f(f...}^{n \text{ times}}(x))))$, so `((repeated square 3) 2) == 256`

```
(define repeated (accumulate compose zero? decr identity))
```

4. Iterative vs. Recursive

Is the above `accumulate` function **iterative** or **recursive**? How would you write a new `accumulate2` function that is the opposite?

```
(define (accumulate2 combine test next ident)
  (lambda (x y)
    (define (iter x y ans)
      (if (test y)
          ans
          (iter x (next y) (combine x ans))))
    (iter x y ident)))
```

What is the **Order of Growth** of `accumulate`?

Time = $\Theta(1)$ Space = $\Theta(1)$

What is the **Order of Growth** of `mult` defined by `accumulate`?

Time = $\Theta(n)$ Space = $\Theta(n)$

What is the **Order of Growth** of `mult` defined by `accumulate2`?

Time = $\Theta(n)$ Space = $\Theta(1)$

How would you need to change accumulate to allow for $\Theta(\log n)$ order of growth of `mult`?