April 19

Office Hours
Discussion: Problem Set #1
Today’s NTO topic: “Big O” Notation

PS1: Algorithm / Not an Algorithm

- Clearly there is no “right” answer here
- I was looking for good explanations of whether or not a problem solution is in the form of an algorithm
  (a) How to find a book:
    - very similar to cooking enchiladas -- if you think a recipe is an algorithm then getting instructions to find a book in a library is an algorithm
  (b) How to find any book:
    - if the librarian gives you instructions for using a catalog you now have an algorithm that helps you find any book (plus you might need a map)
  (c) Balancing a checkbook:
    - this wasn’t worded very well -- what I meant was “if you follow instructions to balance your checkbook...”
    - a method for balancing a checkbook is an algorithm (add up cleared checks, outstanding checks, ...)

PS1: Euclid’s Algorithm

- The best answers to this problem were tables or some other systematic way of showing values of $m$, $n$, and $r$ at each step
- I was slightly fussy -- if the table just ended with $r = 0$ without telling me what the actual output was I took off 3 points (out of 30)
- The toughest part of this problem: finding the remainder of $m / n$
  - calculators don’t seem to have a “remainder” key
- Ruby to the rescue: the expression $m \% n$ means “remainder of $m / n$”

```ruby
[fugu:conery] % irb
>> 858 % 455
=> 403
>> 455 % 403
=> 52
etc
```
Euclid’s Algorithm (cont’d)

- Check your results!
  - only one person gave any explanation for why their answer was correct
  - I was so happy to see this I gave him 5 XC points...
- Example: show that 858 / 13 and 455 / 13 are both integers
- Better yet: show
  - 858 = 2 x 3 x 11 x 13
  - 403 = 13 x 31
  - not too hard once you know 13 is the GCD: 858 / 13 = 66...
- Note: GCD(128,9) shows the algorithm terminates when the two numbers are relatively prime (i.e. GCD = 1)

PS1: Insertion Sort

- A few people got hung up on the second to last step
  - input: A = [14, 6, 9, 26, 3]
  - j key A
  - 1 6 [6, 14, 9, 26, 3]
  - 2 9 [6, 9, 14, 26, 3]
  - 3 26 [6, 9, 14, 26, 3]
  - 4 3 [3, 6, 9, 14, 26]

  - no movements, but the sorted portion has been extended...

  - Notes:
    - key takes on successive values from A (key = A[j] in a loop that increments j)
    - after each step a larger part of the array (shown in red above) is sorted
    - the sorted part is A[0] up to A[j]

Lab 1: Comments from Tom

- #6: how many duplicates are in primes?
  - primes.length tells how many numbers there are
  - primes.uniq.length tells how many unique (non-duplicated) numbers
  - primes.length - primes.uniq.length is the number of duplicates

- #9: a single Ruby expression to reverse a name
  - >> name = "Fred Flintstone"
  - => "Fred Flintstone"
  - >> name.split
  - => ["Fred", "Flintstone"]
  - >> name.split.reverse
  - => ["Flintstone", "Fred"]
  - >> name.split.reverse.join(" ")
  - => "Flintstone Fred"

Lab 1 (cont’d)

- One way to insert a middle initial: find the end of the first name, insert the initial at that location:
  - >> name.index(" ")
  - => 4
  - >> name.insert(name.index(" "), " X.")
  - => "Fred X. Flintstone"

- A second way: split the name into words, insert a new word, re-join the name:
  - >> a = [1, 2, 3]
  - => [1, 2, 3]
  - >> a.insert(1,9)
  - => [1, 9, 2, 3]
  - >> name.split.insert("X.").join(" ")
  - => "Fred X. Flintstone"
Big O Notation

- On to today's topic from NTO....