

# CSC104 fall 2012

## Why and how of computing week 5

Danny Heap

heap@cs.toronto.edu

BA4270 (behind elevators)

<http://www.cdf.toronto.edu/~heap/104/F12/>

416-978-5899

Text: **Picturing Programs**



# What to do with computing machines?

Algorithms!



simple sequence of feasible  
steps to solve a problem  
deterministic (in this course)  
credit **Al-Khwarizmi**

## Examples

- ▶ multiplication
- ▶ PBJ
- ▶ Google page rank

# Sticky algorithm

pbj

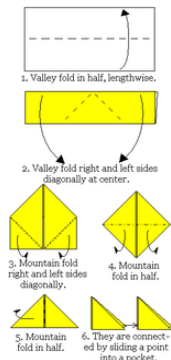


peanut butter bread jam → PBJ sandwich  
could you explain it to a friend  
over the phone, who had  
never made it?



- ▶ which operations are built-in?
- ▶ what if conditions change?
- ▶ name repeated operations
- ▶ does sequence matter?

# paper folding



(ignore the diagram on the left)  
fold over upper surface of paper strip  
after one fold, it has a downward crease  
fold the once-folded strip again  
and it has one upward, two downward  
there are good physical reasons you  
can't experiment far beyond 6 folds  
given the number of folds,  
predict the pattern

For more information, and hints, see [paper folding problem](#)

# 2000+ year-old algorithm

## Euclid's GCD



the largest whole number that divides two non-negative whole numbers is their Greatest Common Denominator (GCD) we could find it by sifting through all the divisors, but there's a quicker way

Euclid noticed that  $(\text{gcd } n_1 \ n_2) = (\text{gcd } n_2 \ (\text{remainder } n_1 \ n_2))$

Also,  $(\text{gcd } 0 \ n_1) = n_1$ . Repeat as needed.

# The way we were grade school multiplication

×	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9
2	0	2	4	6	8	10	12	14	16	18
3	0	3	6	9	12	15	18	21	24	27
4	0	4	8	12	16	20	24	28	32	36
5	0	5	10	15	20	25	30	35	40	45
6	0	6	12	18	24	30	36	42	48	54
7	0	7	14	21	28	35	42	49	56	63
8	0	8	16	24	32	40	48	56	64	72
9	0	9	18	27	36	45	54	63	72	81

We'd memorize, and organize, the algorithm for  $27 \times 38$   
Much better than XXVII  $\times$  XXXVIII

# Notes