CSC104 fall 2013
Why and how of computing
week 1

Danny Heap
heap@cs.toronto.edu
BA4270 (behind elevators)
http://www.cdf.toronto.edu/~heap/104/W13/
416-978-5899

Text: Picturing Programs
http://www.picturingprograms.com
Outline

Introduction

Algorithms

Notes
Who needs computational thinking?

- We all consume computing, the thing is to change it
- Computers and networks change society — privacy, property, democracy, work, education — for better or worse
- We get an insight into computer culture by making some artifacts: programs

This is why Dr Racket
Two tracks in this course

- Insight into computing mindset: problem-solving and programs

- History of computing technology, overview of modern computing OS, social issues
How to do well at this course

- Read the course information sheet as a two-way promise

- Humour me: read your email

- Question, answer, record, synthesize

- Collaborate with respect
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?

careful!

otherwise, explain "scoop"
so we can re-use them
spread PB before putting
bread from bag
paper folding

1 fold: \( \checkmark \)
2 folds: \( \checkmark \)

(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 \((\text{gcd } n1\ n2) = (\text{gcd } n2\ (\text{remainder } n1\ n2))\)
Also, \((\text{gcd } n1\ 0) = n1\). Repeat as needed.
The way we were

grade school multiplication

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

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