Today

- Binary!
- Intro to algorithms and pseudocode
- Discussion of readings

Reminders:
- Leading discussions
- Blog posts!
- Bring laptops with face project ready to show off on Thursday.

Steps in solving a computational task

- Design an algorithm: A precise, unambiguous procedure for solving a computational task. (We will express our algorithms in pseudocode.)
- Turn pseudocode into computer program.

Rules

- Can pick up a cup
- Can compare the price on the cup in hand with the price of a cup on the table.
- Can swap the cup in hand with a cup on the table.
- Should stop when the cup in hand is guaranteed to be the minimum priced one.
- give all the instructions at the beginning.
- should work no matter how many cups there are.

Solution

- Pick up first bottle, check price
- Walk down aisle. For each bottle, do this:
  - If price on bottle is less than price in hand, exchange for one in hand.
How can we describe an algorithm precisely enough so there is no ambiguity?

We will do this using "pseudocode".

Why is arithmetic so important for computers?

Because to a computer everything is a number!

Now let’s solve the problem as it would be done on a computer

There are \( n \) prices stored in memory

Want to find minimum price

Memory

- A simplified view of computer memory: a scratchpad that can be perfectly erased and re-written any number of times
- A variable: a piece of memory with a name; stores a “value”

\[
i = 22.99
\]

Examples

\[
i = 5 \quad \text{Sets } i \text{ to value 5}
\]

\[
\begin{align*}
  j &= i \\
  &\text{Sets } j \text{ to whatever value is in } i. \\
  &\text{Leaves } i \text{ unchanged}
\end{align*}
\]

\[
i = j + 1 \quad \text{Sets } i \text{ to } j + 1. \\
  &\text{Leaves } j \text{ unchanged}
\]

\[
i = i + 1 \quad \text{Sets } i \text{ to 1 more than it was.}
\]
Arrays

- $A$ is an array of $n$ values, $A[i]$ is the $i$'th value

\[
A = \begin{array}{c|c|c|c}
40.99 & 62.99 & 52.99 & - \\
\hline
22.99
\end{array}
\]


Pseudocode

- Variables and arrays
- Assignment
- Simple instructions: involve $+$, $-$, $\times$, $\div$, …
- Tests for $==$, $<$, $>$, …
- Compound instructions
  - Conditionals
  - Loops

Now we can express our solution in pseudocode

- Pick up first bottle, check price
- Walk down aisle. For each bottle, do this:
  - If price on bottle is less than price in hand, exchange for one in hand.

Procedure findmin (in pseudocode)

- Input: $n$ values, stored in array $A$
- Variables are $i$, $best$
- $best \leftarrow 1$
- for ($i = 2$ to $n$)
    - $best \leftarrow i$

Another way to do the same

\[
best = 1;
\]
\[
i = 1
\]
while ($i < n$)
\[
\{
\]
\[
i = i + 1;
\]
\[
if (A[i] < A[best]) then
\]
\[
\{ best \leftarrow i \}
\]
\]
New problem for robot: sorting

Arrange them so prices increase from left to right (or top to bottom).

Solution

Do for $i=1$ to $n-1$

\{ 
  Find cheapest bottle among those numbered $i$ to $n$

  Swap that bottle and the $i$'th bottle.
\}

“selection sort”

Exercise for Thursday: Try to write pseudocode for selection sort.

Swapping

- Suppose $x$ and $y$ are variables. How do you swap their values?
- Need extra variable!

\[
\begin{align*}
  \text{tmp} & \leftarrow x \\
  x & \leftarrow y \\
  y & \leftarrow \text{tmp}
\end{align*}
\]

Efficiency of this sorting algorithm?

Measure the efficiency of an algorithm in terms of the number of elementary operations it performs.

Find a number in a sorted list, if it’s there

- Underneath each cup is a ping pong ball.
- On each ping pong ball is a number.
- The numbers are in sorted order.

Goal: to look at as few ping pong balls as possible!

Efficiency Matters

<table>
<thead>
<tr>
<th>$n$</th>
<th>$k$</th>
<th>$n$</th>
<th>$k$</th>
<th>$n$</th>
<th>$k$</th>
<th>$n$</th>
<th>$k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>2</td>
<td>1000</td>
<td>2</td>
<td>10000</td>
<td>2</td>
<td>10000</td>
<td>2</td>
</tr>
<tr>
<td>0.1 sec</td>
<td>1 sec</td>
<td>0.1 sec</td>
<td>1 sec</td>
<td>0.1 sec</td>
<td>1 sec</td>
<td>0.1 sec</td>
<td>1 sec</td>
</tr>
<tr>
<td>30 min</td>
<td>10 yeas</td>
<td>12,800 years</td>
<td>107 years</td>
<td>108 years</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>very long</td>
<td>very long</td>
<td>very long</td>
<td>very long</td>
<td>very long</td>
<td>very long</td>
<td>very long</td>
<td>very long</td>
</tr>
</tbody>
</table>
How to multiply $2^n$-bit numbers.

Aside: History of Algorithm

- Named for Abu Abdullah Muhammad bin Musa al-Khwarizmi (780-850 AD)
  - His book "Al-Jabr wa-al-Muqabilah" evolved into today’s high school algebra text.
- Notion of algorithm has existed for at least 2000 years (in Hindu, Chinese, and Greek traditions)
- “Variables” in algebra come from the same tradition.

Algorithm defn; revisited

“Pseudocode for turning a set of inputs into outputs in a finite amount of time”

Questions to think about:
- What class of computational tasks can be solved by algorithms?
- How dependent is this class on the exact definition of pseudocode?