Announcements

- Homework 7 online (due Monday)
- TA Applications due today
- Sunday office hours resume (at 3:00)
- Prof. Sorelle Friedler, Haverford College, Friday, 2:30, TCL 202,
  Biased Data, Biased Algorithms: Detecting and Preventing Discrimination in Machine-Learned Decisions
Today’s Plan

- Computing Huffman Code Efficiency
- Extends Explained
- Arrays of things
<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>G</td>
<td>H</td>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>N</td>
<td>O</td>
<td>P</td>
<td>Q</td>
<td>R</td>
<td></td>
</tr>
</tbody>
</table>

-

<table>
<thead>
<tr>
<th></th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>G</td>
<td>H</td>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
<td>G</td>
</tr>
<tr>
<td>N</td>
<td>O</td>
<td>P</td>
<td>Q</td>
<td>R</td>
<td>M</td>
<td></td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th></th>
<th>A-B</th>
<th>B-C</th>
<th>C-D</th>
<th>D-E</th>
<th>E-F</th>
<th>F-A</th>
</tr>
</thead>
<tbody>
<tr>
<td>G-H</td>
<td>H-I</td>
<td>I-J</td>
<td>J-K</td>
<td>K-L</td>
<td>L-G</td>
<td></td>
</tr>
<tr>
<td>M-N</td>
<td>N-O</td>
<td>O-P</td>
<td>P-Q</td>
<td>Q-R</td>
<td>R-M</td>
<td></td>
</tr>
</tbody>
</table>

+    | A  |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>G</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td></td>
<td>A</td>
</tr>
<tr>
<td>---</td>
<td>----</td>
</tr>
<tr>
<td>G</td>
<td>H</td>
</tr>
<tr>
<td>M</td>
<td>N</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>G</td>
<td>H</td>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
<td>G</td>
</tr>
<tr>
<td>N</td>
<td>O</td>
<td>P</td>
<td>Q</td>
<td>R</td>
<td>M</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>A-B</th>
<th>B-C</th>
<th>C-D</th>
<th>D-E</th>
<th>E-F</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>G-H</td>
<td>H-I</td>
<td>I-J</td>
<td>J-K</td>
<td>K-L</td>
<td>L</td>
<td></td>
</tr>
<tr>
<td>M-N</td>
<td>N-O</td>
<td>O-P</td>
<td>P-Q</td>
<td>Q-R</td>
<td>G</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>G</td>
<td></td>
<td>G</td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
</tr>
<tr>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
</tr>
<tr>
<td>G</td>
<td>H</td>
<td>I</td>
<td>J</td>
</tr>
<tr>
<td>M</td>
<td>N</td>
<td>O</td>
<td>P</td>
</tr>
</tbody>
</table>

-

<table>
<thead>
<tr>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
<td>G</td>
</tr>
<tr>
<td>N</td>
<td>O</td>
<td>P</td>
<td>Q</td>
<td>R</td>
<td>M</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A-B</th>
<th>B-C</th>
<th>C-D</th>
<th>D-E</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>G-H</td>
<td>H-I</td>
<td>I-J</td>
<td>J-K</td>
<td>K</td>
<td>L</td>
</tr>
<tr>
<td>M-N</td>
<td>N-O</td>
<td>O-P</td>
<td>P-Q</td>
<td>Q</td>
<td>G</td>
</tr>
</tbody>
</table>

+   | A   |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>G</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>----</td>
<td>----</td>
</tr>
<tr>
<td>G</td>
<td>H</td>
</tr>
<tr>
<td>M</td>
<td>N</td>
</tr>
</tbody>
</table>

-

<table>
<thead>
<tr>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
<td>G</td>
</tr>
<tr>
<td>N</td>
<td>O</td>
<td>P</td>
<td>Q</td>
<td>R</td>
<td>M</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A-B</th>
<th>B-C</th>
<th>C-D</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>G-H</td>
<td>H-I</td>
<td>I-J</td>
<td>J</td>
<td>K</td>
<td>L</td>
</tr>
<tr>
<td>M-N</td>
<td>N-O</td>
<td>O-P</td>
<td>P</td>
<td>Q</td>
<td>G</td>
</tr>
</tbody>
</table>

+ A

  G

  M
going going going gone

- e = 000
- _ = 001
- o = 01
- i = 100
- n = 101
- g = 11
<table>
<thead>
<tr>
<th>Symbol</th>
<th>n</th>
<th>o</th>
<th>i</th>
<th>g</th>
<th>e</th>
<th>space</th>
</tr>
</thead>
<tbody>
<tr>
<td>occurrences</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Symbol</td>
<td>n</td>
<td>o</td>
<td>i</td>
<td>g</td>
<td>e</td>
<td>space</td>
</tr>
<tr>
<td>--------</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>-------</td>
</tr>
<tr>
<td>occurrences</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Symbol</td>
<td>n</td>
<td>o</td>
<td>i</td>
<td>g</td>
<td>e</td>
<td>space</td>
</tr>
<tr>
<td>--------</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>-------</td>
</tr>
<tr>
<td>occurrences</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Symbol</th>
<th>n</th>
<th>o</th>
<th>i</th>
<th>g</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>occurrences</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>Symbol</td>
<td>n</td>
<td>o</td>
<td>i</td>
<td>g</td>
<td>e</td>
</tr>
<tr>
<td>--------</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td><strong>occurrences</strong></td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>Symbol</td>
<td>n</td>
<td>o</td>
<td>i</td>
<td>g</td>
<td>e</td>
</tr>
<tr>
<td><strong>occurrences</strong></td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>3</td>
</tr>
</tbody>
</table>
| Symbol | n | o | i | g | e-
|--------|---|---|---|---|---
<p>| occurrences | 3 | 3 | 2 | 5 | 3 |</p>
<table>
<thead>
<tr>
<th>Symbol</th>
<th>n</th>
<th>o</th>
<th>i</th>
<th>g</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>occurrences</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Symbol</th>
<th>0(^1) _n</th>
<th>o</th>
<th>g</th>
<th>0(^1) _e</th>
</tr>
</thead>
<tbody>
<tr>
<td>occurrences</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>Symbol</td>
<td>Tree 1 (Occurrences: 10)</td>
<td>Tree 2 (Occurrences: 6)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>--------------------------</td>
<td>-------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Symbol</td>
<td>Tree 1 (Occurrences: 16)</td>
<td>Tree 2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
going going gone

- $e = 000$
- $_ = 001$
- $o = 01$
- $i = 100$
- $n = 101$
- $g = 11$

<table>
<thead>
<tr>
<th>Symbol</th>
<th>n</th>
<th>o</th>
<th>i</th>
<th>g</th>
<th>e</th>
<th>space</th>
</tr>
</thead>
<tbody>
<tr>
<td>occurrences</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>
going going gone

- \(e = 000\)
- \(_ = 001\)
- \(o = 01\)
- \(i = 100\)
- \(n = 101\)
- \(g = 11\)

<table>
<thead>
<tr>
<th>Symbol</th>
<th>n</th>
<th>o</th>
<th>i</th>
<th>g</th>
<th>e</th>
<th>space</th>
</tr>
</thead>
<tbody>
<tr>
<td>occurrences</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>
going going gone

- **e** = 000
- **_** = 001
- **o** = 01
- **i** = 100
- **n** = 101
- **g** = 11

<table>
<thead>
<tr>
<th>Symbol</th>
<th>n</th>
<th>o</th>
<th>i</th>
<th>g</th>
<th>e</th>
<th>space</th>
</tr>
</thead>
<tbody>
<tr>
<td>occurrences</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>
going going gone

- $e = 000$
- $\_ = 001$
- $o = 01$
- $i = 100$
- $n = 101$
- $g = 11$

<table>
<thead>
<tr>
<th>Symbol</th>
<th>n</th>
<th>o</th>
<th>i</th>
<th>g</th>
<th>e</th>
<th>space</th>
</tr>
</thead>
<tbody>
<tr>
<td>occurrences</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>
3 + 5 + 6 + 10 + 16 = 40

<table>
<thead>
<tr>
<th>Symbol</th>
<th>n</th>
<th>o</th>
<th>i</th>
<th>g</th>
<th>e</th>
<th>space</th>
</tr>
</thead>
<tbody>
<tr>
<td>occurrences</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Symbol</td>
<td>n</td>
<td>o</td>
<td>i</td>
<td>g</td>
<td>e</td>
<td>space</td>
</tr>
<tr>
<td>------------</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>-------</td>
</tr>
<tr>
<td>occurrences</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Bits used = 0
<table>
<thead>
<tr>
<th>Symbol</th>
<th>n</th>
<th>o</th>
<th>i</th>
<th>g</th>
<th>e</th>
<th>space</th>
</tr>
</thead>
<tbody>
<tr>
<td>occurrences</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Bits used = 0
<table>
<thead>
<tr>
<th>Symbol</th>
<th>n</th>
<th>o</th>
<th>i</th>
<th>g</th>
<th>e</th>
<th>space</th>
</tr>
</thead>
<tbody>
<tr>
<td>occurrences</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Bits used = 0
<table>
<thead>
<tr>
<th>Symbol</th>
<th>n</th>
<th>o</th>
<th>i</th>
<th>g</th>
<th>e</th>
<th>space</th>
</tr>
</thead>
<tbody>
<tr>
<td>occurrences</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Bits used = 0

<table>
<thead>
<tr>
<th>Symbol</th>
<th>n</th>
<th>o</th>
<th>i</th>
<th>g</th>
<th>e</th>
<th>0 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>occurrences</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

Bits used = 3
<table>
<thead>
<tr>
<th>Symbol</th>
<th>n</th>
<th>o</th>
<th>i</th>
<th>g</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>occurrences</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

Bits used = 3
<table>
<thead>
<tr>
<th>Symbol</th>
<th>occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>3</td>
</tr>
<tr>
<td>o</td>
<td>3</td>
</tr>
<tr>
<td>i</td>
<td>2</td>
</tr>
<tr>
<td>g</td>
<td>5</td>
</tr>
</tbody>
</table>

**Bits used = 0**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>5</td>
</tr>
<tr>
<td>n</td>
<td>3</td>
</tr>
<tr>
<td>o</td>
<td>3</td>
</tr>
<tr>
<td>g</td>
<td>5</td>
</tr>
</tbody>
</table>

**Bits used = 3**
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>3</td>
</tr>
<tr>
<td>o</td>
<td>3</td>
</tr>
<tr>
<td>i</td>
<td>2</td>
</tr>
<tr>
<td>g</td>
<td>5</td>
</tr>
<tr>
<td>e</td>
<td>3</td>
</tr>
</tbody>
</table>

**Bits used:**
- For symbol `n`: 3 bits
- For symbol `i`: 2 bits
- Total bits used: $3 + 2 = 5$

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
</tr>
</tbody>
</table>

**Bits used:**
- For symbol `0`: 3 bits
- For symbol `1`: 5 bits
- Total bits used: $3 + 5 = 8$
<table>
<thead>
<tr>
<th>Symbol</th>
<th>(0)</th>
<th>(1)</th>
<th>(0)</th>
<th>(g)</th>
<th>(0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>occurrences</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

Bits used = 3 + 5 = 8
<table>
<thead>
<tr>
<th>Symbol</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td></td>
<td>n</td>
</tr>
<tr>
<td>o</td>
<td></td>
<td></td>
</tr>
<tr>
<td>g</td>
<td></td>
<td></td>
</tr>
<tr>
<td>e</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Symbol</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td></td>
<td>n</td>
</tr>
<tr>
<td>e</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Symbol</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td></td>
<td>n</td>
</tr>
<tr>
<td>e</td>
<td></td>
<td></td>
</tr>
<tr>
<td>o</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Symbol</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td></td>
<td>n</td>
</tr>
<tr>
<td>e</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Symbol</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td></td>
<td>n</td>
</tr>
<tr>
<td>e</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Symbol</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td></td>
<td>n</td>
</tr>
<tr>
<td>o</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Symbol</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td></td>
<td>n</td>
</tr>
<tr>
<td>e</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Symbol</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td></td>
<td>n</td>
</tr>
<tr>
<td>e</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Symbol</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td></td>
<td>n</td>
</tr>
<tr>
<td>e</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Symbol</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td></td>
<td>n</td>
</tr>
<tr>
<td>e</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**occurrences**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>5</th>
<th>3</th>
<th>5</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>o</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>g</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>e</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Symbol</th>
<th>5</th>
<th>6</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>e</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>g</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Bits used = 3 + 5 = 8**

**Bits used = 3 + 5 + 6 = 14**
<table>
<thead>
<tr>
<th>Symbol</th>
<th>0 1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>i n</td>
</tr>
<tr>
<td></td>
<td>ε</td>
</tr>
<tr>
<td>0 1 c</td>
<td>g</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Symbol</th>
<th>0 1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>5</td>
</tr>
</tbody>
</table>

**Bits used** = 3 + 5 + 6 = 14
<table>
<thead>
<tr>
<th>Symbol</th>
<th>occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>5</td>
</tr>
<tr>
<td>n</td>
<td>6</td>
</tr>
<tr>
<td>g</td>
<td>5</td>
</tr>
</tbody>
</table>

Bits used = 3 + 5 + 6 = 14

<table>
<thead>
<tr>
<th>Symbol</th>
<th>occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>10</td>
</tr>
<tr>
<td>n</td>
<td>6</td>
</tr>
<tr>
<td>g</td>
<td>0</td>
</tr>
</tbody>
</table>

Bits used = 3 + 5 + 6 + 10 = 24
<table>
<thead>
<tr>
<th>Symbol</th>
<th>occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>6</td>
</tr>
</tbody>
</table>

**Bits used = 3 + 5 + 6 + 10 = 24**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>16</td>
</tr>
</tbody>
</table>

**Bits used = 3 + 5 + 6 + 10 + 16 = 40**
going going going gone

- $e = 000$
- $=_ = 001$
- $o = 01$
- $i = 100$
- $n = 101$
- $g = 11$
| occurrences | 3 | 3 | 2 | 5 | 1 | 2 |

**Bits used = 0**
<table>
<thead>
<tr>
<th>occurrences</th>
<th>3</th>
<th>3</th>
<th>2</th>
<th>5</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
</table>

Bits used = 0

<table>
<thead>
<tr>
<th>occurrences</th>
<th>3</th>
<th>3</th>
<th>2</th>
<th>5</th>
<th>3</th>
</tr>
</thead>
</table>

Bits used = 3
<table>
<thead>
<tr>
<th>occurrences</th>
<th>3</th>
<th>3</th>
<th>2</th>
<th>5</th>
<th>3</th>
</tr>
</thead>
</table>

Bits used = 3 + 5 = 8
<table>
<thead>
<tr>
<th>occurrences</th>
<th>5</th>
<th>3</th>
<th>5</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bits used</td>
<td></td>
<td>3 + 5 = 8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>occurrences</th>
<th>5</th>
<th>6</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bits used</td>
<td></td>
<td>3 + 5 + 6 = 14</td>
<td></td>
</tr>
<tr>
<td>occurrences</td>
<td>5</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>-------------</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>Bits used</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 + 5 + 6 = 14</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>occurrences</th>
<th>10</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bits used</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 + 5 + 6 + 10 = 24</td>
<td></td>
</tr>
<tr>
<td>occurrences</td>
<td>10</td>
<td>6</td>
</tr>
<tr>
<td>-------------</td>
<td>----</td>
<td>----</td>
</tr>
</tbody>
</table>

Bits used = 3 + 5 + 6 + 10 = 24

<table>
<thead>
<tr>
<th>occurrences</th>
<th>16</th>
</tr>
</thead>
</table>

Bits used = 3 + 5 + 6 + 10 + 16 = 40
AN ARRAY?

\[
\text{occurrences} = \begin{array}{cccccc}
3 & 3 & 2 & 5 & 1 & 2 \\
\end{array}
\]
ARRAYS DON’T SHRINK!!!

\[
\text{occurrences} = \begin{bmatrix}
3 & 3 & 2 & 5 & 1 & 2 \\
5 & 3 & 2 & 3 & 3 & 3 \\
\end{bmatrix}
\]

Bits used = 3
<table>
<thead>
<tr>
<th>occurrences =</th>
<th>3</th>
<th>3</th>
<th>2</th>
<th>5</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>bits used =</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>occurrences =</th>
<th>3</th>
<th>3</th>
<th>2</th>
<th>5</th>
<th>1</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>bits used =</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
occurrences = 

\[
\begin{array}{cccccc}
3 & 3 & 2 & 5 & 1 & 2 \\
\end{array}
\]

Symbols left = 5

Bits used = 3
Holey Cow!

occurrences = 3 3 2 5 3 1

Bits used = 3 + 5 = 8
AVOIDING HOLEY ARRAYS

occurrences =

3 3 2 5 1 2

Symbols left = 6

Bits used = 0
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
</tbody>
</table>

Symbols left = 6
Bits used = 0

Symbols left = 5
Bits used = 3
SWAP THE SMALLEST WITH THE LAST REMAINING

Symbols left = 5
Bits used = 3

Symbols left = 4
Bits used = 3
COMBINE 1 OF THE 2ND SMALLEST WITH SMALLEST

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Symbols left = 4

Bits used = 3 + 5 = 8
occurrences = 3

Symbols left = 4

Bits used = 3 + 5 = 8
SWAP THE SMALLEST WITH THE LAST REMAINING

Symbols left = 3

Bits used = 8
occurrences = 5 3 3 5 2 2

Symbols left = 3

occurrences = 5 5 3 3 2 2

Bits used = 8
COMBINE 1 OF THE 2ND SMALLEST WITH SMALLEST

**occurrences**

<table>
<thead>
<tr>
<th></th>
<th>5</th>
<th>5</th>
<th>3</th>
<th>3</th>
</tr>
</thead>
</table>

Symbols left = 3   

<table>
<thead>
<tr>
<th></th>
<th>5</th>
<th>5</th>
<th>6</th>
<th>3</th>
<th>2</th>
<th>2</th>
</tr>
</thead>
</table>

Bits used = 8+6 = 14
<table>
<thead>
<tr>
<th>occurences</th>
<th>5</th>
<th>5</th>
<th>3</th>
<th>3</th>
<th>2</th>
<th>2</th>
</tr>
</thead>
</table>

Bits used = 8 + 6 = 14

<table>
<thead>
<tr>
<th>occurences</th>
<th>5</th>
<th>5</th>
<th>6</th>
<th>3</th>
<th>2</th>
<th>2</th>
</tr>
</thead>
</table>

Symbols left = 3
SWAP THE SMALLEST WITH THE LAST REMAINING

occurrences = 5 5 6

Symbols left = 2 Bits used = 14
occurrences = 5 5 6

Symbols left = 2  Bits used = 14
COMBINE 1 OF THE 2ND SMALLEST WITH SMALLEST

Symbols left = 2  

Bits used = 14+10 = 24
occurrences = 6

symbols used = 14 + 10 = 24

occurrences = 6

symbols left = 2
SWAP THE SMALLEST WITH THE LAST REMAINING

occurrences = 6 10

Symbols left = 1

Bits used = 24
COMBINE 1 OF THE 2ND SMALLEST WITH SMALLEST

Symbols left = 1   Bits used = 24 + 16 = 40
public class QuantizingFilter {

    // Number of brightness levels desired
    private int levels;

    // Create a filter that will quantize to a specific number of levels
    public QuantizingFilter( int theLevels ) {
        levels = theLevels;
    }

    // Produce an image using a reduced number of brightness levels
    public SImage filter( SImage original ) {
        return new SImage( layerFilter( original, SImage.RED ),
                           layerFilter( original, SImage.GREEN ),
                           layerFilter( original, SImage.BLUE )
        );
    }

    // Given an image and a specified color layer, produce a copy of that color layer with a reduced number of brightness levels
    public int[][] layerFilter( SImage orig, int layer ) {
        return pixelArrayFilter( orig.getPixelArray( layer ),
                                 orig.getWidth(), orig.getHeight() );
    }

    // Return an unchanged array of brightness values
    public int[][] pixelArrayFilter( int[][] shades, int width, int height ) {
        int[][] result = new int[width][height];
        for ( int x = 0; x < result.length; x++ ) {
            for ( int y = 0; y < result[0].length; y++ ) {
                result[x][y] = onePixelFilter( shades, x, y );
            }
        }
        return result;
    }

    // Return an unchanged brightness value
    public int onePixelFilter( int[][] shades, int x, int y ) {
        return shades[x][y];
    }
}

public class ClearFilter {

    // Create a filter
    public ClearFilter( ) {
    }

    // Given an image, produce a copy
    public SImage filter( SImage original ) {
        return new SImage( layerFilter( original, SImage.RED ),
                            layerFilter( original, SImage.GREEN ),
                            layerFilter( original, SImage.BLUE )
        );
    }

    // Extract the specified color pixel array and specify the result size
    public int[][] layerFilter( SImage orig, int layer ) {
        return pixelArrayFilter( orig.getPixelArray( layer ),
                                 orig.getWidth(), orig.getHeight() );
    }

    // Return an unchanged array of brightness values
    public int[][] pixelArrayFilter( int[][] shades, int width, int height ) {
        int[][] result = new int[width][height];
        for ( int x = 0; x < result.length; x++ ) {
            for ( int y = 0; y < result[0].length; y++ ) {
                result[x][y] = onePixelFilter( shades, x, y );
            }
        }
        return result;
    }

    // Return an unchanged brightness value
    public int onePixelFilter( int[][] shades, int x, int y ) {
        return shades[x][y];
    }
}
public class QuantizingFilter {

    // Number of brightness levels desired
    private int levels;

    // Create a filter that will quantize to a specific number of levels
    public QuantizingFilter( int theLevels ) {
        levels = theLevels;
    }

    // Produce an image using a reduced number of brightness levels
    public SImage filter( SImage original ) {
        return new SImage( layerFilter( original, SImage.RED ),
                           layerFilter( original, SImage.GREEN ),
                           layerFilter( original, SImage.BLUE )
        );
    }

    // Given an image and a specified color layer, produce a copy of that color
    // layer with a reduced number of brightness levels
    public int [][] layerFilter( SImage orig, int layer ) {
        return pixelArrayFilter( orig.getPixelArray( layer ),
                                 orig.getWidth(), orig.getHeight() );
    }

    // Given an array of brightness values, produce a copy of that array with
    // a reduced number of brightness levels
    public int [][] pixelArrayFilter( int [] [] shades, int width, int height ) {
        int [] [] result = new int[width][height];
        for ( int x = 0; x < result.length; x++ ) {
            for ( int y = 0; y < result[0].length; y++ ) {
                result[x][y] = onePixelFilter( shades, x, y );
            }
        }
        return result;
    }

    // Map the value of the specified pixel to new brightness level
    public int onePixelFilter( int [] [] shades, int x, int y ) {
        return shades[x][y]*levels/256*256/levels;
    }
}

public class ClearFilter {

    // Create a filter
    public ClearFilter( ) {
    }

    // Given an image, produce a copy
    public SImage filter( SImage original ) {
        return new SImage( layerFilter( original, SImage.RED ),
                           layerFilter( original, SImage.GREEN ),
                           layerFilter( original, SImage.BLUE )
        );
    }

    // Extract the specified color pixel array and specify the result size
    public int [][] layerFilter( SImage orig, int layer ) {
        return pixelArrayFilter( orig.getPixelArray( layer ),
                                 orig.getWidth(), orig.getHeight() );
    }

    // Return an unchanged array of brightness values
    public int [][] pixelArrayFilter( int [] [] shades, int width, int height ) {
        int [] [] result = new int[width][height];
        for ( int x = 0; x < result.length; x++ ) {
            for ( int y = 0; y < result[0].length; y++ ) {
                result[x][y] = onePixelFilter( shades, x, y );
            }
        }
        return result;
    }

    // Return an unchanged brightness value
    public int onePixelFilter( int [] [] shades, int x, int y ) {
        return shades[x][y];
    }
}

public class QuantizingFilter extends ClearFilter {

    // Number of brightness levels desired
    private int levels;

    // Create a filter that will quantize to a specific number of levels
    public QuantizingFilter( int theLevels ) {
        levels = theLevels;
    }

    // Produce an image using a reduced number of brightness levels
    public SImage filter( SImage original ) {
        return new SImage( layerFilter( original, SImage.RED ),
                           layerFilter( original, SImage.GREEN ),
                           layerFilter( original, SImage.BLUE )
        );
    }

    // Given an image and a specified color layer, produce a copy of that color
    // layer with a reduced number of brightness levels
    public int [][] layerFilter( SImage orig, int layer ) {
        return pixelArrayFilter( orig.getPixelArray( layer ),
                                 orig.getWidth(), orig.getHeight() );
    }

    // Given an array of brightness values, produce a copy of that array with
    // a reduced number of brightness levels
    public int [][] pixelArrayFilter( int [] [] shades, int width, int height ) {
        int [] [] result = new int[width][height];
        for ( int x = 0; x < result.length; x++ ) {
            for ( int y = 0; y < result[0].length; y++ ) {
                result[x][y] = onePixelFilter( shades, x, y );
            }
        }
        return result;
    }

    // Map the value of the specified pixel to new brightness level
    public int onePixelFilter( int [] [] shades, int x, int y ) {
        return shades[x][y]*levels/256*256/levels;
    }
}