CS180 - Huffman trees
Idea

We want to transmit information using as few bits as possible.

Standard ASCII: 8 bits
So—how can we do better?
Run length encoding
What if we don’t use every character?

or if some are less common

take more common letters & give them fewer bits
Prefix-free codes

An unambiguous way to send information when we have characters that are not of a fixed length.

No letter's code is the prefix of another letter.

Encode: BAN (end)
1000111101 letters are all leaves
Decode:

$100011011010\overline{1}$

BANANA Eom

Wheel of fortune: S, T, R, E
So how do we do this? With exact frequency counts!

This sentence contains three a’s, three c’s, two d’s, twenty-six e’s, five f’s, three g’s, eight h’s, thirteen i’s, two l’s, sixteen n’s, nine o’s, six r’s, twenty-seven s’s, twenty-two t’s, two u’s, five v’s, eight w’s, four x’s, five y’s, and only one z.

\[
\begin{array}{cccccc}
A & C & D & E & F \\
3 & 3 & 2 & 26 & 5 \\
\end{array}
\]

Goal: Use fewest bits to encode this.
Using frequency counts, build one of these trees.

| A | C | D | E | F | G | H | I | L | N | O | R | S | T | U | V | W | X | Y | Z |
| 3 | 3 | 2 | 26| 5 | 3 | 8 | 13| 2 | 16| 9 | 6 | 27| 22| 2 | 5 | 8 | 4 | 5 | 1 |

Which ones should get few bits?

S and e

flip - who uses lots of bits?
Huffman's algorithm

Take the two least frequent characters.

Merge them into 1 letter, which becomes a new "leaf".

repeat.

make them leaves
Example:

```
| A | C | D | E | F | G | H | I | L | N | O | R | S | T | U | V | W | X | Y |
| 3 | 3 | 2 | 26| 5 | 3 | 8 | 13| 2 | 16| 9 | 6 | 27| 22| 2 | 5 | 8 | 4 | 5 | 1 |
```

Merge D & Z:

```
| A | C | E | F | G | H | I | L | N | O | R | S | T | U | V | W | X | Y | Z |
| 3 | 3 | 26| 5 | 3 | 8 | 13| 2 | 16| 9 | 6 | 27| 22| 2 | 5 | 8 | 4 | 5 | 3 |
```

Diagram:

[Diagram of network or connections between elements]
Next?
In end, build a tree:
Using the tree:
How many bits?

| freq. | 3 | 3 | 2 | 26| 5 | 3 | 8 | 13| 2 | 16| 9 | 6 | 27| 22| 2 | 5 | 8 | 4 | 5 | 1 |
| depth | 6 | 6 | 7 | 3 | 5 | 6 | 4 | 4 | 7 | 3 | 4 | 4 | 2 | 4 | 7 | 5 | 4 | 6 | 5 | 7 |
| total | 18| 18| 14| 78| 25| 18| 32| 52| 14| 48| 36| 24| 54| 88| 14| 25| 32| 24| 25| 7 |

\[ \text{total} = 646 \text{ bits} \]

How many bits would ASCII use to send these 170 letters?

\[ 170 \times 8 = 1360 \]
Exercise: 01001111 000010 000010 0001

Message: Hello

How many bits? 26 versus $5 \times 9 = 40$
Thm: Huffman codes are optimal in the sense that they use the fewest # of bits possible.

(60 take 314 to see the proof, or read supplemental notes on the schedule page.)

This is a greedy algorithm.
Next program: Decode

Given an input which describes a tree and a set of bits which are a message:

1) Create the tree
2) Use it to decode the message