Exercise 2.70.

The following eight-symbol alphabet with associated relative frequencies was designed to efficiently encode the lyrics of 1950s rock songs. (Note that the "symbols" of an "alphabet" need not be individual letters.)

A	2	NA	16
BOOM	1	SHA	3
GET	2	YIP	9
JOB	2	WAH	1

Use `generate-huffman-tree` (exercise 2.69) to generate a corresponding Huffman tree, and use `encode` (exercise 2.68) to encode the following message:

Get a job
Sha na na na na na na na
Get a job
Sha na na na na na na na
Wah yip yip yip yip yip yip yip yip yip
Sha boom

How many bits are required for the encoding? What is the smallest number of bits that would be needed to encode this song if we used a fixed-length code for the eight-symbol alphabet?

Answer.

The corresponding Huffman tree generated by `generate-huffman-tree` procedure is:

```
((leaf na 16)
 ((leaf yip 9)
  (((leaf a 2)
    ((leaf wah 1) (leaf boom 1) (wah boom) 2)
    (a wah boom)
    4)
   ((leaf sha 3)
    ((leaf job 2) (leaf get 2) (job get) 4)
    (sha job get)
    7)
   (a wah boom sha job get)
   11)
  (yip a wah boom sha job get)
  20)
 (na yip a wah boom sha job get)
 36)
```

Figure 1 provides a more intuitive view on the encoding Huffman tree.

By using `encode` to to encode the designated message, we obtain:

1 1 1 1 1 1 1 0 0 1 1 1 1 0
1 1 1 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 0 0 1 1 1 1 0
1 1 1 0 0 0 0 0 0 0 0 0 0 0
1 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
1 1 1 0 1 1 0 1 1

It takes up 84 bits for this encoding.

On the other hand, if we use a fixed-length code for the eight-symbol alphabet to encode this song, we will need to use $\log_2 8 = 3$ bits per symbol. As the lyrics contains 36 symbols, hence we need to take at least $36 \times 3 = 108$ bits.

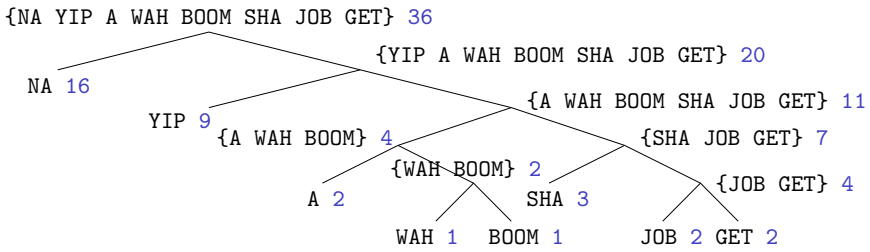

Figure 1. Huffman tree generated from the eight-symbol alphabet.