$ cryptography.py

Cryptography Challenge

// before we can break codes, let's see how data gets encoded in the first place

— Dr. Hughes

What Does "Encoding" Mean?

- Encoding just means writing the same information in a different way.
- You already do it every day: "lol", "omw", 👍 — shorter or different symbols that still carry the same meaning.
- Morse code, sign language, traffic lights, binary — all different ways of encoding a message.
- **Compression** is encoding with a goal: say the same thing using as few symbols as possible.

Computers Encode Everything

- Text, pictures, sound, video — a computer stores all of it as numbers. Turning something into numbers (and back again) is encoding.
- Different jobs use different encodings: the code for a letter isn't the code for a pixel's colour, isn't the code for a musical note.
- Every app, file format, and website you use is built on encodings like this — mostly invisible, running underneath everything.

What About the Internet?

- Every website, video call, and game sends data as a stream of encoded bits over a network.
- Networks encode data into **packets** — small chunks with an address label, split up and reassembled at the other end. That's organising data for the journey, not shrinking it.
- The internet compresses a lot of what it sends, too — just with cleverer methods than RLE, built for messy data like photos and video.
- Same big idea though: find a pattern, then use it to send data more efficiently.

Why Compress Data?

- Computers store and send everything as bits — more data means more storage space and slower transfers.
- Compression re-writes data so it takes up less space. Some methods throw a little detail away to save more room (that's how a small MP3 or a compressed photo works) — others don't lose anything at all.
- **Run-length encoding (RLE)** is one of the simplest tricks, and it's one of the second kind: instead of storing every repeated value one by one, store how many times it repeats, **without losing any information**.

What Is Run-Length Encoding?

> **Run-length encoding (RLE)** replaces a repeated run of the same value with just two things: **what** is being repeated, and **how many times**.

> AAAABBBCCCCD → 4A 3B 4C 1D

A Few Things Worth Knowing

- RLE is **lossless** — decoding it gives back the exact original data, nothing is thrown away.
- It works best on **repetitive** data: long runs of the same value, like blank space or flat colour.
- It works badly on **noisy** data with few repeats — the "compressed" version can end up bigger than the original.
- It's still used today: fax machines, BMP/TIFF/PCX images, and as a step inside many other compression formats.

Compressing a Sentence

RLE isn't just for pictures — it works on any run of repeated symbols, including text.

```
Y  E  SSSSS  !!!!
```

```
↓ run-length encode ↓
```

```
1Y  1E  5S  4!
```

11 characters → 8. Long runs of the same character compress well.

...But Not Every Sentence Compresses

H	A	H	A	H	A

↓ run-length encode ↓

1H	1A	1H	1A	1H	1A

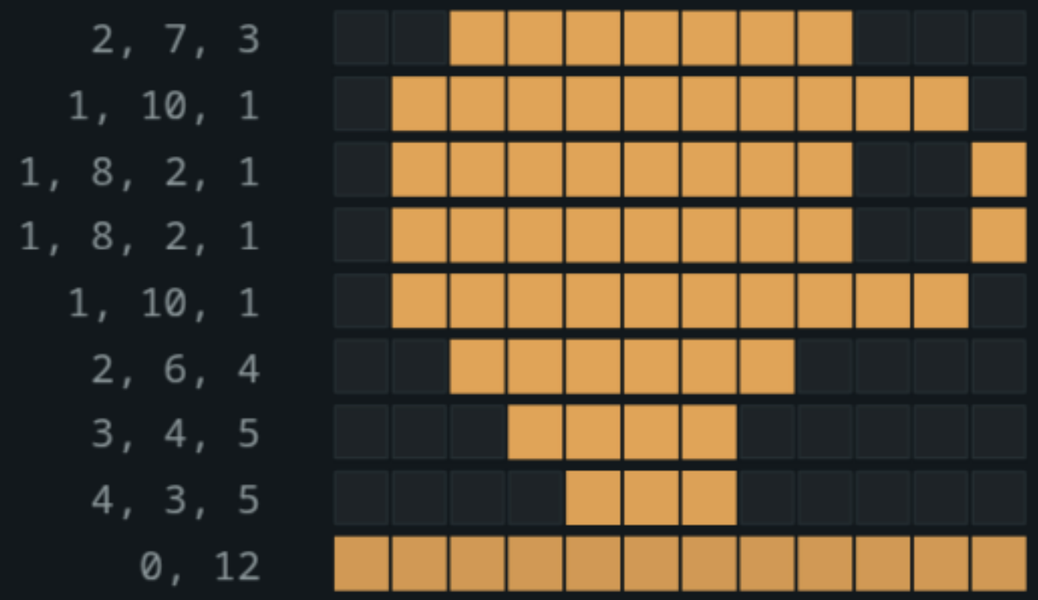
6 characters → 12. No repeats means RLE makes it **bigger**, not smaller.

Compressing a Picture

> # Instead of storing every pixel, store **how many blanks, then how many coloured pixels** — one row at a time.

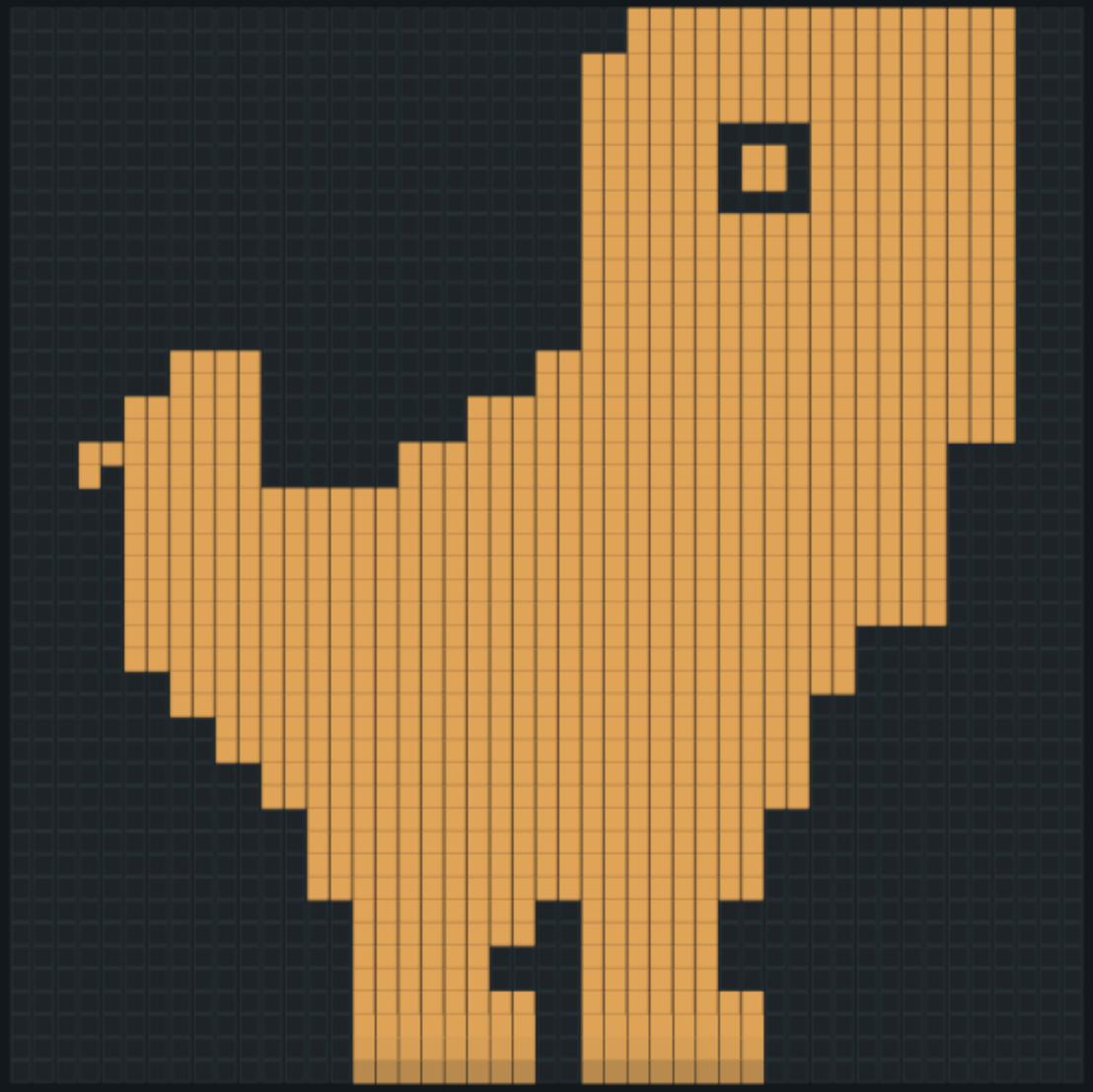
> 6, 4 = 6 blanks, then 4 coloured pixels.

Decoding messages #1

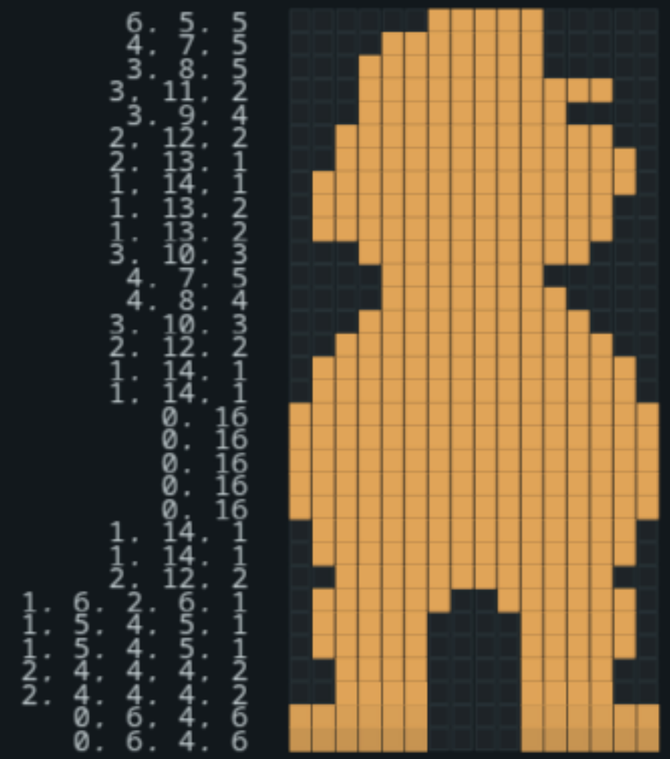


Decoding messages #2

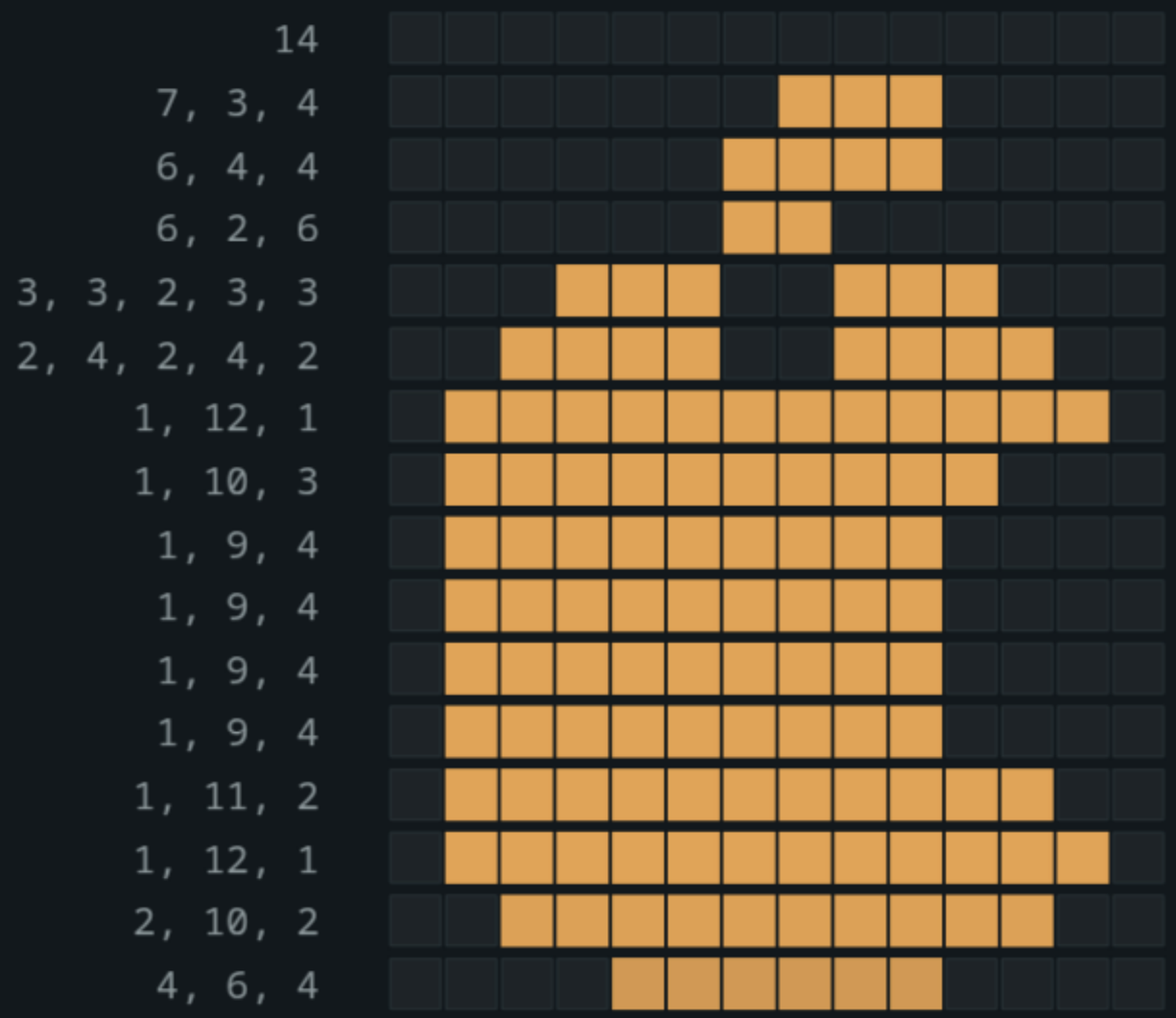
```
          27. 17. 3
          27. 17. 3
          25. 19. 3
          25. 19. 3
          25. 19. 3
     25. 6. 4. 9. 3
25. 6. 1. 2. 1. 9.
25. 6. 1. 2. 1. 9.
     25. 6. 4. 9. 3
          25. 19. 3
          25. 19. 3
          25. 19. 3
          25. 19. 3
          25. 19. 3
          25. 19. 3
          25. 19. 3
    7. 4. 12. 21. 3
    7. 4. 12. 21. 3
     5. 6. 9. 24. 3
     5. 6. 9. 24. 3
     3. 8. 6. 24. 6
3. 1. 1. 6. 6. 24.
          5. 36. 6
          5. 36. 6
          5. 36. 6
          5. 36. 6
          5. 36. 6
          5. 36. 6
        5. 32. 10
        5. 32. 10
        7. 30. 10
        7. 28. 12
        9. 26. 12
        9. 26. 12
       11. 24. 12
       11. 24. 12
       13. 20. 14
       13. 20. 14
       13. 20. 14
       13. 20. 14
  15. 8. 2. 6. 16
  15. 8. 2. 6. 16
  15. 6. 4. 6. 16
  15. 6. 4. 6. 16
  15. 8. 2. 8. 14
  15. 8. 2. 8. 14
  15. 8. 2. 8. 14
  15. 8. 2. 8. 14
```



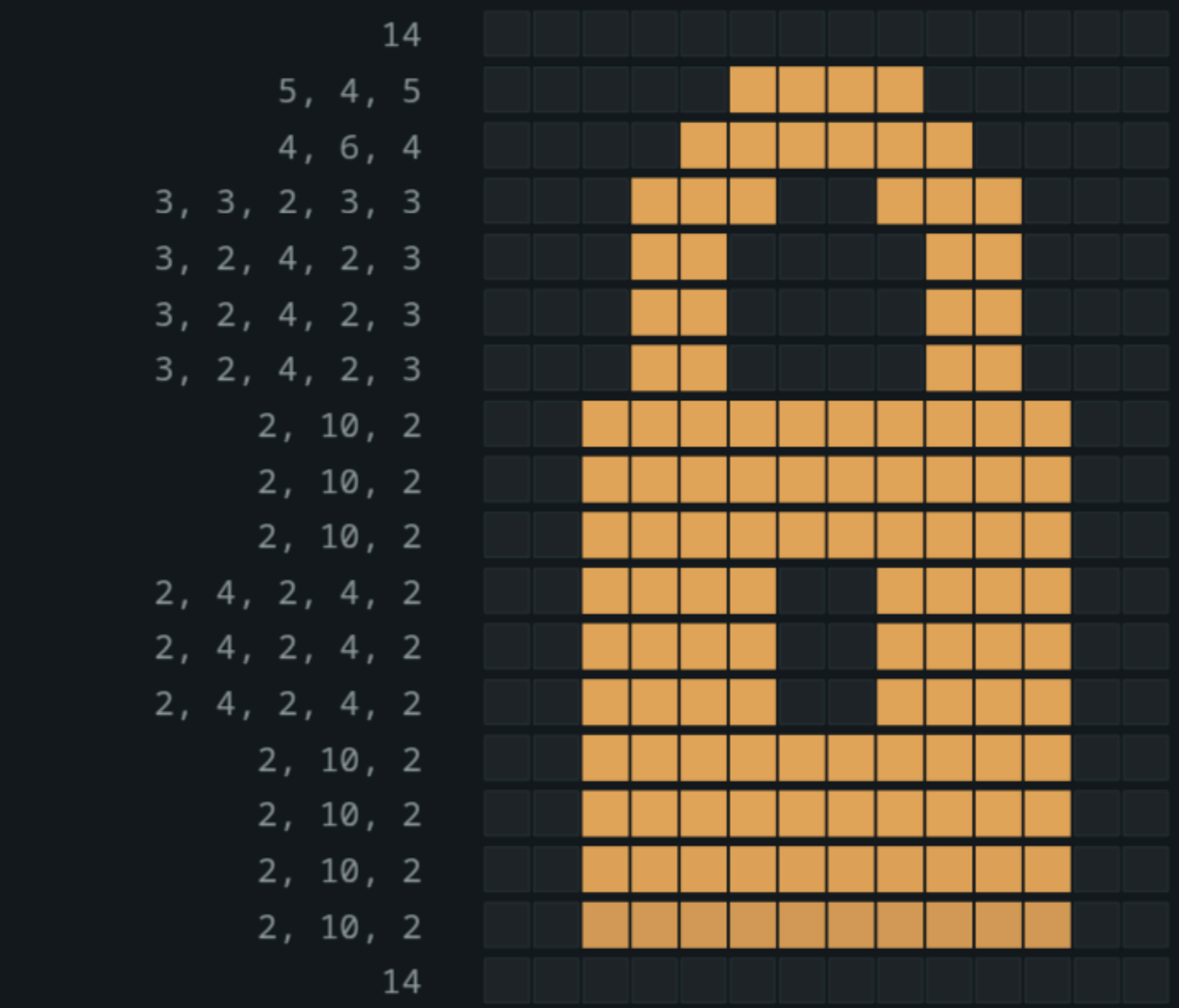
Decoding messages #3



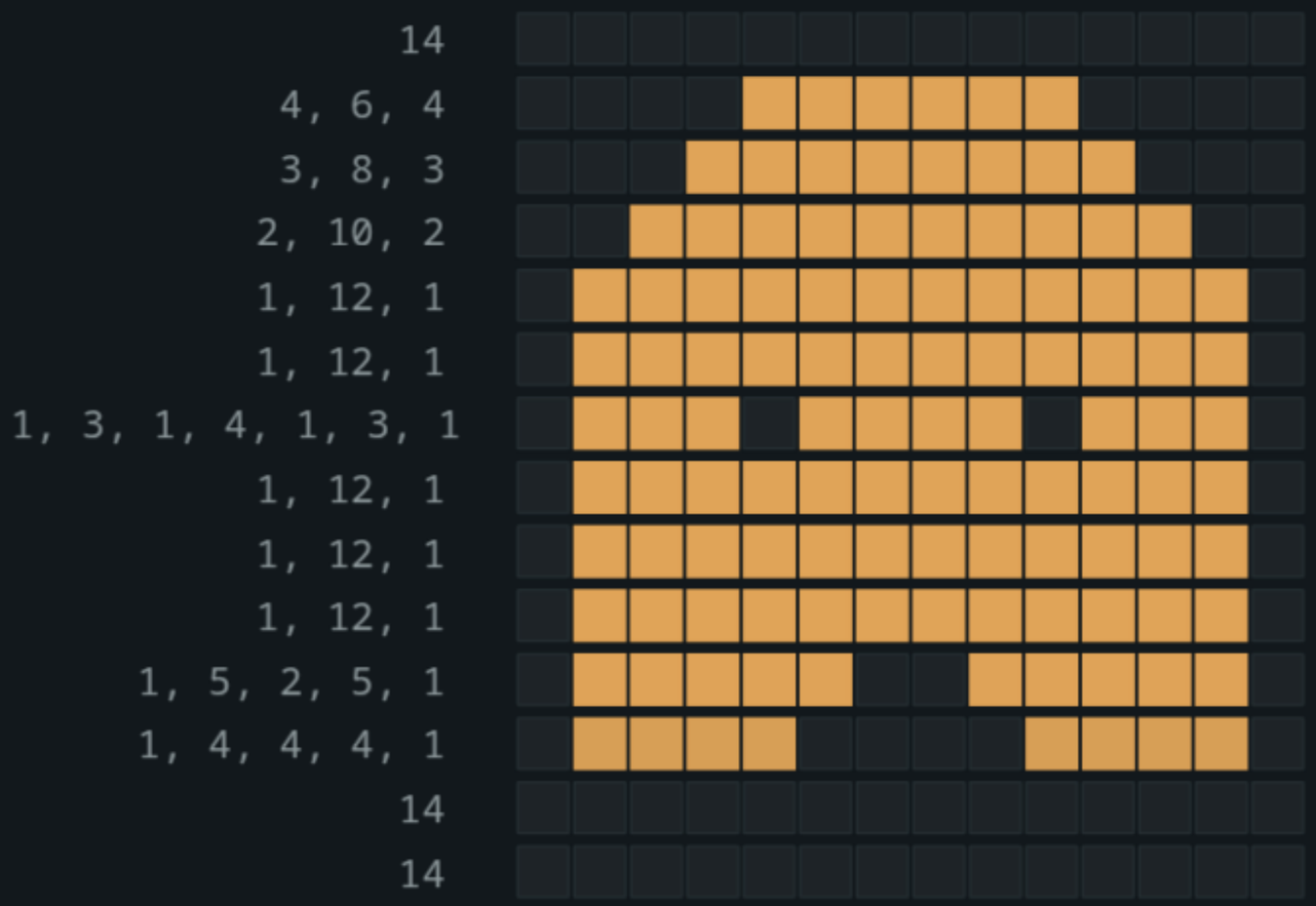
Decoding messages #4



Decoding messages #5



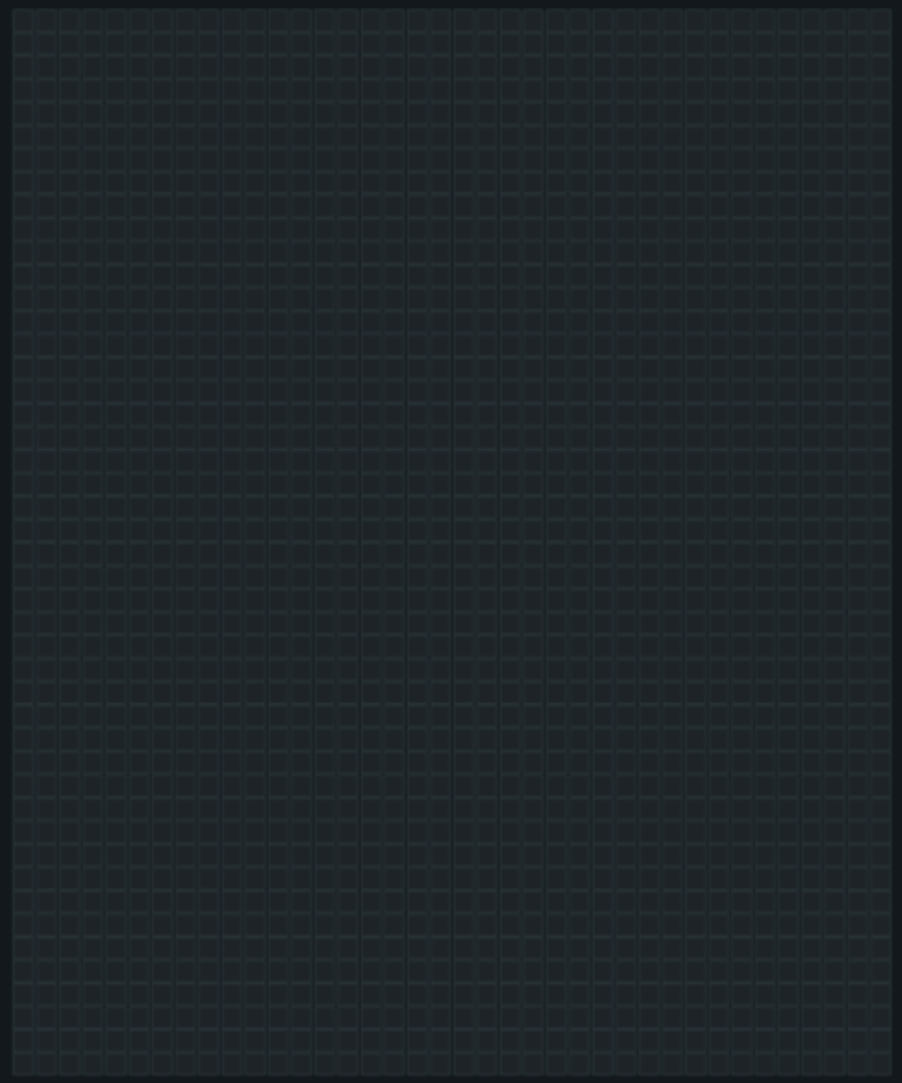
Decoding messages #6



Instructions & Decoded

Instructions & Decoded

```
                38
                38
          17. 6. 15
          16. 6. 16
          16. 5. 17
          15. 5. 18
   15. 5. 1. 5. 12
         15. 11. 12
         15. 11. 12
         15. 11. 12
         14. 11. 13
         14. 11. 13
          16. 8. 14
          15. 9. 14
          15. 9. 14
    15. 1. 1. 9. 12
   15. 1. 1. 10. 11
    15. 1. 2. 9. 11
9. 2. 1. 2. 3. 10.
   7. 7. 3. 11. 10
   7. 8. 2. 11. 10
    6. 10. 5. 8. 9
     7. 9. 5. 9. 8
    2. 13. 7. 9. 7
    3. 11. 9. 8. 7
    3. 11. 9. 9. 6
    3. 11. 9. 9. 6
   2. 12. 8. 10. 6
   4. 10. 5. 14. 5
   4. 10. 4. 14. 6
   4. 10. 5. 17. 2
   3. 11. 6. 17. 1
   3. 10. 6. 18. 1
   2. 11. 6. 17. 2
    2. 8. 7. 19. 2
    2. 6. 8. 19. 3
   3. 3. 13. 15. 4
         21. 14. 3
         19. 17. 2
         18. 17. 3
   18. 10. i. 4. 5
   17. 5. 2. 1. 13
         16. 5. 17
         15. 2. 21
                38
                38
```



What You Learned Today

- ✓ encoding = writing information a different way
- ✓ compression = encoding to use fewer symbols
- ✓ run-length encoding: store what repeats, and how many times
- ✓ it works great on repetitive data, badly on noisy data
- ✓ you decoded a mug, a dinosaur, Mario, and four more, by hand

You Can Read Code Now

> Every file on your phone or laptop is just numbers, encoded — you just decoded some by hand.

» Challenge: grab a blank worksheet and design your own picture as an RLE clue for a partner to decode.