Mathematicians like to model uncertainty about a particular circumstance by introducing the concept of a random variable.

For our application, we'll always be dealing with circumstances where there are a finite number N of distinct choices, so we'll be using a discrete random variable that can take on one of N possible values: x_1, x_2, and so on up to x_N.

The probability that X will take on the value x_1 is given by the probability p_1, the value x_2 by probability p_2, and so on.

The smaller the probability, the more uncertain it is that X will take on that particular value.

Claude Shannon, in his seminal work on the theory of information, defined the information received when learning that X had taken on the value x_i as the log-base-2 of 1/p_i.

Note that uncertainty of a choice is inversely proportional to its probability, so the term inside of the log is basically the uncertainty of that particular choice.

We use the log-base-2 to measure the magnitude of the uncertainty in bits -- where a bit is a quantity that can take on the value 0 or 1 -- think of the information content as the number of bits we would require to encode this choice.

Suppose the data we receive doesn't resolve all the uncertainty.

For example, when earlier we received the data that the card was a heart: some of uncertainty has been resolved since we know more about the card than we did before the receiving the data, but we don't yet know the exact card, so some uncertainty still remains.

We can still use the formula for information content from the previous slide, using the probability we received to compute the information content.

In our example the probability of learning that a card chosen randomly from a 52-card deck is a heart is 13/52, the number of hearts over the total number of choices.

So p_data is 13/52, or 1/4 and the information content is computed as log-base-2 of 1/(1/4), which figures out to be 2 bits.

This example is one we encounter often -- we receive partial information about N equally-probable choices (each choice has probability 1/N) that narrows the number of choices down to M.

The probability of receiving such information is M*(1/N), so information received is log-base-2 of N/M bits.

Let's look at some examples.

If we learn the result (HEADS or TAILS) of a flip of a fair coin, we go from 2 choices to a single choice.

So the information received is log-base-2 of 2/1, or a single bit.

This makes sense: it would take us one bit to encode which of the two possibilities actually happened, say, "1" for heads and "0" for tails.

Reviewing the example from the previous slide: learning that a card drawn from a fresh deck is a heart gives us log-base-2 of 52/13, or 2 bits of information.

Again this makes sense: it would take us two bits to encode which of four possible card suits had turned up.

Finally consider what information we get from rolling two dice, one red and one green.

Each die has six faces, so there are 36 possible combinations.

Once we learn the exact outcome of the roll, we've received log-base-2 of 36/1 or 5.17 bits of information.

Hmm. What do those fractional bits mean?

Our circuitry will only deal with whole bits!

So to encode a single outcome, we'd need to use six bits.

But suppose we wanted to record the outcome of 10 successive rolls.

At 6 bits per roll, we would need a total of 60 bits.

What this formula is telling us is that we would need not 60 bits, but only 52 bits to unambiguously encode the results.

Whether we can come with an encoding that achieves this lower bound is an interesting question, which we will take up later in the chapter.

To wrap up, let's return to our initial example.

Here's table showing the different choices for the data received, along with the probability of that event and the computed information content.

We've already talked about learning that the card was a heart.

The probability of this event is 13/52 with an information content of 2 bits.

Learning that a card is not the Ace of spades is quite likely, since there's only one chance in 52 that it is the Ace of spades.

So we only get a small amount of information from this event -- .028 bits.

There twelve face cards in a card deck, so the probability of this event is 12/52 and we would receive 2.115 bits.

A bit more information than learning about the card's suit since there's slightly less residual uncertainty.

Finally, we get the most information when all uncertainty is eliminated -- a bit more than 5.7 bits.

The results line up nicely with our and Mr. Blue's intuition: the more uncertainty is resolved, the more information we have received.

Now try your hand at computing the information for a few more examples in the following exercises.