# Turing’s card problem

Turing's 1950 article on Computing Machinery and Intelligence. In it, Turing proposes what he calls the 'Imitation Game' (now called the Turing Test). A human 'interrogator' is talking by teletype to two 'witnesses', one human and one a computer. The interrogator's job is to figure out which is which. The human witness's job is to help the interrogator, the computer's job is to confuse the interrogator (that is, pretend to be the human). Turing proposes the ability, if achieved, to fool most human interrogators as an operational definition of intelligence.

Much of the article consists of Turing considering and wittily rebutting nine different objections to his proposal. The last of these is that humans are allegedly capable of extra-sensory perception, but computers are not. Turing's discussion includes the following passage.

Let us play the imitation game, using as witnesses a man who is good as a telepathic receiver, and a digital computer. The interrogator can ask such questions as 'What suit does the card in my right hand belong to?' The man by telepathy or clairvoyance gives the right answer 130 times out of 400 cards. The machine can only guess at random, and perhaps get 104 right, so the interrogator makes the right identification.

Getting the card suit (clubs (♣), diamonds (♦), hearts (♥) or spades (♠)) right 130 times in 400 trials does not sound so impressive. But let's test it:

Your programs will start to get more complicated and take longer to run. We therefore strongly recommend you to save each program as a separate file (for this problem as 3\_turing.py for example).

## Step 1: the simulation game

Write a program that simulates guessing the suits (clubs (♣), diamonds (♦), hearts (♥) or spades (♠)) of 400 cards (from an unlimited pool of cards).

There are different ways to do this, the most important aspect being that a correct guess has a probability of 1/4. Thus, you can use the numpy.random.randint() function for this, but you could also use np.random.uniform(0,1), which generates a float between 0 and 1 (including 0, excluding 1) from a uniform distribution.

Now we know for how many of the 400 cards our machine is randomly guessing the right suit (likely around 100 times). As it is guessing at random, every time we run the program (or "play the game") the number of correct guesses is different.