Formal definition of computer

To prove a theorem, we'll need some kind of definition.

The first theoretical description of a ("normal") computer is a **Turing Machine**

Turing Machines have:

1. A finite control
   - Think: a DFA-like object to represent program state, what is the program I'm executing, what line number am I on, etc.
2. As much memory as we need
   - Stored on an infinite "tape"
3. A "focus-of-attention"
   - The bit of memory on the tape we're paying attention to right now. Can read and write at this spot. The finite control can move the focus.

The Halting Problem

The Halting Problem

Given: source code for a program P and x an input we could give to P
Return: True if P will halt on x, False if it runs forever (e.g. goes in an infinite loop or infinitely recurses)

This would be super useful to solve!

We **can't** solve it…let's find out why.

A Very Tricky Program

Run `Diagonal.java` given String input x (which could be a Java program)

```
class Diagonal { // taking input x, a Java program
   ...
   result = H(x, x)
   if (result == true) // H says "x halts on x"
      while (true) { // Go into an infinite loop
         int x = 2 + 2
      }
   else // H says "x doesn't halt on x"
      return; // Halt
   }
}
```

Fun (Scary?) Fact

Rice's Theorem

Says any "non-trivial" input-output behavior of programs cannot be computed (in finite time).