DUOLINGO OPEN TO THE PUBLIC
Luis von Ahn wants to translate the whole web into every major language. Duolingo is doing it using a GWAP: You get points for completing language lessons, and also for translating text from the web. They claim people learn as well as with Rosetta Stone, but Duolingo is free.

http://nyti.ms/Lee58T

Generalization (in CS10) REVIEW
- You are going to learn to write functions, like in math class:
  \( y = \sin(x) \)
  - \( \sin \) is the function
  - \( x \) is the input
  - It returns a single value, a number

Function basics
- Functions take in 0 or more inputs and return exactly 1 output
- The same inputs MUST yield same outputs.
  - Output function of input only
- Other rules of functions
  - No state (prior history)
  - No mutation (no variables get modified)
  - No side effects (nothing else happens)

Which is NOT a function?
- a) \( \text{pick random} \) to
- b) \( \leq \)
- c) length of
- d) sort of
- e) true

More Terminology (from Math)
- Domain
  - The "class" of input a function accepts
- Examples
  - Sort of
    - Positive numbers
    - Length of
      - Sentence, word, number
      - \( \_ \leq \_ \)
      - Both Sentence, word, number
    - Boolean
  - Letter of
    - Number from 1 to input length
    - Sentence, word, number

- Range
  - All the possible return values of a function
- Examples
  - Sort of
    - Non-negative numbers
    - Length of
      - Non-negative integer
    - Boolean (true or false)
  - Letter

## Why functions are great!

- If a function only depends on the information it gets as input, then nothing else can affect the output.
  - It can run on any computer and get the same answer.
- This makes it incredibly easy to parallelize functions.
  - Functional programming is a great model for writing software that runs on multiple systems at the same time.

---

## Scratch ➔ BYOB (Build Your Own Blocks)

- **Scratch**
  - Invented at MIT
  - Maintained by MIT
  - Huge community
  - Sharing via Website
  - No functions
  - Scratch 2.0 in Flash
    - No iOS devices
    - scratch.mit.edu

- **BYOB (to be "S I")**
  - Based on Scratch code
  - Maintained by jens & Cal
  - Growing community
  - No sharing (yet)
  - Functions! "Blocks"
  - BYOB 4.0 in HTML5
    - All devices
    - byob.berkeley.edu

---

## Why use functions? (1)

**The power of generalization!**

---

## Why use functions? (2)

They can be composed together to make even more magnificent things.

They are literally the building blocks of almost everything that we create when we program.

We call the process of breaking big problems down into smaller tasks **functional decomposition**

---

## Types of Blocks

- **Command**
  - No outputs
  - Used for side-effects

- **Reporter (Often a Function)**
  - Any type of output

- **Predicate (Function)**
  - Boolean output
    - True or false

---

## Types of input (there are more)

- **Sentences**
  - spaces, N ≥ 0
  - E.g., CS 10 is great

- **Word**
  - Length ≥ 1, no spaces
  - E.g., 42, CS/10

- **Character**
  - Length = 1
  - E.g., A, 3, #

- **Digit**
  - 0-9 only
  - E.g., 7
Quick Preview: Recursion

Recursion is a technique for defining functions that use themselves to complete their own definition.

We will spend a lot of time on this.

M. C. Escher: Drawing Hands

Functional Programming Summary

- Computation is the evaluation of functions
  - Plugging pipes together
  - Each pipe, or function, has exactly 1 output
  - Functions can be input!

- Features
  - No state
  - E.g., variable assignments
  - No mutation
  - E.g., changing variable values
  - No side effects
  - Need BYOB not Scratch

\[ f(x) = (x+3) \times \sqrt{x} \]