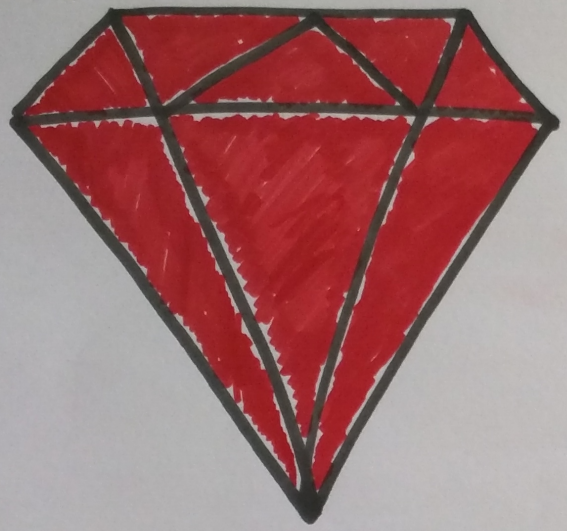


Ruby:

A (pointless) Workshop

@frmendes
@fribmendes

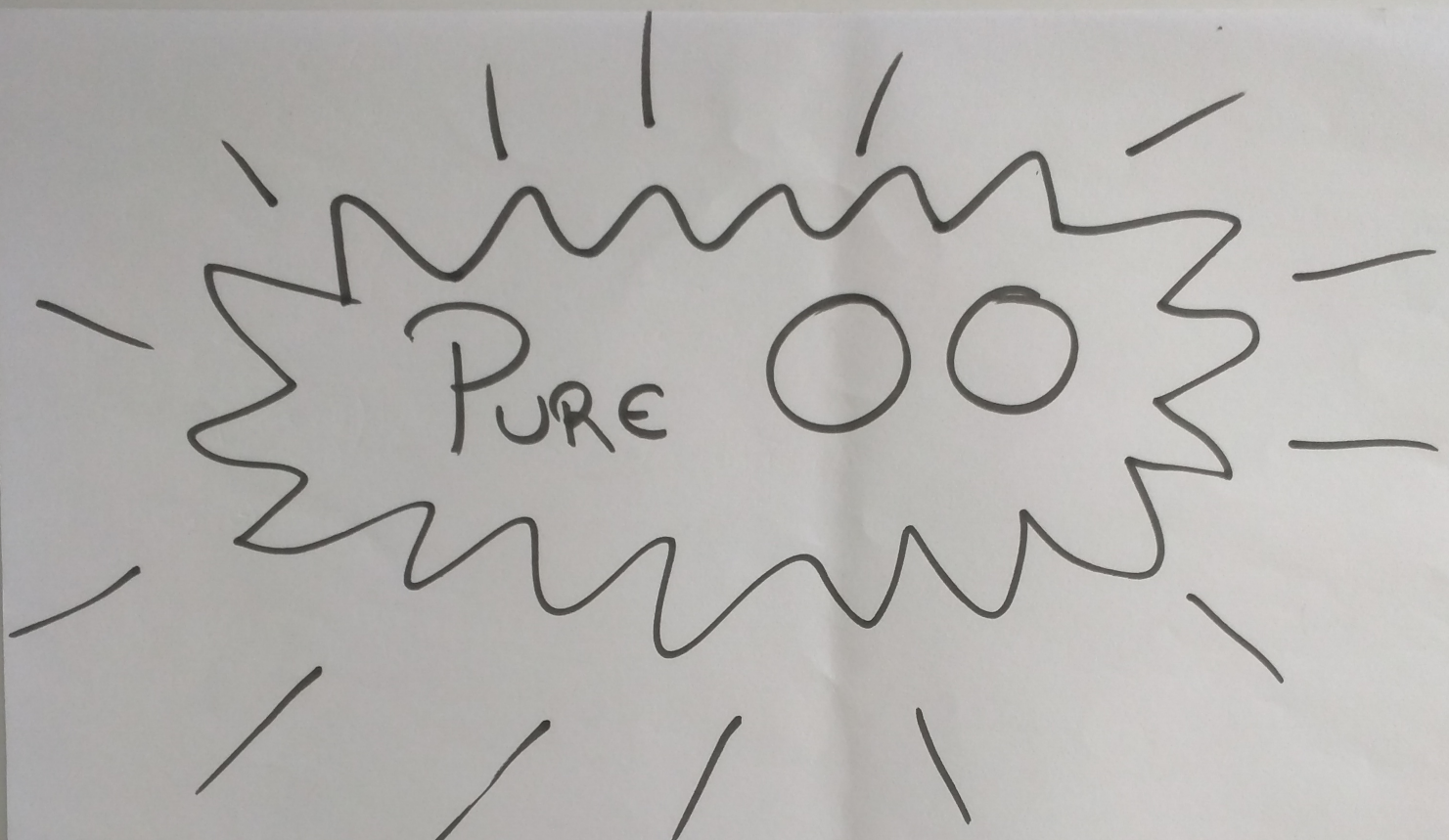
CeSIUM

Ruby: wat the wtf?

↳ Dynamic Awesome
Little Scripting Language

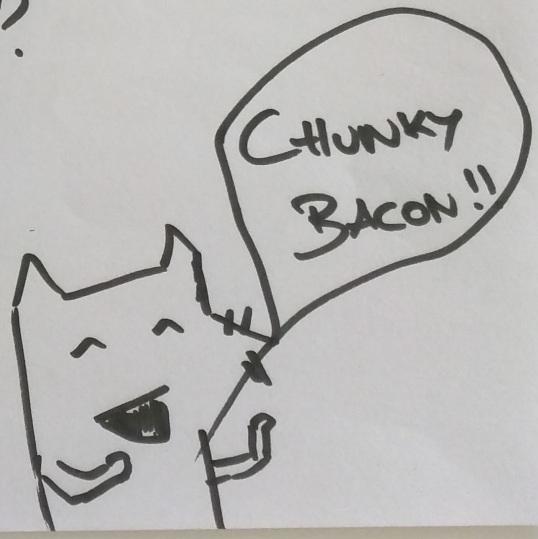
PROGRAMMER HAPPINESS!

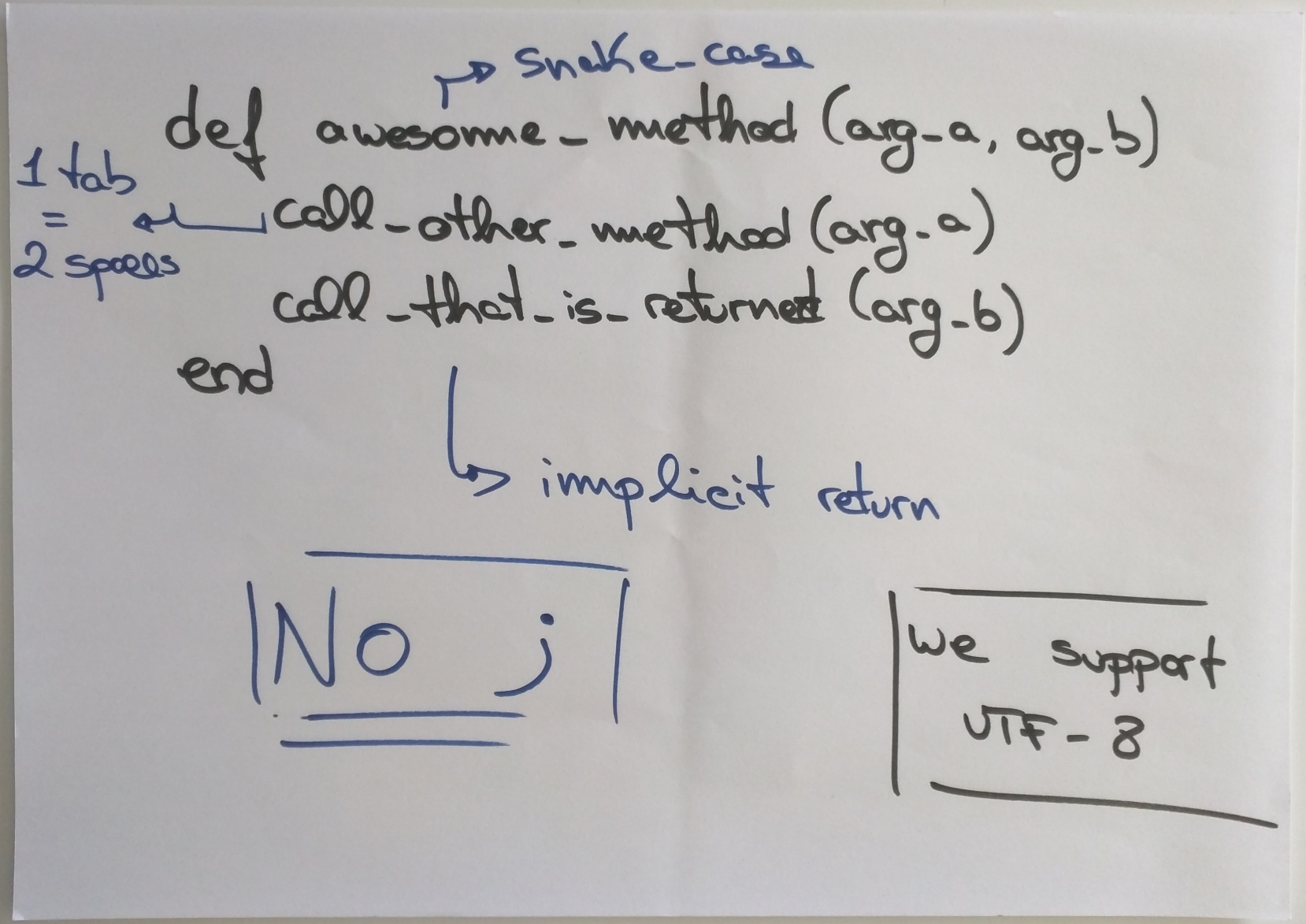
:D :D


PURE OO
5. times { print "CHUNKY BACON" }

```
restaurants.each do |r|
    Puts r.menu
end

exit unless food.free?
```

READS LIKE ENGLISH™

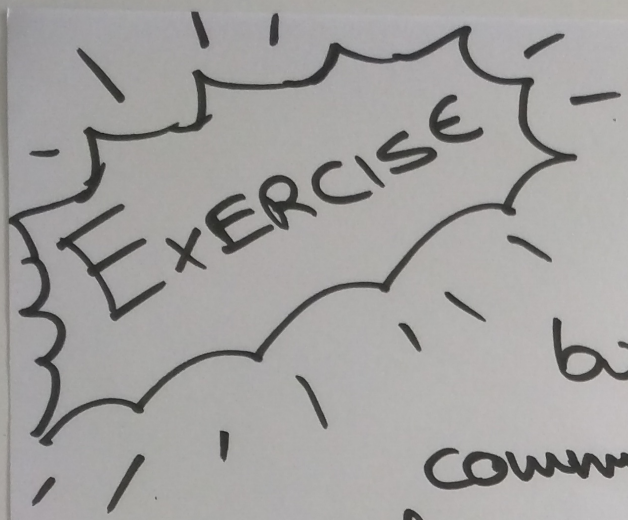



snake_case
def awesome_method(arg_a, arg_b)
1 tab = 2 spaces
call_other_method(arg_a)
call_that_is_returned(arg_b)
end
implicit return
No ;
we support UTF-8

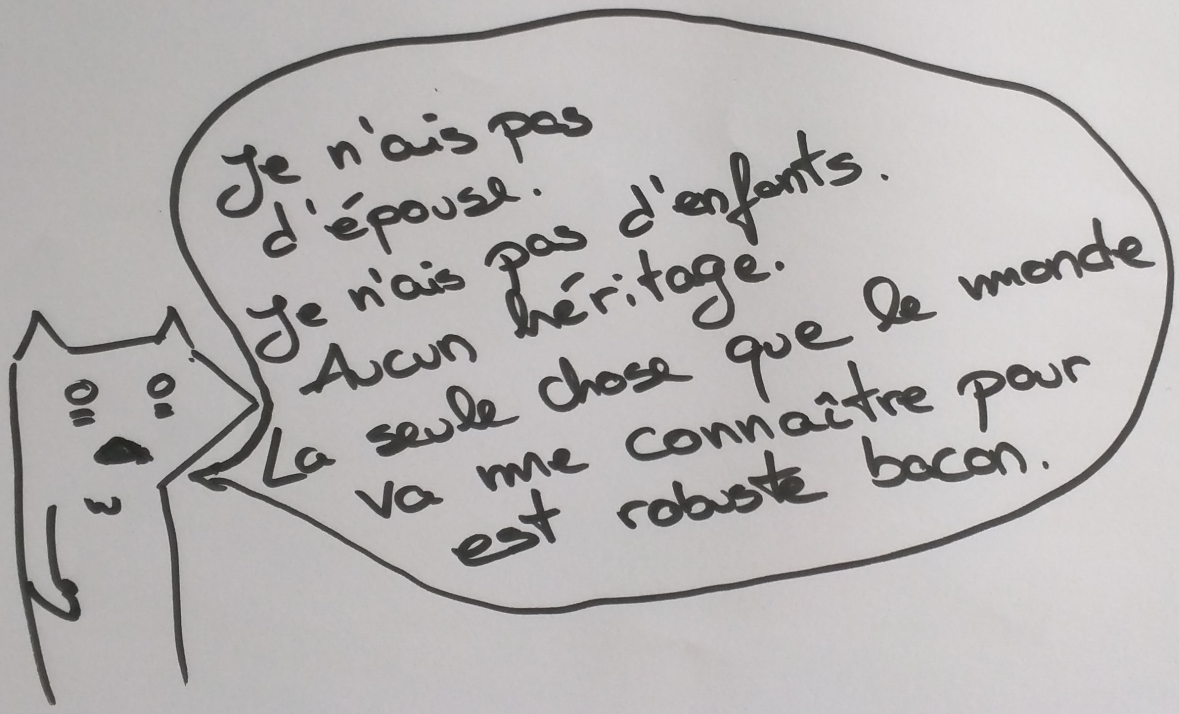
```
irb> 1.class
=> Fixnum

irb> (1.0).class
=> Float

irb> ("CHUNKY" + "BACON").class
=> String
```

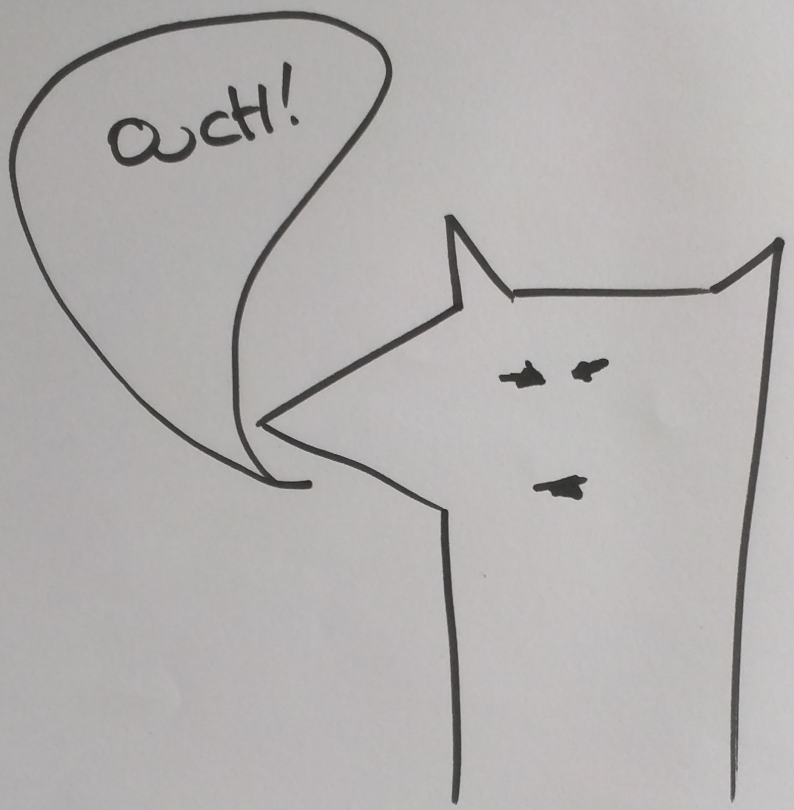
```
irb> {"a" => 1, true => 2}["a"]
=> 1
irb> h = {:a => "a", b: "b"}
irb> h[:a]
=> "a"
```

```
irb> :chunky.class
=> Symbol

irb> (1..3).to_a
=> [1, 2, 3]

irb> (1...3).to_a
=> [1, 2]
```

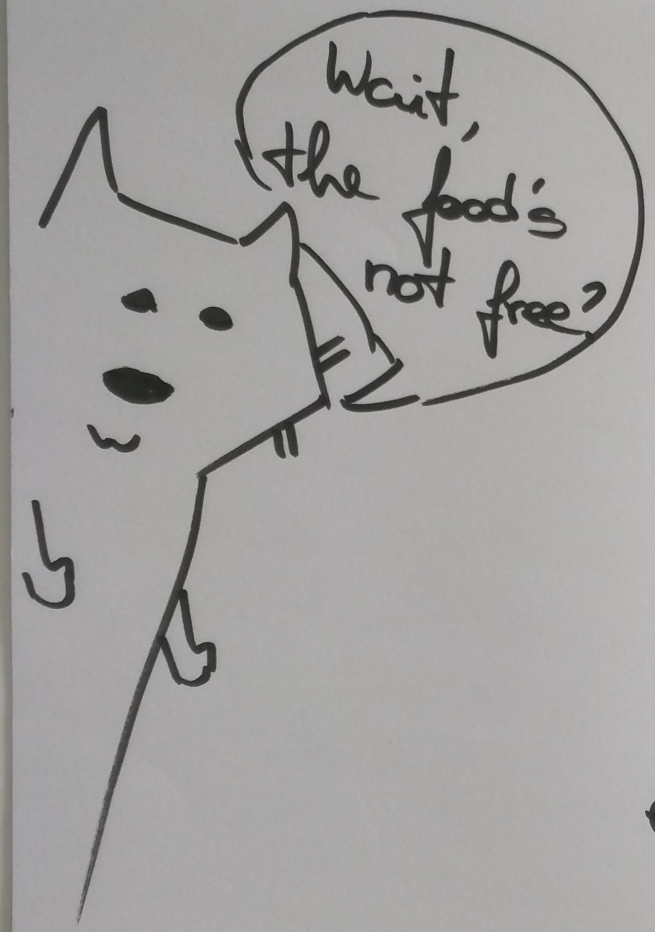


These are _why's buddies. They used to communicate with him in a foreign and misterious language. Luckily, before parting _why left us a translation, at least for some words. Use your awesome ruby skills to translate


Je n'ais pas d'épouse.
Je n'ais pas d'enfants.
Aucun héritage.
La seule chose que le monde va me connaître pour est robuste bacon.


Ouch!

```
if food.free?
   eat_everything!
elsif food.cheap?
   eat_a_lot
else
   exit
end
```




```
if !food.free?
  exit
end
```

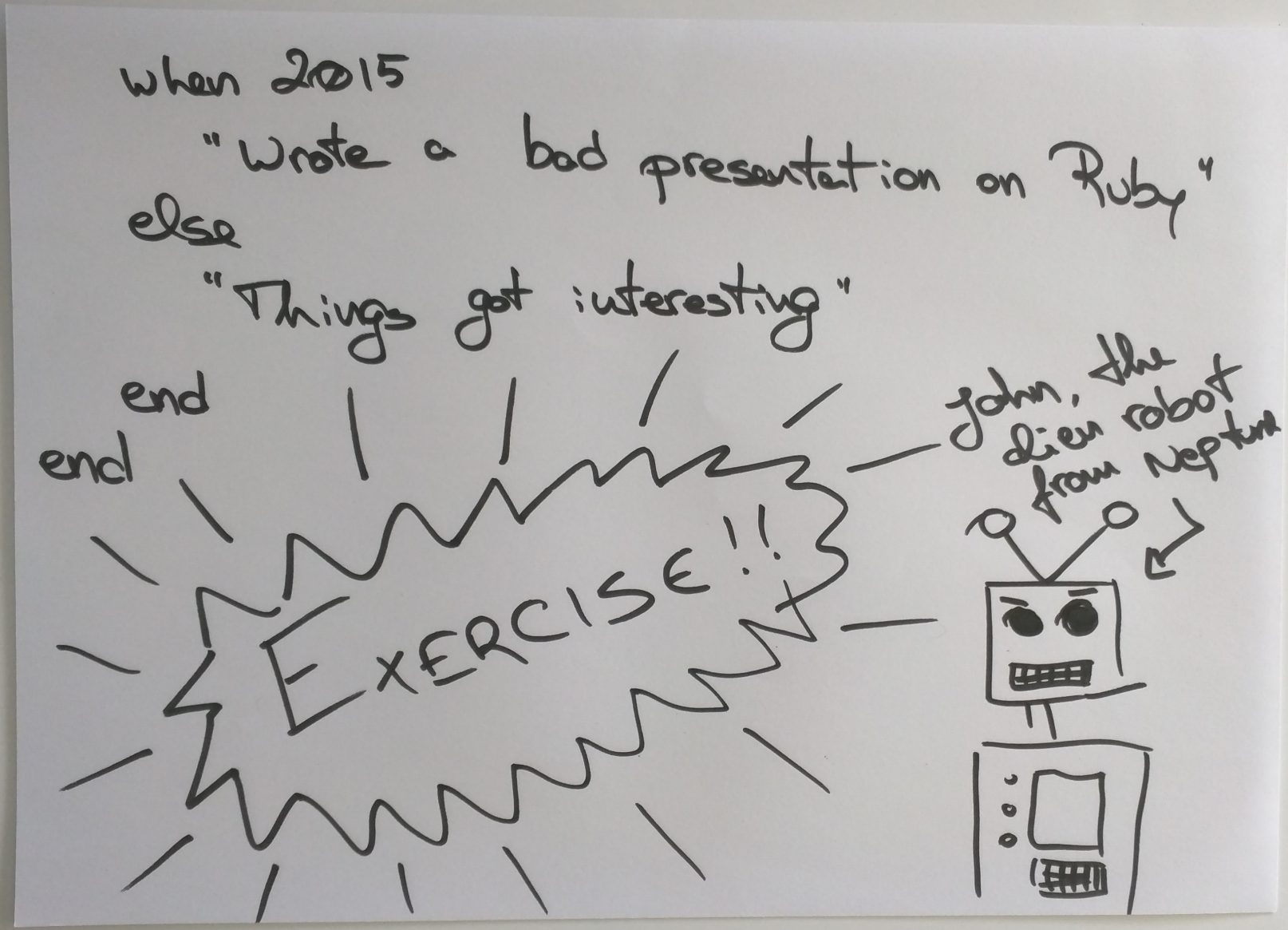
<=>

```
exit if !food.free?
```

<=>

```
exit unless food.free?
```

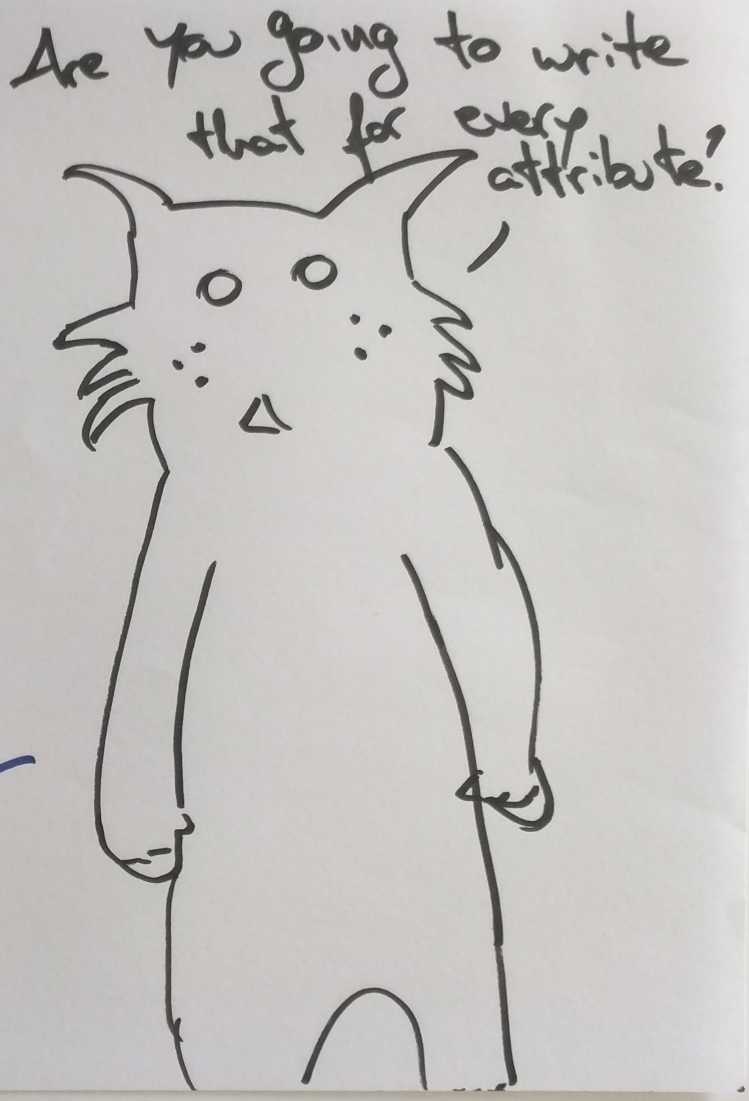
```
def biography(year)
  case year
  when 1994
    "Born"
  when 1998
    "Saw a dog for the first time"
  when 2001..2007
    "Had a birthday a couple of times"
  when 2010
    "Got into a boxing fight with a
      renegade Kangaroo
```


when 2015
"Wrote a bad presentation on Ruby"
else
"Things got interesting"
end
end
EXERCISE!!
John, the alien robot from Neptun

```
class Neptunian
    def initialize(name, personality)
        @name = name
        @personality = personality
    end
end

irb> john = Neptunian.new("John", "angry")
irb> john.name ??
```

```
def name
  @name
end

def name=(str)
  @name = str
end
```

John's cousin Alfred, from Pluto



```
class Neptunian
    attr_writer :name
    attr_reader :name
end
```

attr_writer + attr_reader > attr_accessor :name

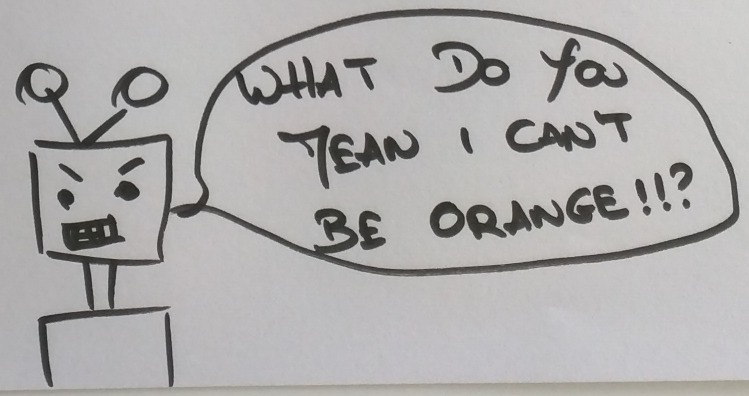
```
irb > john.name = "John, the Neptunian"
irb > john.name
 => "John, the Neptunian"
```

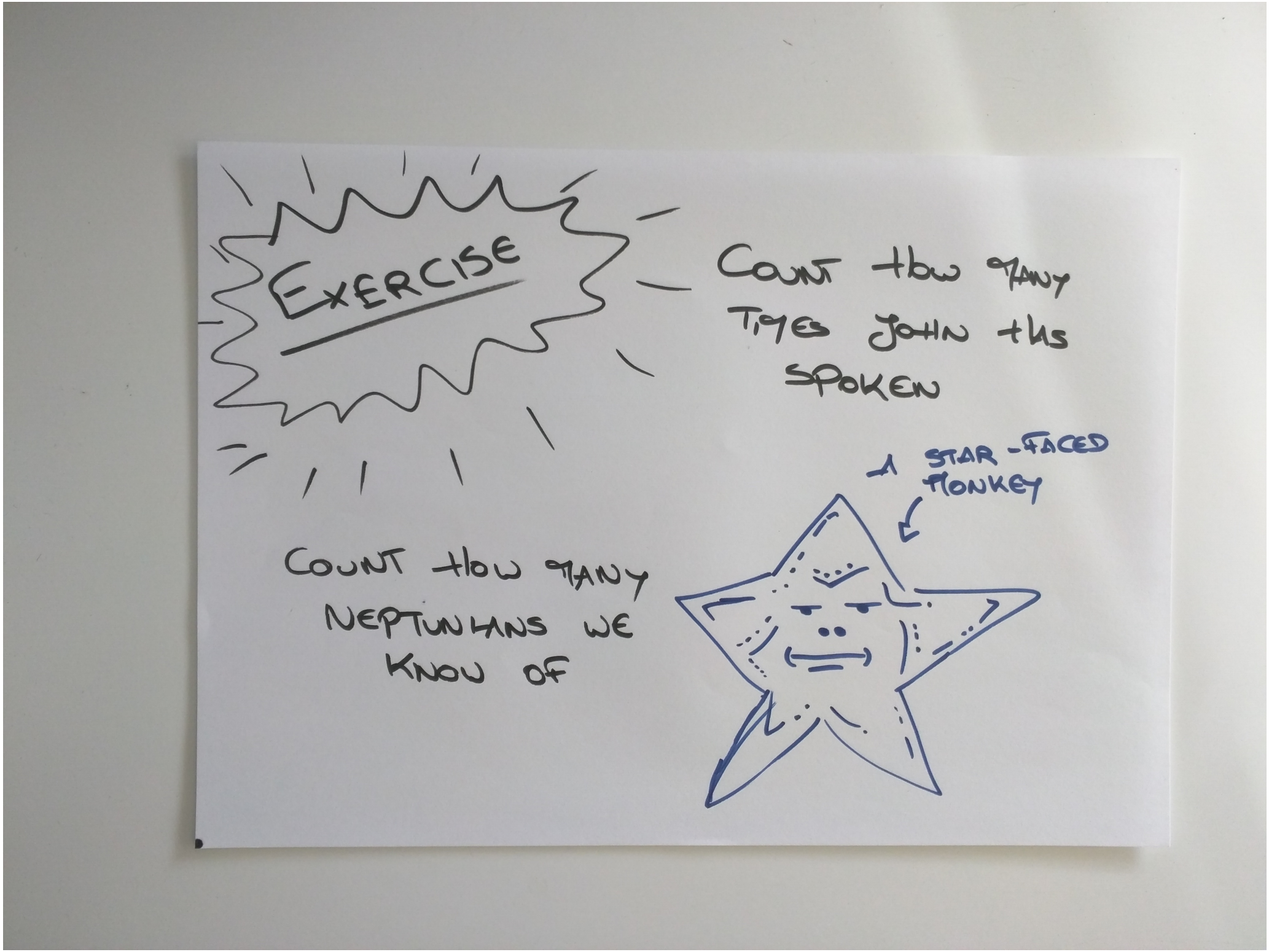
```
class Neptunian
  attr_accessor :personality
  def initialize(name, personality)
    super("Neptune", name)
    @personality = personality
  end
end
```

```
irb> john = Neptunian.new("John", "angry")
irb> john.name + " is " + john.personality
 => "John is angry"
```

```
def speak
  case @personality
    when "happy"
      "BEEP BOOP!"
    when "angry"
      # ...
    else
      "I don't know what to say!
       DON'T PUT THAT MUCH PRESSURE ON ME!"
  end
end
```

```
def speak!
  case @personality
    #...
  else
    raise ArgumentError,
      "Yeah, #{@personality} is not a thing"
  end
end
```




EXERCISE
COUNT HOW MANY TIMES JOHN HAS SPOKEN
A STAR-FACED MONKEY
COUNT HOW MANY NEPTUNIANS WE KNOW OF


```
class Alien
  attr_reader :planet
  attr_accessor :name

  def initialize(planet, name)
    @planet = planet
    @name = name
  end
end
```

```
ary = (1..100).to_a
i=0
while i< ary.length do
   puts ary[i]
   i+=1
end


loop do   # while true do
  puts "CHUNKY BACON!!"
end
```

```
ary = (1..100).to_a
ary.each {|n| puts n}
```

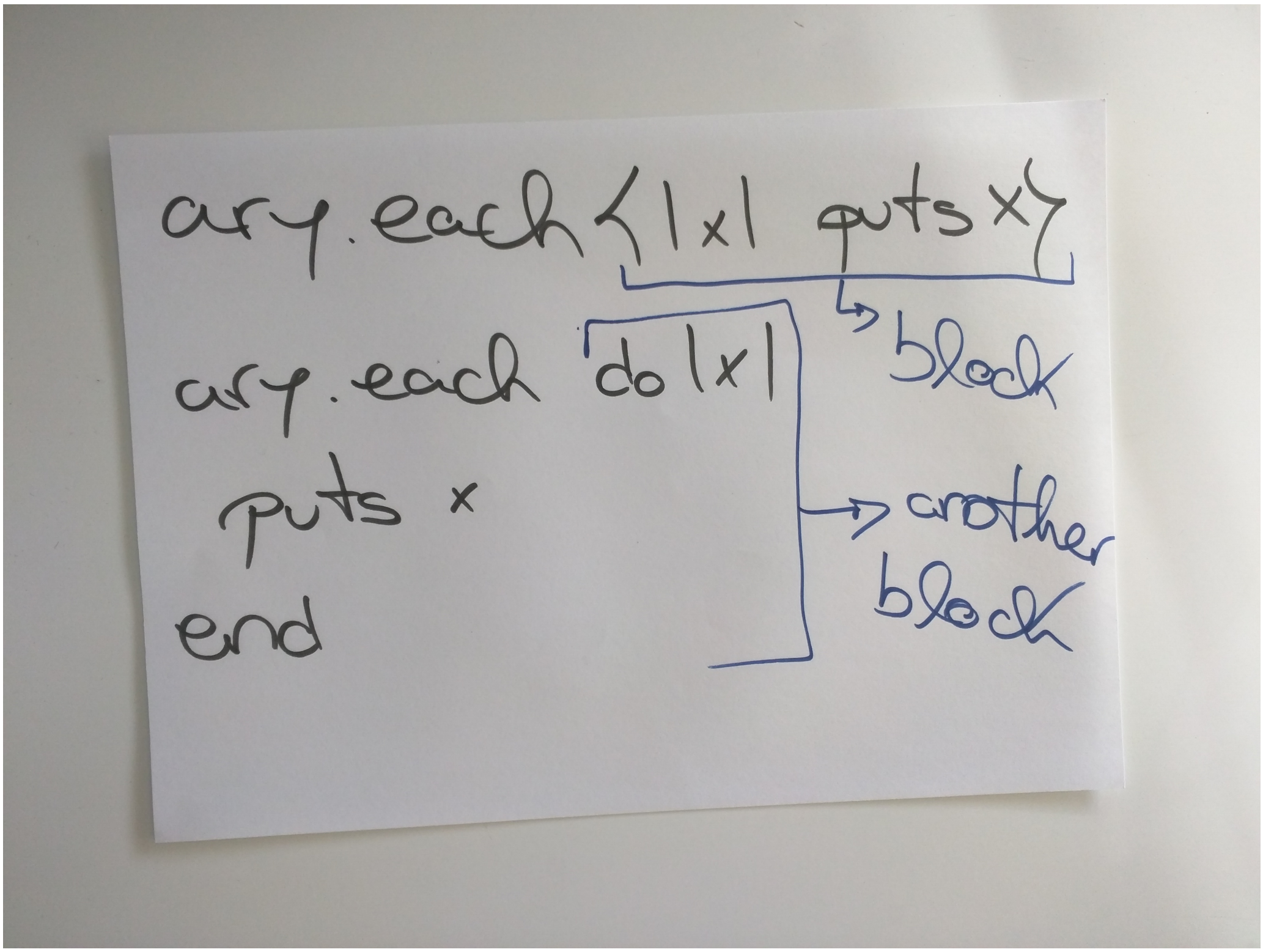
↳ See? Much better than while

PROGRAMMER HAPPINESS™

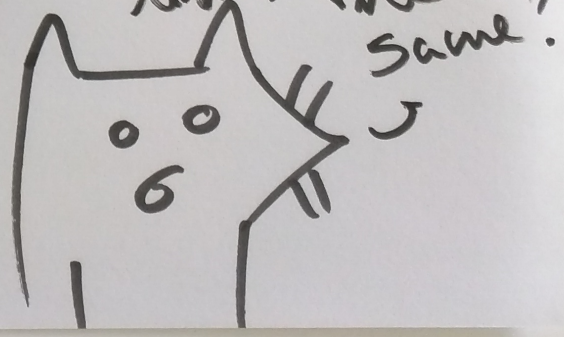
```
until self.happy? do
    eat
end

    <=>

eat until self.happy?

    <=>

eat while !self.happy?
```

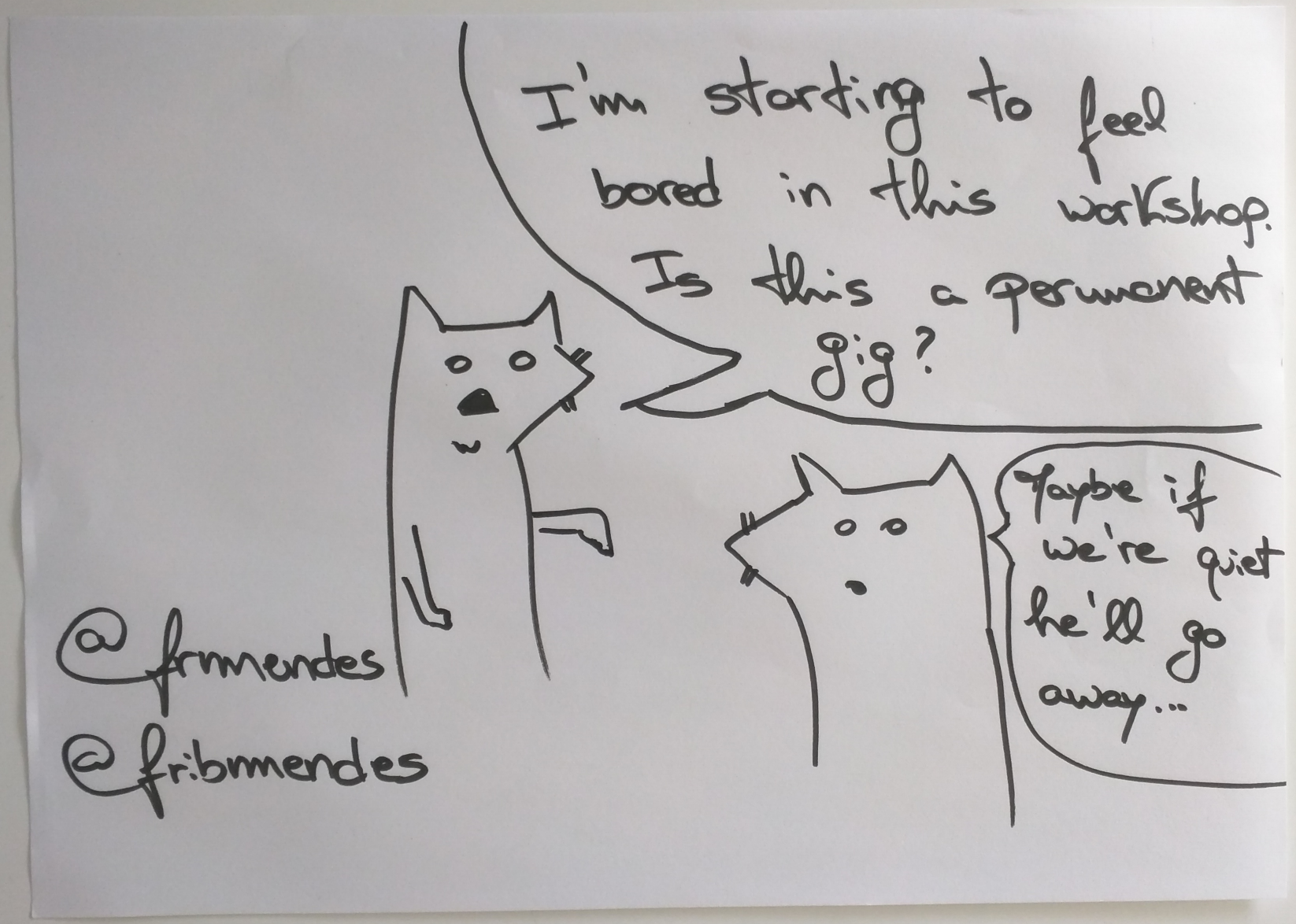
```
ary = (1..100).to_a

ary.map {|n| n+1}

ary.select {|n| n.odd?}

ary.inject(0) {|sum, n| sum + n}
```

```
ary = (1..100).to_a
ary.map(&:next)
ary.select(&:odd?)
ary.reduce(:+)
```


ary.each {|x| puts x}
block
ary.each do |x|
puts x
end
another
block

```
l = lambda {|x, y| x + y}

p = Proc.new {|x, y| x + y}

l.call(1, 2)

p.call(1, 2)
```




I'm starting to feel bored in this workshop. Is this a permanent gig?
Maybe if we're quiet he'll go away...
@frmendes
@fribrmendes