¡Bienvenidos a Ingeniería de Software!

# AYUDANTÍA RUBY

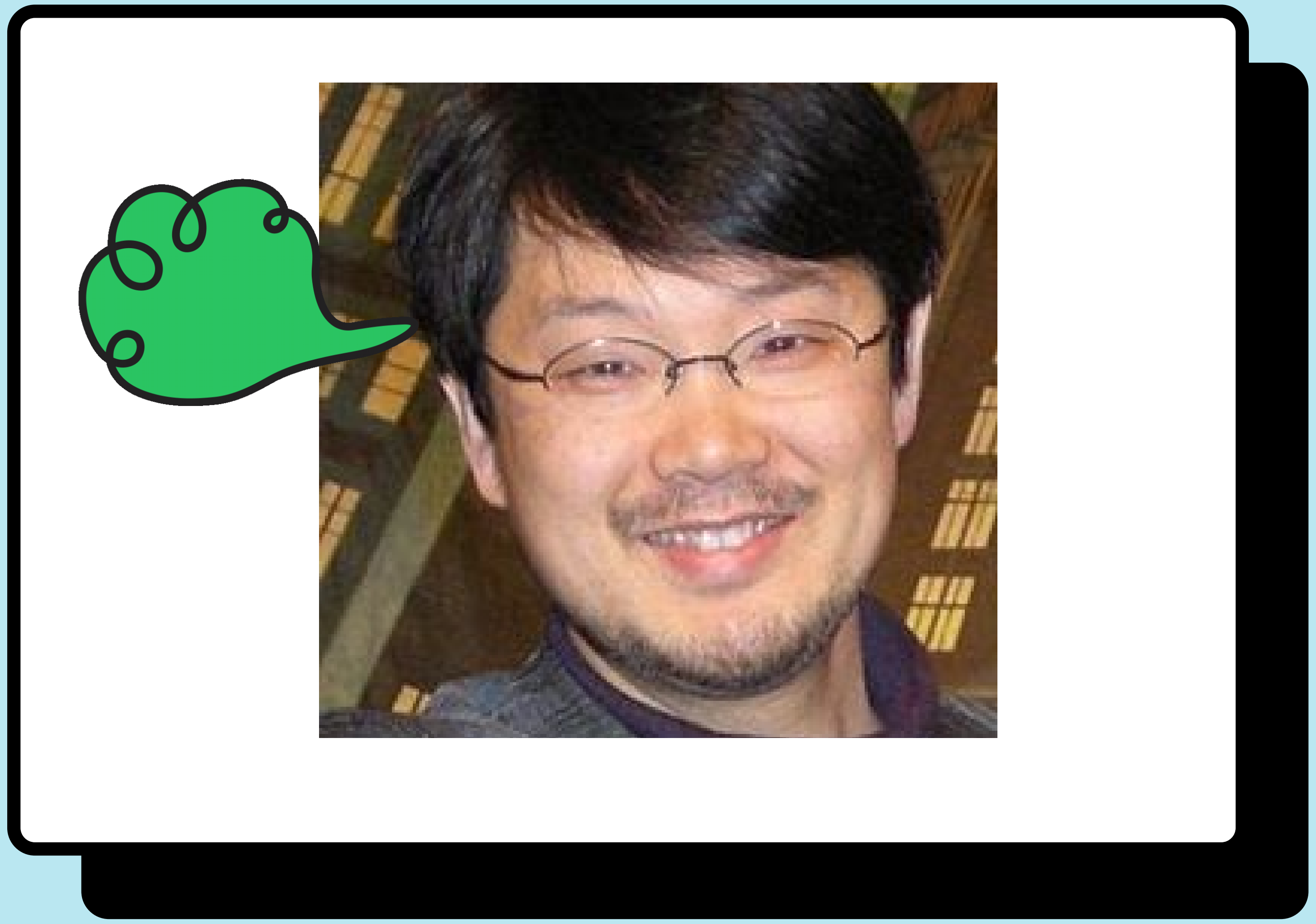
## ¿QUÉ VEREMOS?

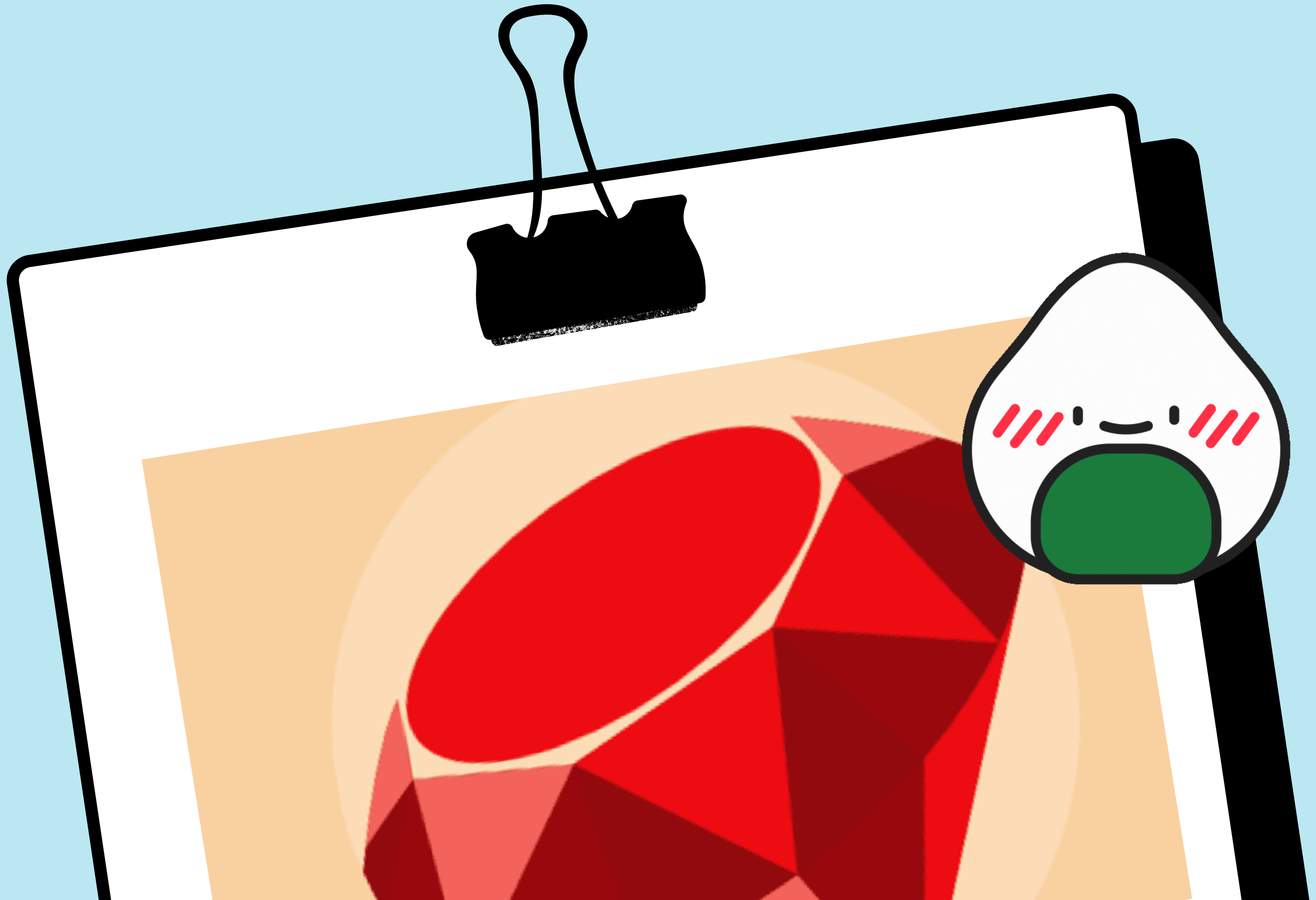
## WHAT YOU'LL LEARN

- I/O
- String Interpolation
- Console Arguments
- Type Conversions
- Require
- Implicit Returns
- Method Calls
- Hashes
- Classes
- Iterations
- Control Flow
- File Handling

## ¡UN POCO DE HISTORIA!

¿Quién fue el creador de Ruby?





- Yukihiro Matsumoto en 1995
- Sintaxis inspirada en Python y Perl

## ¡ A PRACTICAR!

- Input

gets → pide lo que ingresa el usuario.

chomp → similar a strip.

- Output

puts → imprime con "\n".

print → puts pero sin el salto de línea.

- Strings

"hola como estan"

'hola'

"ho" + "la" = "hola" → concatenación.

"estamos en el año #{año}" → string interpolation.

STRING INTERPOLATION

- ARGVS

arreglo con los argumentos dados por consola → separador: espacio

```
ruby archivo.rb argumento1 argumento2 argumento3
```

ARGV: ["argumento1", "argumento2", "argumento3"]
ARGV[1] : "argumento2"

ARGVS

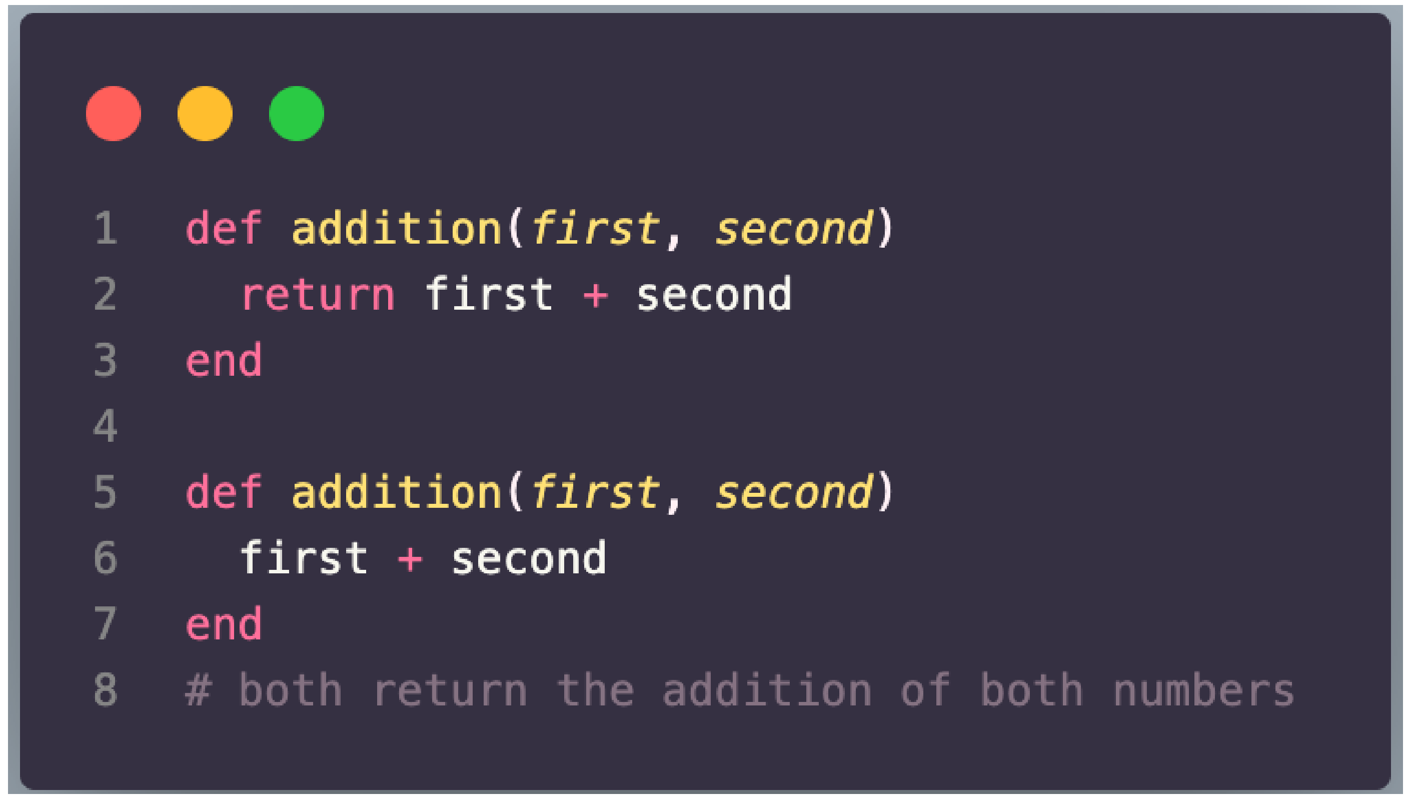
- require

require_relative → importar un archivo del directorio.
require → importar una librería externa.

REQUIRE

- **Implicit returns**

**una función siempre retorna lo último dado si no aparece un return :o**

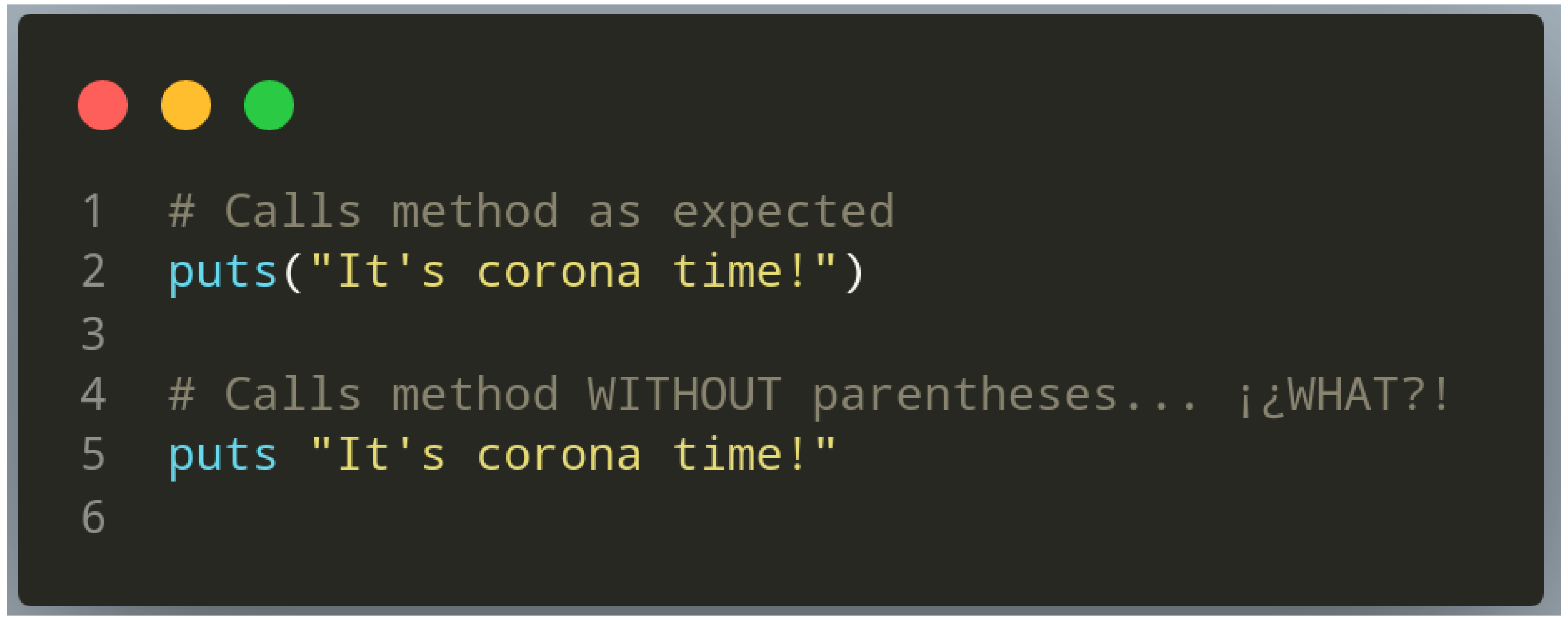


```ruby
def addition(first, second)
  return first + second
end

def addition(first, second)
  first + second
end
# both return the addition of both numbers
```

IMPLICIT RETURNS

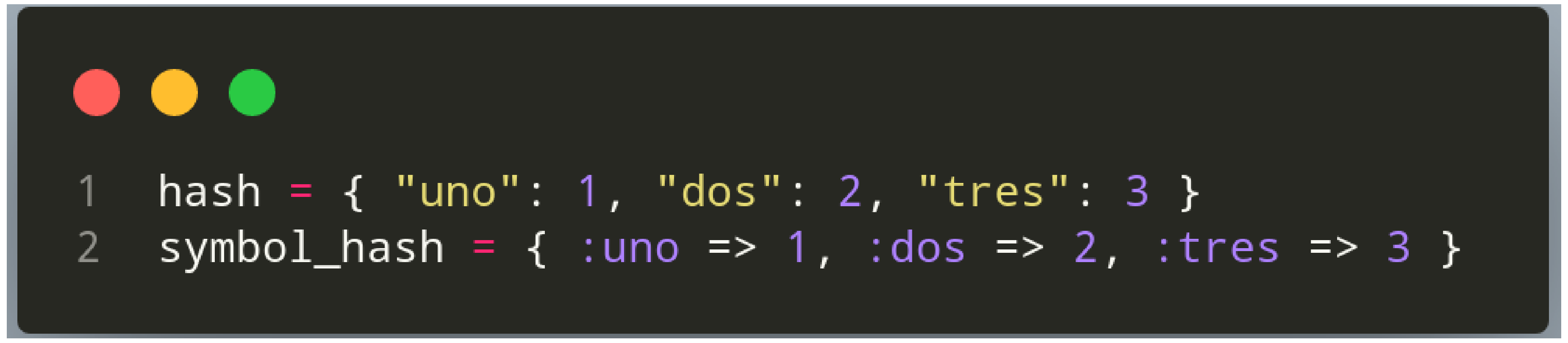
- **Method calls**

**podemos hacer llamadas a funciones / métodos sin usar paréntesis :)**



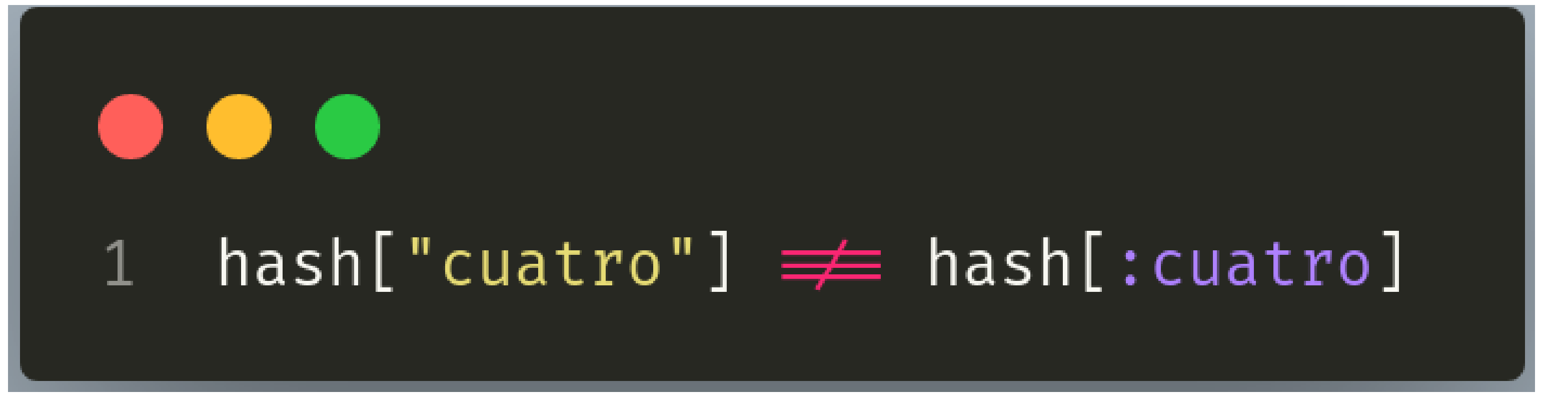
```ruby
# Calls method as expected
puts("It's corona time!")

# Calls method WITHOUT parentheses... ¡¿WHAT?!
puts "It's corona time!"

```

## • Hashes



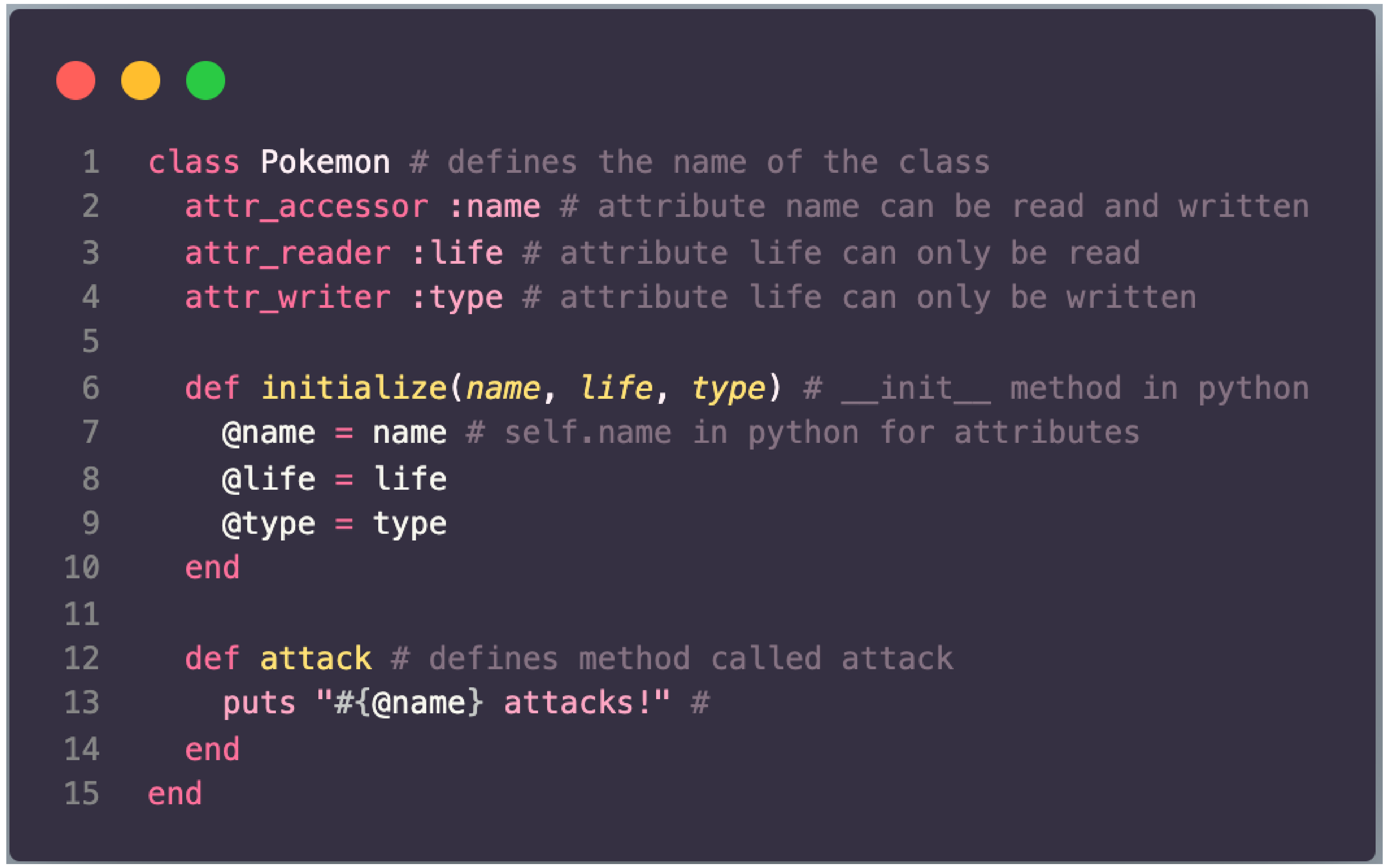
```ruby
hash = { "uno": 1, "dos": 2, "tres": 3 }
symbol_hash = { :uno => 1, :dos => 2, :tres => 3 }
```

**:uno → symbols usan menos memoria.**



```ruby
hash["cuatro"] ≢ hash[:cuatro]
```

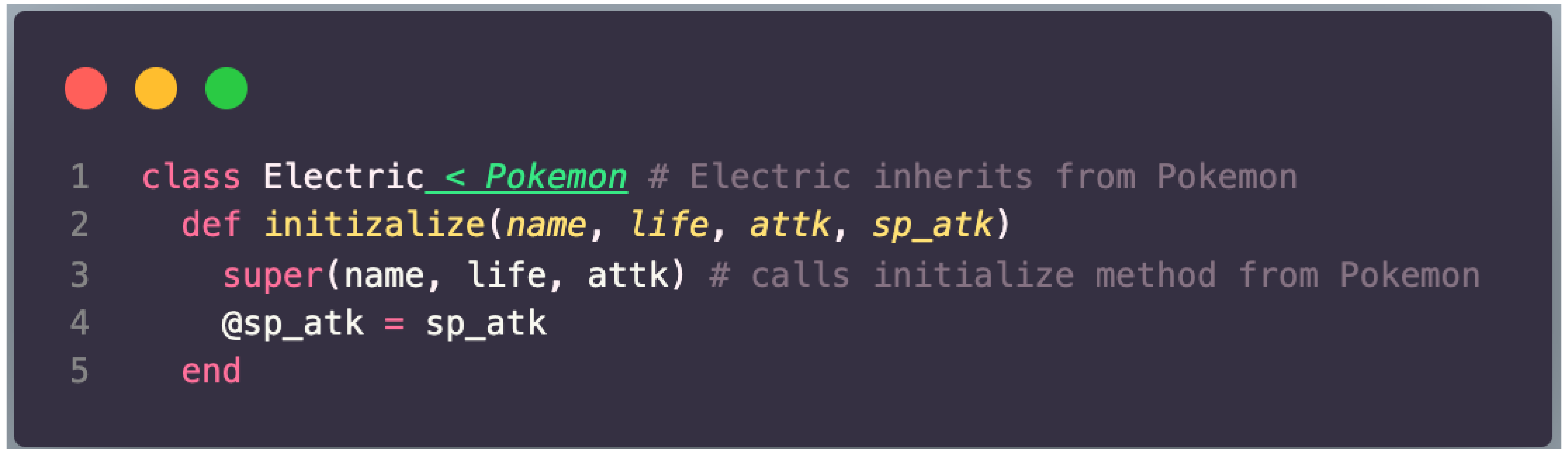
## • Clases



```ruby
class Pokemon # defines the name of the class
  attr_accessor :name # attribute name can be read and written
  attr_reader :life # attribute life can only be read
  attr_writer :type # attribute life can only be written

  def initialize(name, life, type) # __init__ method in python
    @name = name # self.name in python for attributes
    @life = life
    @type = type
  end

  def attack # defines method called attack
    puts "#{@name} attacks!" #
  end
end
```

CLASS

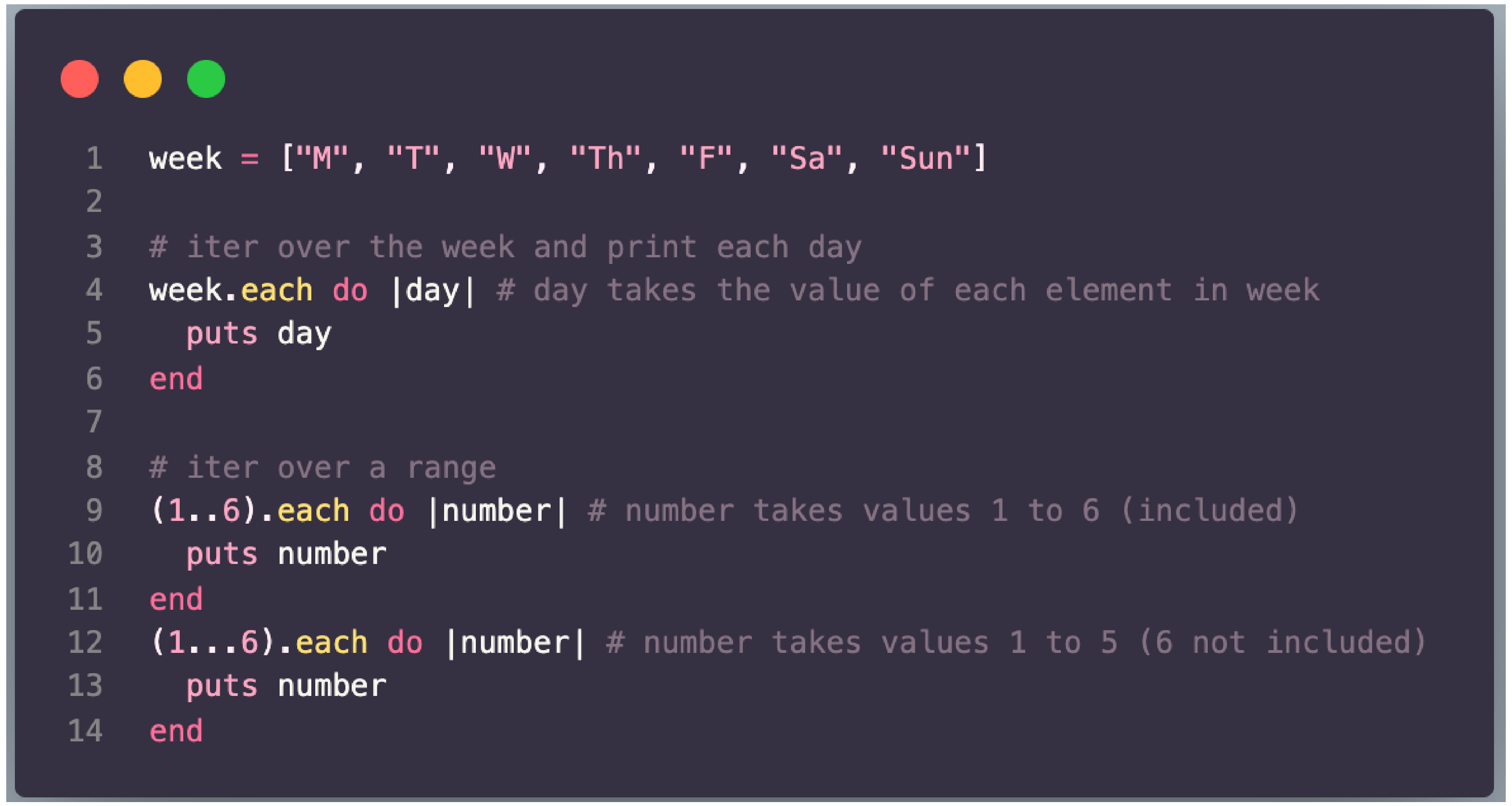
- **Herencia**



```ruby
class Electric < Pokemon # Electric inherits from Pokemon
  def initizalize(name, life, attk, sp_atk)
    super(name, life, attk) # calls initialize method from Pokemon
    @sp_atk = sp_atk
  end
```

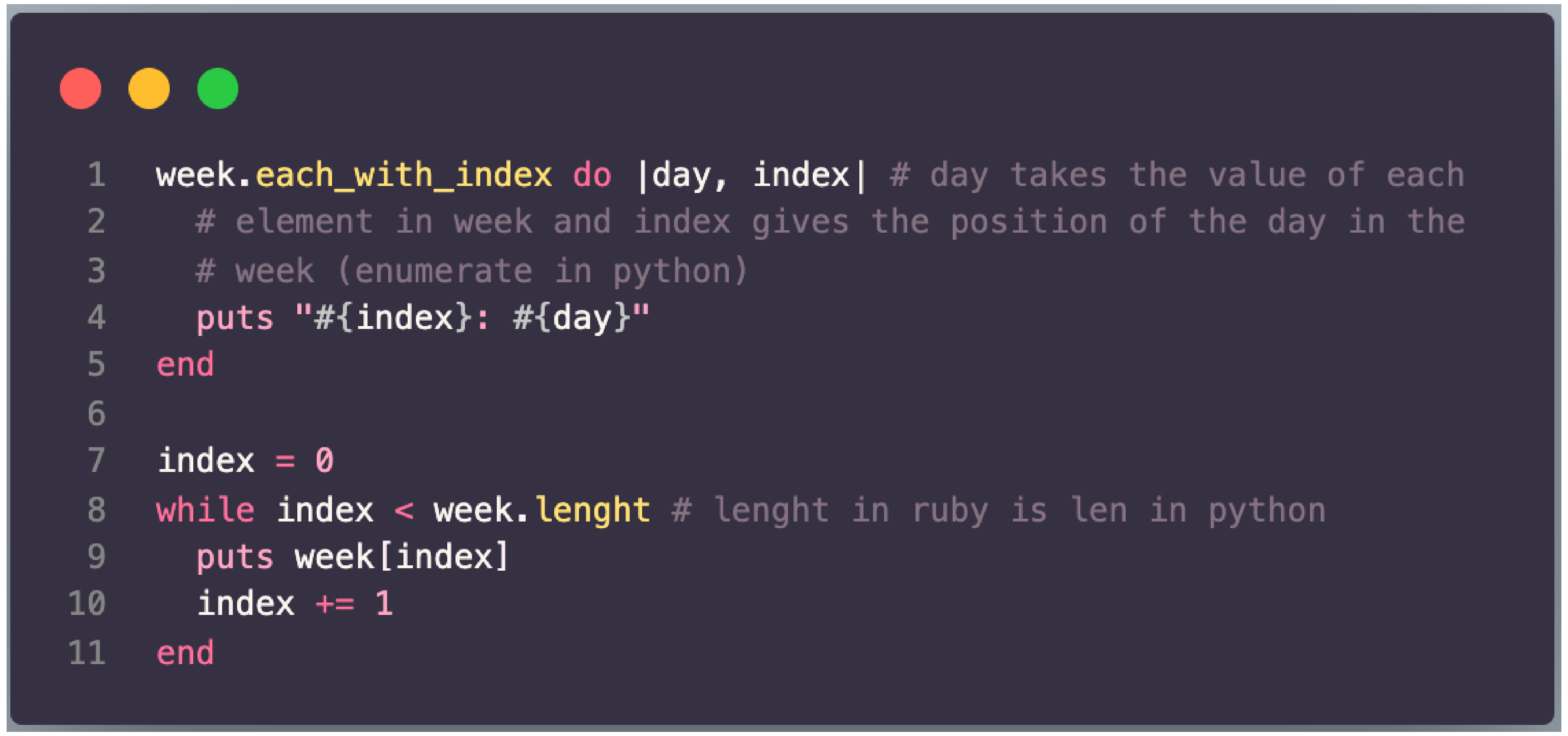
CLASS

## • Iterar



```ruby
week = ["M", "T", "W", "Th", "F", "Sa", "Sun"]

# iter over the week and print each day
week.each do |day| # day takes the value of each element in week
  puts day
end

# iter over a range
(1..6).each do |number| # number takes values 1 to 6 (included)
  puts number
end
(1...6).each do |number| # number takes values 1 to 5 (6 not included)
  puts number
end
```

ITERATE



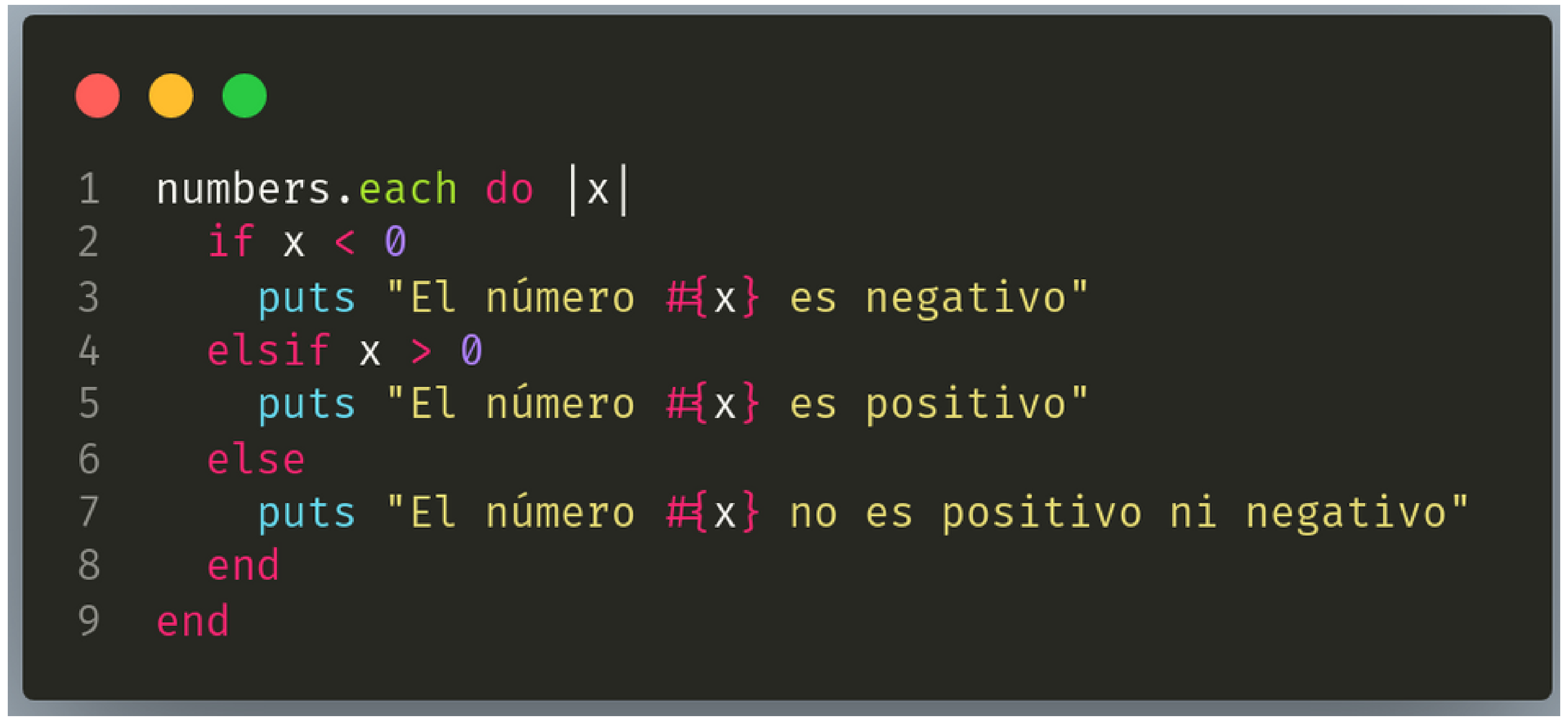
```ruby
week.each_with_index do |day, index| # day takes the value of each
  # element in week and index gives the position of the day in the
  # week (enumerate in python)
  puts "#{index}: #{day}"
end

index = 0
while index < week.lenght # lenght in ruby is len in python
  puts week[index]
  index += 1
end
```

ITERATE

- Control flow

list.each do |element|

unless → if not



```ruby
numbers.each do |x|
  if x < 0
    puts "El número #{x} es negativo"
  elsif x > 0
    puts "El número #{x} es positivo"
  else
    puts "El número #{x} no es positivo ni negativo"
  end
end
```

- File Handling

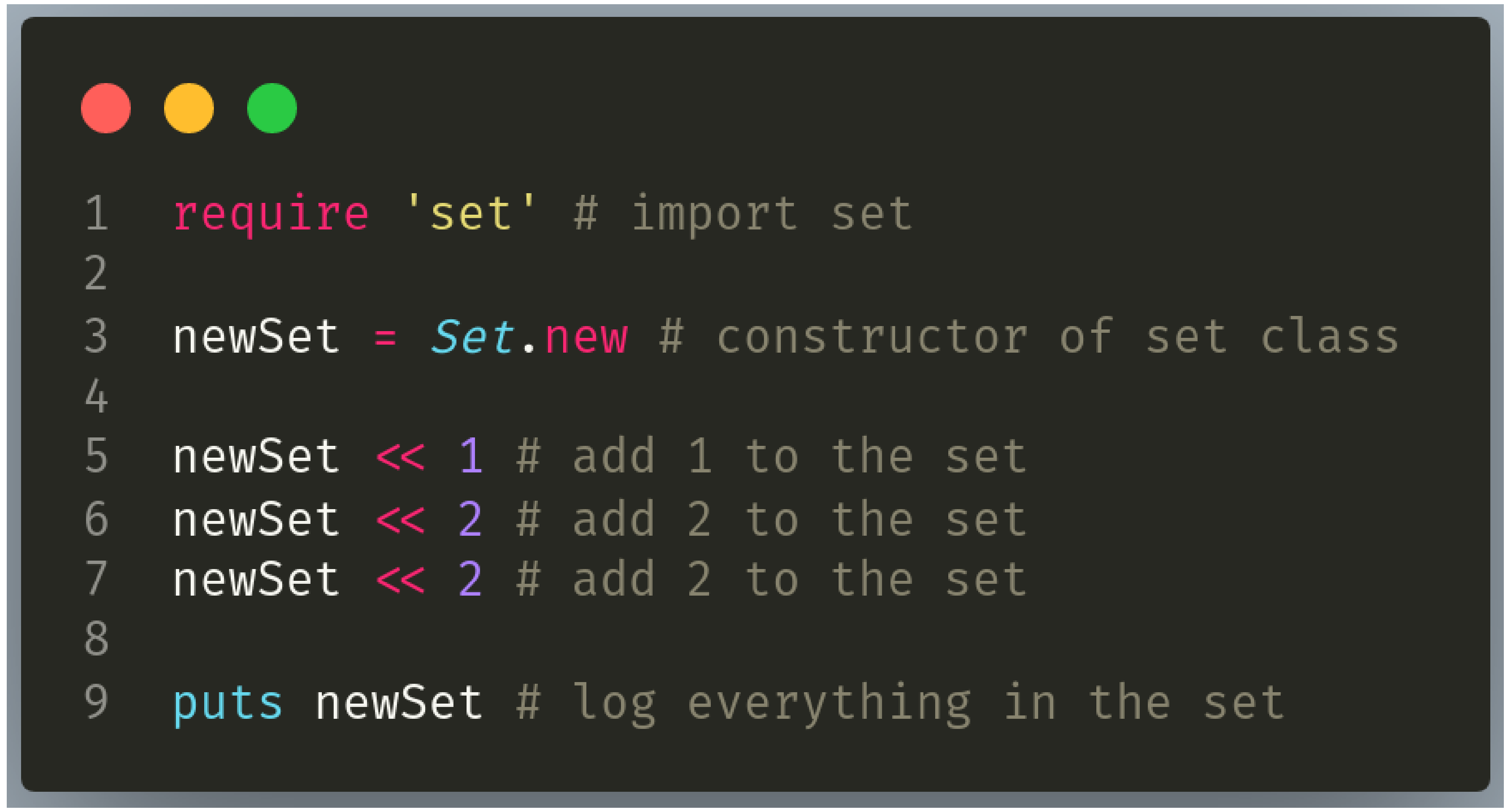
require "csv" → librería para leer csv.

File.open(name, "type") do |f|

- "a" → append
- "w" → write
- "r" → read

FILE HANDLING

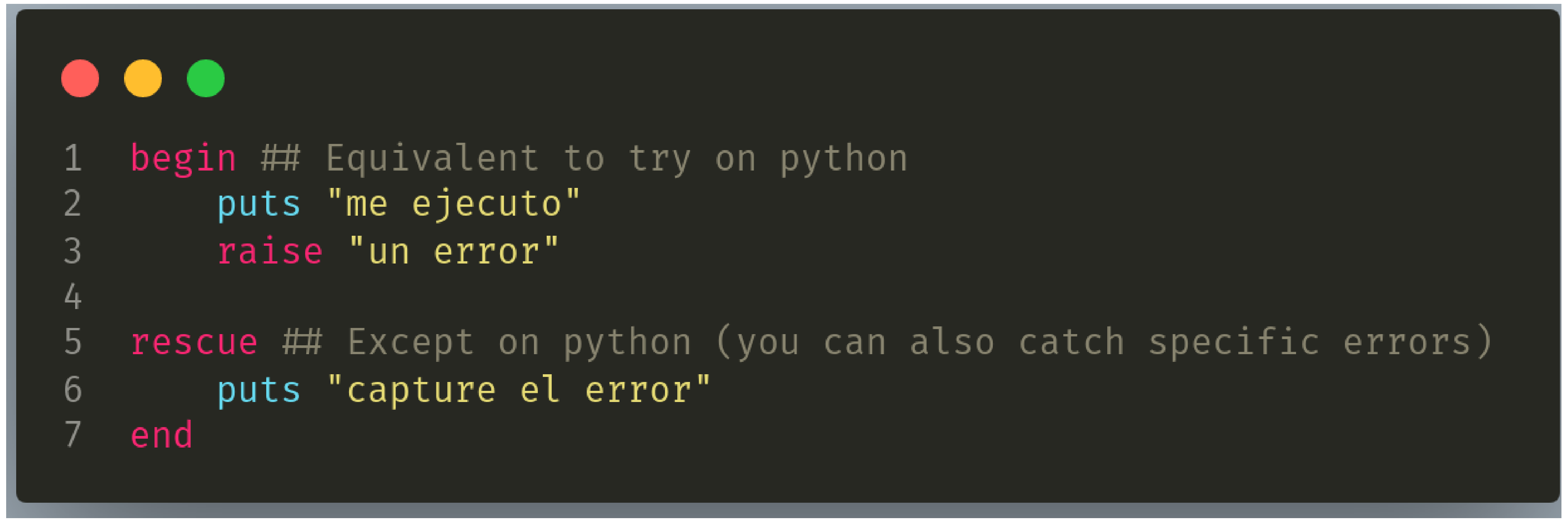
- **Sets**



```ruby
require 'set' # import set

newSet = Set.new # constructor of set class

newSet << 1 # add 1 to the set
newSet << 2 # add 2 to the set
newSet << 2 # add 2 to the set

puts newSet # log everything in the set
```

**SETS**

- Exceptions



```ruby
begin ## Equivalent to try on python
    puts "me ejecuto"
    raise "un error"

rescue ## Except on python (you can also catch specific errors)
    puts "capture el error"
end
```

EXCEPTIONS

## ¡USEN LAS ISSUES!