

# CONTRACTS.RUBY

Contracts for Ruby



@dodecaphonic

Come linguagens de café-da-manhã  
É o pai do Mathias (que hoje faz dois anos)  
Usa Emacs com toda sua ginga morena

**A INDÚSTRIA DE SOFTWARE PRODUZ  
RESULTADOS DE BAIXA QUALIDADE**

# Volkswagen delays recall of 160,000 cars due to software glitch

By Reuters | 12 Apr, 2016, 12.06AM IST

Post a Comment



READ MORE ON » Volkswagen | Skoda | Passat | German Federal Motor Transport Authority | europe

Ads by Google

## Free Woodworking Videos

[www.wwgoa.com](http://www.wwgoa.com) - Professional HD How-to Videos, tips & more. Join our newsletter Free.

HAMBURG: Volkswagen will slightly delay the recall of 160,000 cars in Europe aimed at fixing a software bug that could

# Knight Capital Reports Net Loss After Software Error

Checks  
Transpo  
consum  
engines  
update t  
for the c

by Whitney Kisling

October 17, 2012 – 12:58 PM BRT



# Heartbleed

From Wikipedia, the free encyclopedia

For other uses, see [Heartbleed \(disambiguation\)](#).

**Heartbleed** is a security bug disclosed in April 2014 in the OpenSSL cryptography library, which is a widely used implementation of the Transport Layer Security (TLS) protocol. It may be exploited regardless of whether the party using a vulnerable OpenSSL instance for TLS is a server or a client. It results from improper input validation (specifically, a length check) in the implementation of the TLS heartbeat extension,<sup>[3]</sup> thus the bug's name derives from "heartbeat".<sup>[4]</sup> The vulnerability is classified as a buffer overflow, and it allows an attacker to read data that was not intended to be read.

Heartbleed is registered in the Common Vulnerabilities and Exposures system as CVE-2014-0160.<sup>[5]</sup> The federal Canadian Cyber Incident Response Centre issued an alert to advise system administrators about the bug.<sup>[7]</sup> A fixed version of OpenSSL was released on April 7, 2014, on the same day Heartbleed was publicly disclosed.

At the time of disclosure, some 17% (around half a million) of the Internet's secure web servers certified by trusted authorities were believed to be vulnerable to the bug.<sup>[6]</sup> The servers' private keys and users' session cookies and passwords.<sup>[8][9][10][11][12]</sup> The Electronic Frontier Foundation,<sup>[13]</sup> Ars Technica,<sup>[14]</sup> and Bruce Schneier<sup>[15]</sup> described the bug as "catastrophic". Forbes cybersecurity columnist Joseph Steinberg wrote, "Some might argue that [Heartbleed] is the worst vulnerability found (at least) since commercial traffic began to flow on the Internet."<sup>[16]</sup>

A British Cabinet spokesman recommended that "People should take advice on changing passwords from the websites they use... Most websites have now placed to advise what action, if any, people need to take."<sup>[17]</sup> On the day of disclosure, the Tor Project advised anyone seeking "strong anonymity or privacy" to disconnect from the Internet entirely for the next few days while things settle.<sup>[18]</sup>

"FirstEnergy's power system's conditions," the report noted. FirstEnergy's operators "were working normally, but did not notice that the system was in jeopardy. However, they were in further jeopardy because they did not know that they were operating in a state of emergency, so that they did not realize that system conditions were changing." The blackout eventually cut off electricity to 50 million people in eight states and Canada. The cause of the outage was a time when the Blaster computer worm was wreaking havoc across the Internet. Some analysts triggered some speculation that the virus may have played a role in the outage. FirstEnergy had given credence after SecurityFocus reported that two systems at a nuclear power plant had been impacted by the Slammer worm earlier in the year.

## Medical Devices: The Therac-25\*

Nancy Leveson  
University of Washington

### 1 Introduction

Between June 1985 and January 1987, a computer-controlled radiation therapy machine, called the Therac-25, massively overdosed six people. These accidents have been described as the worst in the 35-year history of medical accelerators [6].

NÃO É ENGRAÇADO, MAS A GENTE  
NÃO SE TOCA E RI ASSIM MESMO

# Programação Orientada a Gambiarra

**Nota:** Seu navegador poderá apresentar uma certa lentidão durante o processamento e decodificação das gamas.



**Nota:** o trecho seguinte está "compactado" de modo a despoluir visualmente o contexto da página toda.

“Corrigir todos os problemas em 2 minutos!”

Você após colocar em comentário todas as linhas que estavam com problemas

“ERRO! PASSEI POR AK!!!.”

Maker sobre POG.

“Esta é a solução ótima, não temos tempo para isso. Faça a solução boa! O maior inimigo do bom é o ótimo.”

Líder de Projeto sobre POG

“É só fazer do jeito que eu to falando que entregamos na data correta.”

Diretor TI POG sobre projeto com tempo recorde!

“A culpa é do Hardware!”

desenvolvedor de software sobre POG

## Índice [ocultar]

[1 Introdução](#)

[2 A Origem do POG](#)

[3 Definição de POG](#)

[4 Exemplos de POG's softwares famosos projetados usando esta técnica](#)

[5 Conceitos de POG](#)

[5.1 Comentário é o que realmente importa](#)

[5.2 Enjambração](#)

[5.3 Reflexão](#)

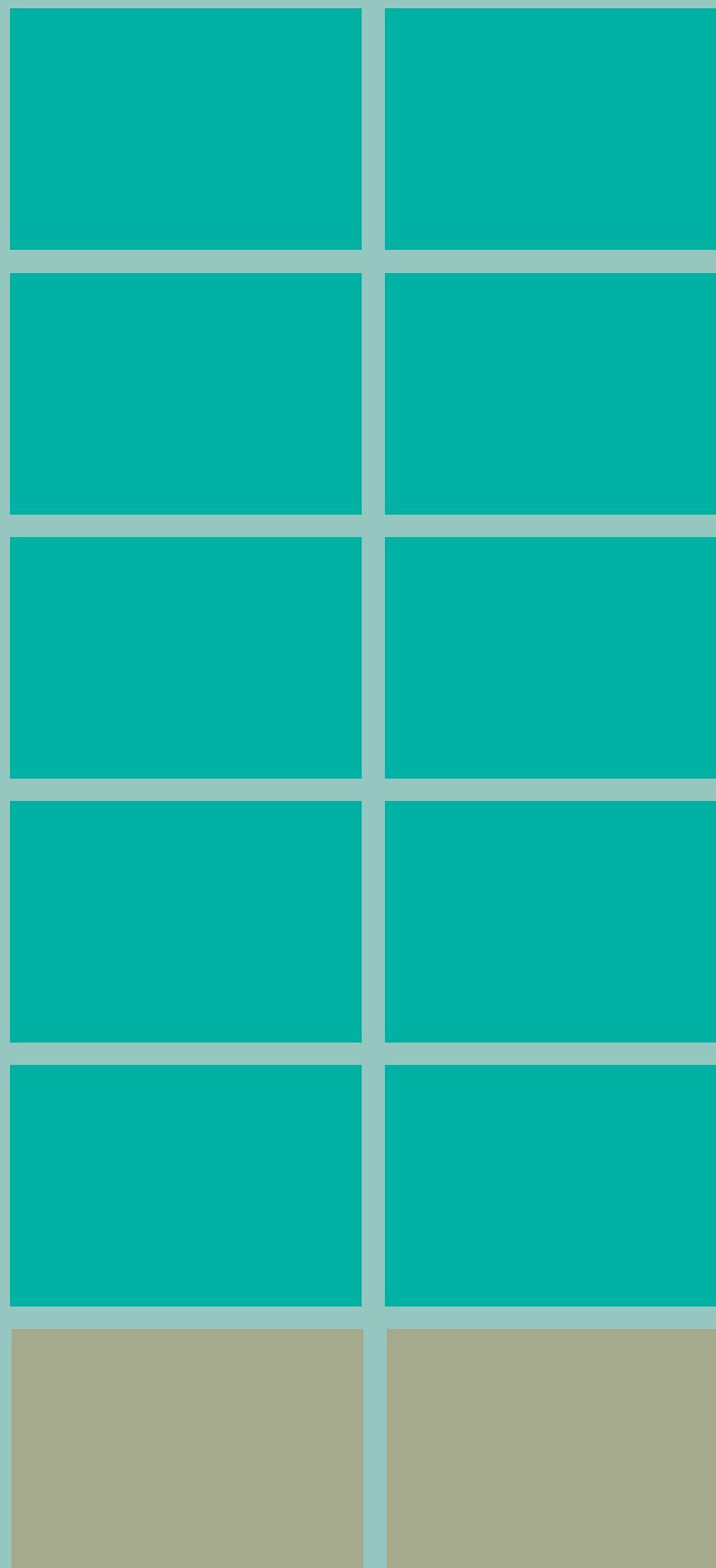
[5.4 Redireção](#)

[5.5 Insistimento](#)

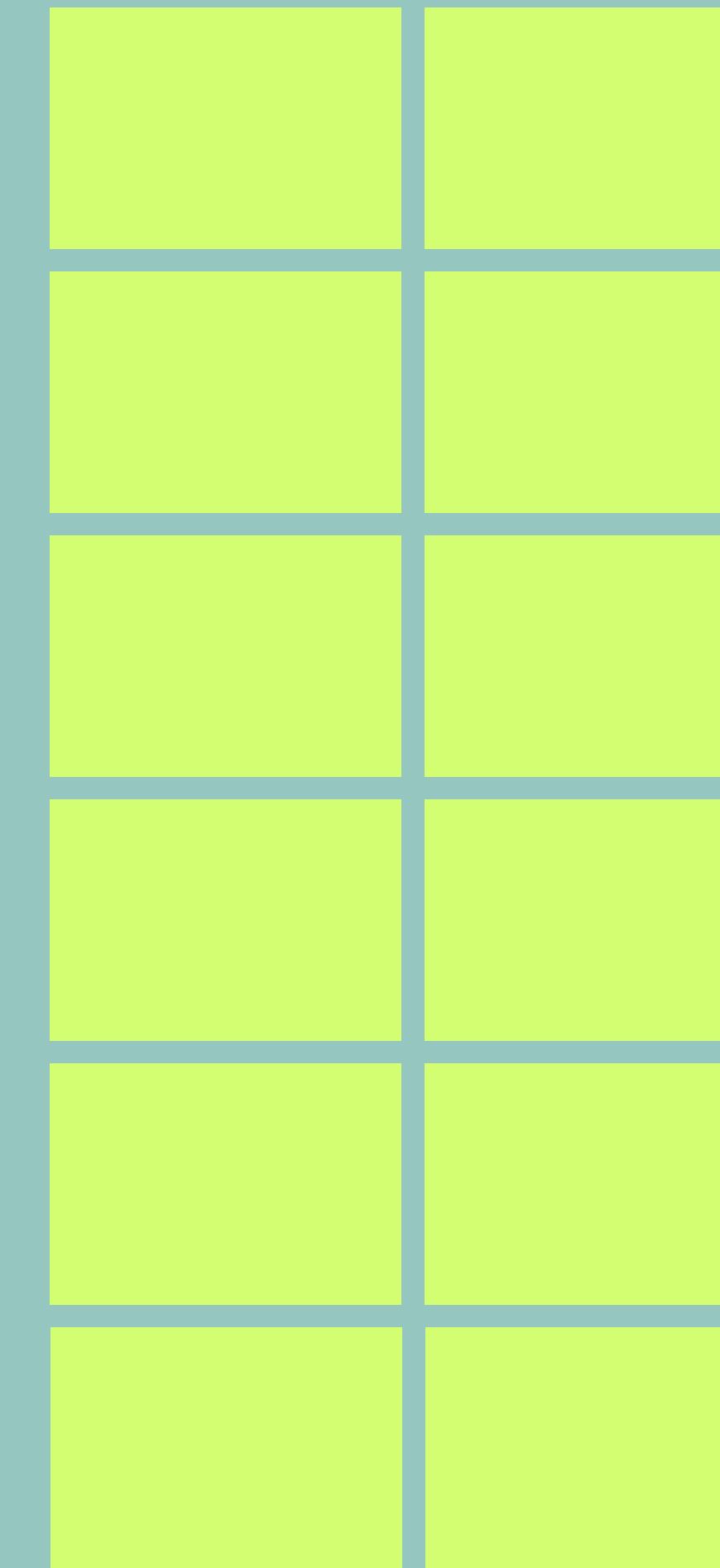
[6 PPOG \(Princípios da Programação Orientada a Gambiarra\)](#)

**CUSTO X BENEFÍCIO**

# CUSTO X BENEFÍCIO

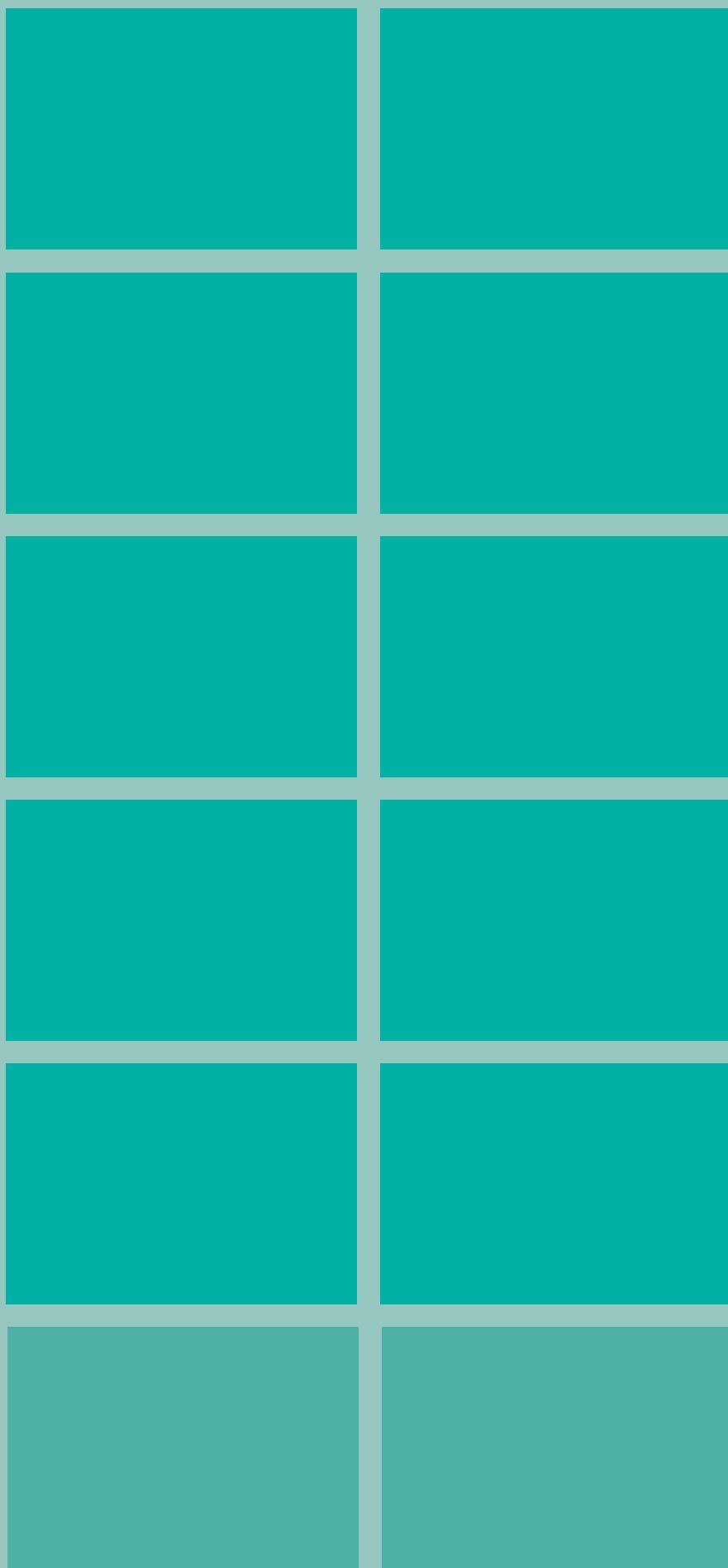


Qualidade

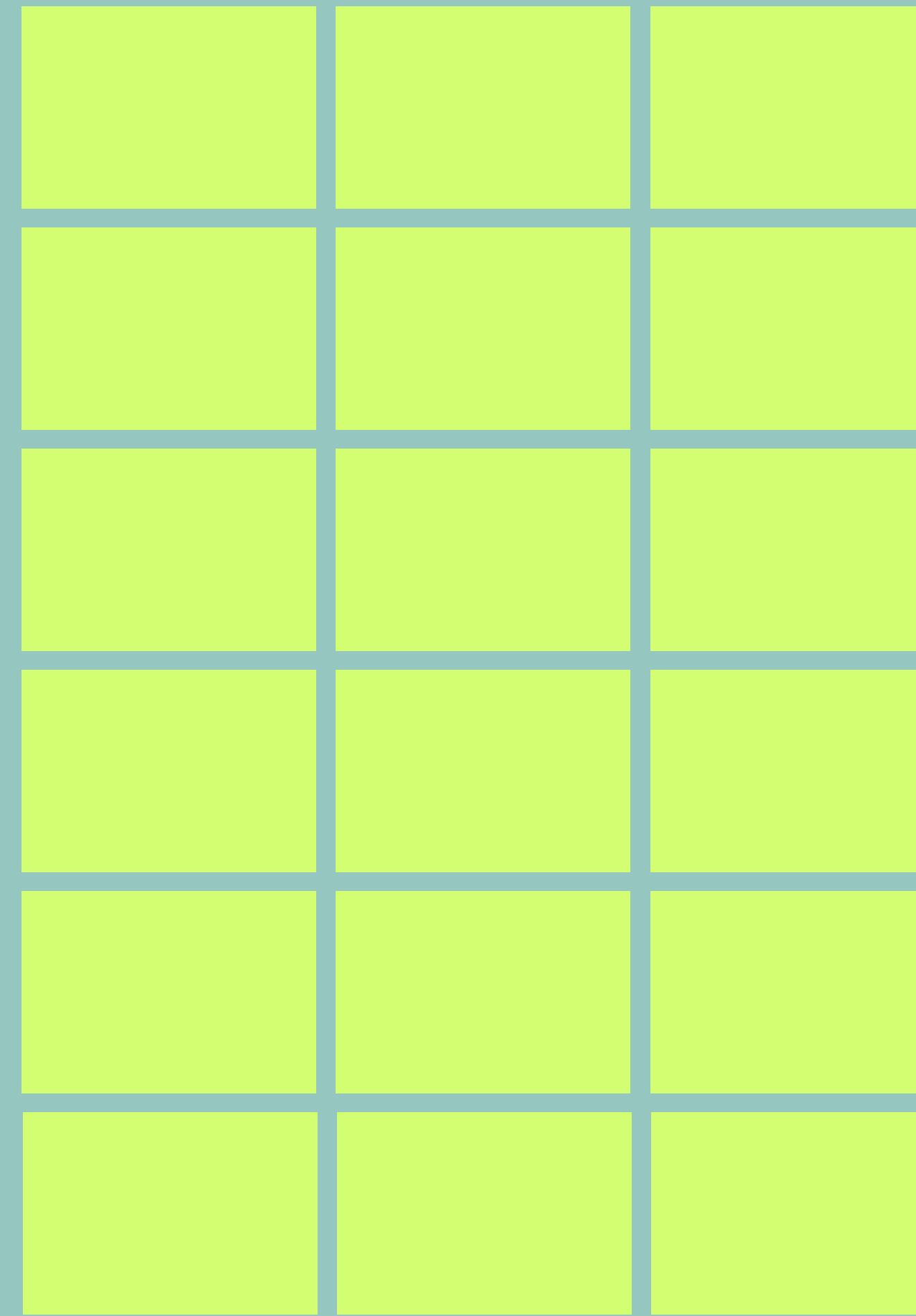


Custo Total

# CUSTO X BENEFÍCIO

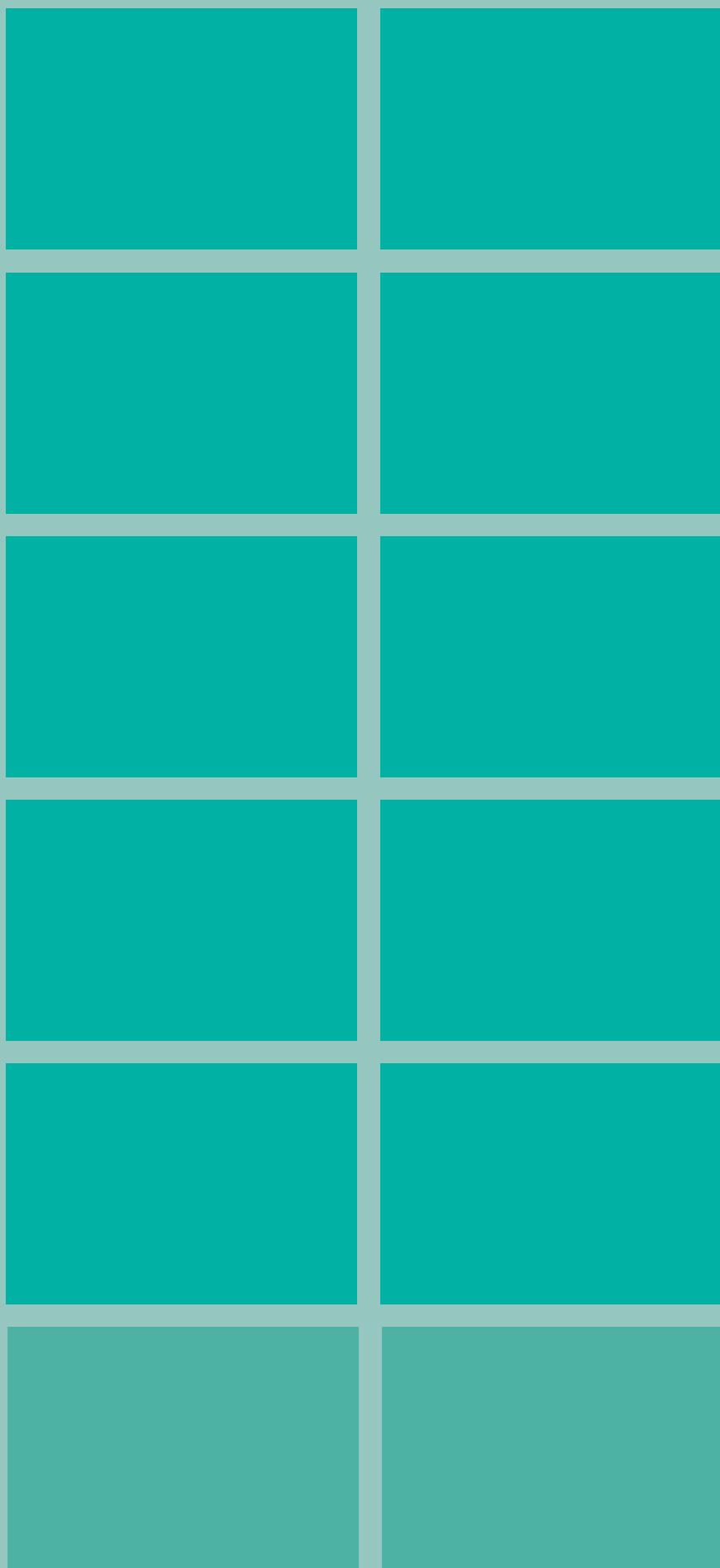


Qualidade

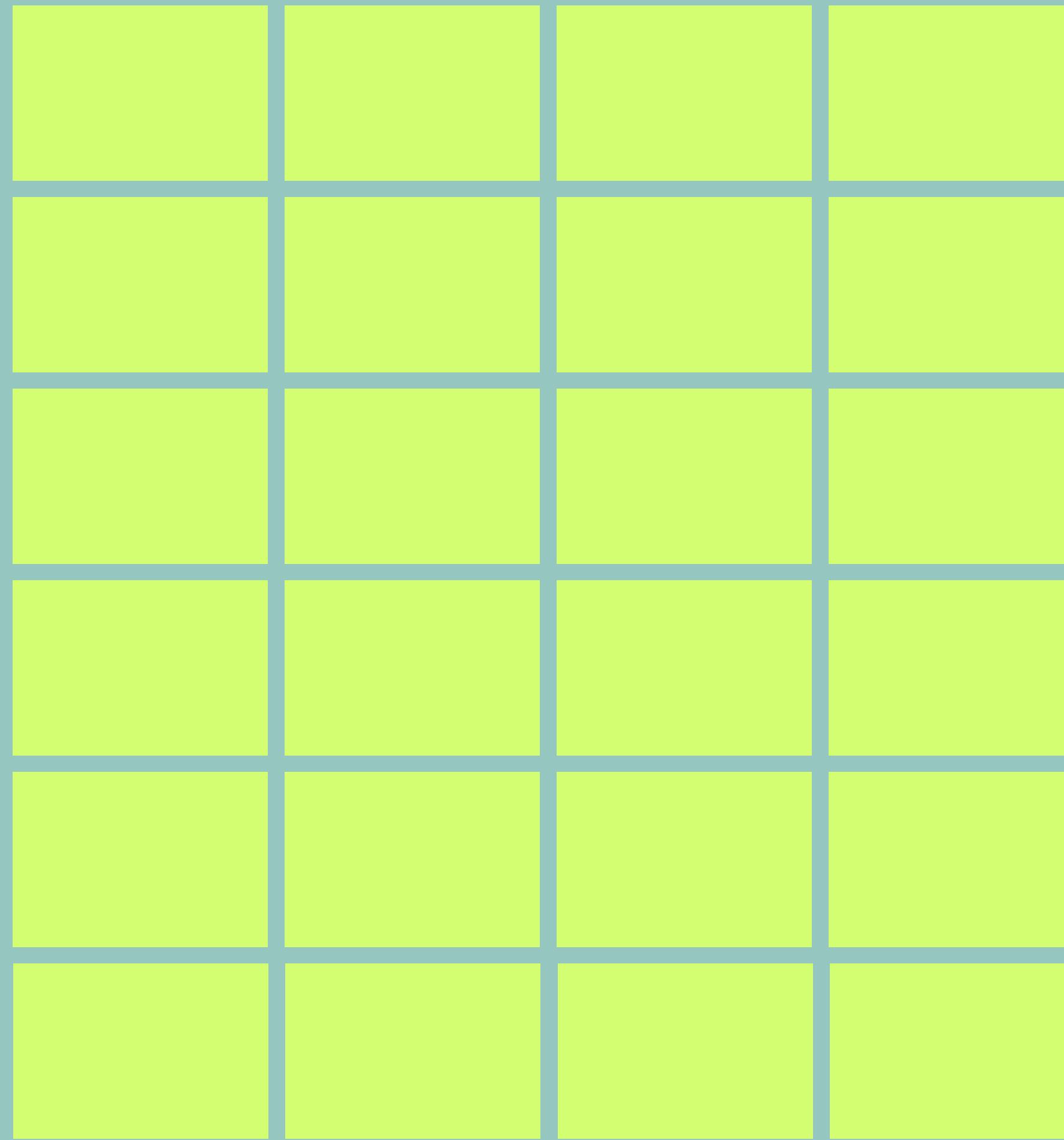


Custo Total

# CUSTO X BENEFÍCIO

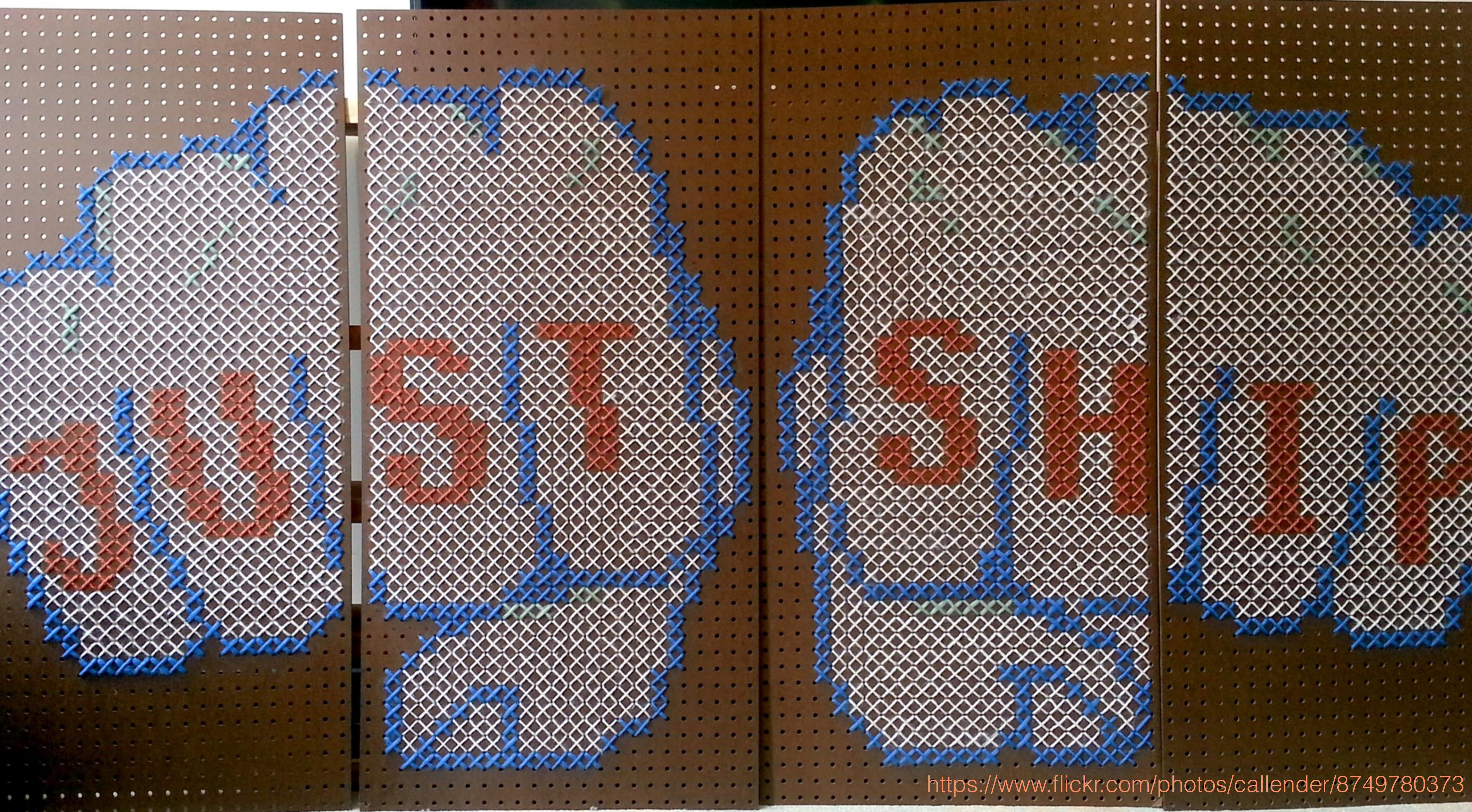


Qualidade



Custo Total

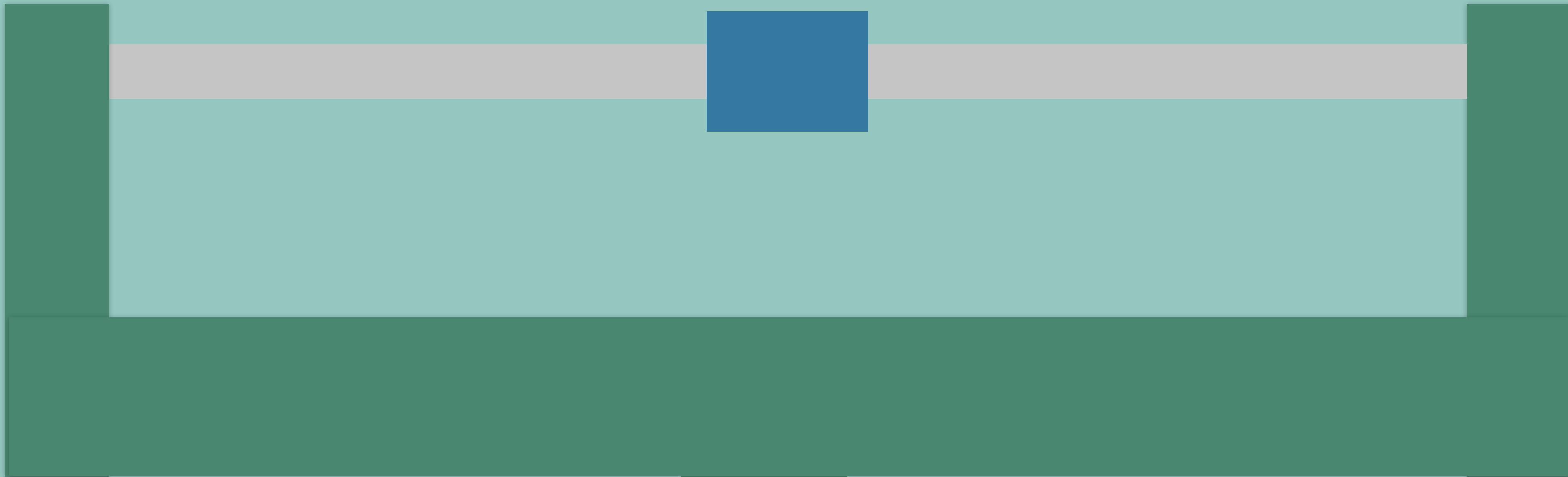
**CULTURA DE STARTUPS = STATUS QUO**



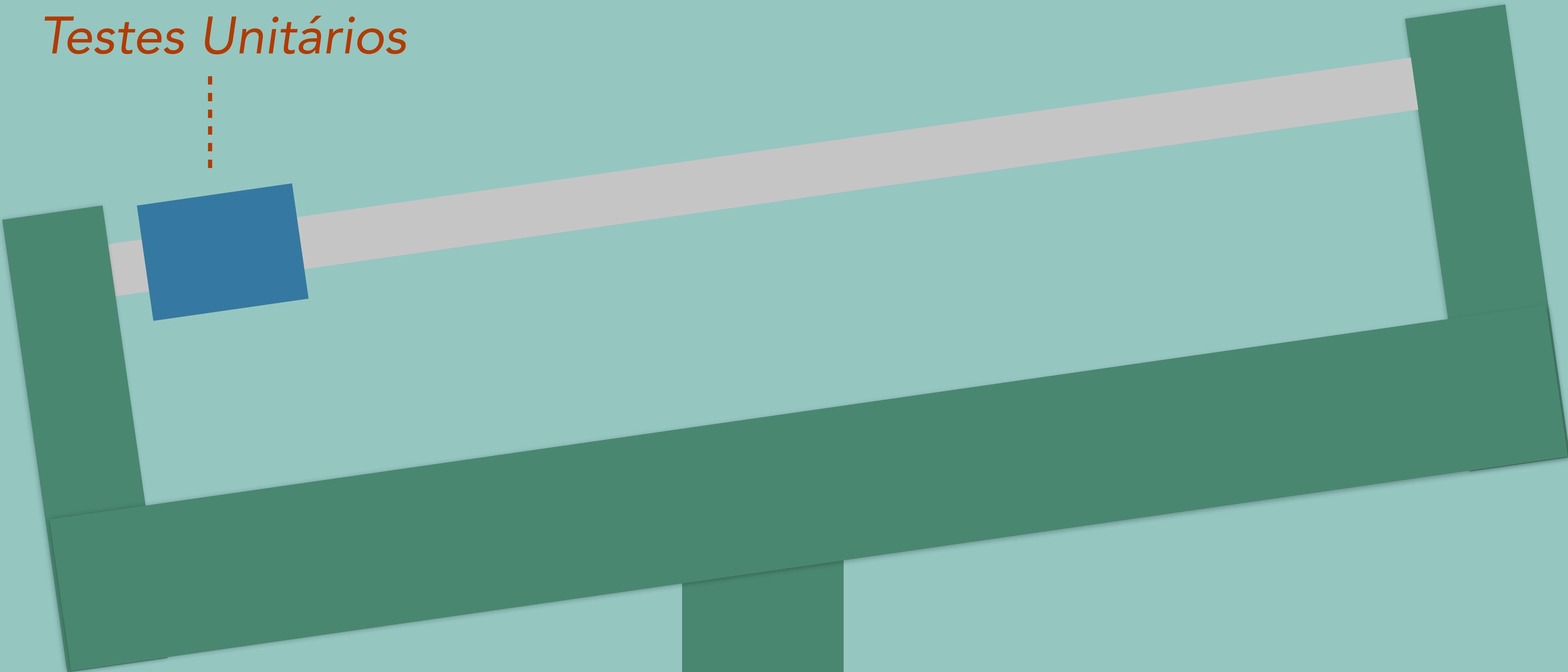
<https://www.flickr.com/photos/callender/8749780373>

**RUBY, PHP, PYTHON, JAVASCRIPT:  
PROCESSOS HUMANOS >  
COMPUTADORES E ALGORITMOS**

# **VERIFICAÇÃO FORMAL**



# *Testes Unitários*



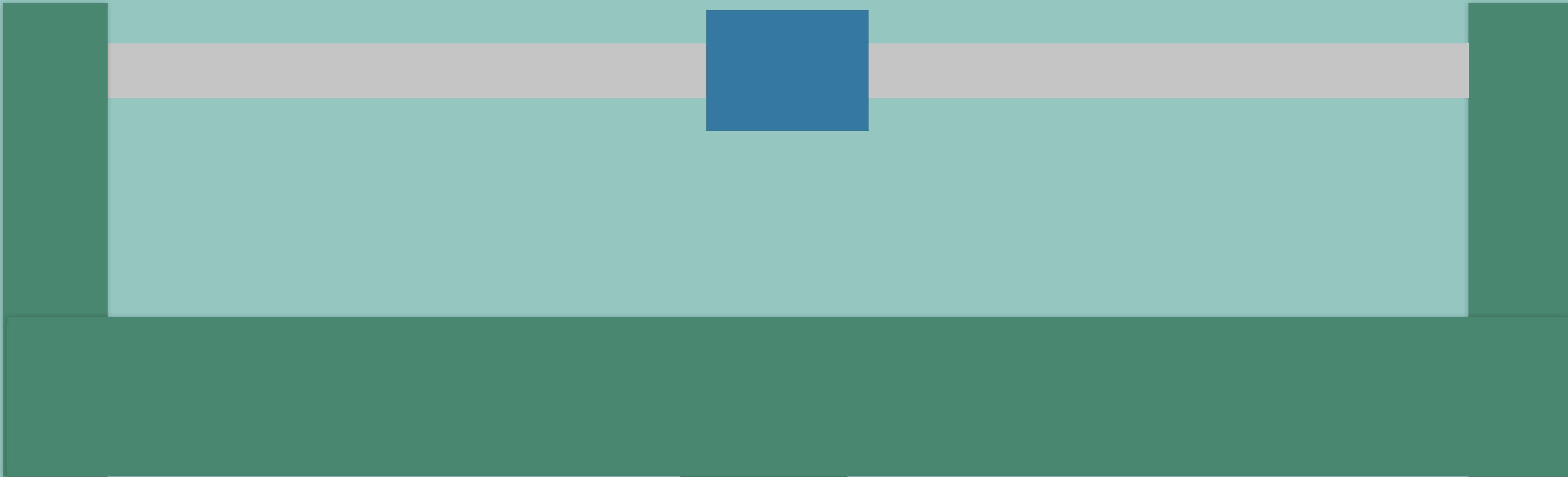
# Property-based testing



# Especificações Formais / Provas Matemáticas



???



# **DESIGN BY CONTRACT**

# TIPOS DE ENTRADA

# **TIPOS DE ENTRADA**

# **TIPOS DE SAÍDA**

# TIPOS DE ENTRADA

# TIPOS DE SAÍDA

# ERROS E EXCEÇÕES

**TIPOS DE ENTRADA**

**TIPOS DE SAÍDA**

**ERROS E EXCEÇÕES**

**EFEITOS COLATERAIS**

**TIPOS DE ENTRADA**

**PRÉ-CONDIÇÕES**

**TIPOS DE SAÍDA**

**ERROS E EXCEÇÕES**

**EFEITOS COLATERAIS**

**TIPOS DE ENTRADA**

**PRÉ-CONDIÇÕES**

**TIPOS DE SAÍDA**

**PÓS-CONDIÇÕES**

**ERROS E EXCEÇÕES**

**EFEITOS COLATERAIS**

**TIPOS DE ENTRADA**

**PRÉ-CONDIÇÕES**

**TIPOS DE SAÍDA**

**PÓS-CONDIÇÕES**

**ERROS E EXCEÇÕES**

**INVARIANTES**

**EFEITOS COLATERAIS**

**TIPOS DE ENTRADA**

**TIPOS DE SAÍDA**

**ERROS E EXCEÇÕES**

**EFEITOS COLATERAIS**

**PRÉ-CONDIÇÕES**

**PÓS-CONDIÇÕES**

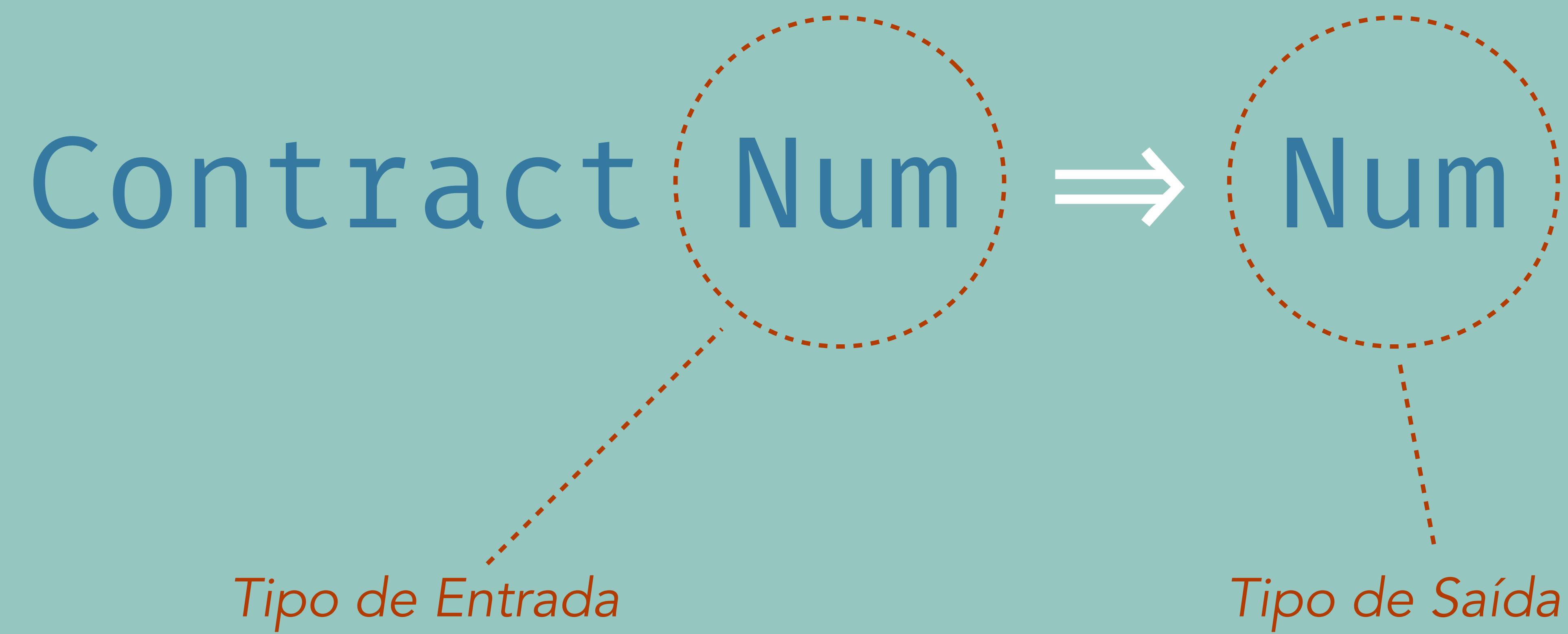
**INVARIANTES**

**PERFORMANCE**

```
put (x: ELEMENT; key: STRING) is
    -- Insert x so that it will be retrievable through key.
require
    count ≤ capacity
    not key.empty
do
    ... Some insertion algorithm ...
ensure
    has (x)
    item (key) = x
    count = old count + 1
end
```

# CHECAGEM EM TEMPO DE EXECUÇÃO

**CONTRACTS.RUBY**



A screenshot of an Emacs window titled "example1.rb - emacs". The window has a dark background and a light gray header bar. The title bar shows the file name "example1.rb" and the mode "emacs". The main buffer area is completely blank, showing only the cursor at the top center. The bottom status bar is visible, displaying the file name "example1.rb", the line number "(1,0)", and a list of minor modes and packages: "(Ruby rt RubyRef RuboCop FlyC YARD yas Projectile[-]) [ ?? ]".

# DOCUMENTAÇÃO

```
*ruby* - emacs  
1 require "contracts"  
2  
3 class Example  
4   include Contracts::Core  
5   include Contracts::Builtin  
6  
7   Contract Num => Num  
8   def double(x)  
9     x * 2  
10  end  
11 end  
  
U:--- example1.rb      (7,21)  (Ruby rt RubyRef RuboCop FlyC YARD yas Projectile[-]) [Example]  
40 irb(main):019:0*  
  
U:**- *ruby*      (40,17)  (Inf-Ruby:run yas Projectile[-] Shell-Compile)
```

Contract `ArrayOf[String] ⇒ Fixnum`

```
def average_length(strings)
  strings.map(&:size).reduce(0, :+) / strings.size
end
```

The screenshot shows an Emacs window with two buffers:

- Top Buffer:** File name: `example2.rb`. Major mode: `emacs`. Status bar information: `(1,0) (Ruby rt RubyRef RuboCop FlyC YARD yas Projectile[-]) [??]`. The buffer content starts with `1 irb(main):001:0>` .
- Bottom Buffer:** File name: `*ruby*`. Major mode: `Inf-Ruby:run yas Projectile[-] Shell-Compile`. Status bar information: `(1,17)`.

Contract `ArrayOf[RespondTo[:size]]`  $\Rightarrow$  `Fixnum`

```
def average_length(values)
  values.map(&:size).reduce(0, :+) / strings.size
end
```

```
*ruby* - emacs  
1 require "contracts"  
2  
3 module Example3  
4   include Contracts::Core  
5   include Contracts::Builtin  
6  
7   Contract ArrayOf[RespondTo[:size]] => Fixnum  
8   def self.average_length(values)  
9     values.map(&:size).reduce(0, :+) / values.size  
10  end  
11 end  
  
U:--- example3.rb      (9,0)      (Ruby rt RubyRef RuboCop FlyC YARD yas Projectile[-]) [Example3]  
1 irb(main):001:0>  
  
U:**- *ruby*      (1,17)      (Inf-Ruby:run yas Projectile[-] Shell-Compile)
```

# **CONTRATOS PERSONALIZADOS**

The screenshot shows an Emacs window with a dark theme. At the top, the title bar reads "example4.rb - emacs". The main buffer contains the following Ruby code:

```
1 require "contracts"
2
3 Person = Struct.new(:name, :gender)
4
5 module Example4
6   include Contracts::Core
7   include Contracts::Builtin
8
9   def self.boy_names(boys)
10    boys.map(&:name)
11  end
12 end
```

Below the code, the status bar displays the file name "U:\*\*\* example4.rb" and the position "(9,0) (Ruby rt RubyRef RuboCop FlyC YARD yas Projectile[-]) [Exa". In the bottom left, an inferior Ruby shell is running, indicated by the prompt "1 irb(main):001:0>". In the bottom right, another inferior Ruby shell is running, indicated by the prompt "U:\*\*\* \*ruby\* (1,17) (Inf-Ruby:run yas Projectile[-] Shell-Compile)".

```
*ruby* - emacs
```

```
1 require "contracts"
2
3 Person = Struct.new(:name, :gender) []
4
5 module Example4
6   include Contracts::Core
7   include Contracts::Builtin
8
9 Contract ArrayOf[→(p) { p.gender == :male }] => ArrayOf[String]
10 def self.boy_names(boys)
11   boys.map(&:name)
12 end
13 end
```

```
U:***- example4.rb      (3,35)  (Ruby rt RubyRef RuboCop FlyC YARD yas Projectile[-]) [ ???
1 irb(main):001:0>
```

```
U:***- *ruby*      (1,17)  (Inf-Ruby:run yas Projectile[-] Shell-Compile)
```

# **INARIANTES**

```
example5.rb - emacs
```

```
1 require "contracts"
2
3 Person = Struct.new(:name, :gender) do
4   include Contracts::Core
5   include Contracts::Builtin
6   include Contracts::Invariants
7
8   Contract Symbol => Symbol
9   def reassigned_gender(new_gender)
10    self.gender = new_gender
11  end
12 end
```

```
U:--- example5.rb          (8,0)      (Ruby rt RubyRef RuboCop FlyC* YARD yas Projectile[-]) [?]
9         from /Users/dodecaphonic/.rubies/ruby-2.3.0/bin/irb:11:in `<main>'
10 irb(main):002:0> require "./example5"
11 => true
12 irb(main):003:0>
13
14 Process ruby finished
15 irb(main):001:0>
```

```
U:**- *ruby*          (15,17)  (Inf-Ruby:run yas Projectile[-] Shell-Compile)
Wrote /Users/dodecaphonic/Projects/Personal/lightning-talks/contracts.ruby/examples/example5.rb
```

# MAIS DETALHES

[https://en.wikipedia.org/wiki/Design\\_by\\_contract](https://en.wikipedia.org/wiki/Design_by_contract)

<https://www.eiffel.com/values/design-by-contract/introduction/>

<https://egonschiele.github.com/contracts.ruby>

**OBRIGADO.**