

Práticas de Desenvolvimento de Software

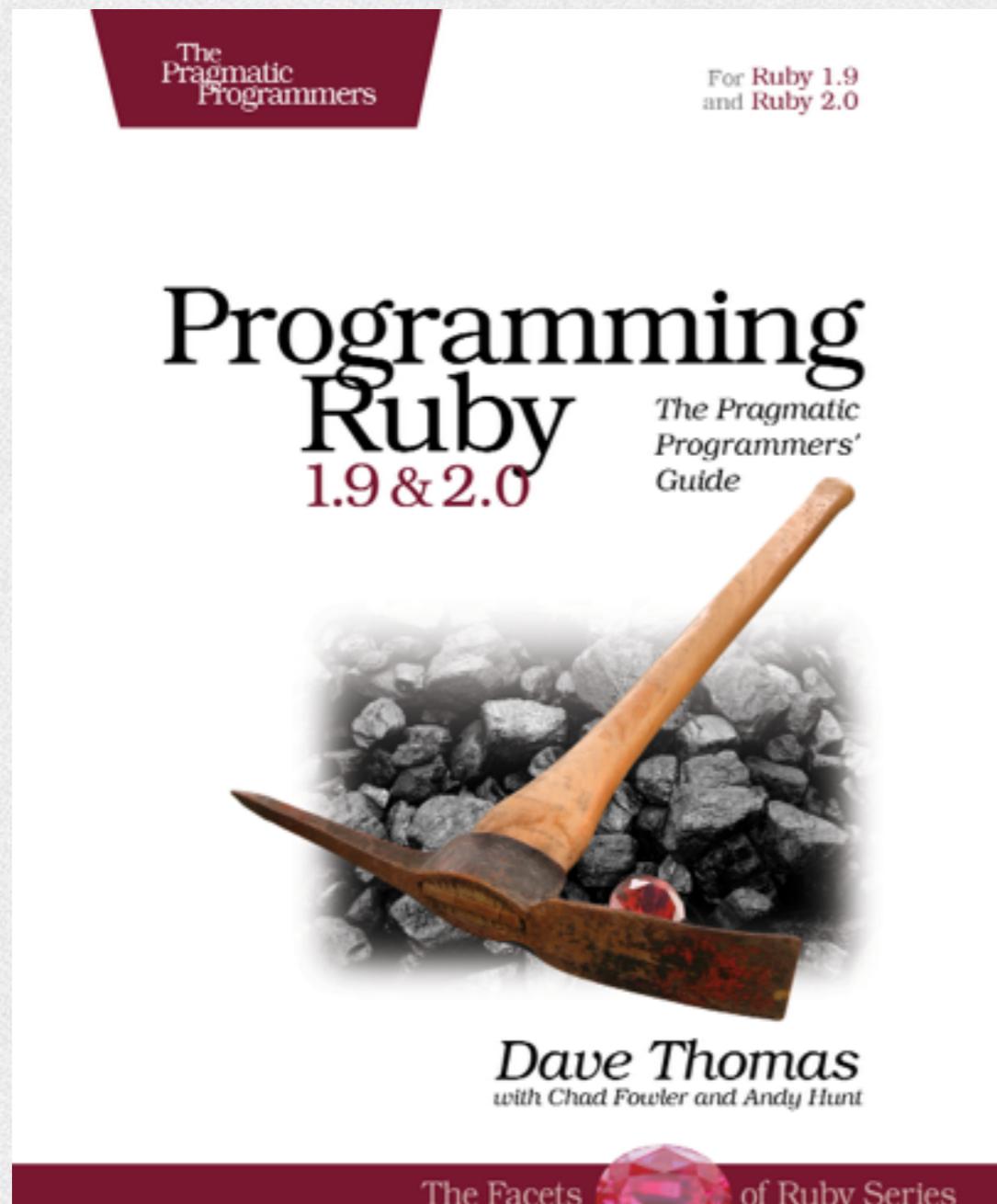
#

Aula 03

Introdução à linguagem de programação Ruby

Introdução à linguagem de programação Ruby

Bibliografia



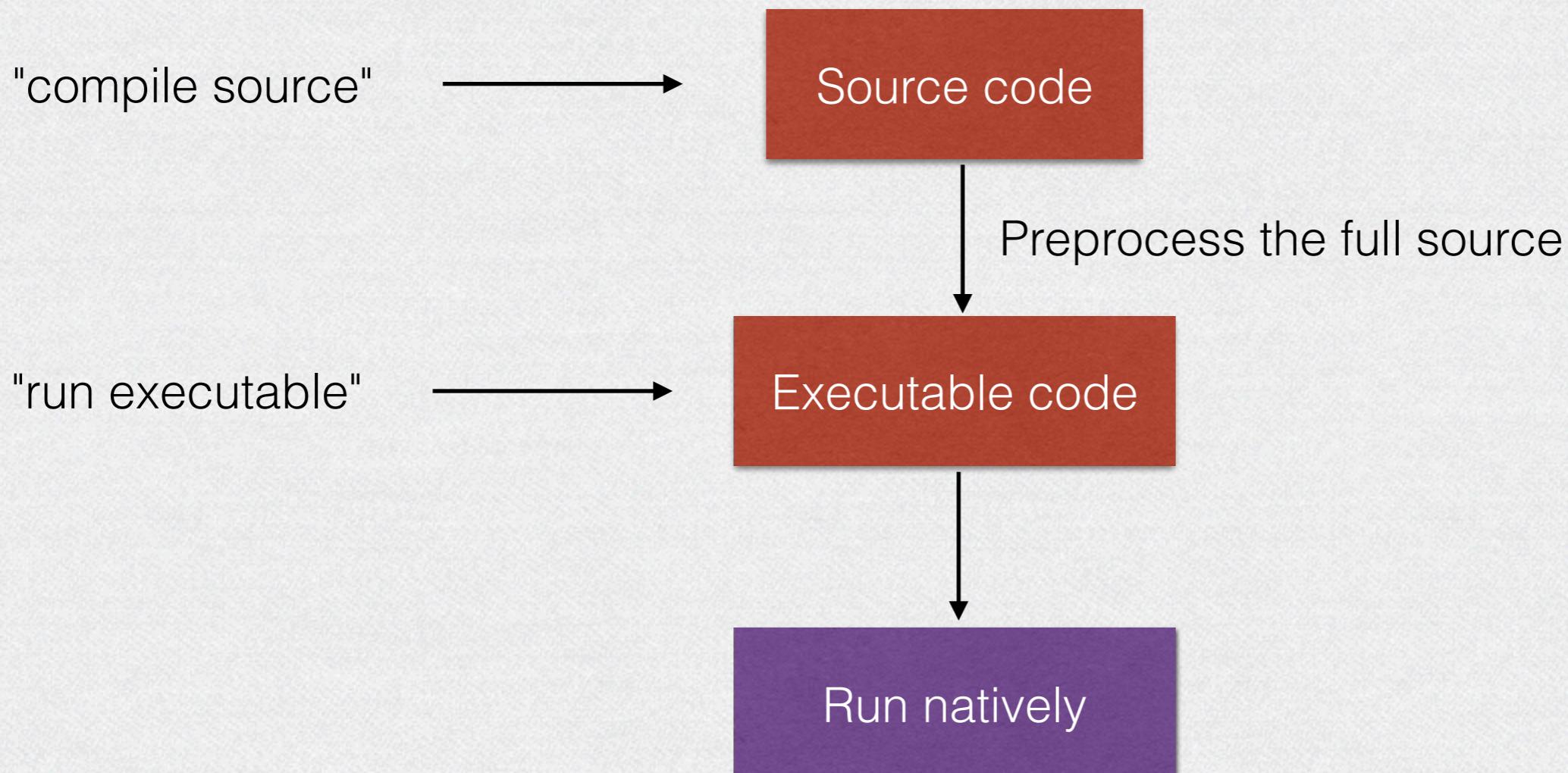
Programming Ruby 1.9 & 2.0
4th edition
by Dave Thomas, with
Chad Fowler and Andy Hunt

Ruby

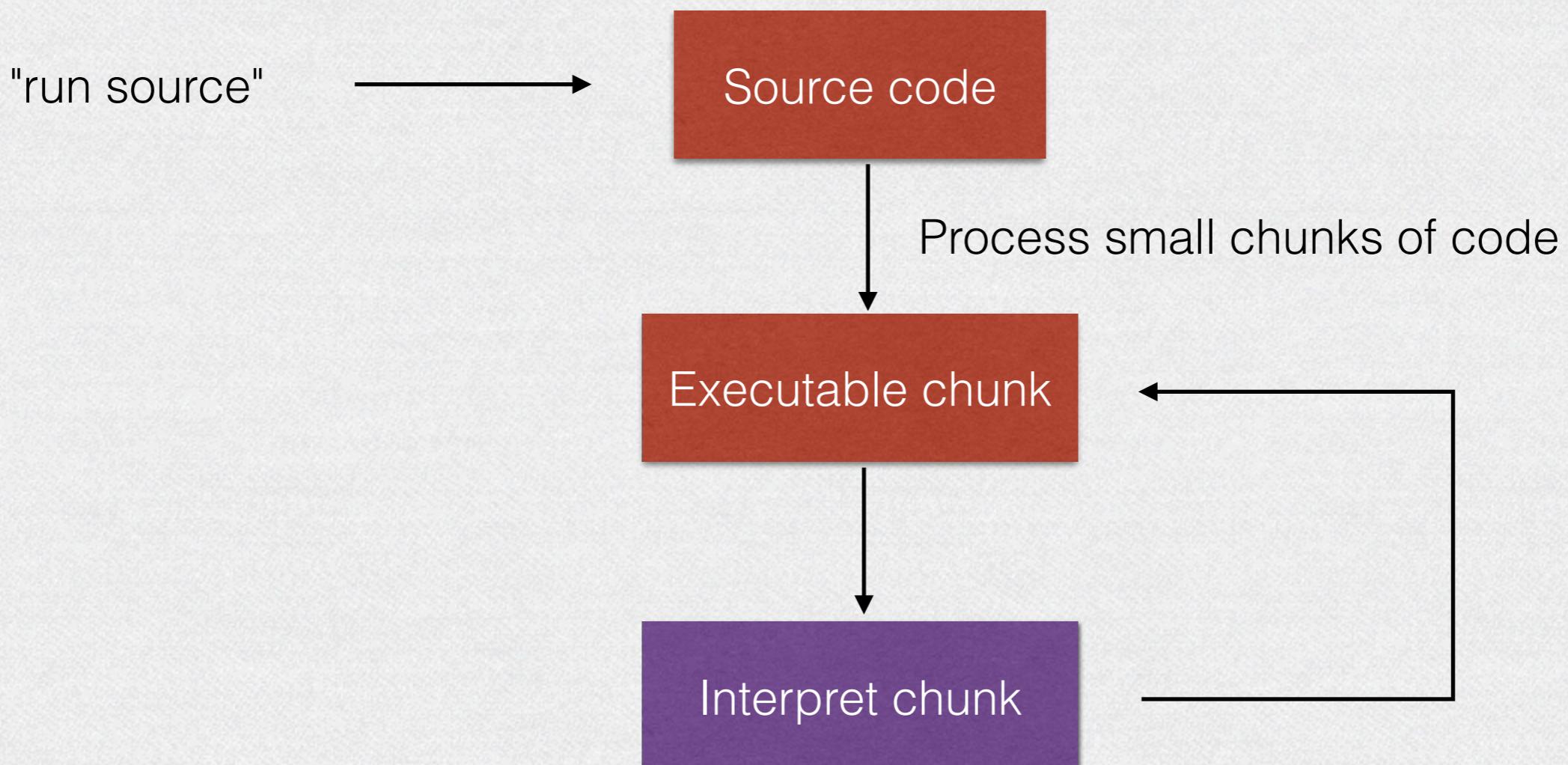
- Criada por Yukihiro Matsumoto em 1995
 - “I was thinking about the possibility of an object-oriented scripting language. Perl was ugly. Python wasn’t a true OO language.”
- Popularizada em 2005 com o framework web **Ruby on Rails**
- Sintaxe simples, mas elegante, com foco em produtividade
- Código aberto (<https://github.com/ruby/ruby>)
- Multiparadigma (orientação a objeto, procedural, funcional)
- Multiplataforma (Linux, Mac OS X, Windows, ...)
- Linguagem interpretada

Linguagem compilada vs interpretada

Linguagem compilada vs interpretada



Linguagem compilada vs interpretada



Interpretadores Ruby

- **MRI** (Matz Ruby Interpreter)
- jRuby (implementação em Java)
- ironRuby (implementação em .Net)
- Rubinius (implementação em Ruby)

`ruby -v`

ruby 2.0.0p0 (2013-02-24 revision 39474) [x86_64-darwin12.3.0]

Instalação

- Sugestão para gerenciar múltiplas versões de Ruby em um computador: RVM
 - <http://rvm.io/rvm/install>
- Se você usa windows, boa sorte!
- No box do Vagrant e no servidor remoto, Ruby já está instalado! Yey \o/

Introdução à linguagem de programação Ruby

Quem usa Ruby?

Twitter	Hulu	Basecamp	Groupon
Airbnb	SoundCloud	ZenDesk	Scribd
SlideShare	GitHub	Heroku	Infosimples

Introdução à linguagem de programação Ruby

irb

permite interpretar código passo a passo

```
[ubuntu] ~ (ruby-2.1.1)
$ irb
2.1.1 :001 > 2 * 3
=> 6
2.1.1 :002 > 5.times { |n| puts n }
0
1
2
3
4
=> 5
2.1.1 :003 > _
```

Introdução à linguagem de programação Ruby

Execução de um script

```
# my_script.rb  
# exemplo de um script em Ruby  
5.times do |i|  
    puts i**10  
end
```

```
[ubuntu] ~ (ruby-2.1.1)  
$ ruby my_script.rb  
0  
10  
20  
30  
40
```

Introdução à linguagem de programação Ruby

Tipos de dados mais comuns

Tipo	Exemplos
Nulo	nil
Booleano	true, false
Inteiro	1, 100, 3000
Ponto flutuante	2.53, 3.1415
String	"duplas", 'simples'
Símbolos	:username, :age, :password
Array	[1, 2, 3]
Hash	{ 'a' => 1, 'b' => 2, 'c' => 3 }

operadores aritméticos

Operador	Exemplos
+	1 + 1 # 2
-	9 - 5 # 4
*	3 * 2 # 6
/	10 / 3 # 3
%	10 % 3 # 1

operadores relacionais

Operador	Exemplos
>	<code>5 > 3 # true</code>
<code>>=</code>	<code>10 >= 11 # false</code>
<	<code>3 < 3 # false</code>
<code><=</code>	<code>3 <= 3 # true</code>
<code>==</code>	<code>5 == 5.5 # false</code>
<code>!=</code>	<code>9 != 2 # true</code>

Introdução à linguagem de programação Ruby

operadores condicionais

Operador	Exemplos
&& and	true && false # false
 or	true true # true
!	!true # false

operadores de atribuição

Operador	Exemplos
=	var = 1 # 1
+=	var = 1 var += 1 # 2
-=	var = 1 var -= 1 # 0
*=	var = 2 var *= 3 # 6
/=	var = 4 var /= 2 # 2
%=	var = 3 var %= 2 # 1

Comentário

```
# exemplos de linha comentada  
=begin  
# exemplo de trecho de código comentado  
def todo  
end  
=end  
  
def done  
  puts "Ready to go"  
end
```

Introdução à linguagem de programação Ruby

String

exemplo de concatenação

```
a = "Hello"
```

```
b = "World"
```

```
c = a + b # => "HelloWorld"
```

exemplo de interpolação

```
s = "2 * 3 = #{2*3}" # => "2 * 3 = 6"
```

Introdução à linguagem de programação Ruby

Array

```
# exemplos de lista
a = [ 1, 2, 3, 4, 5 ]
a[0] # 1
a[4] # 5
a[-1] # 5
a.pop # retira o último elemento: 5
a.shift # retira o primeiro elemento: 1
a.push(10) # adiciona 10 no fim da lista
a.unshift(20) # adiciona 20 no começo da lista
a.size # 5 (tamanho da lista)
a.index(2) # 1 (posição da primeira ocorrência do elemento 2)
a.count(3) # 1 (quantidade de ocorrências do elemento 3)
```

Introdução à linguagem de programação Ruby

Hash

```
# exemplos de uma tabela associativa
h = { "nome" => "Rafael", "sobrenome" => "Barbolo", "idade" => 27 }
h["nome"] # "Rafael"
h["sobrenome"] # "Barbolo"
h["idade"] # 27
a.size # 3 (quantidade de elementos no hash)
a.keys # [ "nome", "sobrenome", "idade" ]
a.values # [ "Rafael", "Barbolo", 27 ]
```

Declaração de função

```
def nome_da_funcao arg1, arg2 = 3
    puts “Esta é uma função que recebeu arg1 = #{arg1}...”
    puts “e pode receber arg2=#{arg2}, mas por padrão arg2 = 3.
    return arg1 * arg2 # “return” pode ser omitido
end

def nome_da_funcao
    puts “Esta é uma função que não recebe argumentos e retorna 1000”
    1000
end

# é convenção em Ruby usar snake_case para nomes de função e de variável
```

Impressão

```
# exemplos de impressão
print "imprimindo sem quebra de linha"
puts "imprimindo com quebra de linha"
p "p é um alias de puts"
puts 2**5 # => 10
```

Controle condicional: if/else

```
# exemplo de controle condicional
if condicao_1
    puts "condicao 1"
elsif condicao_2
    puts "condicao 2"
else
    puts "outro caso"
end
```

Controle condicional: case/when

```
# exemplo de controle condicional
case condicao
when 1
    puts "condicao == 1"
when 2
    puts "condicao == 2"
else
    puts "outro caso"
end
```

Controle de iteração: while

```
# exemplo de controle de iteração
while condicao
    next # pula para próxima iteração
    redo # reexecuta iteração atual
    break # cancela próximas iterações (sai do laço)
end
```

Controle de iteração: each

```
# exemplo de como navegar em elementos de um Array  
a = [ 1, 2, 3, 4, 5 ]  
a.each do |elemento|  
    puts elemento * 10  
end
```

não é comum utilizar **for** em Ruby para navegar em um conjunto de elementos

Ler/escrever em arquivo

exemplo de leitura

```
contents = File.open("caminho/do/arquivo", "r").read
```

exemplo de escrita

```
contents = "conteudo a ser gravado"  
f = File.open("caminho/do/arquivo", "w")  
f.write(contents)  
f.close
```

Rubygems: instalação de bibliotecas

deve ser executado na linha de comando unix
gem install nome_da_biblioteca

exemplo de instalação do Rails
gem install rails

Introdução à linguagem de programação Ruby

Exemplo

```
# fibonacci: 1 1 2 3 5 8 13 ...
def fibonacci n
    return n if [0, 1].include? n
    fibonacci( n - 1 ) + fibonacci( n - 2 )
end

puts fibonacci( 6 )
# => 8
```

Exercícios

1. Crie um script ruby ex3.1.rb que conta quantas letras tem o seu nome (n) e imprime n vezes o caractere “-” (hífen).
2. Crie um script ruby ex3.2.rb com a função **fatorial** que calcula o fatorial de um número inteiro positivo. Calcule o fatorial de 20.
3. Crie um script ruby ex3.3.rb com a função **ordena** que recebe um array como entrada e retorna esse array ordenado. Ordene [20, 0, 50, 30, 34, 33, 35, 22, 1].
4. Crie um script ruby ex3.4.rb com a função **upcase** que recebe uma string alfanumérica como entrada e retorna essa string com letras maiúsculas. Chame a função com a string “introducao a ruby finalizada”. Dica: use um Hash.