



Ingeniería de Software

1 - Ruby & Active Record

IIC2143-3

Josefa España

jpespana@uc.cl



Contenidos

- Comentarios
- Aspectos básicos
- Strings
- Input
- Números
- Comparación
- Métodos
- Condicionales
- Arrays
- Hashes
- OOP
- Herencia
- Convenciones

```
1  # imprimir "Hello, World!" en la consola
2  puts "Hello, World!"
```

```
~/Projects/ruby (0.148s)
ruby ruby.rb
Hello World!
```



Ruby



Vamos a ir programando juntos, para esto en la terminal escriban **irb** y accederemos al interactive ruby.





Comentarios

```
1  # Comentarios de una línea
2  puts 'Hello World!'
3  puts 'Hello Ruby!' # Comentarios
4
5  =begin
6  Comentario de múltiples líneas
7  puede tener código de ruby
8  y no será ejecutado dentro del comentario
9  =end
```



Aspectos básicos

```
1  address = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
2
3  # imprimir
4  p address
5
6  # index
7  p address[5]
8
9  new_address = address.reverse!
10 p new_address
```



Strings

```
1  # Concatenar strings
2  first_name = 'Josefa'
3  last_name = 'España'
4  puts first_name + ' ' + last_name
5
6  # Interpolación de strings
7  # solo funciona con comillas dobles
8  puts "Mi primer nombre es #{first_name} y mi apellido es #{last_name}"
9
```



Strings

```
10  # Cómo encontrar métodos
11  puts first_name.class
12  puts 10.class
13  puts 10.0.class
14
15  puts first_name.methods
16
17  # Métodos comunes
18  first_name.empty?
19  "".nil?
20
21  sentence = 'Estamos aprendiendo Ruby'
22  sentence.sub('Ruby', 'programación')
```



Strings

```
24  # Asignación de variables
25  first_name = 'Josefa'
26  new_first_name = first_name
27  first_name = 'John'
28  new_first_name # sigue siendo Josefa
29
30  # Escapar caracteres
31  # con \ podemos escapar símbolos
32  puts "My first name is \#{first_name} and my last name is \#{last_name}"
33  puts 'Hey John, \'How are you doing?\''
```




Input

```
1  puts "Qué te gustaría comer?"
2  food = gets.chomp
3  puts "Compraré #{food} entonces!"
4
5  puts "Ingresa un número y lo multiplicaré por 2"
6  input = gets.chomp
7  puts input.to_i * 2
```



Números

```
1 puts 1 + 2
2
3 puts 10 / 4 # retorna un integer, 2.
4 puts 10.0 / 4 # retorna un float
5 puts 10 / 4.to_f
6
7 x = 5
8 y = 10
9 puts y / x
```



Números

```
11  "5" * "5" # Error: no implicit conversion of String into Integer
12  "5" * 2   # "55"
13  2 * "5"   # Error: String can't be coerced into Integer
14
15  x = "5".to_i
16  y = "10".to_f
17  x * y
18
19  "hello".to_i # returns 0 when you can't convert to a number
```



Números

```
19  "hello".to_i # retorna 0 cuando no puedes convertir a un número
20
21  puts "Soy una línea"
22  puts "-" * 20
23  puts "Soy una línea diferente después de un divisor"
24  20.times {print "-"}
25
26  20.times {puts rand(10)} # no incluye al 10
```



Comparación

```
1  # Comparación
2  x = y
3  1 = 2
4  3 = 3
5  5 < 2
6  2 ≤ 5
7  5 > 2
8  5 && 6
9  5 || 6
10
11  10.eql?(10.0) # false, compara el tipo de dato
12  10 = 10.0 # true
```



Métodos

```
1  def multiply(first_num, second_num)
2      # no necesitamos usar return si queremos devolver
3      # el resultado de la última línea
4      first_num.to_f * second_num.to_f
5  end
6
7  def divide first_num, second_num
8      first_num.to_f / second_num.to_f
9  end
10
11 def subtract(first_num, second_num)
12     first_num.to_f - first_num.to_f
13 end
14
15 def sum(first_num, second_num)
16     first_num.to_f + first_num.to_f
17 end
18
19 def remaining(first_num, second_num)
20     first_num.to_f % second_num.to_f
21 end
```



Métodos

```
23 puts "Calculadora simple"
24 25.times{ print "-" }
25 puts
26 puts "Ingresa el primer número"
27 num_1 = gets.chomp
28 puts "Ingresa el segundo número"
29 num_2 = gets.chomp
30 puts "El primer número multiplicado por el segundo número es #{multiply(num_1, num_2)}"
31 puts "El primer número dividido por el segundo número es #{divide(num_1, num_2)}"
32 puts "El primer número restado por el segundo número es #{subtract(num_1, num_2)}"
33 puts "El primer número más el segundo número es #{sum(num_1, num_2)}"
34 puts "El resto de la división del primer número por el segundo número es #{remaining(num_1, num_2)}"
35
```



Métodos

```
1  def call_block
2      puts "Start of method"
3      yield
4      yield
5      puts "End of method"
6  end
7
8  call_block { puts "In the block" }
```




Condicionales

```
1  condition = true
2  another_condition = false
3
4  if condition && another_condition # y
5      puts "esto se evaluó como verdadero"
6  else
7      puts "esto se evaluó como falso"
8  end
9
10 if condition || another_condition # o
11     puts "Hola"
12 else
13     puts "Adiós"
14 end
```



Condicionales

```
39 puts "Calculadora simple"
40 20.times{ print "-" }
41 puts
42 puts "Por favor ingresa tu primer número"
43 first_number = gets.chomp
44 puts "Por favor ingresa tu segundo número"
45 second_number = gets.chomp
46 puts "¿Qué quieres hacer?"
47 puts "Ingresa 1 para multiplicar, 2 para sumar, 3 para restar"
48 user_entry = gets.chomp
49 puts "Has seleccionado #{user_entry}"
50 if user_entry == "1"
51     puts "Has elegido multiplicar"
52     puts "El resultado es #{multiply(first_number, second_number)}"
53 elsif user_entry == "2"
54     puts "Has elegido sumar"
55     puts "El resultado es #{add(first_number, second_number)}"
56 elsif user_entry == "3"
57     puts "Has elegido restar"
58     puts "El resultado es #{subtract(first_number, second_number)}"
59 else
60     puts "Entrada inválida"
61 end
```



Arrays

```
1  a = [1, 2, 3, 4, 5, 6, 7, 8, 9]
2  # índices: 0, 1, 2, 3, 4, 5, 6, 7, 8
3
4  # puts a    # imprimir con salto de línea
5  # print a   # imprimir sin salto de línea y como arreglo
6  # p a       # imprimir con salto de línea e índices
7
8  a.last # 9
9  a.first # 1
```



Arrays

```
11 # crear un rango de números entre 1 y 100
12 x = 1..100 # objeto rango
13 x = (1..100).to_a # objeto arreglo
14
15 x.to_a.shuffle # mezclar los números
16 z = x.to_a.shuffle! # mezclar los números y guardarlos en z como arreglo
17
18 x.reverse! # invertir los números y guardarlos en x como arreglo
19 # ! : operador bang, modificará el objeto original
20
```



Arrays

```
21 a = ("a" .. "z").to_a # crear un arreglo de letras de a a z
22 # agregar elemento al arreglo
23 a << 10
24 a.unshift("Josefa") # agregar elemento al principio del arreglo
25 a.append("Josefa") # agregar elemento al final del arreglo
26 a.uniq # eliminar duplicados
27 a.uniq! # eliminar duplicados y guardarlos en a como arreglo
28
29 a.empty? # verificar si el arreglo está vacío
30 a.include?("Josefa") # verificar si el arreglo contiene a Josefa
31 a.push("new item") # agregar elemento al final del arreglo
32 a.pop # eliminar el último elemento del arreglo
33 a.join # convertir arreglo a cadena de texto
34 a.join("-") # convertir arreglo a cadena de texto con - entre elementos
35 a.split("-") # convertir cadena de texto a arreglo con - entre elementos
```



Arrays

```
37 %w(mi nombre es josefa) # convertir cadena de texto a arreglo con espacio entre elementos
38 z = _ # guardar la expresión anterior en z como arreglo
39
40 z.each { |food| print food + " " } # iterar sobre el arreglo e imprimir cada elemento con espacio entre ellos
41 z.each { |food| print food.capitalize + " " } # iterar sobre el arreglo e imprimir cada elemento con espacio entre ellos y capitalizar cada elemento
42
43 x = (1..100).to_a.shuffle
44 x.select { |number| number.odd? } # seleccionar los números impares del arreglo
45
46 p x.methods # imprimir todos los métodos disponibles para x
```




Hashes

```
1 sample_hash = {'a' => 1, 'b' => 2, 'c' => 3}
2 my_details = {'name' => 'mashrur', 'favcolor' => 'red'}
3 p my_details['favcolor']
4 p sample_hash['b']
5
6 another_hash = {a: 1, b: 2, c: 3} # asignar pares clave valor usando símbolos
7 p another_hash[:a]
8
9 sample_hash.keys
10 sample_hash.values
11 sample_hash.each do |key, value|
12     puts "The class for key is #{key.class} and the value is #{value.class}"
13 end
14
15 sample_hash[:e] = "Mashrur"
16 sample_hash[:c] = "Ruby"
17
18 sample_hash.each { |some_key, some_value| puts "The key is #{some_key} and the value is #{some_value}" }
19 sample_hash.select { |k, v| v.is_a?(String) }
20 sample_hash.each { |k, v| sample_hash.delete(k) if v.is_a?(String) }
21
```



OOP

```
1  class Student
2      attr_accessor :first_name, :last_name, :email, :username
3
4      def initialize(first_name, last_name, username, email, password)
5          @first_name = first_name
6          @last_name = last_name
7          @username = username
8          @password = password
9          @email = email
10     end
11
12     def to_s
13         "First name: #{@first_name}, Last name: #{@last_name}, Username: #{@username}, Email: #{@email}"
14     end
15 end
16
17 felipe = Student.new("Felipe", "Muñoz", "felipemunoz1", "felipe@email.com", "password")
18 john = Student.new("John", "Doe", "john1", "john@email.com", "password1")
19 puts felipe
20 felipe.last_name = john.last_name
21 puts "felipe is altered"
22 puts felipe
```




Herencia

```
1  class Song
2    def initialize(name, artist, duration)
3      @name = name
4      @artist = artist
5      @duration = duration
6    end
7
8    def duration=(duration)
9      @duration = duration
10   end
11
12   def duration
13     @duration
14   end
15
16   def to_s
17     "Song: #{@name}--#{@artist} (#{@duration})"
18   end
19 end
20
21 song = Song.new('Anti-Hero', 'Taylor Swift', 3.2)
22 song.duration= 10
23 song.duration # => 10
24 song.duration = 20
```

```
26 class KaraokeSong < Song
27   def initialize(name, artist, duration, lyrics)
28     super(name, artist, duration)
29     @lyrics = lyrics
30   end
31
32   def to_s
33     super + " [#{@lyrics}]"
34   end
35 end
36
37 karaoke_song = KaraokeSong.new('Anti-Hero',
38                                'Taylor Swift',
39                                3.2,
40                                "I'm the problem")
41 karaoke_song.to_s
```



Convenciones

1. `snake_case` para métodos y variables.
2. `PascalCase` para nombres de clases.
3. Si el método retorna booleanos, debe ir con un signo de interrogación al final.
4. Identación de 2 espacios.

```
4  def is_even?(number)
5      number.even?
6  end
```



Ingeniería de Software

1 - Ruby

IIC2143-3

Josefa España

jpespana@uc.cl