Operaciones aritméticas

Objetivos

- Construir aplicaciones de tipo calculadora, donde el usuario ingresa valores y le entregamos resultados.

Para lograr esto tenemos que:

- Conocer los operadores aritméticos
- Conocer la precedencia de los operadores aritméticos
- Utilizar paréntesis para priorizar operaciones
- Operar sobre números enteros y flotantes

Introducción a operaciones y operadores

En Ruby existen herramientas que nos permiten, entre otras cosas, sumar, restar, asignar valores, comparar, etc. Todo esto -y mucho más- es posible gracias a los operadores.

Motivación

¿Por qué debemos aprender de operaciones matemáticas si estamos interesados en crear aplicaciones web o videojuegos?

Los operadores aritméticos los ocuparemos **todo el tiempo**, ya sea para calcular el total de un carro de compras o cambiar la posición de un personaje en un videojuego.

Operadores aritméticos

Los operadores aritméticos nos permiten realizar operaciones matemáticas sobre números.

Operador	Nombre	Ejemplo => Resultado
+	Suma	2 + 3 => 5
-	Resta	2 - 3 => -1
*	Multiplicación	3 * 4 => 12
/	División	12 / 4 => 3
**	Potencia	2 ** 4 => 16

Operaciones con variables

El proceso es exactamente igual si guardamos los valores en variables

```
a = 2
b = 3
puts a + b # 5
```

Creando un calculadora

Esto nos permite que el usuario ingrese los valores, transformarlos a números y luego operar.

```
a = gets.to_i
b = gets.to_i
puts "a + b es: #{a + b}"
puts "a * b es: #{a * b}"
```

Este código podemos guardarlo en el archivo `calculadora1.rb` y ejecutarlo con `ruby calculadora1.rb`

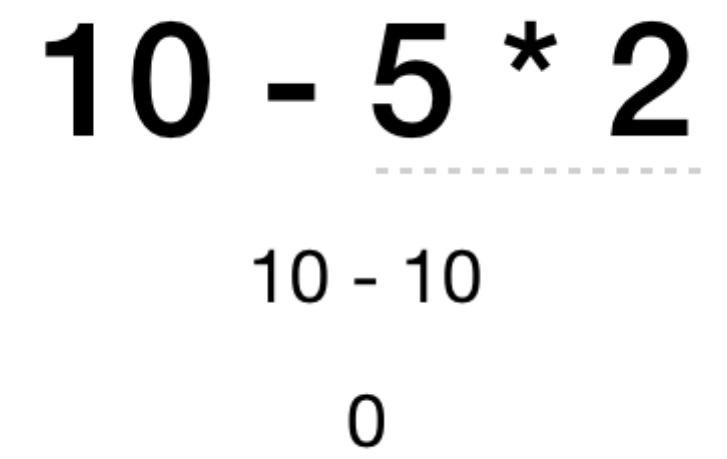
Precedencia de operadores

Un concepto muy importante que debemos conocer es el de precedencia. Dicho de otro modo, **saber en qué orden se realiza un grupo de operaciones** .

Ejemplo de precedencia

Por ejemplo, en la siguiente instrucción ¿cuál es el resultado?

```
10 - 5 * 2
```




```
10 - 5 * 2 # 0
```

Orden de las operaciones

Veamos una tabla simplificada de precedencia. Esta tabla está ordenada de mayor a menor prioridad, esto quiere decir que la operación de exponenciación precede a (se realiza antes que) la suma.

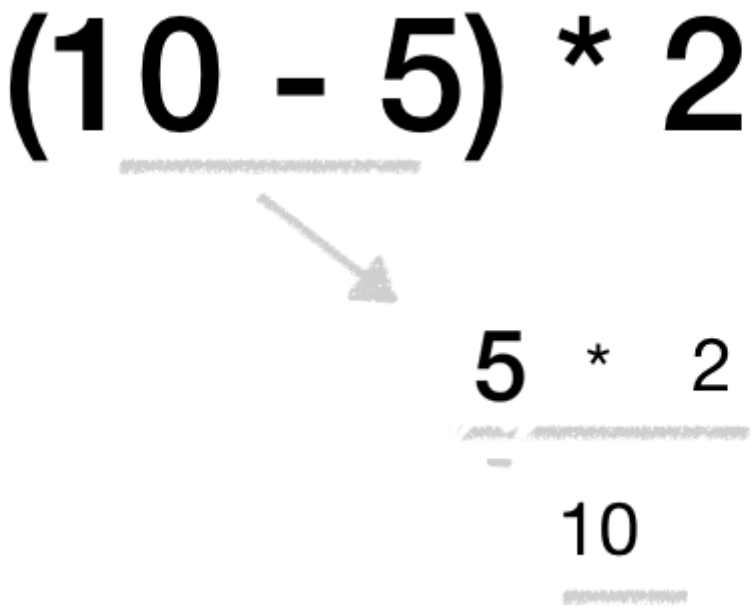
Operador	Nombre
**	Exponenciación (potencia)
* , %	Multiplicación, división y módulo
+ , -	Suma y resta

Cuando dos operaciones tienen el mismo nivel de precedencia se resuelven de izquierda a derecha

Operaciones y paréntesis

Al igual que en matemáticas, los paréntesis cambian el orden en que preceden las operaciones. Dando prioridad a las operaciones que estén dentro de los paréntesis.

```
(10 - 5) * 2
```



¡¡¡ Los paréntesis importan !!!

Operaciones con números enteros y decimales

Si dividimos números enteros nos encontraremos con una sorpresa.

```
5 / 3 # => 1
```

Esto es muy común, ocurre en casi todos los lenguajes de programación, pero normalmente esperamos una respuesta diferente. Para obtenerla, debemos ocupar otro tipo de dato, el float.

Float

En el capítulo anterior mencionamos que existía el tipo de dato asociado a los números **decimales**, llamado float.

```
(3.1).class # float
```

Enteros y floats

La división entre entero y float, o float y entero da como resultado un float.

```
5.0 / 3.0 # 1.6666666666666667
5 / 3.0 # 1.6666666666666667
```

La división entre floats también es un float.

Los floats son muy importantes dentro de la programación y tienen propiedades curiosas, una de las más importantes es que solo almacenan una representación aproximada de los números.

```
(10 / 3.0)
 => 3.3333333333333335
```

También podemos transformar a float ocupando el método to_f, esto será especialmente útil cuando estemos trabajando con variables que contengan enteros.

```
a = gets.to_i # => 1
b = gets.to_i # => 2
puts a / b.to_f # 0.5
```

En algunos casos, también podemos transformar la variable a float desde un inicio```ruby a = gets.to_f