{desafío} latam_

Profundizando en flujo

Objetivo

- Aprenderemos a resolver problemas donde las posibles alternativas -a una decisión- sean más de dos.
- Analizar situaciones más allá de opción y caso contrario.
- Utilizar `ifs` anidados
- Utilizar la instrucción `elsif` en Ruby

Abordemos el problema desde un ejemplo

En el capítulo anterior, creamos un algoritmo que determinaba el mayor de dos números ingesados por el usuario. Utilizaremos el mismo ejemplo manejando el flujo cuando ambos números ingresados sean iguales.

Antes:

- caso1: `A >= B` => Decimos que A es mayor
- caso2: `A < B` => Decimos que A es menor

Ahora:

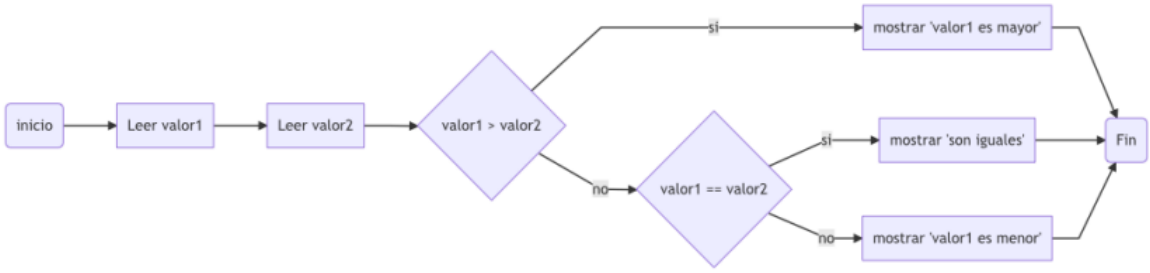
- caso1: `A > B` => Decimos que A es mayor
- caso2: `A == B` => Decimos que A y B son iguales
- caso3: `A < B` => Decimos que A es menor que B

El mayor de dos números: Escenario donde se ingresan dos números iguales

¿Cómo hacemos para mostrar que ambos son iguales? Ante cualquier problema de este tipo lo mejor es ir paso a paso y analizar el problema.

- Si el primero es mayor -> Mostramos mensaje.
- Si NO es mayor -> Tenemos dos opciones:
    - El primer valor es menor
    - Ambos son iguales

Con esta descomposición del problema, tenemos toda la información necesaria para construir nuestro diagrama:



Transcribiendo el diagrama

Para escribir este código crearemos el archivo `mayor_menor_o_igual.rb` y dentro agregaremos nuestro código

```
puts 'Ingrese valor1:'
valor1 = gets.to_i #asignación

puts 'Ingrese valor2:'
valor2 = gets.to_i #asignación

if valor1 > valor2 #comparación
   puts "valor1 #{valor1} es mayor"
else
   if valor1 == valor2 #comparación
      puts 'ambos valores son iguales'
   else
      puts "valor2 #{valor2} es mayor"
   end
end
```

Un código siempre debe ser probado en todos los casos, por lo que probaremos el programa 3 veces, ingresando los siguientes valores:

- 2, 1
- 2, 2
- 2, 3

Existe otra solución sin tener que agregar un `if` dentro de otro.

La instrucción ELSIF

La instrucción `elsif` nos permite capturar el flujo y volver a realizar una evaluación condicional cuando no se cumplió una evaluación previa. Con esta instrucción podemos reescribir el algoritmo del ejemplo anterior, pero conservaremos el archivo y crearemos uno nuevo, llamado `mayor_menor_o_igual2.rb`.

```
puts 'Ingrese valor1:'
valor1 = gets.chomp.to_i

puts 'Ingrese valor2:'
valor2 = gets.chomp.to_i

if valor1 > valor2
   puts "valor1 #{valor1} es mayor"
elsif valor1 == valor2
   puts 'son iguales'
else
   puts "valor1 #{valor2} es menor"
end
```

Podemos tener tantos elsif como necesitemos

La instrucción `elsif` nos permite continuar realizando tantas evaluaciones condicionales como necesitemos. Finalmente podemos utilizar la instrucción `else` cuando no se cumpla ninguna condición anterior.

Cabe destacar que dos instrucciones `if` **no son lo mismo** que utilizar `elsif`, porque la condicón de un `elsif` se evalúa sólo si no se cumple la correspondiente al `if`. `elsif` es en realidad una abreviación de "else if ".

Clasificación según rangos

Es normal estar en situaciones donde nos pidan clasificar algún elemento según un rango.

Por ejemplo supongamos que tenemos una palabra y queremos clasificarla en corta, mediana y larga:

- 4 letras o menos será corta.
- 5 a 10 será mediana.
- Más de 10 será larga.

Crearemos el archivo `clasificador.rb` para crear nuestro código.

```
puts 'Ingresa una palabra'
palabra = gets.chomp
largo = palabra.size

if largo <= 4
   puts 'Pequeña'
elsif largo < 10
   puts 'Mediana'
else
   puts 'Larga'
end
```

Ejercicio

Modifica el código anterior para poder distinguir palabras muy largas, cuando tengan 15 o más caracteres.

Solución

```
puts 'Ingresa una palabra'
palabra = gets.chomp
largo = palabra.size

if largo <= 4
   puts 'Pequeña'
elsif largo < 10
   puts 'Mediana'
elsif largo < 15
   puts 'Larga'
else
   puts 'Muy larga'
end
```

¿Cuál es la diferencia entre estos códigos?

```
puts 'Ingresa una palabra'
palabra = gets.chomp
largo = palabra.size

if largo <= 4
   puts 'Pequeña'
elsif largo < 10
   puts 'Mediana'
else
   puts 'Larga'
end
```

y:

```
puts 'Ingresa una palabra'
palabra = gets.chomp
largo = palabra.size

if largo <= 4
   puts 'Pequeña'
end
if largo < 10
   puts 'Mediana'
else
   puts 'Larga'
end
```

Probemos con la palabra 'hola', en el primer código obtendremos 'Pequeña', pero en el segundo 'Pequeña' y 'Mediana'

Una vez que una condición `if` o `elsif` se evalúa como verdadera, el flujo entra en esa dirección y no evalúa el resto de las condiciones. Con dos if se evalúa el primero y luego el segundo.