Condiciones de borde

¡Comencemos!

¿Qué aprenderás?

- Conocer la importancia de las condiciones de borde.
- Analizar condiciones de borde.
- Evaluar comportamiento de operadores >, >=, < y <= en Ruby.

¡Vamos con todo!

Introducción

En este capítulo aprenderemos a analizar código y buscar errores semánticos.

Es decir, cuando el código está bien escrito pero no hace exactamente lo que queremos que haga. Este tipo de errores es muy típico y es el que más tiempo consume.

La falta de un símbolo, un ; o algo similar, son errores sintácticos, y son bastante fáciles de detectar. Cuando un programa no funciona como debería, puede llegar a ser bastante más complejo ya que no existe ningún mensaje de error asociado, ni una línea de código en específico que esté fallando.

En este capítulo aprenderemos un tipo de análisis que nos ayuda a descubrir ese tipo de problemas. Se llama análisis de condiciones de borde.

Cuando evaluamos los casos específicos alrededor de una condición estamos hablando de condiciones de borde. Estos son, precisamente, los puntos donde debemos tener más cuidado.

Analizando una condición de borde

Revisemos algunos problemas que hemos visto hasta ahora y analicemos sus condiciones de borde.

Por ejemplo, nuestro programa donde se pregunta si una persona es mayor de edad.

```
puts "¿Qué edad tienes?"
edad = gets.chomp.to_i

if edad >= 18
   puts "Eres mayor de edad"
else
   puts "Eres menor de edad"
end
```

En este ejemplo, el punto que define una rama de la otra es el 18. A este punto se le suele llamar punto crítico, mientras que los bordes son los números que "bordean" el 18, es decir, analizar los bordes corresponde a probar con los valores 17, 18 y 19.

En el siguiente algoritmo intenta analizar los bordes:

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

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

En este caso, los puntos críticos son 4 y 10, por lo que nuestros puntos a analizar deberían ser 3, 4, 5, 9, 10 y 11

Veamos un caso un poco distinto, en que el punto crítico no es un número exacto:

```
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
```

En este ejemplo, el caso crítico es cuando ambos valores son iguales, y los bordes son cuando el primer número es mayor que el segundo y cuando el primer número es menor que el segundo.

En este caso podemos escoger un par de números para hacer la prueba, digamos 2 y 2, y luego los bordes serían 3 y 2, y 1 y 2.