Logical Operators

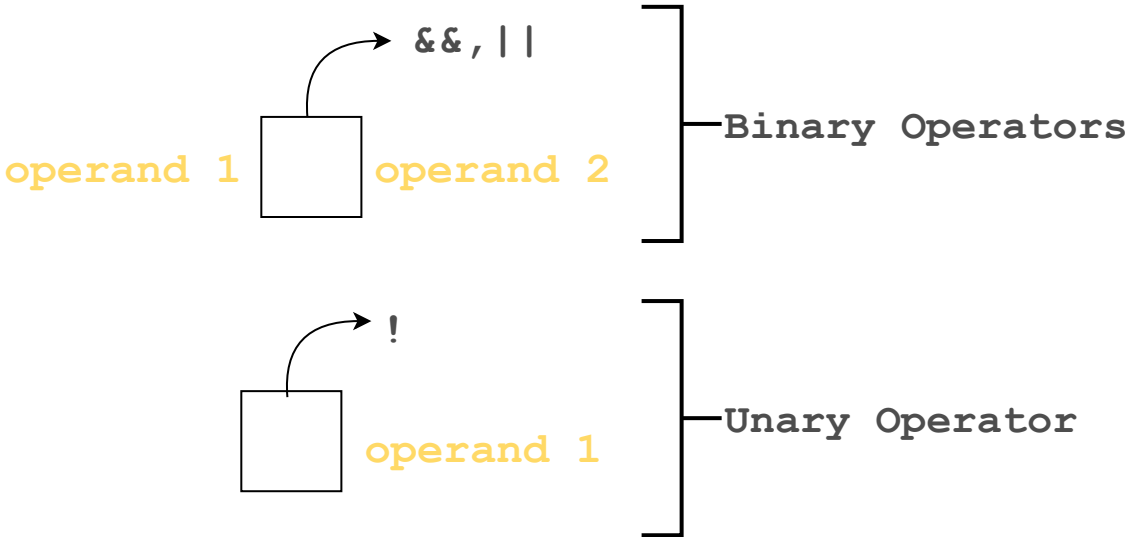
This lesson teaches logical operators in Rust.

We'll cover the following

- What Are Logical Operators?
  - Types
- Quiz

What Are Logical Operators?

Logical operators operate on true / false values.



Types

The following table summarizes the types and functions of the logical operators:

Operator	Operation	Explanation
operand 1 && operand 2	AND	Evaluates to true if operand 1 and operand 2 both evaluate to be true
operand 1 || operand 2	OR	Evaluates to true if operand 1 or operand 2 true evaluates to be true
! operand 1	NOT	Negates the value of a single operand

Logical Operators

> The logical `AND` and `OR` are known as **Lazy Boolean** expressions because the left-hand side operand of the operator is first evaluated. If it is false, there is no need to evaluate the right-hand side operand in case of AND. If it is true, there is no need to evaluate the right-hand side operand in case of OR.

The following example shows the use of logical operators in a program:

```
fn main() {
  let a = true;
  let b = false;
  println!("Operand 1:{}, Operand 2:{}", a , b);
  println!("AND:{}", a && b);
  println!("OR:{}", a || b);
  println!("NOT:{}", ! a);
}
```

Quiz

Test your understanding of logical operators in Rust!

Quick Quiz on Logical Operators!

Q

What is the output of the following code?

```
fn main() {
  let mut a = false;
  let mut b = true;
  a = a && b || ( ! a);
  b = !b;
  println!("a:{}", a);
  println!("b:{}", b);
}
```

Retake Quiz

In the next lesson, you'll learn about comparison operators in Rust.