Goal: Use the AND operator to combine two conditions and adjust your path if both are true.

The logical AND operator (&&) combines two Boolean conditions and runs your code only if *both* are true. For example, in the following code, `isBlocked` AND `isOnClosedSwitch` must both be true.

Example

```
if isBlocked && isOnClosedSwitch {
    toggleSwitch()
}
```

New condition!

The Boolean condition `isBlockedLeft` is **true** if you *can't* move 1 tile to the left and **false** if you can make that move.

1. Add an `if` statement in the `for` loop, then add a condition to check whether your character is on a gem.
2. In the shortcut bar, select `&&`, then add a second condition.
3. If your character is on a gem AND blocked on the left, turn right and toggle the switch. Otherwise, if on a gem, collect it.

```
    // Movement Functions & For Loop w/ if Statements
func program() {
    func turnAround() {
    turnLeft()
    turnLeft()
}
func stairs() {
    for i in 1 ... 2 {
        moveForward()
    }
}
for i in 1 ... 7 {
    moveForward()
    if isOnGem && isBlockedLeft {
        collectGem()
        turnRight()
        stairs()
```

```
        toggleSwitch()
        turnAround()
        stairs()
        turnRight()
    }
    else if isOnGem {
        collectGem()
    }
    else if isOnClosedSwitch {
        toggleSwitch()
    }
    else if  isOnOpenSwitch {
        turnAround()
        moveForward()
        moveForward()
        turnRight()
    }
}
}

// Solve Challenge
program()
```