Challenge: Implement the most efficient algorithm to collect the gems and activate the switches.

For the last challenge of Learn to Code 1, you'll test your algorithm design skills. There are many different algorithms you could use to solve the puzzle, and many different ways to structure your code.

If you're not able to find a solution right away, that's okay! Coding often requires trying different solutions to a problem until you find the one that works best. When you're ready, you can move on to Learn to Code 2.

```
func navigateMaze() {
    if isOnGem && isBlockedLeft && isBlocked {
        collectGem()
        turnRight()
        moveForward()
    }else if isOnGem && isBlockedLeft {
        collectGem()
        moveForward()
    }else if isBlockedRight && isBlocked {
        turnLeft()
    }else if isOnClosedSwitch && !isBlocked && isBlockedRight {
        toggleSwitch()
        moveForward()
    }else if isOnClosedSwitch && isBlocked {
        toggleSwitch()
        turnLeft()
        moveForward()
    }else if isOnClosedSwitch && !isBlocked {
        toggleSwitch()
        turnLeft()
        moveForward()
    }else if isBlocked && isBlockedLeft && !isBlockedRight {
        turnRight()
    }else {
        moveForward()
    }
```

```
}

while !isOnOpenSwitch {
    navigateMaze()
}


if isBlocked && isBlockedLeft && !isBlockedRight {
    turnRight()
}
```