# Converting a Biased Coin to an Unbiased Coin

## The Problem

We have a **biased coin** (called BIASED-RANDOM) that:

- Returns 1 with probability $p$
- Returns 0 with probability $1-p$
- We **don't know** what $p$ is, but we know $0 < p < 1$

**Goal**: Create an algorithm that returns 0 and 1 with **exactly equal probability** (50% each), using only this biased coin.

## The Key Insight

Even though individual coin flips are biased, we can find **pairs of outcomes** that have equal probability!

### The Magic of Opposite Pairs

Consider flipping the biased coin **twice**:

| First Flip | Second Flip | Probability |
|---|---|---|
| 0 | 0 | $(1-p) \times (1-p) = (1-p)^2$ |
| 0 | 1 | $(1-p) \times p = p(1-p)$ |
| 1 | 0 | $p \times (1-p) = p(1-p)$ |
| 1 | 1 | $p \times p = p^2$ |

**Notice**: $P(01) = P(10) = p(1-p)$

These two outcomes have **exactly the same probability**, regardless of what $p$ is!

## The Algorithm Explained

```
for all eternity do
    a = BiasedRandom()
    b = BiasedRandom()
```

```
  if a > b then
    return 1
  end if
  if a < b then
    return 0
  end if
end for
```

### Step-by-Step Breakdown

1. **Get two random bits**: Call the biased function twice to get `a` and `b`
2. **Check for opposite outcomes**:
   - If `a = 1` and `b = 0` (i.e., `a > b`): Return 1
   - If `a = 0` and `b = 1` (i.e., `a < b`): Return 0
   - If `a = b` (both 0 or both 1): Try again
3. **Why this works**: We only return a result when we get the two equally-likely opposite patterns (01 or 10)

## Why This Algorithm is Unbiased

### Probability Analysis

When we return a result, it's either:

- **Return 1**: When we see pattern (1,0) with probability p(1-p)
- **Return 0**: When we see pattern (0,1) with probability p(1-p)

**Key insight**: Both outcomes have the same probability!

Therefore:

```
P(return 1 | we return something) = p(1-p) / [p(1-p) + p(1-p)] = 1/2
P(return 0 | we return something) = p(1-p) / [p(1-p) + p(1-p)] = 1/2
```

### The Math Behind the Solution

The solution shows: `p(1-p) = 1/2`

Wait, that's not right! Let me correct this:

The actual reasoning is:

- We only return when we get outcomes (1,0) or (0,1)
- P(1,0) = p(1-p)
- P(0,1) = (1-p)p = p(1-p)
- Since these are equal, when we condition on getting one of these two outcomes, each has probability 1/2

## Examples with Different Bias Levels

### Example 1: Slightly Biased (p = 0.6)

- P(1,0) = 0.6 × 0.4 = 0.24
- P(0,1) = 0.4 × 0.6 = 0.24
- P(0,0) = 0.4 × 0.4 = 0.16
- P(1,1) = 0.6 × 0.6 = 0.36

We ignore the (0,0) and (1,1) cases and only use the equal-probability (1,0) and (0,1) cases.

### Example 2: Heavily Biased (p = 0.9)

- P(1,0) = 0.9 × 0.1 = 0.09
- P(0,1) = 0.1 × 0.9 = 0.09
- P(0,0) = 0.1 × 0.1 = 0.01
- P(1,1) = 0.9 × 0.9 = 0.81

Even with heavy bias, the two opposite outcomes still have equal probability!

## Why the "Forever" Loop?

The loop runs indefinitely because:

- We might get (0,0) or (1,1), which we ignore
- The more biased the coin, the more often we get these "unusable" outcomes

- But eventually we'll always get a usable (0,1) or (1,0) pair

**Expected number of iterations**: 1 / [2p(1-p)]

- Most iterations when p = 0.5 (unbiased): 1 iteration on average
- Fewest iterations when p is close to 0 or 1: many iterations needed

## The Beautiful Result

This algorithm **perfectly converts any biased coin into an unbiased one**, using only the biased coin itself. No matter how biased the original coin is, the output will be perfectly fair!

This is a classic result in probability theory, often called the "von Neumann technique" for bias elimination.