Pseudorandom Generators

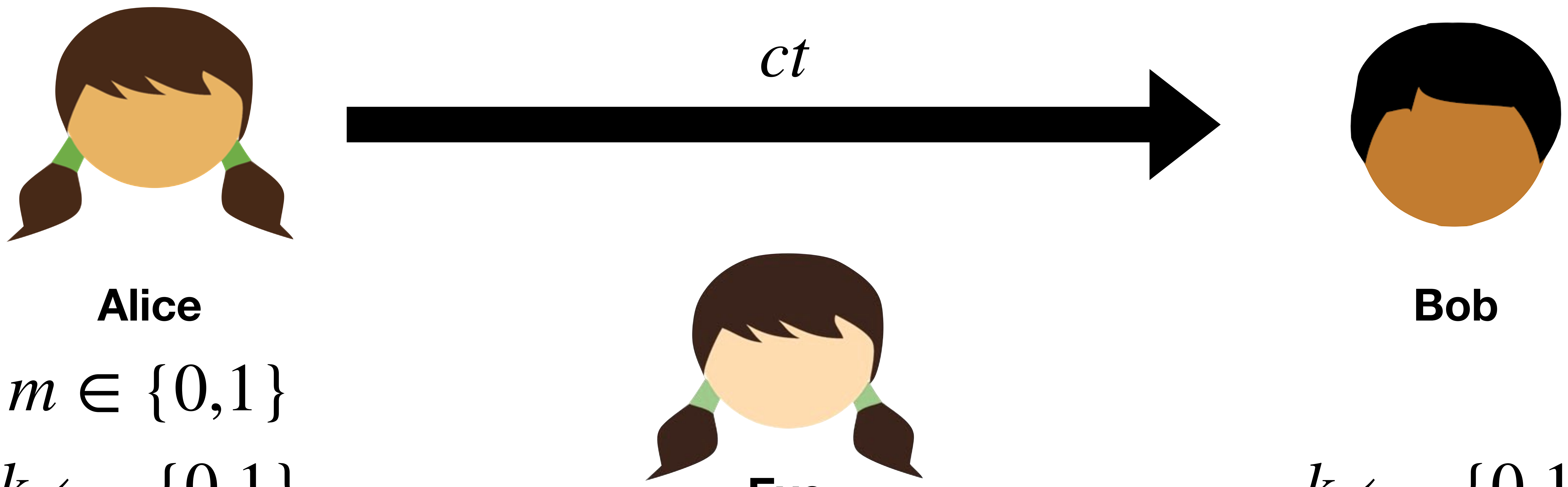
CS/ECE 407

Today's objectives

Describe pseudorandomness/pseudorandom generators

Define negligible functions

Understand security of PRGs



ct

Alice

$m \in \{0,1\}$

$k \leftarrow_{\$} \{0,1\}$

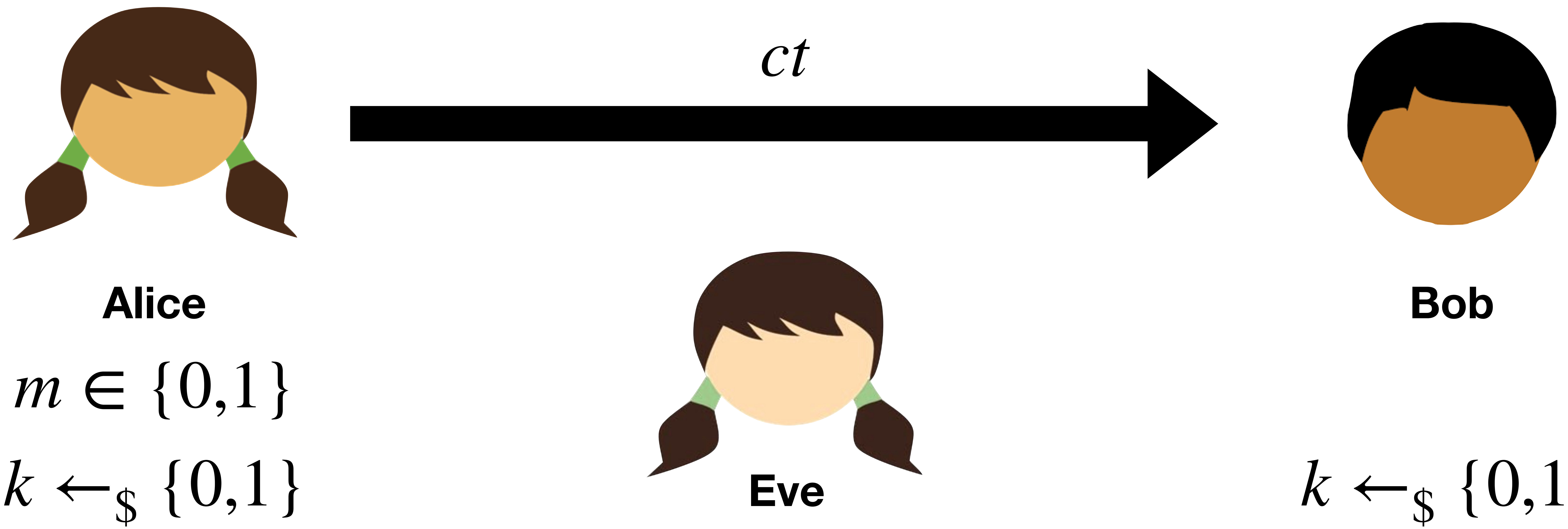
$ct \leftarrow m \oplus k$

Eve

Bob

$k \leftarrow_{\$} \{0,1\}$

$m' \leftarrow ct \oplus k$



ct

Alice

$m \in \{0,1\}$

$k \leftarrow_{\$} \{0,1\}$

$ct \leftarrow m \oplus k$

Eve

Bob

$k \leftarrow_{\$} \{0,1\}$

$m' \leftarrow ct \oplus k$

Question: *what if Alice wants to send more than one bit?*

Perfect Secrecy:

For every pair of messages $m_0, m_1 \in \mathrm{M}$ and every cipher text $c \in \mathrm{C}$:

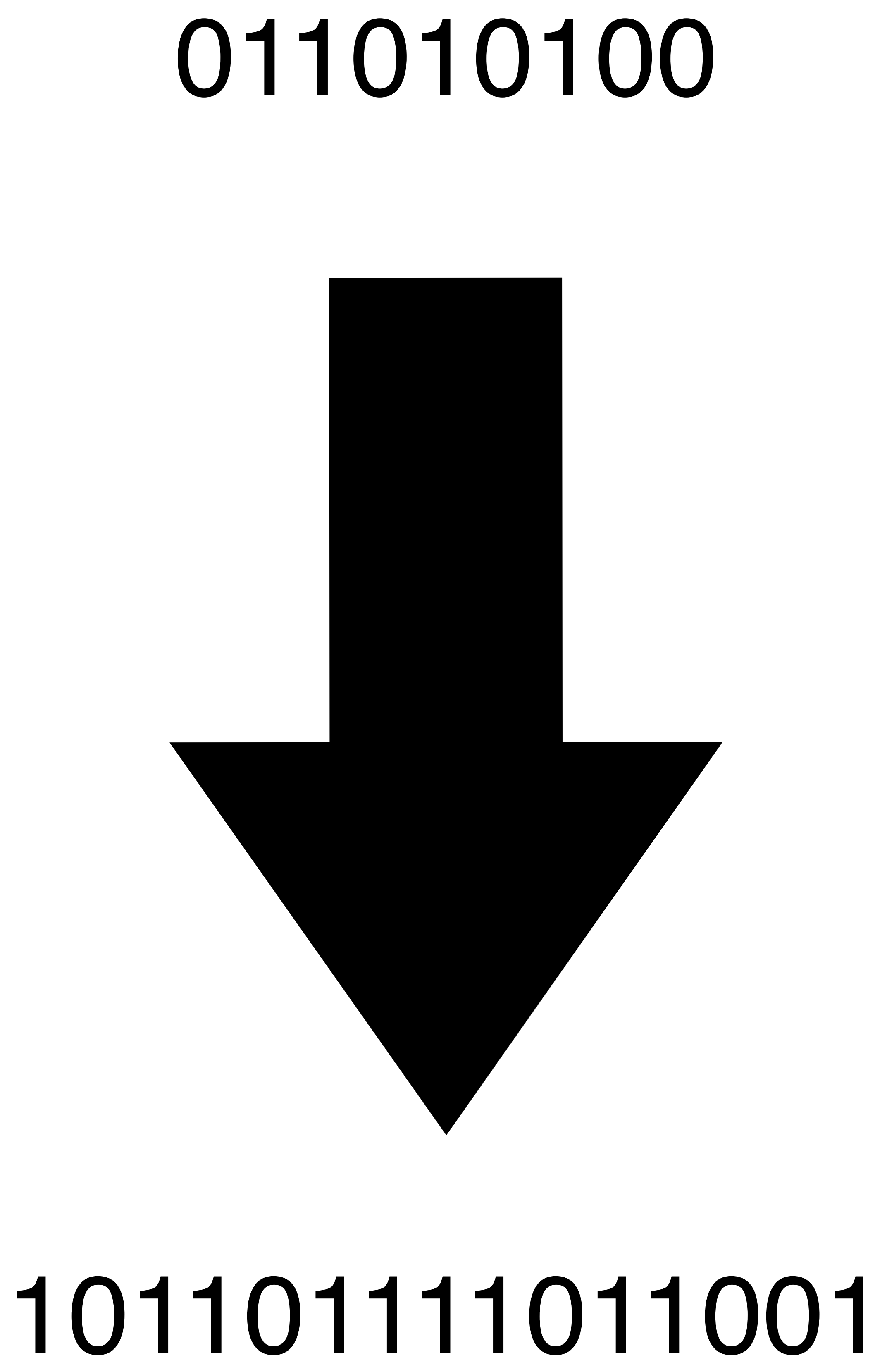
$$\Pr_{k \leftarrow \mathrm{K}}[\ Enc(k, m_0) = c\ ] = \Pr_{k \leftarrow \mathrm{K}}[\ Enc(k, m_1) = c\ ]$$

Theorem [Shannon 1949]: *Any cipher achieving perfect secrecy requires that* $|\mathrm{K}| \geq |\mathrm{M}|$*.*

“If we want to encrypt more stuff, we need more randomness”

Theorem [Shannon 1949]: *Any cipher achieving perfect secrecy requires that* $|\mathrm{K}| \geq |\mathrm{M}|$.

“If we want to encrypt more stuff, we need more randomness”



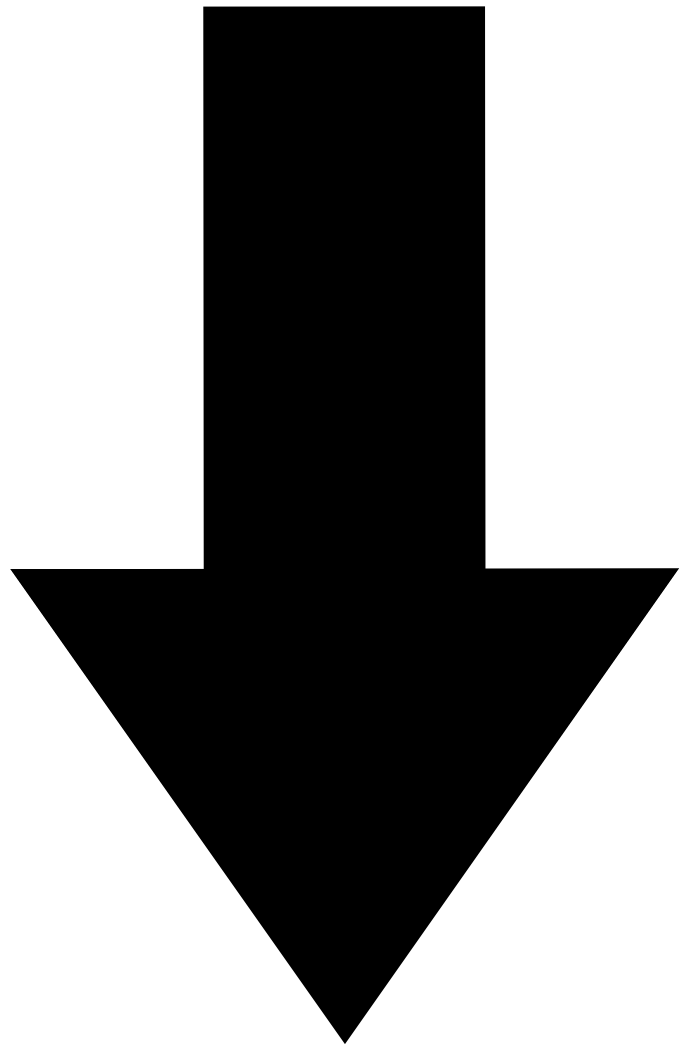
011010100

101101111011001

Q: Can we turn a short random string into a long random string?

“If we want to encrypt more stuff, we need more randomness”

011010100

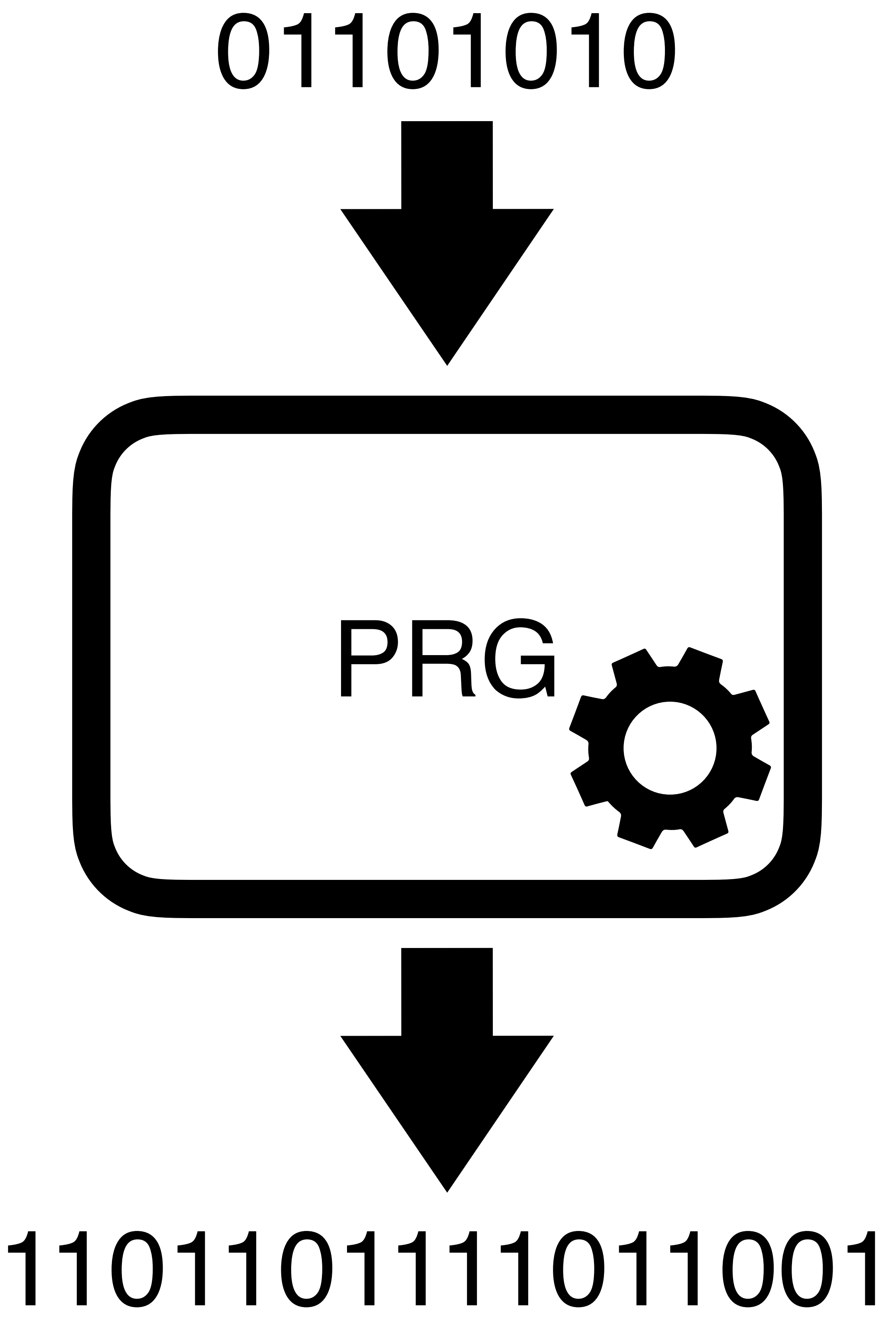


101101111011001

Q: Can we turn a short random string into a long random string?

A: No, this is provably impossible

"If we want to encrypt more stuff, we need more randomness"



Q: Can we turn a short random string into a long random string?

A: No, this is impossible

Q: Can we turn a short random string into a long string that <u>looks</u> random?

A: Yes! Use a pseudorandom generator!

Pseudorandom Generator (PRG)

A PRG is a function $G : \{0,1\}^n \rightarrow \{0,1\}^{n+s}$

Pseudorandom Generator (PRG)

A PRG is a function $G : \{0,1\}^n \to \{0,1\}^{n+s}$

Security?

Pseudorandom Generator (PRG)

A PRG is a function $G : \{0,1\}^n \to \{0,1\}^{n+s}$

Security?

Informal: *"no program can tell the difference between the output of* G *and truly random strings"*

Hardness as a basis for cryptography

Security?

Informal: *"no program can tell the difference between the output of* G *and truly random strings"*

Modern Cryptography

State assumptions

Define security

Design system

Prove: if assumption holds, system meets definition

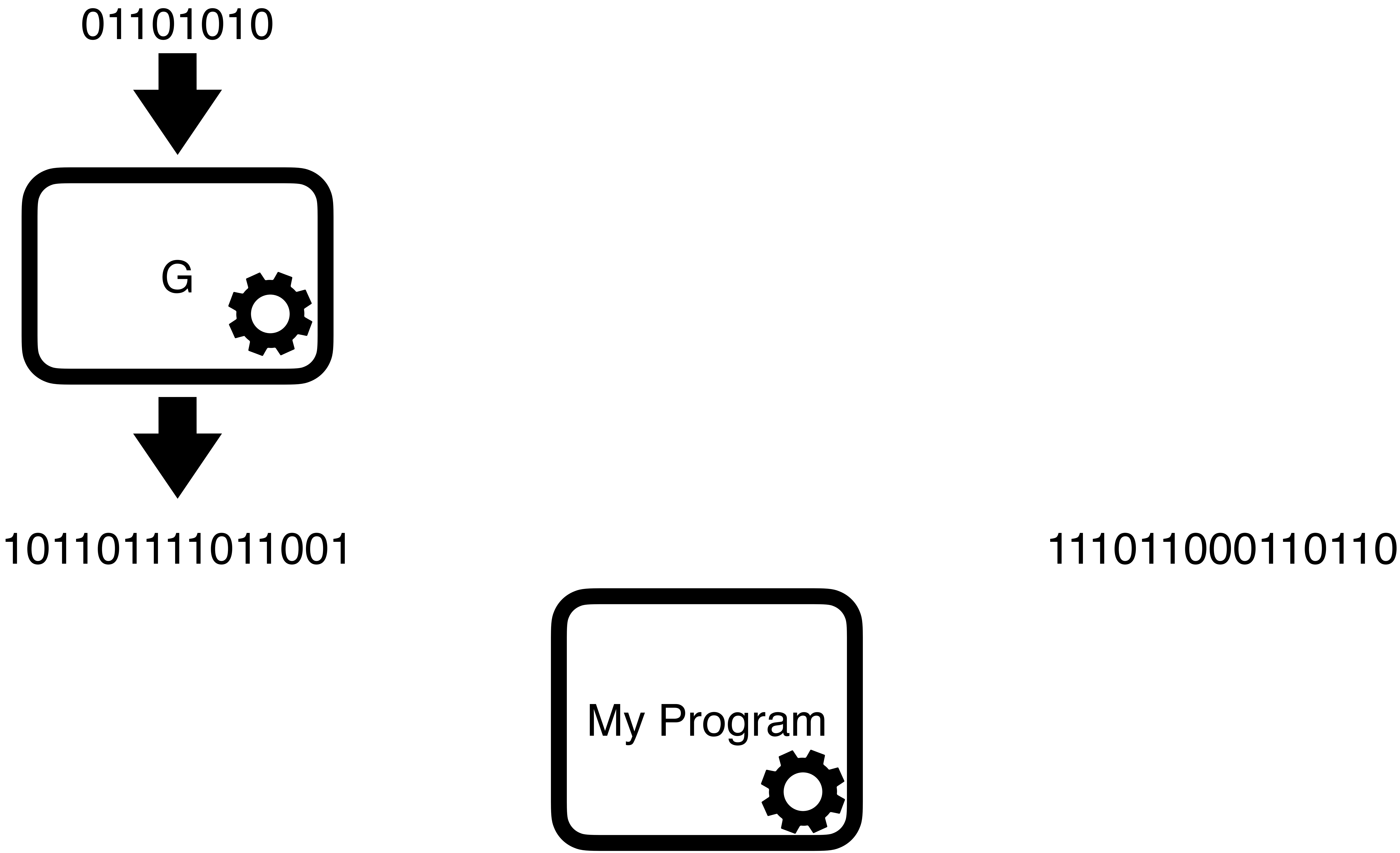
Modern Cryptography

State assumptions **PRGs exist**

Define security

Design system

Prove: if assumption holds, system meets definition



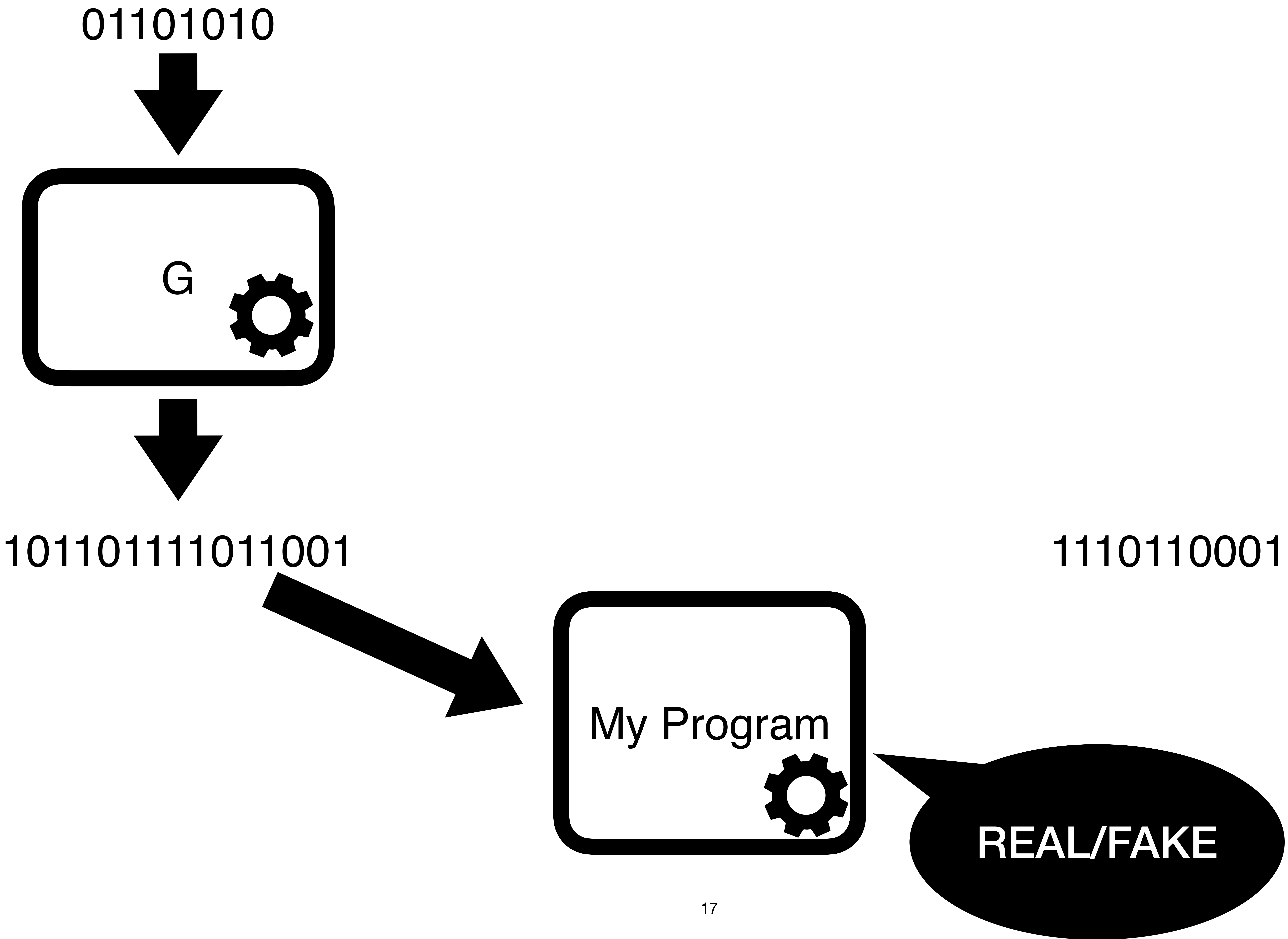
01101010

G

101101111011001

111011000110110

My Program



01101010

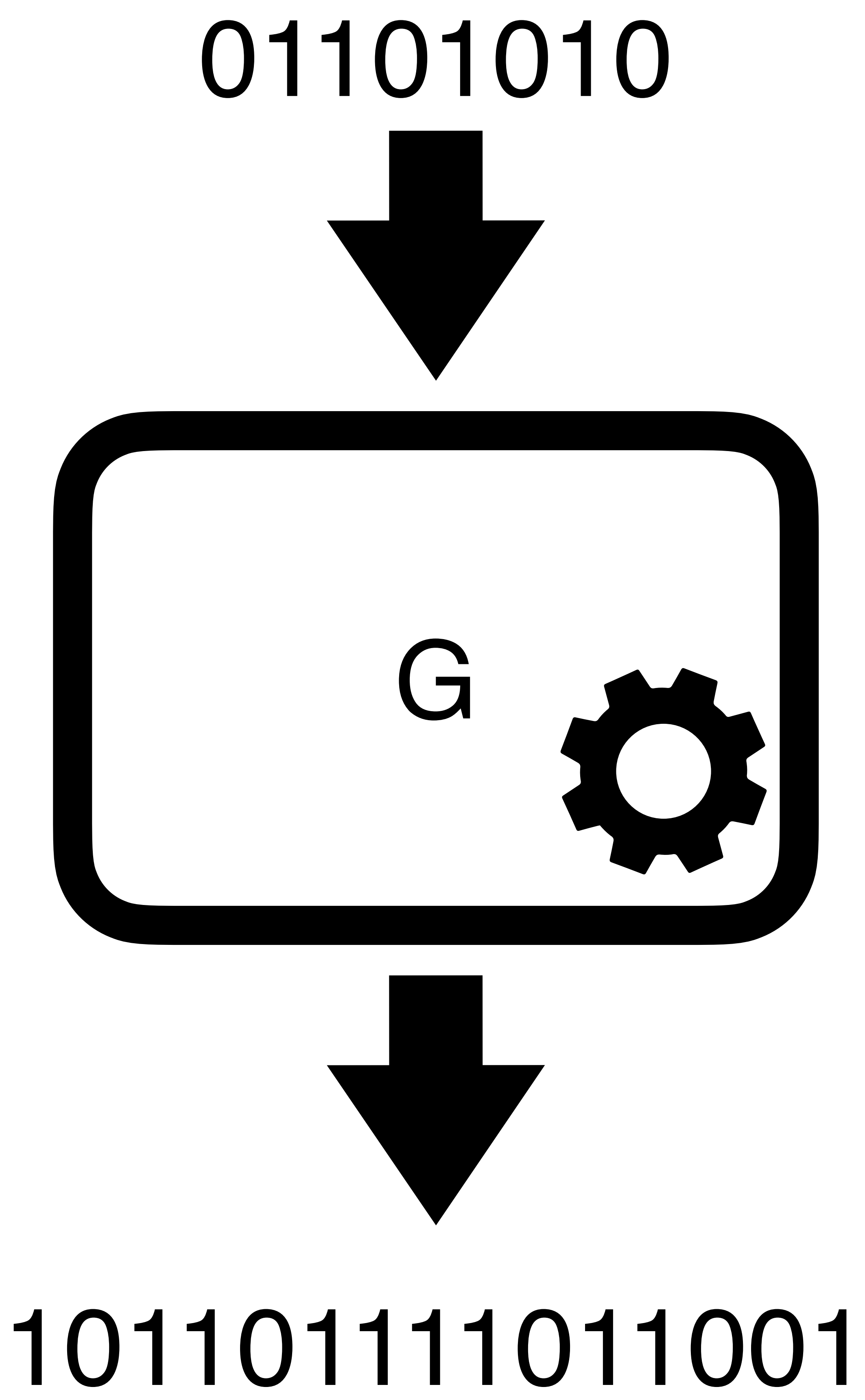
G

101101111011001

111011000110110

My Program

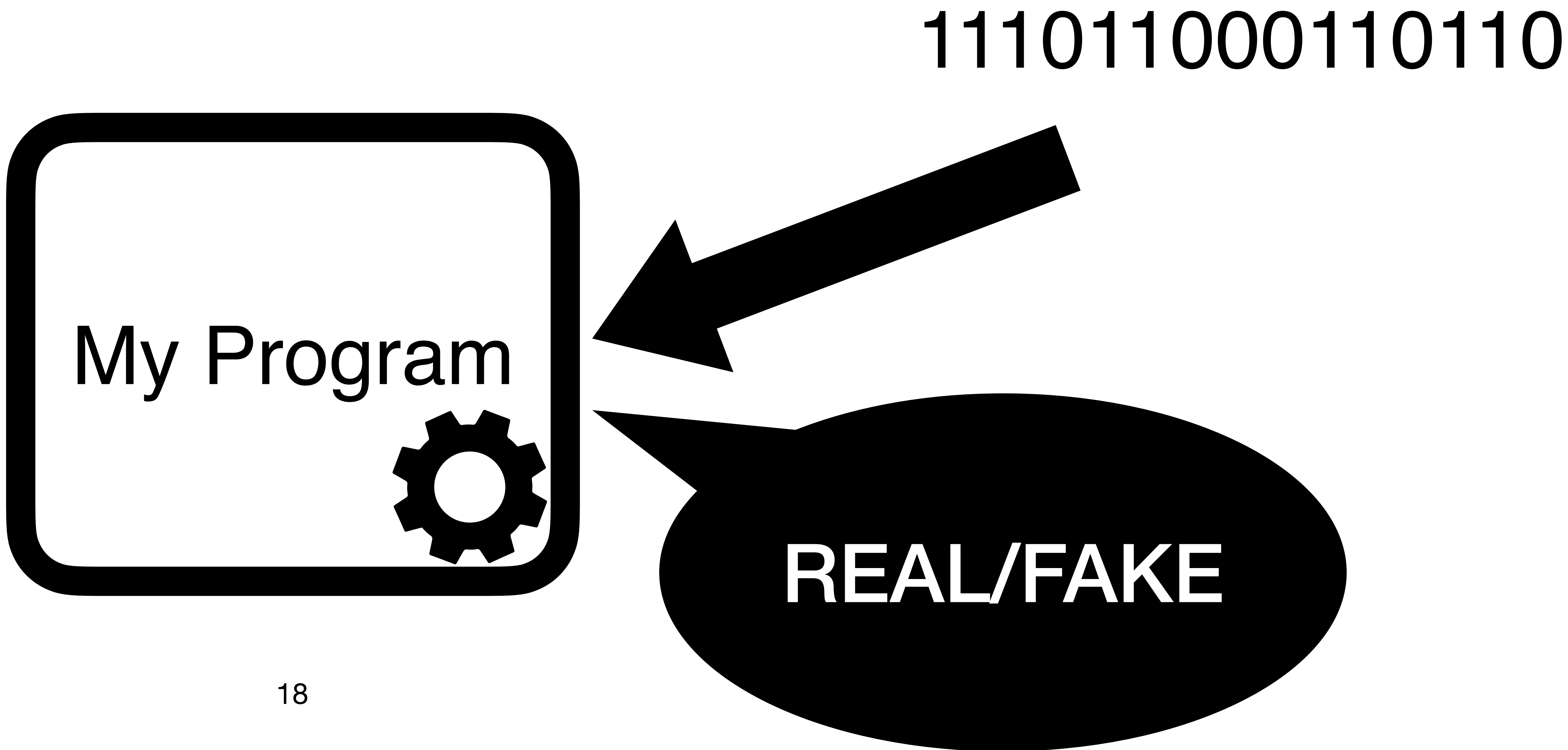
REAL/FAKE



01101010

G

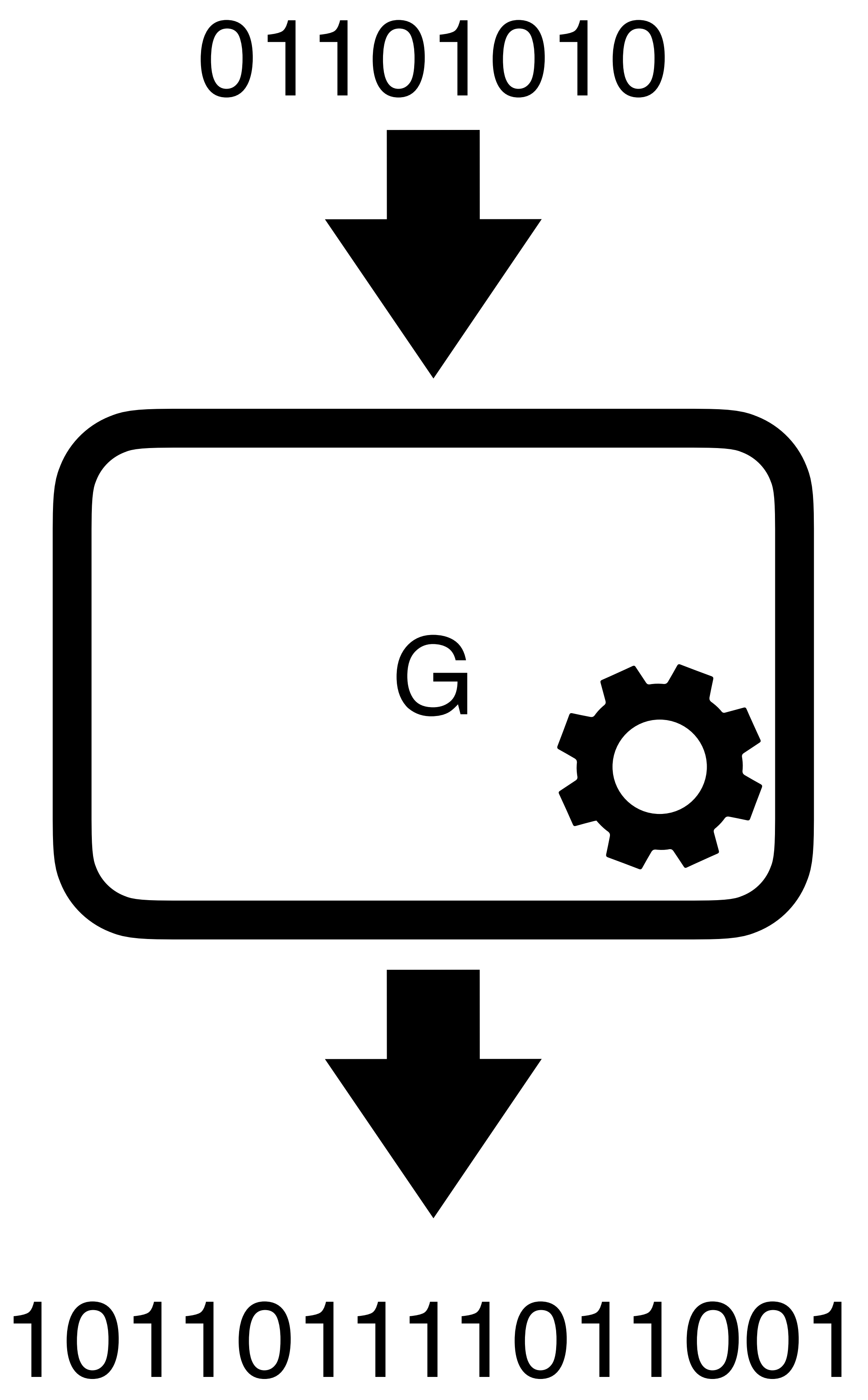
101101111011001



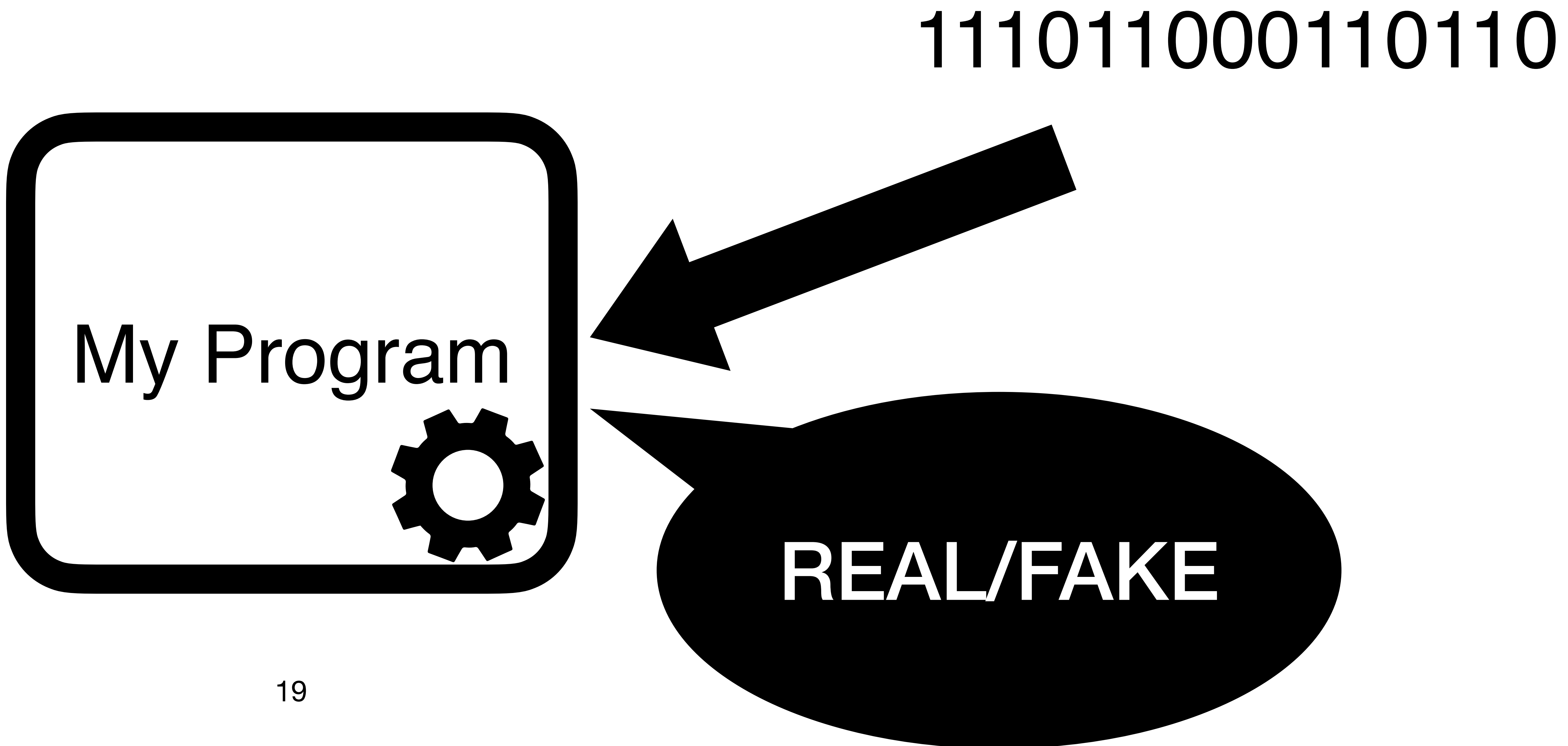
111011000110110

My Program

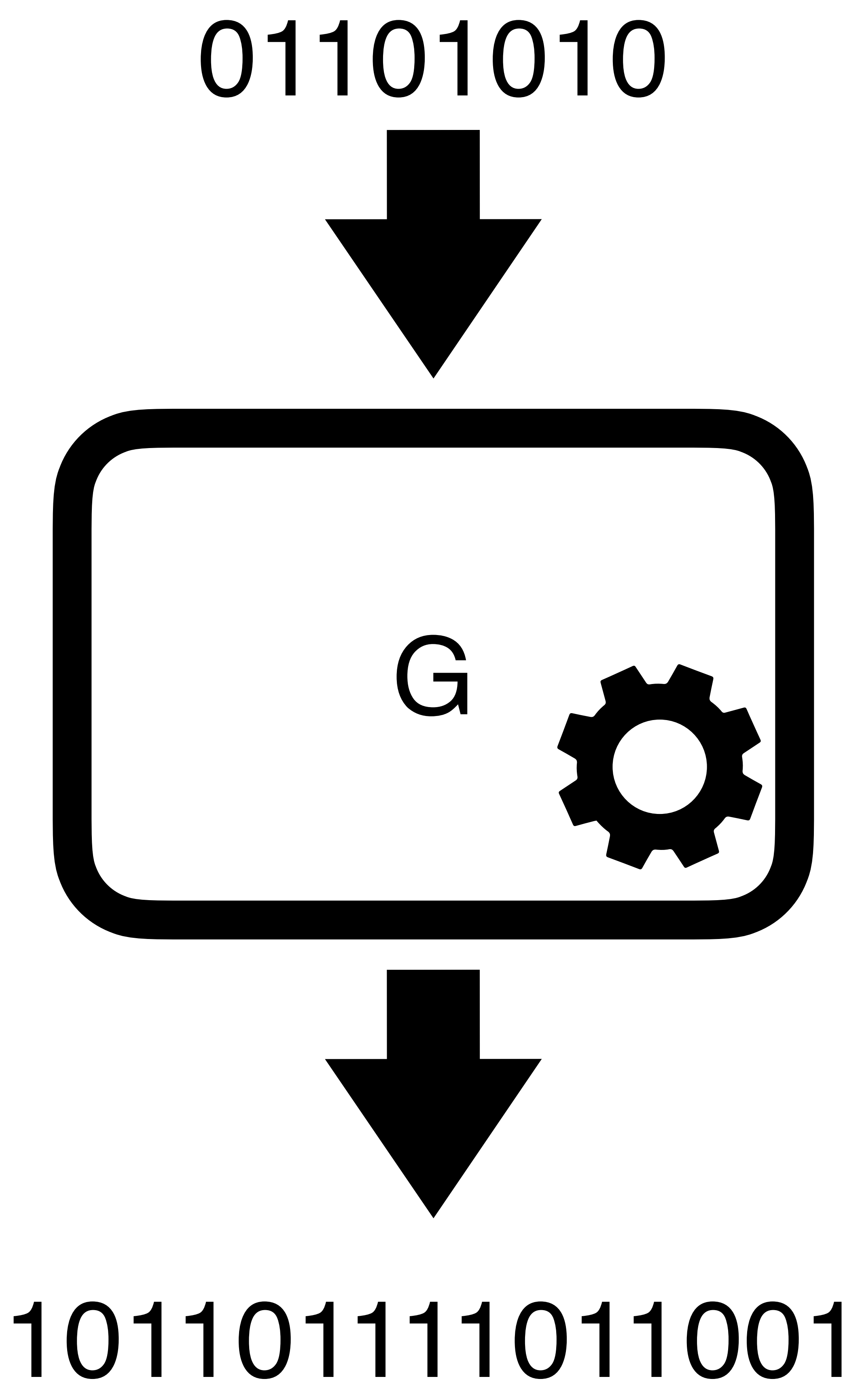
REAL/FAKE



G is a PRG if *no* program can reliably win this game



G is a PRG if *no* program can reliably win this game

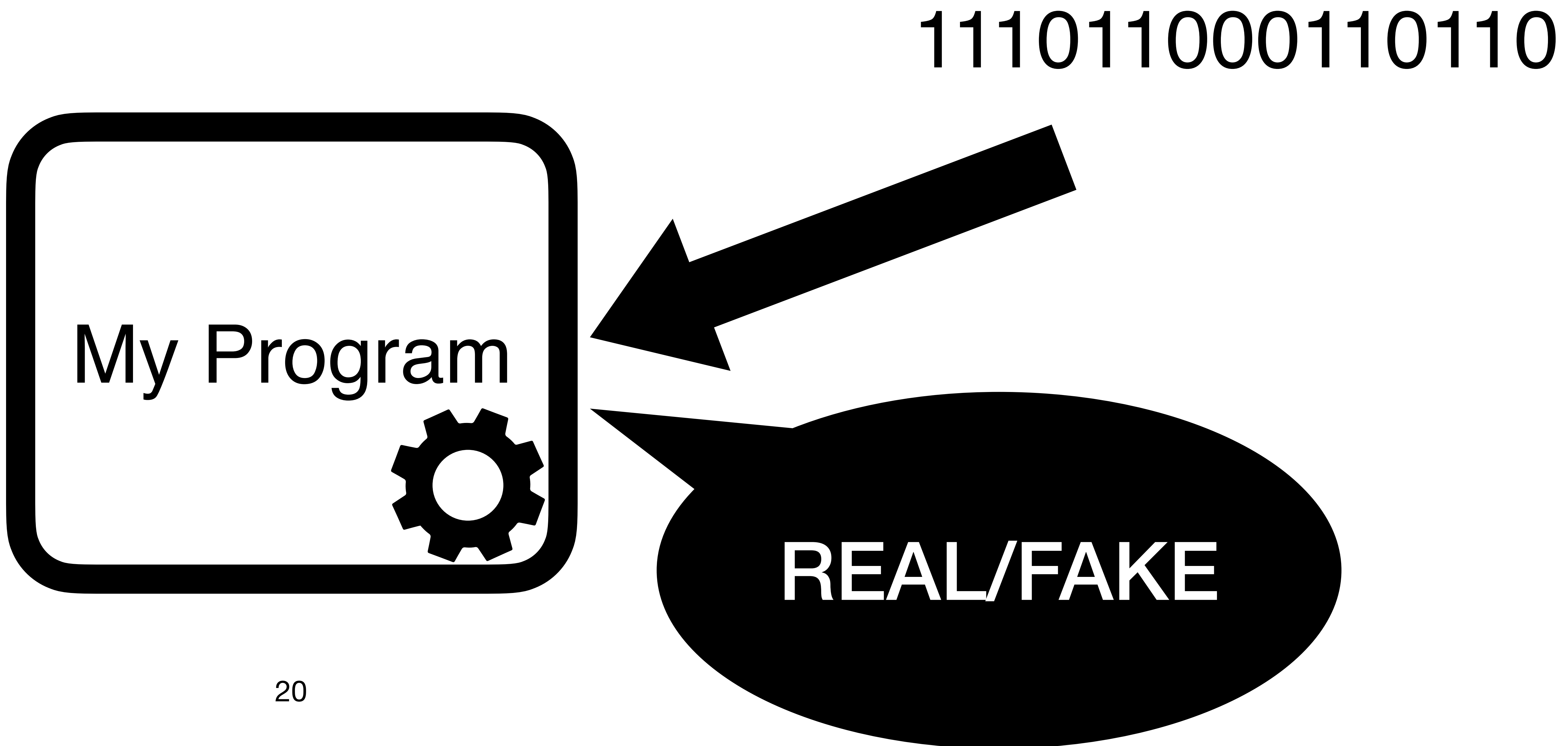


01101010

G

101101111011001

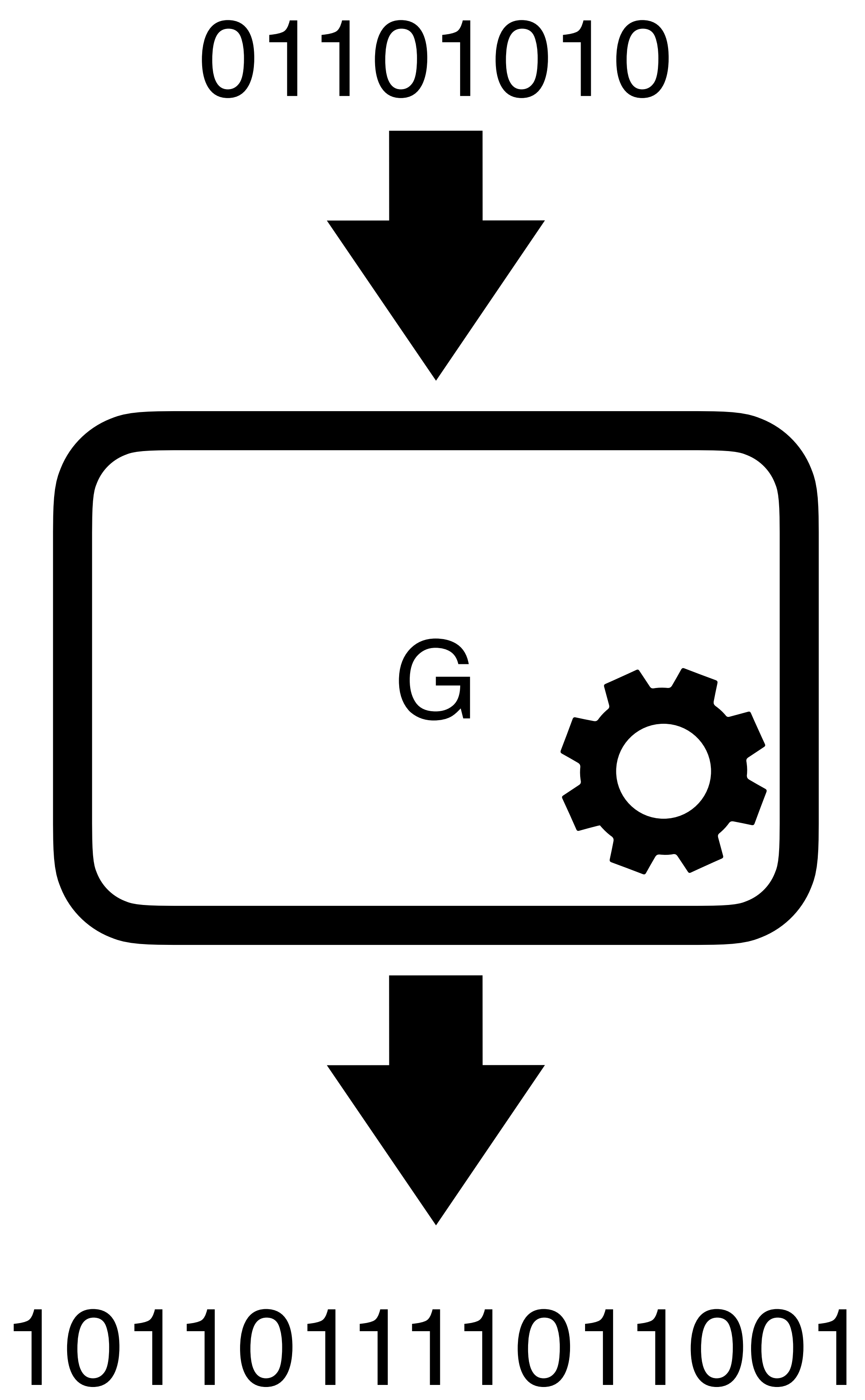
We believe that PRGs exist



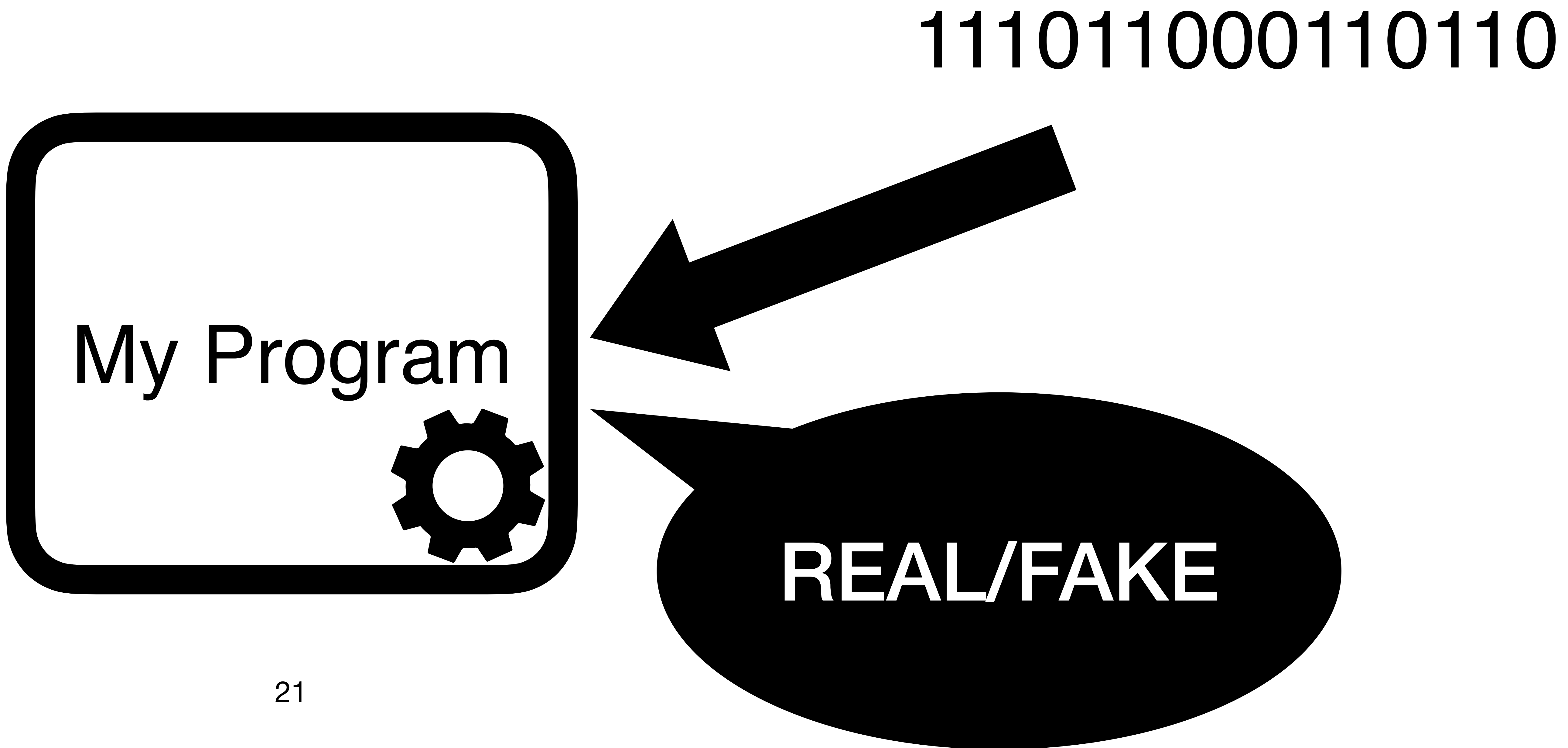
11101100110110

My Program

REAL/FAKE



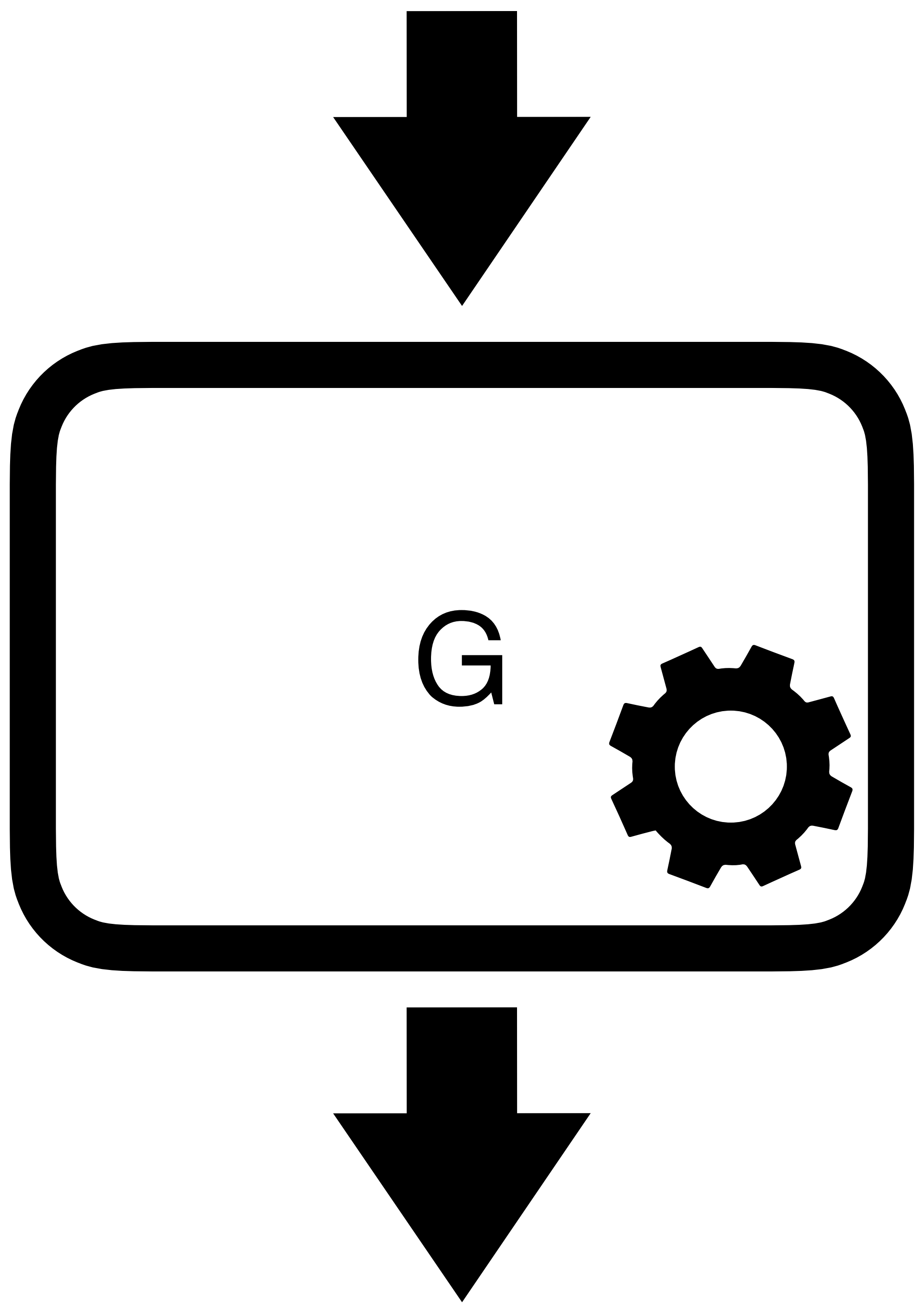
G is a PRG if *no* program can reliably win this game



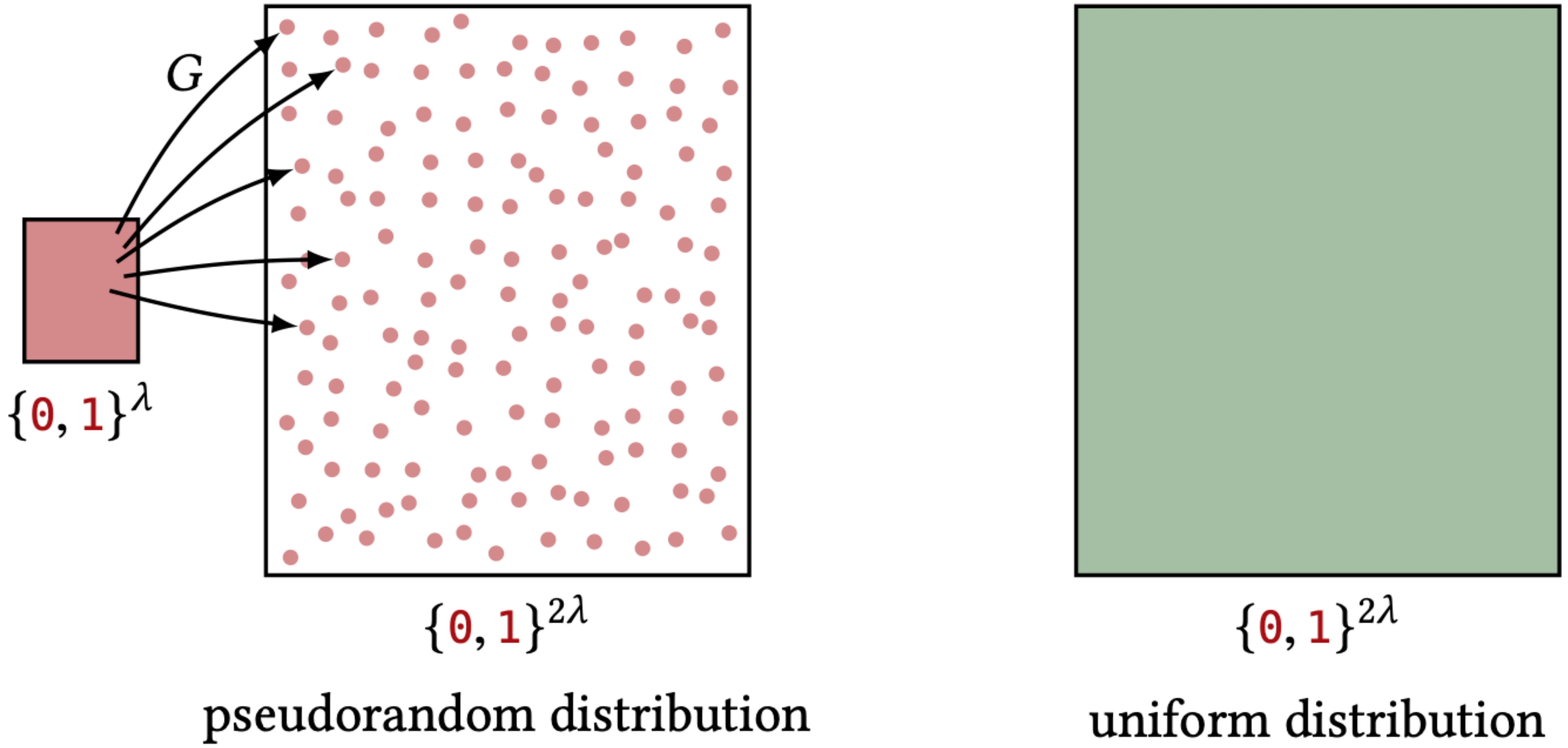
We believe that PRGs exist

If they do, $P \neq NP$

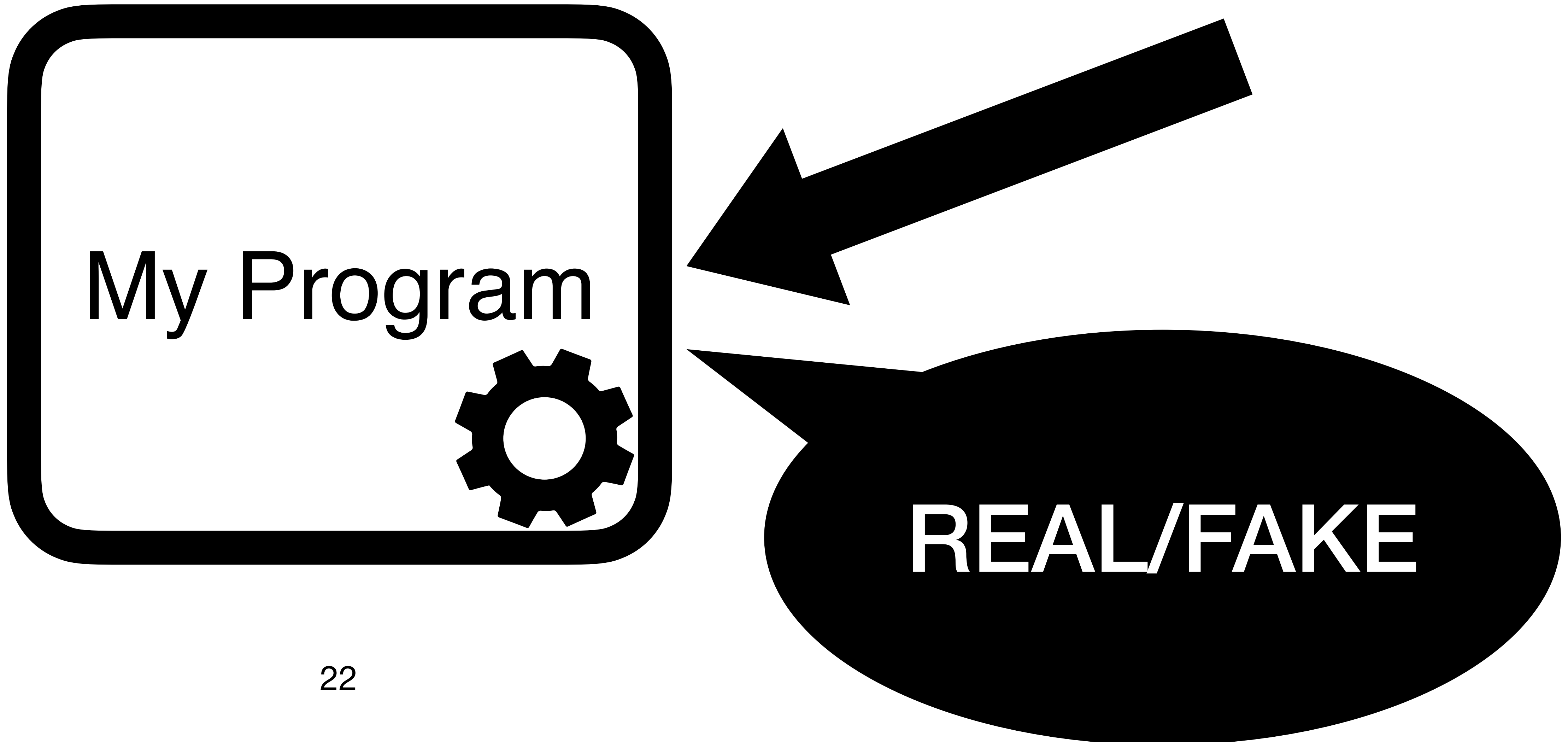
01101010



101101111011001



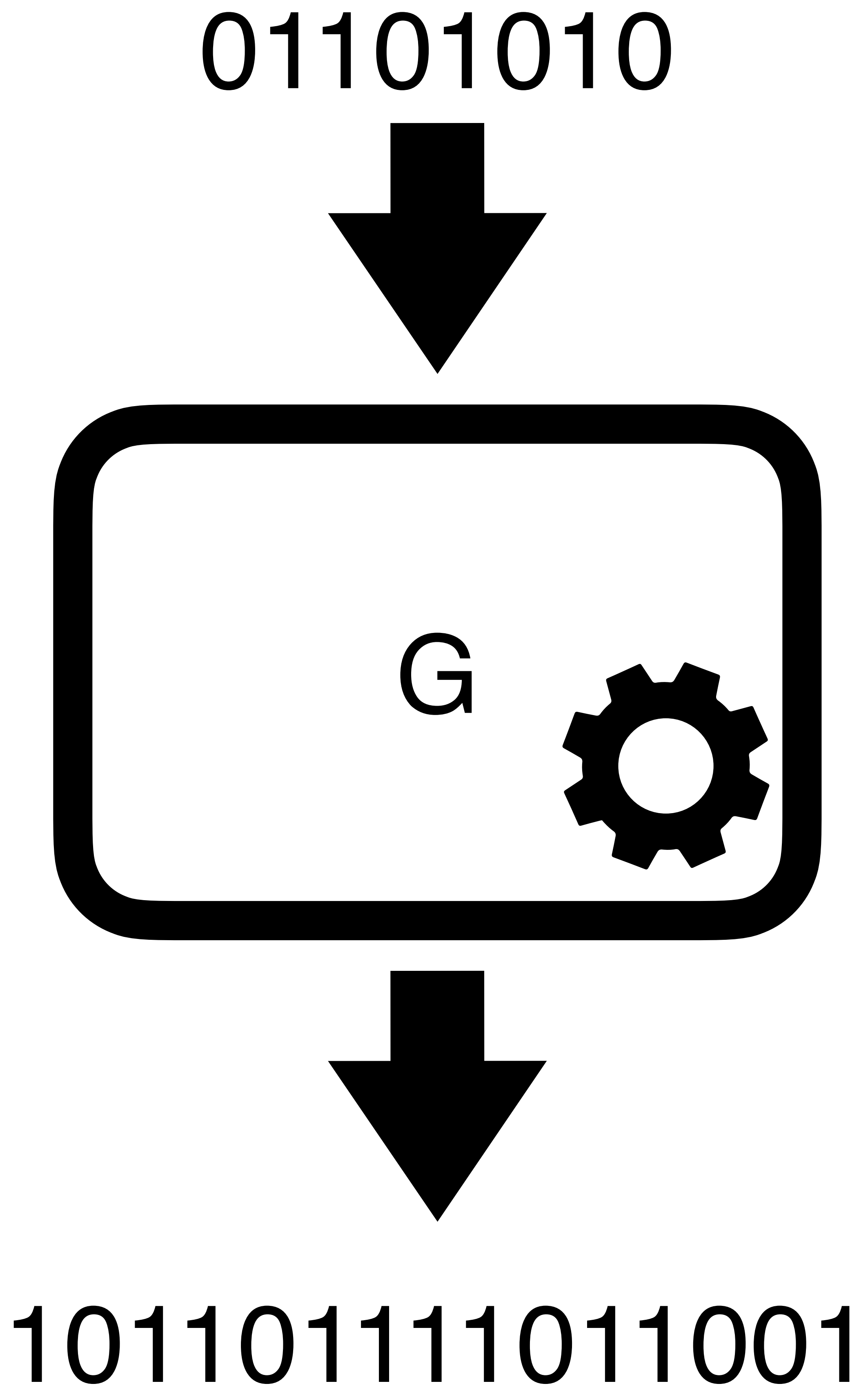
111011000110110

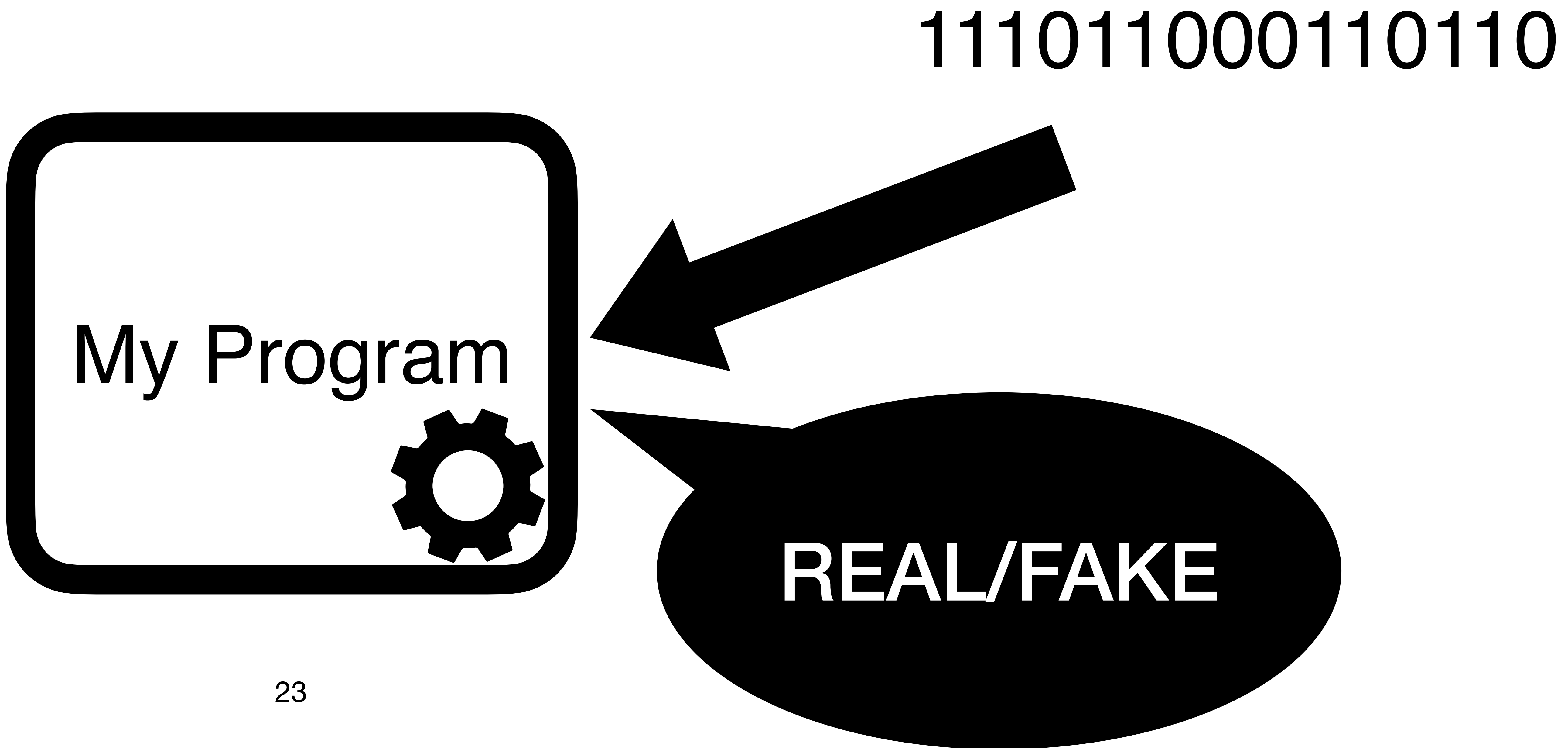


We believe that PRGs exist

If they do, $P \neq NP$

Goal: Make this more precise





We believe that PRGs exist

If they do, $P \neq NP$

Negligible Function

A function μ is **negligible** *if for any positive polynomial p there exists an N such that for all $n > N$:*

$$\mu(n) < \frac{1}{p(n)}$$

“μ approaches zero really fast”

PRG security

Game 0

```
seed ←$ $\{0,1\}^n$
y := G(seed)
b := A(y)
```

Game 1

```
y ←$ $\{0,1\}^{n+s}$
b := A(y)
```

PRG security

Game 0

```
seed ←$ $\{0,1\}^n$
y := G(seed)
b := A(y)
```

Game 1

```
y ←$ $\{0,1\}^{n+s}$
b := A(y)
```

For *any* PPT algorithm A outputting a bit, the following quantity is **negligible** (in n):

$$|\Pr[\ b = 1 \mid \text{Game } 0\ ] - \Pr[\ b = 1 \mid \text{Game } 1\ ]|$$

PRG security

```
b ←$ $\{0,1\}$
if b = 0:
    seed ←$ $\{0,1\}^n$
    y := G(seed)
else
    y ←$ $\{0,1\}^{n+s}$
b’ := A(y)
```

For *any* PPT program A outputting a bit, the following quantity is **negligible** (in n):

$$\left| \Pr\left[\ b = b'\ \right] - \frac{1}{2} \right|$$

PRG security

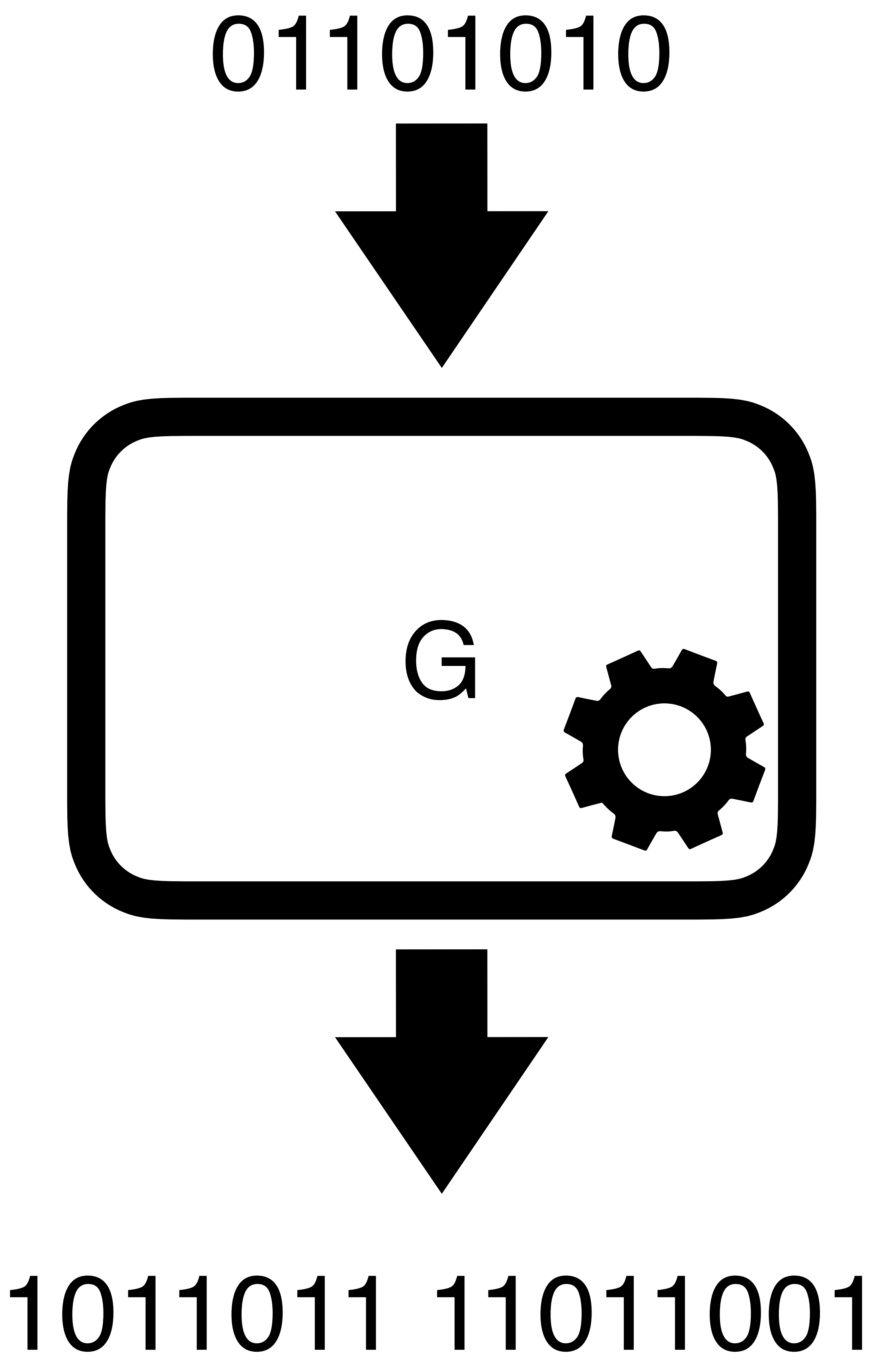
```
b ←$ $\{0,1\}$
if b = 0:
    seed ←$ $\{0,1\}^n$
    y := G(seed)
else
    y ←$ $\{0,1\}^{n+s}$
b' := A(y)
```

For *any* PPT program A outputting a bit, the following quantity is **negligible** (in n):

$$\left| \Pr\left[\ b = b'\ \right] - \frac{1}{2} \right|$$

In other words, the best possible strategy is only negligibly better than simply guessing

Stretching the output of a PRG

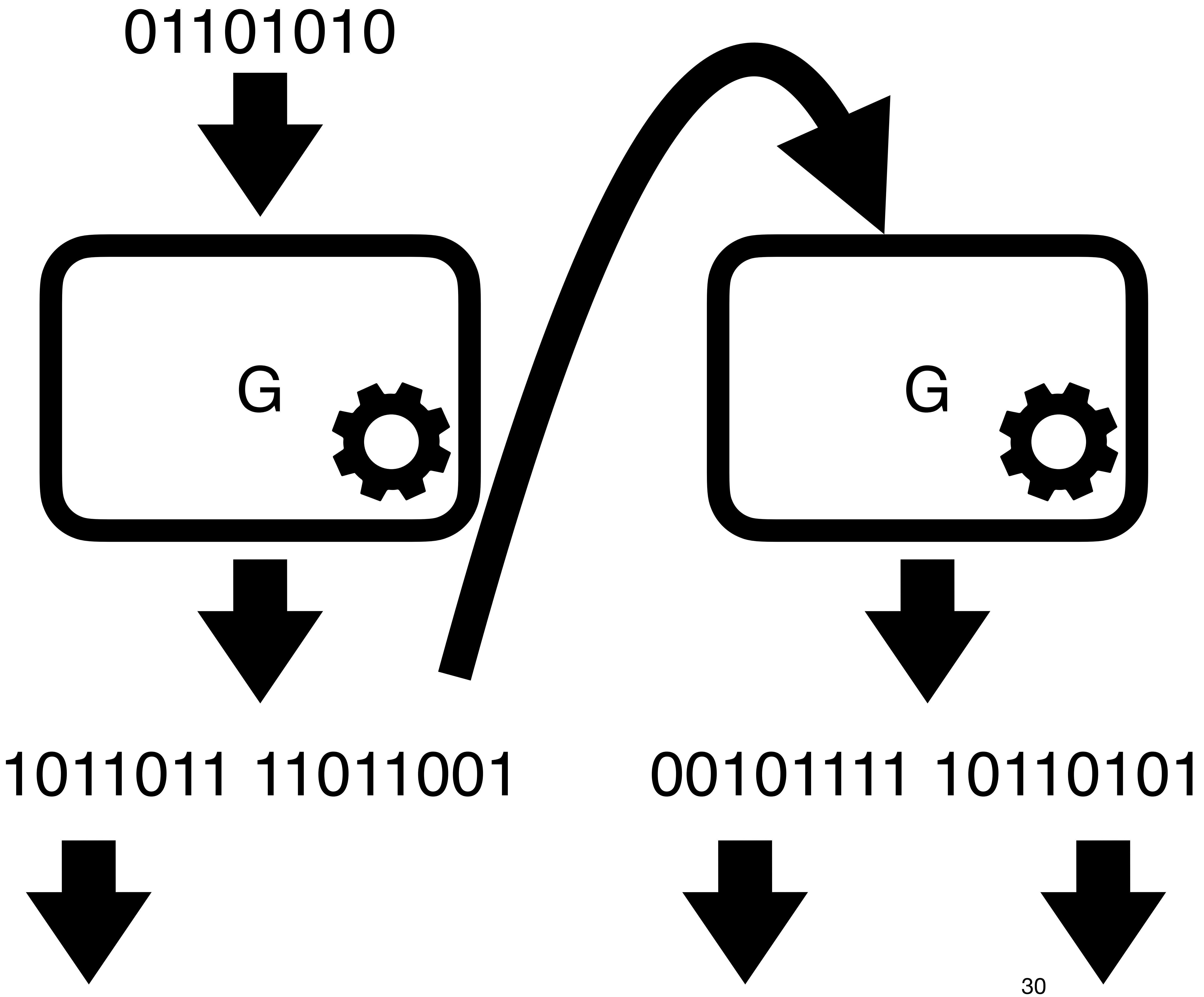


01101010

G

1011011 11011001

Stretching the output of a PRG



01101010

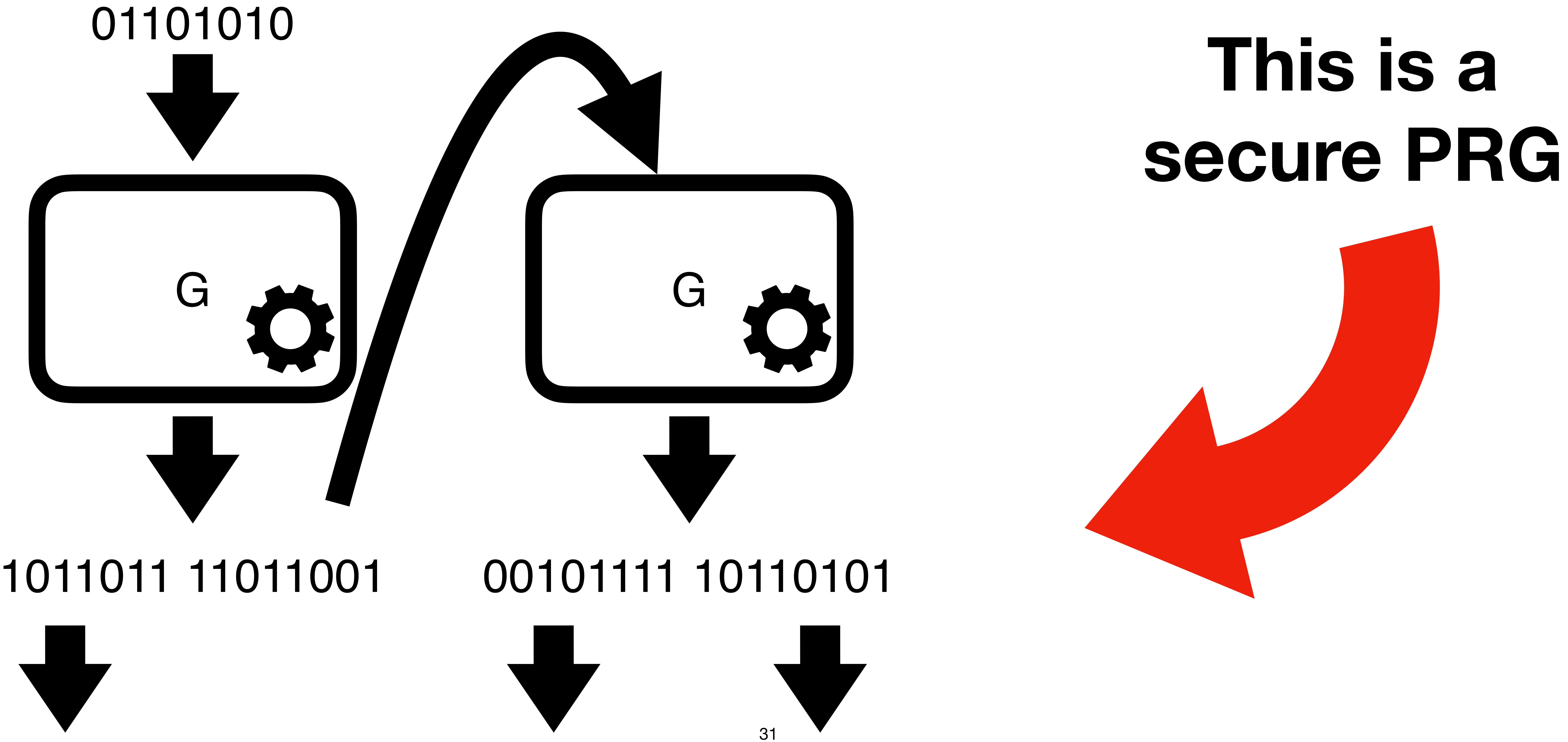
G

G

1011011 11011001

00101111 10110101

Stretching the output of a PRG



01101010

G

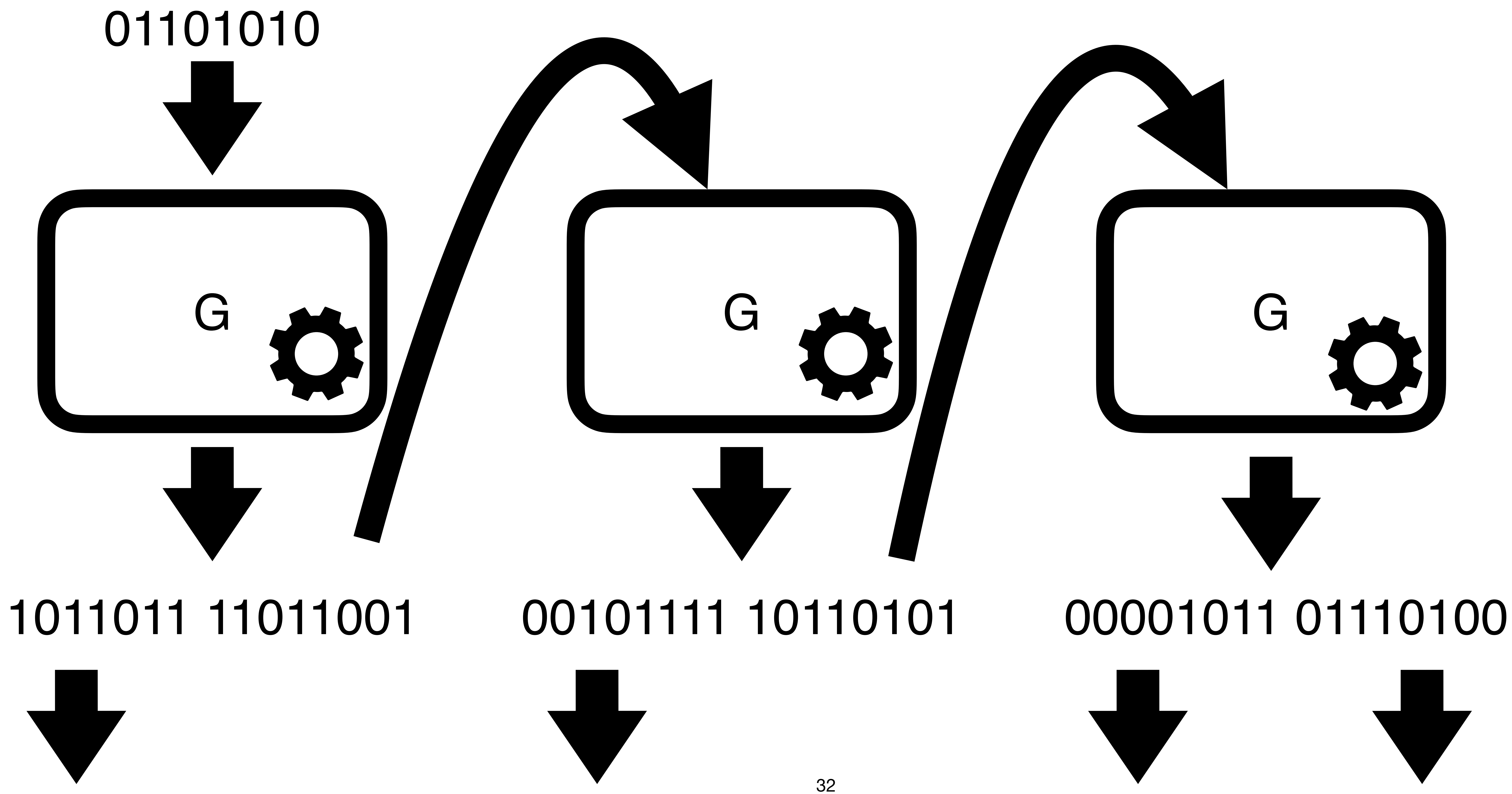
1011011 11011001

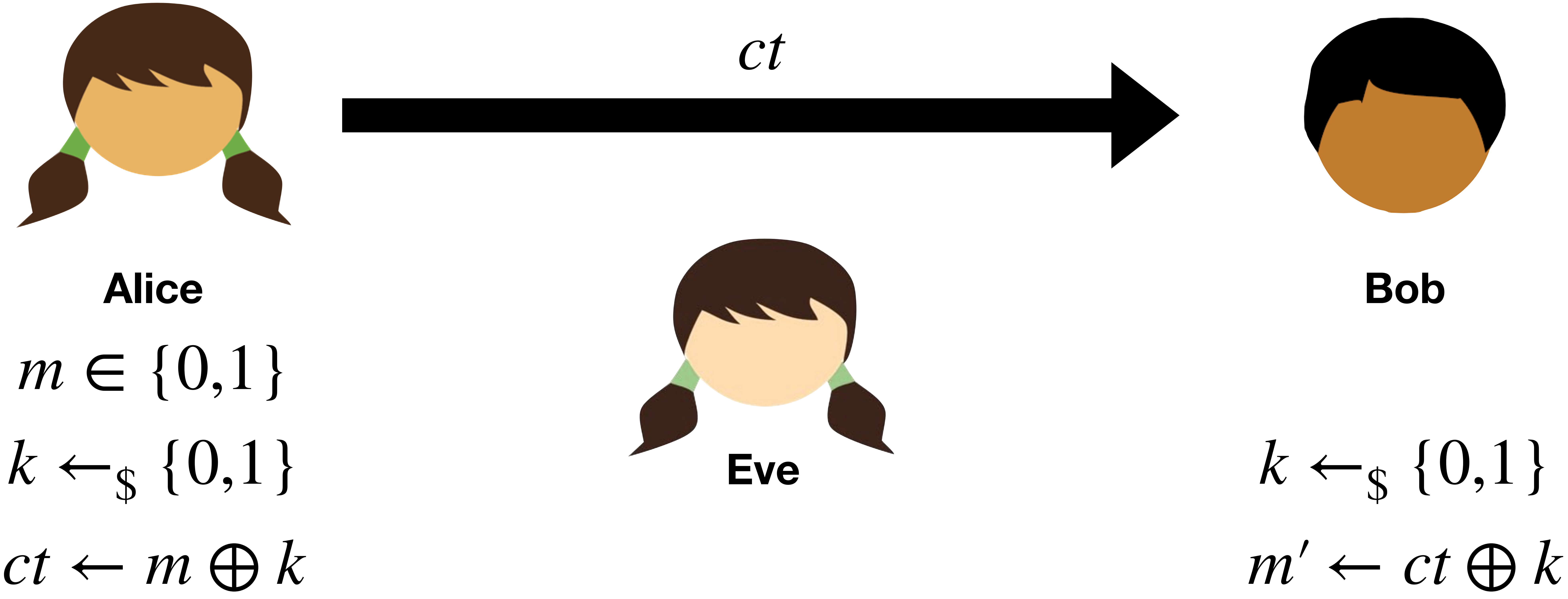
G

00101111 10110101

This is a secure PRG

Repeatable any polynomial number of times





ct

Alice

$m \in \{0,1\}$

$k \leftarrow_{\$} \{0,1\}$

$ct \leftarrow m \oplus k$

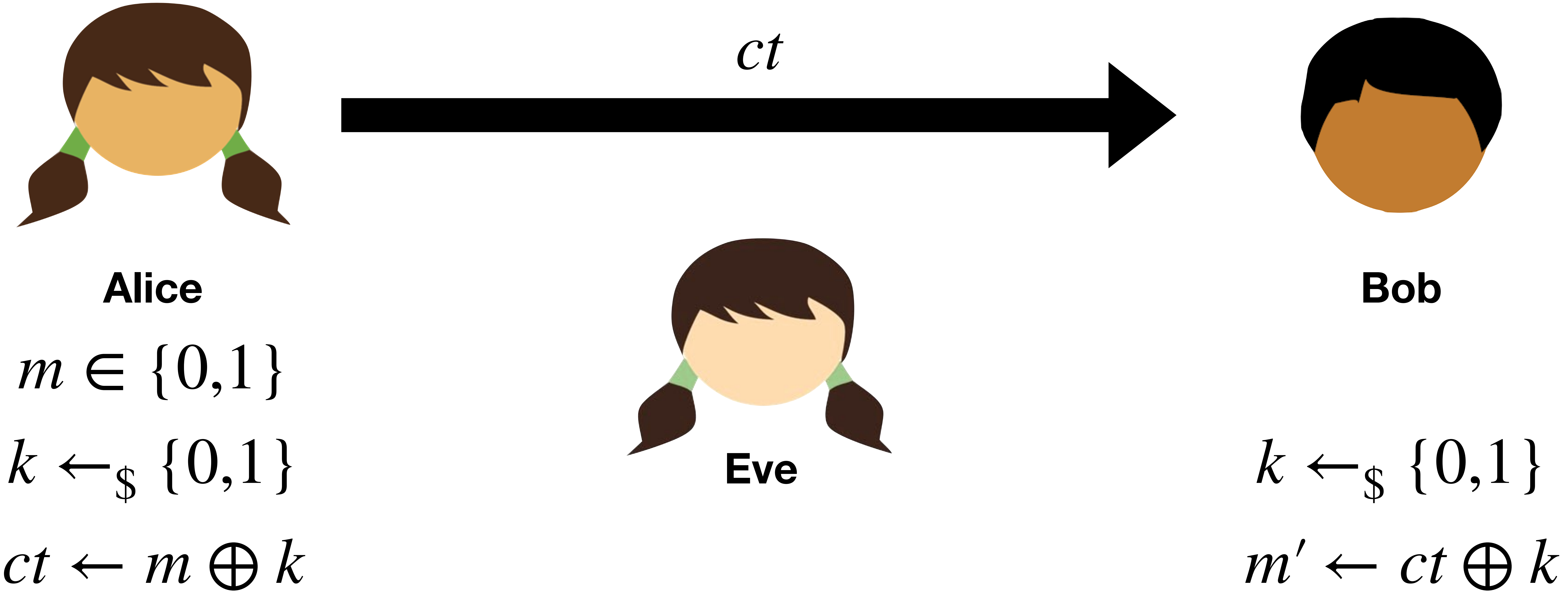
Eve

Bob

$k \leftarrow_{\$} \{0,1\}$

$m' \leftarrow ct \oplus k$

Question: *what if Alice wants to send more than one bit?*



ct

Alice

$m \in \{0,1\}$

$k \leftarrow_{\$} \{0,1\}$

$ct \leftarrow m \oplus k$

Eve

Bob

$k \leftarrow_{\$} \{0,1\}$

$m' \leftarrow ct \oplus k$

Question: *what if Alice wants to send more than one bit?*

Answer: *Alice and Bob can exchange a short PRG seed, then expand it (effectively) indefinitely*

Today’s objectives

Describe pseudorandomness/pseudorandom generators

Define negligible functions

Understand security of PRGs