Randomized also: uses $n$ bits
How do we supply this? Can we derandomize the algorithm?

Cryptography: Randomness crucial
- One-time pad
- Semantic security
  \[ \text{Enc}(m) = m \oplus k \rightarrow k \] secret key

Shannon: No \[ \xrightarrow{\text{Bypass this lower bound?}} \]

Computational vs Information-theoretic security
Information leaks in a way that a computationally bounded adversary can't detect it.

Key concept: "Pseudorandomness"
"A random looking string"
Computational Indistinguishability

$D_1$ & $D_2$ on $\mathbb{S}^{0,1^n}$

$D_1$ & $D_2$ are statistically close:

$D_1$, $D_2$ are $\varepsilon$-close if $\forall$ tests $T: \mathbb{S}^{0,1^n} \to \{0,1\}$

$$\left| \Pr_{x \in D_1} [T(x) = 1] - \Pr_{x \in D_2} [T(x) = 1] \right| \leq \varepsilon$$

$$\Delta_{TV}(D_1;D_2) := \max_{T \text{ of above}} \left| \Pr_{x \in D_1} [T(x) = 1] - \Pr_{x \in D_2} [T(x) = 1] \right|$$

Exercise: $\Delta_{TV}(D_1;D_2) = \frac{1}{2} \|D_1 - D_2\|_1$

$$= \frac{1}{2} \sum_{x \in \mathbb{S}^{0,1^n}} |D_1(x) - D_2(x)|$$

$\Delta_{TV}(D_1;D_2) \in [0,1]$
**Defy Dist** $D$ is said to be $(C, \varepsilon)$-pseudorandom if $D \sim \text{Uniform Dist}$ (randomized)

Claim: If an algorithm $A$ with “complexity” $C(n)$ using $n$ random bits is fed a pseudorandom string from a $(C, \varepsilon)$-pseudorandom distribution, then its accuracy deviates by at most $\varepsilon$.

$$\left| \Pr_{x \sim \text{Unif}} [A(x) = 1] - \Pr_{x \in D} [A(x) = 1] \right| \leq \varepsilon$$

---

How to produce a pseudorandom sample. **Pseudorandom Generator**: $G : \{0,1\}^S \rightarrow \{0,1\}^n$

is an $(C, \varepsilon)$-PRG if

$G(\text{Unif}(\{0,1\}^S))$ is $(C, \varepsilon)$-pseudorandom

$G$ stretches a random string of size $S$ into a "random looking" string of length $n$.
Even for $s = n - 1$, this is impossible statisically.

Computationally seek $s < n$, (like $n^{0.01}$, or even $O(lg n)$)

**Food for Thought:** If $G$ is computable in $G: \{0,1\}^{cn} \rightarrow \{0,1\}^n$ in $20(n)$ time and it is $(C(n), 1/10)$-pseudorandom, then a randomized alg with complexity $C(n)$ & success prob $2/3$ can be derandomized into a det. alg with run time $O(5(n)). C(n)$

Run $A$ on $G(a)$ ($a \in \{0,1\}^S$) as its "random" stay, take majority.
So if \( s(n) = O(\log n) \)
then randomized polynomial also
\[
\begin{align*}
\text{compute} & \quad \text{det polynomial also} \\
\text{Intuitively, output of } G & \quad \text{is } (\mathbb{Z}_2)\text{-pseudorandom} \\
\text{First 5 bits are random } (x \text{ is random}) \\
\text{Link between hardness & pseudorandomness}
\end{align*}
\]
Great, but there's a small problem.

\[ G(x) = (x, g(x)) \]

hard, so how can G compute it.

Two possible avenues:

1. Allow G more time than the tests it
   "fools". (Ok for derandomization
   Not ok for cryptographic uses)

2. Twist the PRG construction
   to create some asymmetry that gives
   G an edge compared to tests it has to
   fool.

Few words about 2:

\[ G(x) = (x, h(x)) \]

\( h: \{0,1\}^9 \rightarrow \{0,1\}^3 \)

hard to predict
Given \( \Pi(x) \) to a bounded adversary

\[ h(x) \text{ looks random} \]

\[ G(x) = (\pi(x), h(x)) \]

**One-way permutation**

\[ x \xrightarrow{\text{easy}} \pi(x) \]

\[ \pi(x) \xrightarrow{\text{hard}} h(x) \]

\[ h : \text{hard core predicate for permutation } \pi \]

\[ x \mapsto h(x) \text{ easy} \]

\[ \pi(x) \mapsto h(x) \text{ hard} \]
Example: Let $\mathbb{G} = \mathbb{Z}/p\mathbb{Z}$ for some prime $p$. For any $x \in \mathbb{G}$, $\mathbb{G}$ is a cyclic group of order $|\mathbb{G}| = p - 1$. Let $\mathcal{H} = \text{Hash}(\text{RSA exponentiation})$.

Comment: $1 \leq |S| \leq \frac{|\mathbb{G}|}{2}$

$\mathcal{H}_0 \leftarrow \mathcal{H}(x)$. Let $y = \mathcal{H}_0^{-1} : y \in \mathbb{G}$, where $\mathcal{H}_0^{-1}$ is the most significant half of $\mathcal{H}(x)$. The invertibility of $\mathcal{H}$ is characterized by the discrete logarithm $\mathcal{H}(x) = \text{eval} \circ \text{hash} \circ \text{exponentiation}(x)$. For $\mathcal{H}(x)$ to be hard-core for $\mathcal{L}$, the following must hold for all $x \in \mathbb{G}$:

$$\text{Eval}(x) = \mathcal{H}(x) \mod p \ (p \text{ prime})$$

$$\exists \mathcal{N}(x) \in \mathbb{Z}/2\mathbb{Z} \ \text{such that} \ h(x) \cdot \mathcal{N}(x) \equiv 0 \ (p \text{ prime})$$

Example of one-way permutation:}
Stretchily by more bits

Cryptography setting

\[ \text{Poly}(s) \rightarrow s \text{ bits} \]

\[ \text{One-way permutation} \rightarrow \text{PRG with seed keys} \]

Output \((b_0, b_1, b_2, \ldots)\)
Approach 1: $G$ takes more time than test it fools

\[ x \rightarrow x' \quad q \rightarrow \{011\} \rightarrow \{011\} \]

More bits?

\[ x \in \{011\}^* \]

Choose windows so that they have small intersection i.e. Design Error-correcting code