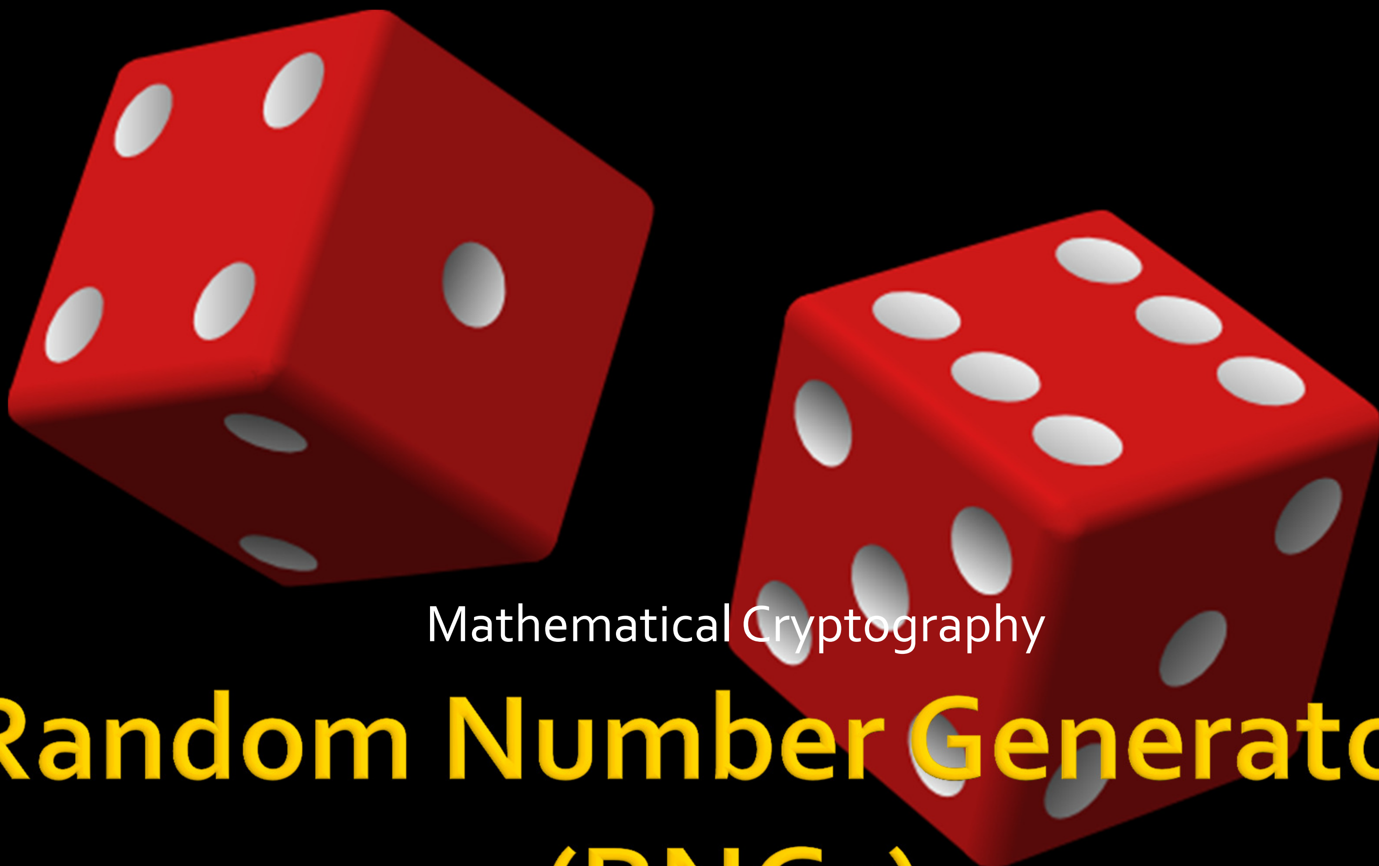


Mathematical Cryptography

Random Number Generators (RNGs)

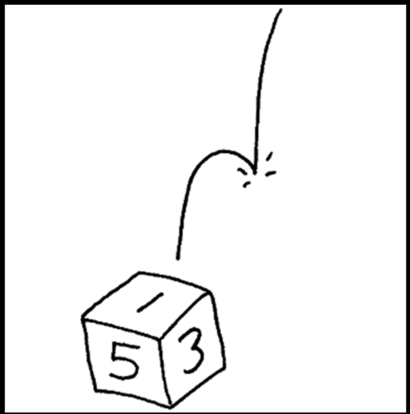
Muhammad Usman Akram
10030053

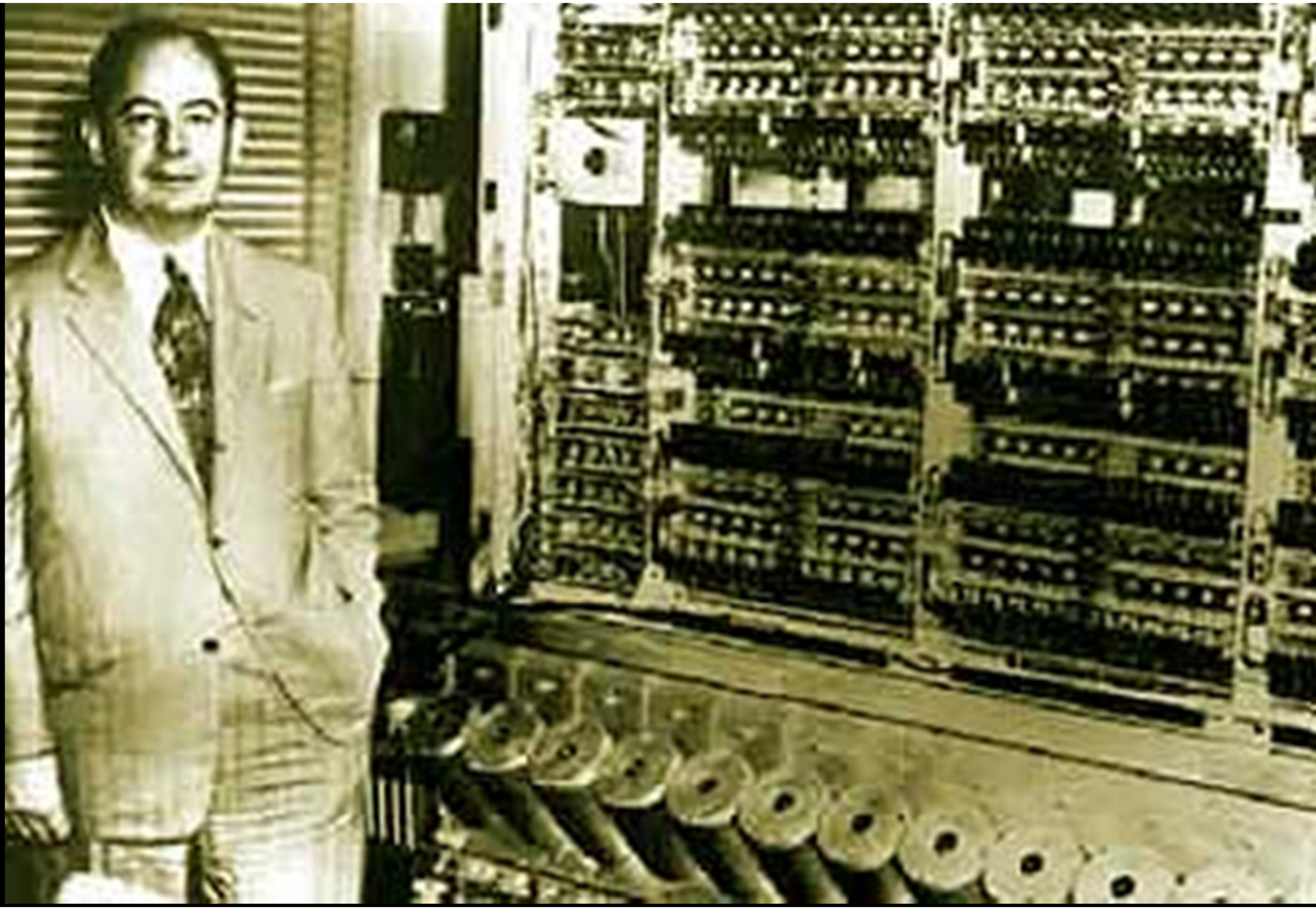
Overview

- Random Numbers
- Applications
- Desired Attributes
- Random Number Generators (RNGs)
- Pseudo Random Number Generators (PRNGs)
- Empirical Statistical Tests
- Cryptographically Secure RNGs

Random

- Lacking a definite plan, purpose, or pattern
- A set where each of the elements has equal probability of occurrence
- *A sequence in which each term is unpredictable -D. H. Lehmer (1951)*





Any one who considers arithmetical methods of producing random digits is, of course, in a state of sin.

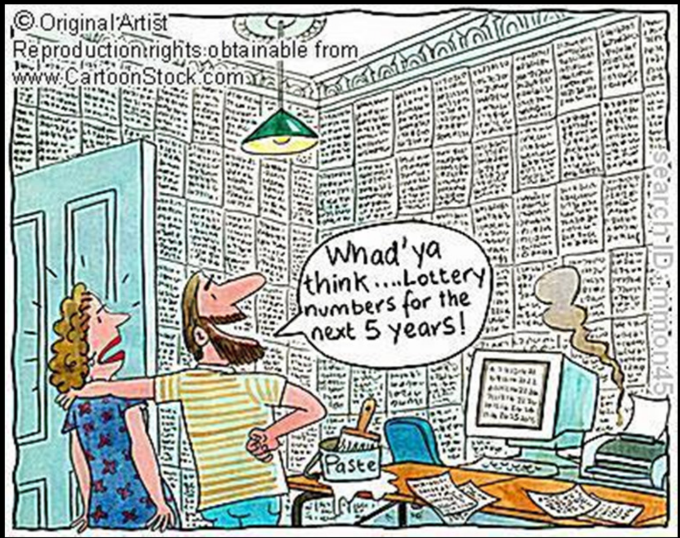
John von Neumann

Random Numbers

- True Random
  - Show "true" randomness
  - For Example: readings of a Geiger counter
- Pseudo Random (aka Deterministic Random)
  - Have some repeating pattern but show certain degree of randomness
- Quasi Random (aka Low-discrepancy)
  - more uniformly than uncorrelated random numbers

Applications

Application	Most Suitable Generator
Lotteries and Draws	TRNG
Games and Gambling	TRNG
Random Sampling	TRNG
Simulation and Modeling	PRNG
Security (e.g., generation of keys)	TRNG



Attributes

- Uniform distribution
- Uncorrelated / Independent
- Efficiency / Portability
- Replicable
- Long Period (before pattern starts repeating)

Desired Attributes for RNGs

Random Number Generators

- True Random Number Generators
  - Uses physical phenomena
  - With Quantum-random properties
    - Nuclear decay, Geiger counters exposed to radioactive material
    - Shot noise, a quantum mechanical noise source in electronic circuits
  - Without Quantum-random properties
    - Snapshots of lava lamps
    - Thermal noise from a resistor
    - Atmospheric noise

Random Number Generators

- Pseudo Random Number Generators
  - Using deterministic algorithms
    - Need a “seed” for initialization
    - Uses output of an iteration as input to next

Pseudorandom Number Generators

- According to Pierre L'Ecuyer, a RNG is:
  - *RNG = (S, s_O, T, U, G)*
    - S is a finite set of states
    - s_o is initial state (or seed)
    - Mapping T: S -> S is transformation function
    - U is finite set of output states
    - G: S -> U is output finction

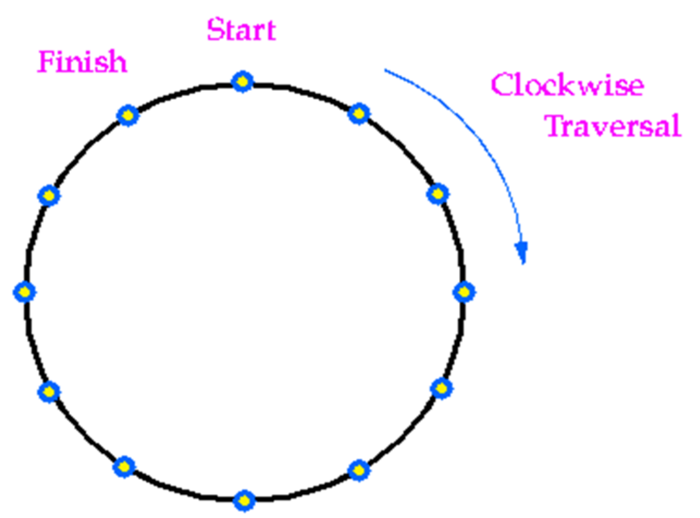
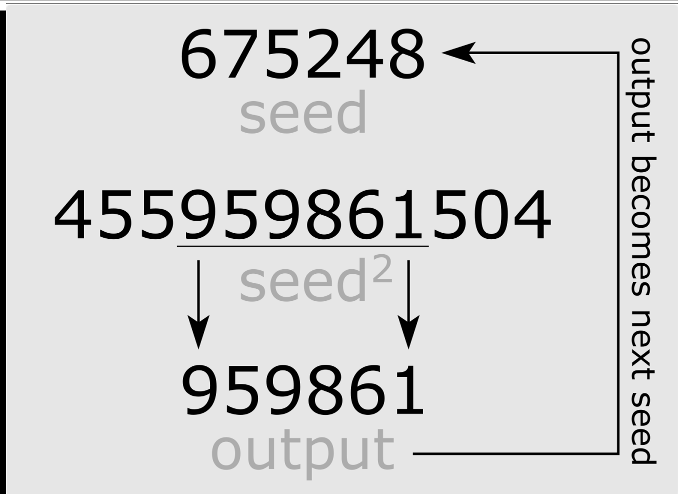


Figure 2: Illustration of Random Number Cycle

Pseudorandom Number Generators



- Mid Square RNG
- Congruential RNGs
  - Linear Congruential Generators
    - $X_{i+1} = a*X_i + c \bmod m$
  - Lehmer / Park–Miller RNG
    - $X_{i+1} = a*X_i \bmod m$
    - Multiplicative LCG (special case of LCG, with c = 0)
- Lagged Fibonacci RNG
  - $X_i = X_{i-J} + X_{i-K} \bmod m$

Pseudorandom Number Generators

- Blum Blum Shub RNG
  - $X_{i+1} = X_i^2 \bmod m$
- Xorshift class of RNGs designed by G. Marsaglia
  - repeatedly uses XOR on a number with a bit shifted version of itself
- MWC

$$x_n = (a x_{n-1} + c_{n-1}) \bmod b$$

$$c_n = \left\lfloor \frac{a x_{n-1} + c_{n-1}}{b} \right\rfloor$$

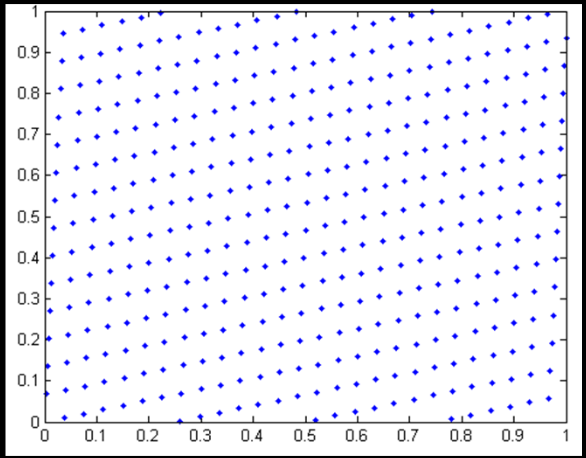
$$n \geq r$$

Pseudorandom Number Generators

- Cryptographic Random Number Generators
  - Strong Hash functions
  - Cryptographic algorithms

George Marsaglia (*Guru of RNGs*)

- "Random numbers fall mainly in the planes"
- Developed some of the most commonly used methods for generating random numbers
  - RNGs
    - multiply-with-carry
    - subtract-with-borrow
    - Xorshift
    - Mother
    - KISS
  - Ziggurat algorithm for generating normally distributed random numbers
- Diehard RNG tests Battery (part of Marsaglia CDROM)

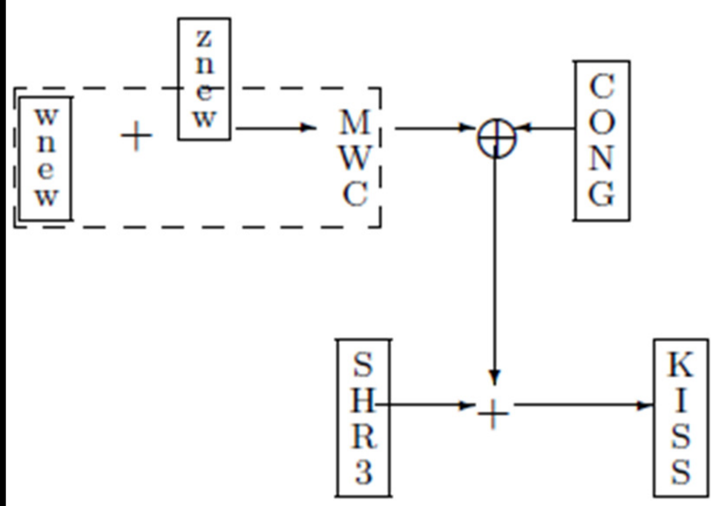


Keep It Simple Stupid Generator

- KISS generator is an efficient pseudo-random number generator by George *Marsaglia* and Arif *Zaman* in 1993
  - KISS consists of a combination of four sub-generators each with 32 bits of state, of three kinds:
    - one linear congruential generator modulo 2^{32}
    - one general binary linear generator over the vector space $GF(2)^{32}$
    - two multiply-with-carry generators modulo 2^{16}, with different parameters

KISS Generator (G. *Marsaglia & A. Zaman*)

The KISS generator, (Keep It Simple Stupid), is designed to combine the two multiply-with-carry generators in MWC with the 3-shift register SHR3 and the congruential generator CONG, using addition and exclusive-or. Period about 2^123.



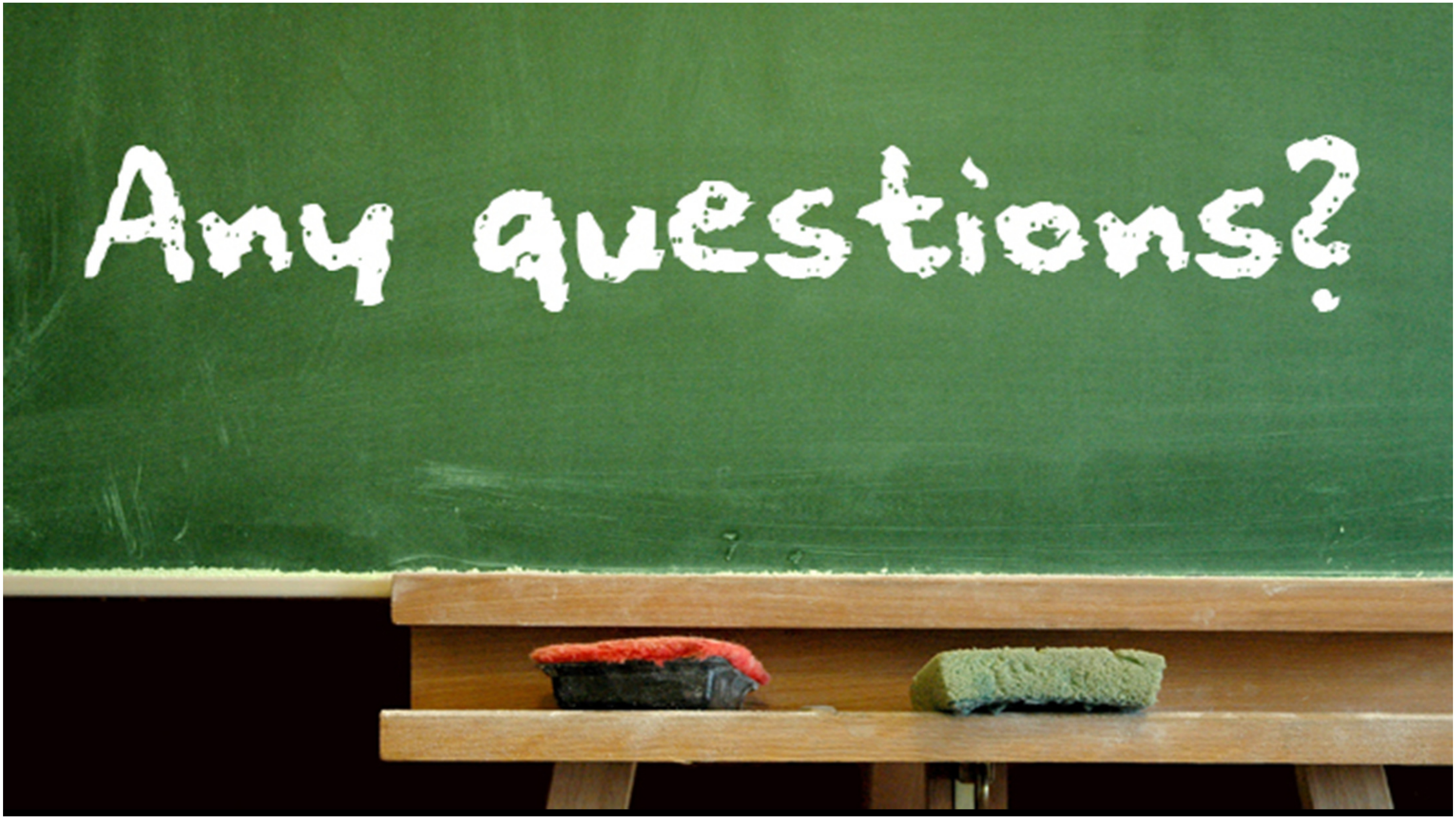

```
#define znew (z=36969*(z&65535)+(z>>16))
#define wnew (w=18000*(w&65535)+(w>>16))
#define MWC ((znew<<16)+wnew )
#define SHR3 (jsr^=(jsr<<17), jsr^=(jsr>>13), jsr^=(jsr<<5))
#define CONG (jcong=69069*jcong+1234567)
#define KISS ((MWC^CONG)+SHR3)
```

Statistical Tests

- Diehard tests are a battery of tests, developed by G. Marsaglia
- Includes, following tests
  - Birthday spacings
  - Overlapping permutations
  - Ranks of matrices
  - Monkey tests
  - Count the 1s
  - Parking lot test
  - Minimum distance test
  - Random spheres test
  - The squeeze test
  - Overlapping sums test
  - Runs test
  - The craps test

Cryptographically Secure RNGs

- Where we need random numbers
  - Key generation
  - Nonces
  - One-time pads
  - Salts in certain signature schemes
- Should satisfy "next-bit test"
- Should with stand "state compromise extension"


Any questions?

•Hoffstein, Pipher, and Silverman, "*An Introduction to Mathematical Cryptography*"
•P. L'Ecuyer, "*Random Number Generation*", Chapter 4 of the Handbook on Simulation, Jerry Banks Ed., Wiley, 1998
•P. L'Ecuyer, "*Uniform random number generators: a review*", 29th conference on Winter simulation, IEEE, 1997
•G. Marsaglia, "*The Marsaglia Random Number CDROM including the Diehard Battery of Tests of Randomness*"
•Greg Rose, "*KISS: A Bit Too Simple*", Cryptology ePrint Archive, 2011
•http://en.wikipedia.org/wiki/Diehard_tests
•http://en.wikipedia.org/wiki/Cryptographically_secure_pseudorandom_number_generator
•http://www.boallen.com/random-numbers.html
•http://en.wikipedia.org/wiki/Lehmer_random_number_generator
•http://www.iro.umontreal.ca/~lecuyer/papers.html
•http://www.h-online.com/security/news/item/Random-numbers-from-entangled-atoms-995780.html

Young man, in mathematics you don't understand things. You just get used to them.
John von Neumann