Randomness

Randomness is easy to take for granted. Quarters make for great tie breakers and anyone who's played a card game knows how fun, or defeating, a random outcome can be. Since there are no coins or cards inside your laptop, pseudo-random systems are instead used to synthesize the randomness observed in nature. These systems, called Random number generators (RNGs), are used in many applications relating to cryptography.

Achieving true randomness on a computer is not straightforward. The algorithms created and implemented vary drastically and are often impressively complicated. The quality of RNGs are judged on a few criteria including:

- Statistical Properties – Are numbers generated are evenly distributed?
- Geometric properties – Do numbers lie on any plane of N dimensions?
- Period Length – Large amount of numbers can be generated before the algorithm duplicates number patterns

With the emphasis on cyber security research, judging specific RNGs may fall outside the scope of this crash course. For everyday purposes, RNGs have come a long way after decades of cryptographic research and are usually adequate for DIY and learning applications. For secure implementations, there are hardware chips dedicated to generating random numbers. Consider researching and implementing dedicated hardware to produce good randomness.

For this lab, we will be looking at two different RNG algorithms. One, RANDU, is an algorithm developed and used in the 1960s before the importance of randomness and cryptography was demonstrated. The other is Python's random class. Two python scripts are included comparing two different RNGs. Please run, modify accordingly, and answer the questions within the python files.

RANDU - https://en.wikipedia.org/wiki/RANDU

Python Random Class - https://docs.python.org/2/library/random.html