Praudyogikee Lab – Randomization Test

Java uses PRNG “Pseudo-Random Number Generator” in Math.random() Algorithm.
We decided to make a test to check how random is Math.random().

Date: 26/1/2021

Test Description:
We used standard deviation formula “σ” as a scale for randomness, where R∝ σ.
We made a lot of random numbers with PRNG and a specific formula,
f(x) = 2 * y^2
when 2≥x>0,y=sin(x)
when 4≥x>2,y=cos(x)
when 6≥x>4,y=tan(x)
when 8≥x>6,y=log(x)
when 10≥x>8,y=log10(x)
x is generated using multiple solutions such as PRNG,System.getNanoTime(),etc..

Results:

x is generated by PRNG, 100 values * 100 tries.
Average deviation for Math.Random() : 0.28850880240284
Average deviation for MyRand() : 0.259040230836988

x is generated by PRNG, 300 values * 100 tries
Average deviation for Math.Random() : 0.288875328666978
Average deviation for MyRand() : 0.256947333228367

x is generated by PRNG, 10000 values * 150 tries
Average deviation for Math.Random() : 0.288721994855821
Average deviation for MyRand() : 0.25687806192743

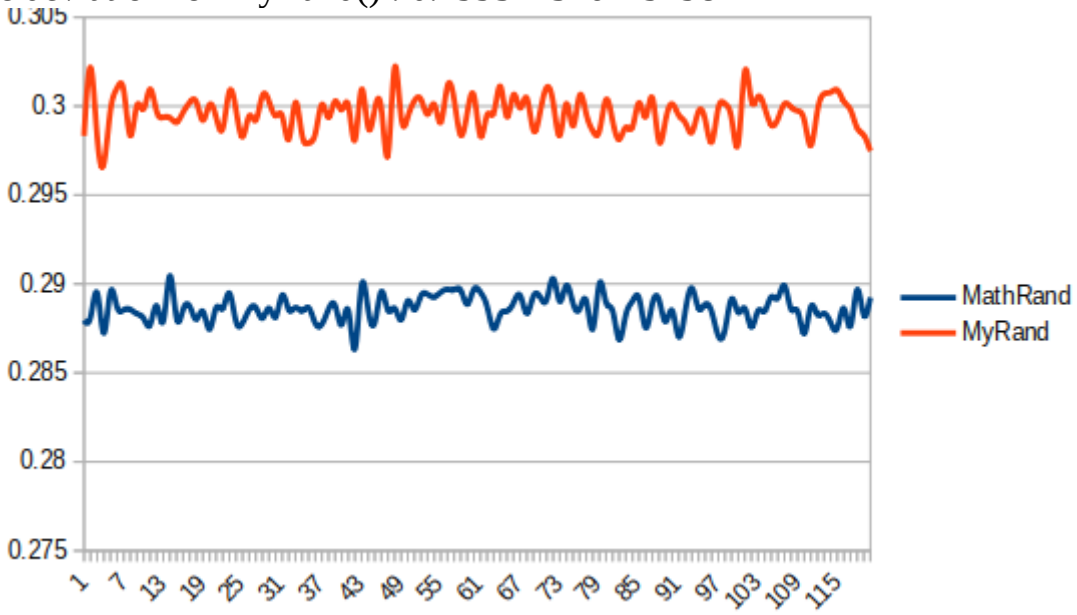
We noticed that something was wrong, we are trying to beat PRNG. Instead, we put more rules on PRNG. Let’s change that.

f(x) = y
when x<100,y=sin(x)
when x<200,y=cos(x)
if not both,y=tan(x)

x is generated by PRNG, 10000 values * 150 tries
Average deviation for Math.Random() : 0.288418600769118
Average deviation for MyRand() : 0.299531359836622

x is generated by PRNG, 25000 values * 120 tries “that’s 3Million”

Average deviation for Math.Random() : 0.288604399794008
Average deviation for MyRand() : 0.299571526419498



Conclusion: The less the rules, the more random the numbers will be. That’s an important conclusion if we want to make a complete random “not pseudo-random” algorithm that can generate random numbers for encryption/security purposes.
Also, It seems hard for any randomness algorithm to be σ>0.31.but if we made it, we can build more secure systems.