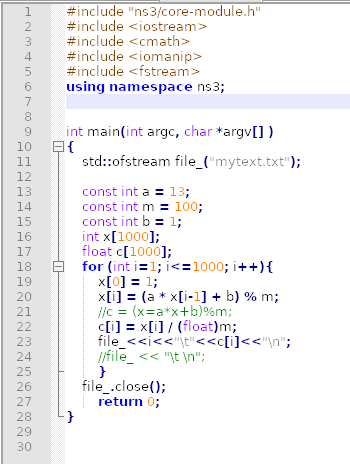
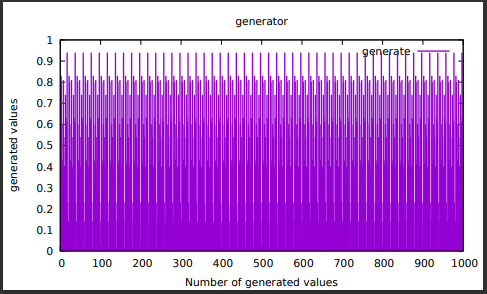
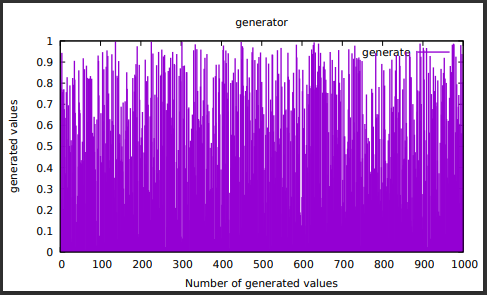
**Part I: PRNG and Random Variables**

1. Create a function that implements a linear congruential generator (LCG), accepting as input the parameters: seed, m, a, and c. Hint: It is better if you do not attempt to modify the rng module of ns-3; instead, create a function in your simulation file (e.g., mysimulation.cc) and call the function from the main.

2. Generate 1000 values uniformly distributed in the range [0,1] using your PRNG. For this case use m=100, a=13 c=1 and seed =1;



3. Compare the distribution of your values with the distribution of values generated using the UniformRandomVariable() of ns-3. Hint: To compare the distribution of values, you can use a histogram plot.



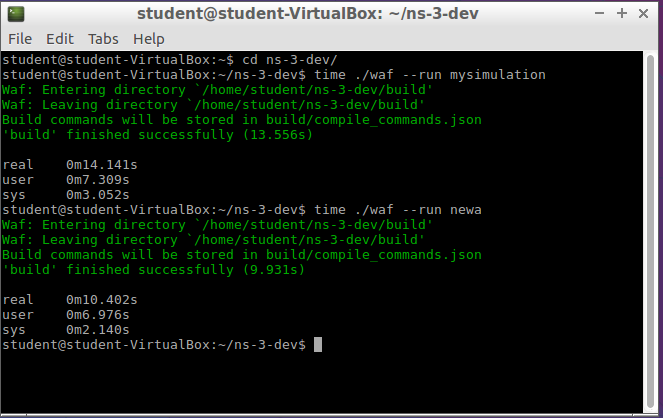
4. Comment on the difference in the results and propose values of m, a, and c which gives you better results.

To get a better result as for random numbers then if m is a prime number the generated number will never be 0.

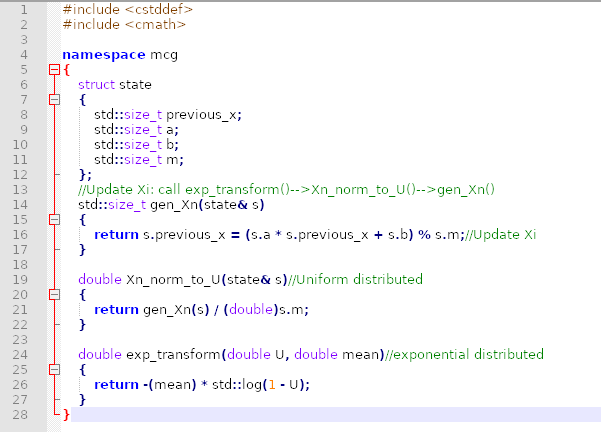
5. What PRNG does ns-3 use? What method does ns-3 use to generate a normal random variable?

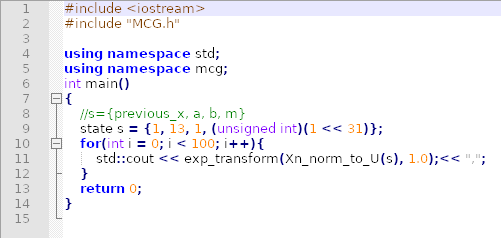
*ns-3* contains a built-in pseudo-random number generator (PRNG)

each RandomVariableStream used in *ns-3* has a virtual random number generator associated with it; all random variables use either a fixed or random seed based on the use of the global seed

6. Using the time system command of Linux compare the execution time for the generation of the uniform distribution using your function and ns-3 function. 

7. Write a second function that generates an exponential distribution with mean 𝛽 > 0 from a uniform distribution generated using the LCG; Choose one of the methods for generating RV covered in the course and motivate your choice with respect to the specific task.





8. Compare your exponential distribution with ns-3 ExponentialRandomVariable()and the theoretical expression of the probability density function.

