Name Adam Schaible

CS 4050

11/17/19

**Programming Assignment #4 (Random Numbers)**

In order for a number to random every number must have an equal probability of being generated, which by extension also means that sequences of different values also have an equal probability of generated. (Haahr) This means that all numbers randomly generated should have an even distribution that does not favor the generation of one number over the other and by extension this also means that all sequences that can be generated from these numbers should also have even distribution. The chance of generating any number between 0-9 is 1 in 10, since random events are independent events. (Haahr) A 1 in 10 chance of getting a number combined with the fact that random numbers are evenly distributed means that one should not see any digit more than on average 10% of the time.

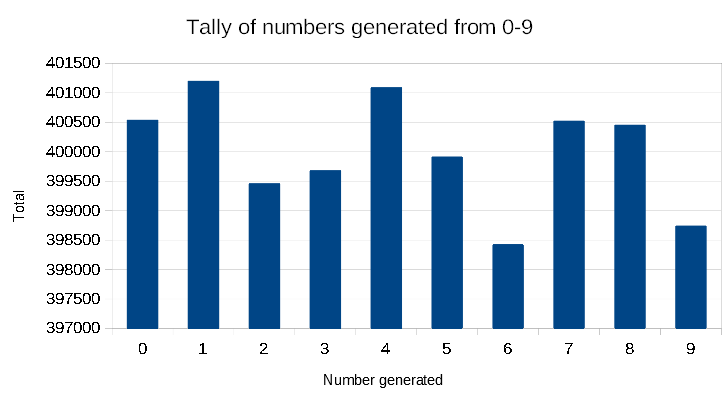
If one had a sequence of two digits there are a 100 ways to make a two digit combo (e.g 10 digits\*10 digits = 100 combinations for a sequence). A 100 ways to make a random two digit sequence combined with that sequences should be uniformly distributed means that any unique combination has a 1 in 100 chance of being randomly generated and any random number generator should be reflective of this. In order for a random number generator to be considered good it must have both an even distribution of both digits generated and sequences as well as the probability of generating any digit should be 0.10(1 in 10). With randomly generated numbers and their sequence being uniformly distributed it means that both the randomly generated numbers and their sequences should have a statistical standard deviation of zero for a perfect random number generator. The random number generator must be able to generate all numbers from 0-9.

For this project two different random number generators will be compared in a Java program I wrote named ComparingTwoRandomGenerators.java. One random generator that will be tested is Java’s built in random generator from the class Random. The other random generator is one that I built that will generate numbers by sequentially reading each byte in a wav file and then getting absolute value of the byte%10 (byte modulus 10). The website in which I got the sound from is <http://soundbible.com/grab.php?id=1574&type=wav>. Both random number generators will generate 4 million numbers that will be be between 0-9. The sequences generated will consist of two digits where the current digit generated is associated to the previous digit generated. After the numbers and sequences are generated they will be evaluated in LibreOffice in terms of their standard deviation to determine how uniformly distributed they are, ability to generate all numbers, raw data will be given in table form and graphed in order to visually illuminate any skewing (lack of unformal distribution) in terms of sequences or numbers generated. The standard deviation will be calculated using LibreOffice Calc’s built in standard deviation calculator.

**Results from Java’s Random Number generator:**

The tally of numbers generated from 0-9

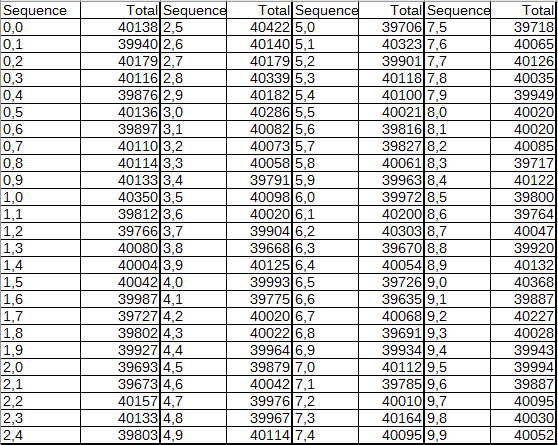


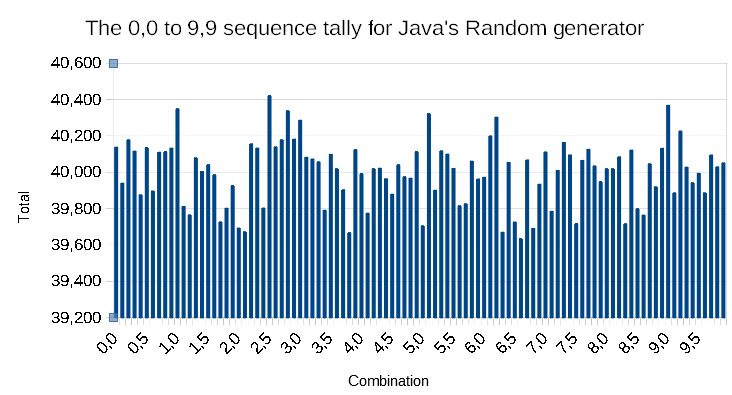


Standard Deviation of numbers generated using Java’s Random number generator = 933.763829289231

**Results from Java’s Random Number generator (continued):**

Tally of the number of sequences from 0,0 to 9,9 generated from Java’s Random number generator



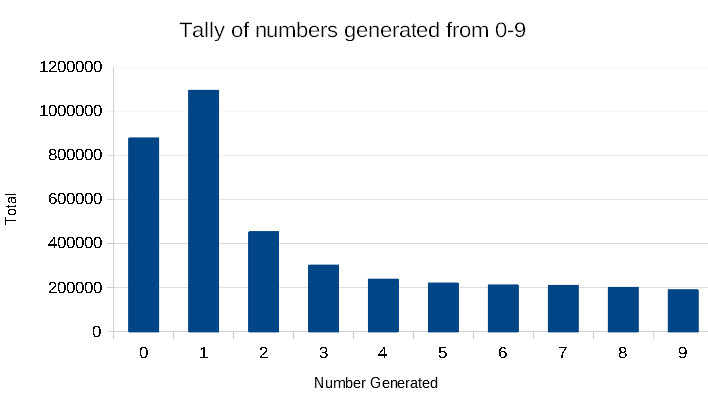


Standard Deviation of two digit sequences generated using Java’s Random number generator = 220.070374419116

**Results from the custom number generator that I made:**

The tally of numbers generated from 0-9



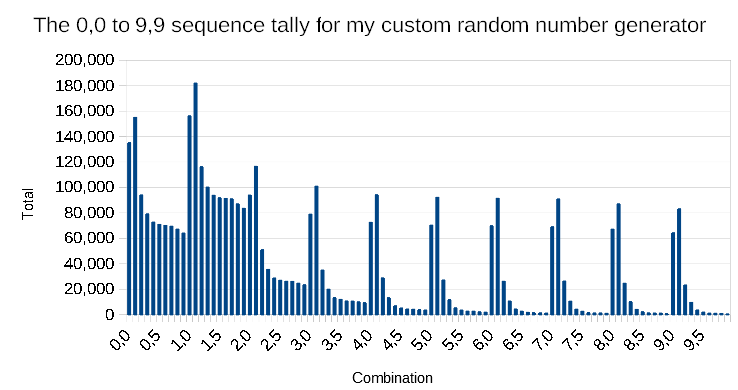


Standard Deviation of numbers generated using the custom number generator I made = 322655.375620973

**Results from the custom number generator that I made (Continued):**

Tally of sequences from 0,0 to 9,9 generated from my own custom made number generator





Standard Deviation of two digit sequences generated using my custom number generator = 43523.8566737551

**Analysis of Java’s built in random number generator:**

Java’s built in random number generator was able to successfully generate all individual digits and all 2 digit sequences and because of that it passes the first litmus test in terms of being considered as a successful random number generator. The graph representations of the tally counts for the individual numbers generated and the associated sequences did not show any left/right skewing or anything that could be considered a well formed bell curve, so that infers that both the individual numbers generated and the associated sequences are fairly evenly distributed. The tally of the individual digits generated had a standard of 933.763829289231, which is a small in comparison to each digit having an average count of 400,000 and this means that the standard deviation also says that the individual numbers generated is also fairly evenly distributed. The sequences standard deviation is 220.07, which is small in comparison to an average sequence tally count of 40,000, so in terms of generating 2 digit sequences the generator has a fairly even distribution. Overall Java’s random generator has a fairly good at generating numbers and 2 digit sequences that are evenly distributed, even though the generator does not have a standard deviation of zero for either individual numbers or 2 digit sequences.

**Analysis of the random number generator that I built:**

My random number generator was able to successfully generate all individual digits and all 2 digit sequences and because of that it passes the first litmus test in terms of being considered as a successful random number generator. The graph representation of the tally counts for the individual numbers generated is right skewed, so this means that individually generated numbers are not evenly distributed. The graph representation of the 2 digit sequences does have some right skewing as well as a few bell curves, which means that the sequences generated are also not evenly distributed. The standard deviation of the individual numbers generated is 322655.375620973 and this is very high in comparison to the average tally of each number generated is 400,000 and this means that individual numbers generated are not evenly distributed. The standard deviation of the 2 digit sequences is 43523.8566737551 and this is very high in comparison to the average tally of each sequence generated is 40,000 and this means that the sequences generated are not evenly distributed. Because this number generator does not produce an even distribution of individually generated numbers or sequences it cannot be considered an effective random number generator.

Citations:

Haahr, Mads. “True Random Number Service.” *RANDOM.ORG - Introduction to Randomness and Random Numbers*, https://www.random.org/randomness/.