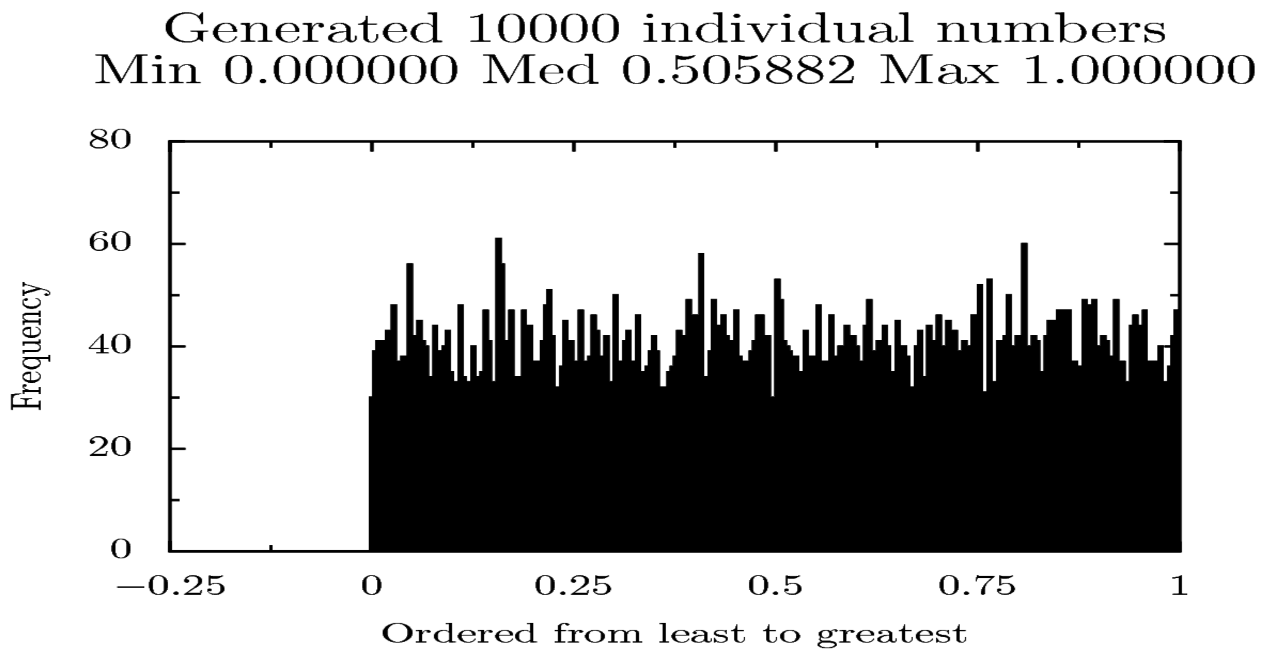
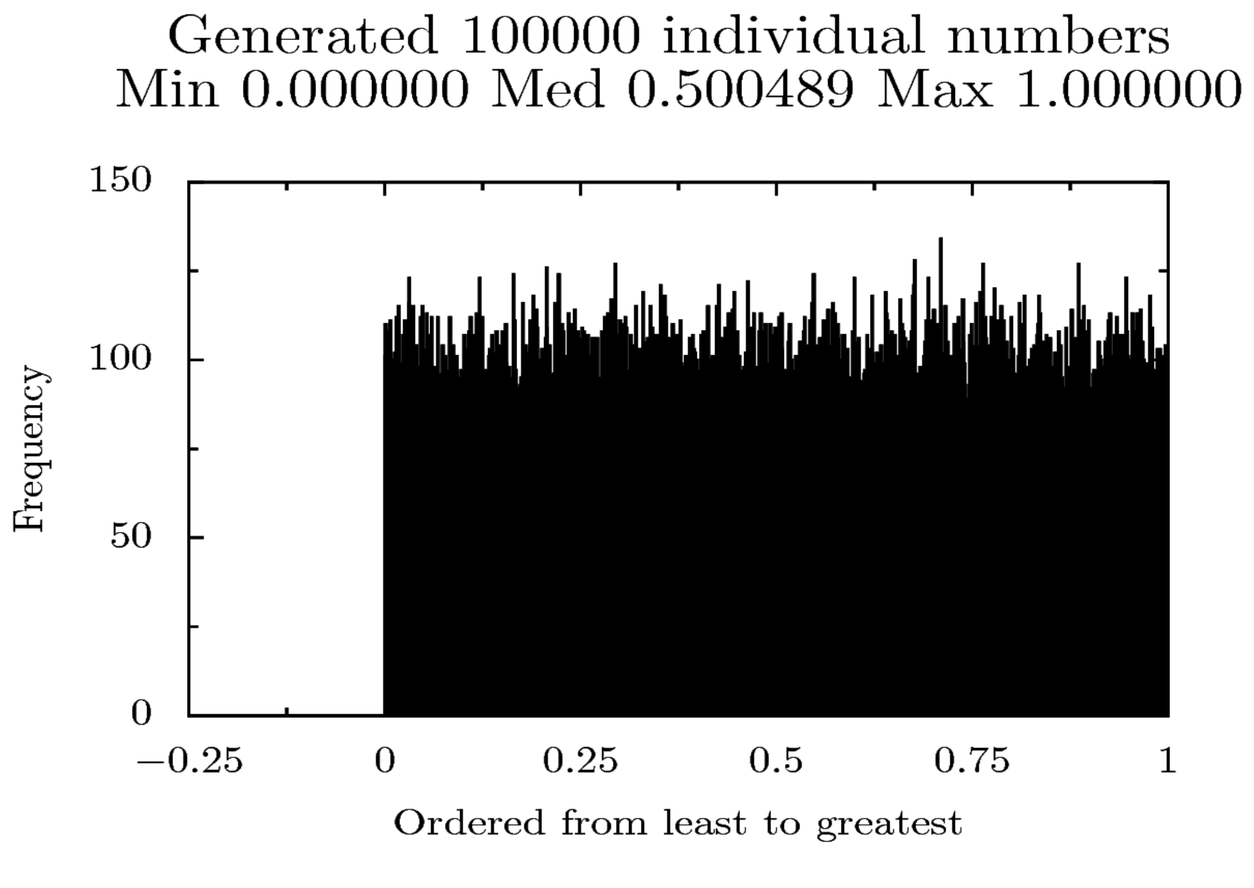
```python
def getRandom(N):
  """
  Returns an random number.
  Note N bit number results should be [0, 2**N - 1] inclusive.
   For a 10bit number.
   Min should be greater or equal to 0
   Max should be less than or equal to 1023

  """
  randfunc = os.urandom

  S = randfunc(N/8)

  odd_bits = N % 8
  if odd_bits != 0:
    char = ord(randfunc(1)) >> (8-odd_bits)
    S = chr(char) + S

  value = 0L
  length = len(S)

  for i in range(0, length):
    value = (value << 8)
    value = value + ord(S[i])

  #value |= 2L ** (N-1)            # Ensure high bit is set
  return value

def getFloat(N_bit):
  """
  Will return a float of N_bit resolution.

  This is meant to duplicate the behavior of random.random() but use
  or.urandom as a source instead of a PRNG
    Return the next random floating point number in the range [0.0, 1.0).

  TODO - experiment with size of denomenator to ensure 1.0 does not return.
  """

  # numberator is a number between 0 and 2**(N_bit) - 1
  numerator = float(getRandom(N_bit))

  # Should use a number larger then the numerator to ensure
  # that the float is never equal to 1.0
  denomenator = float(2**(N_bit) - 1)

  return numerator / denomenator
```

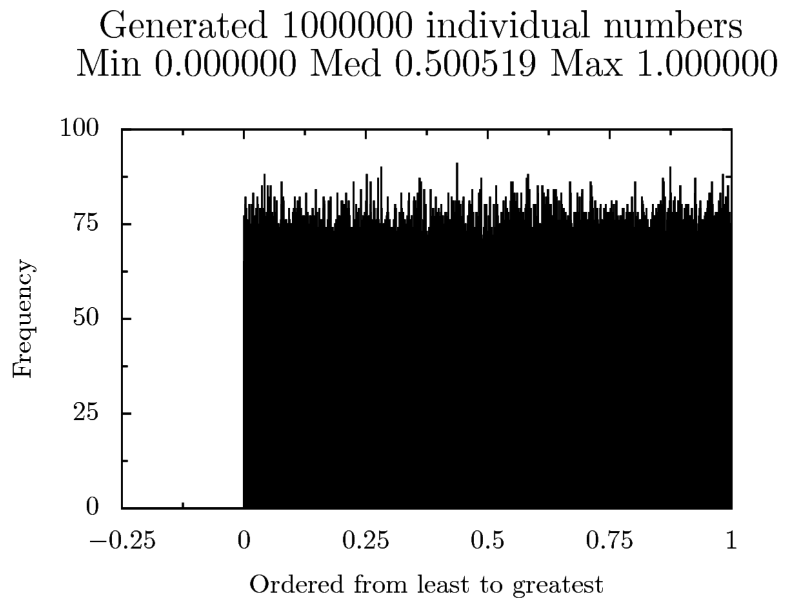
```
Generated 10000 individual floats with 8 bit resolution
Min 0.000000 Med 0.505882 Max 1.000000
The maximum frequency was 61 for number 0.156863
```



```
Generated 10000 individual floats with 8 bit resolution
Min 0.000000 Med 0.505882 Max 1.000000
The maximum frequency was 61 for number 0.156863
```



```
Generated 1000000 individual floats with 14 bit resolution
Min 0.000000 Med 0.500519 Max 1.000000
The maximum frequency was 91 for number 0.436794
```


Generated 1000000 individual numbers
Min 0.000000 Med 0.500519 Max 1.000000
Frequency
Ordered from least to greatest