111550113 謝詠晴

Cryptography Engineering Quiz. 5

Problem 1

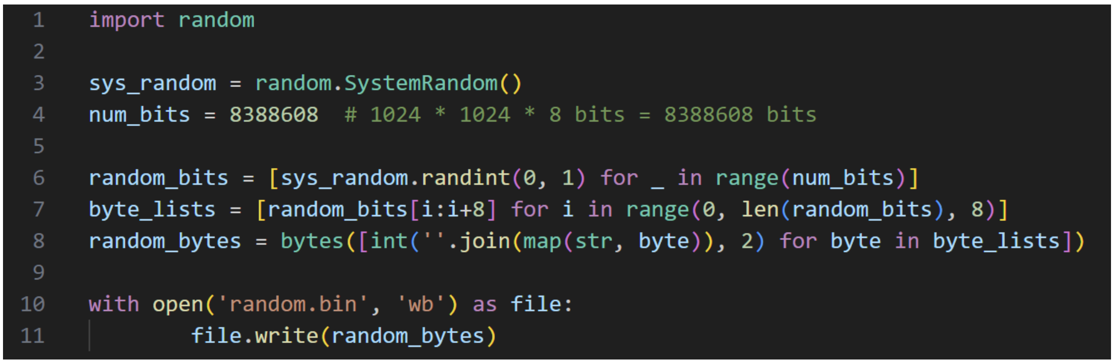
a) Python code to generate 1M bytes random numbers:

(1) Import ‘random’ module and use *‘sys_random’* to represent random.SystemRandom().

(2) First, generate 8388608 number of random 0 or 1 and store them as random_bits.

(3) Convert each sublist of 8 bits into a byte and form a byte object using bytes() and store them as *‘random_bytes’*.

(4) Write *‘random_bytes’* to a binary file named "random.bin" in binary mode.

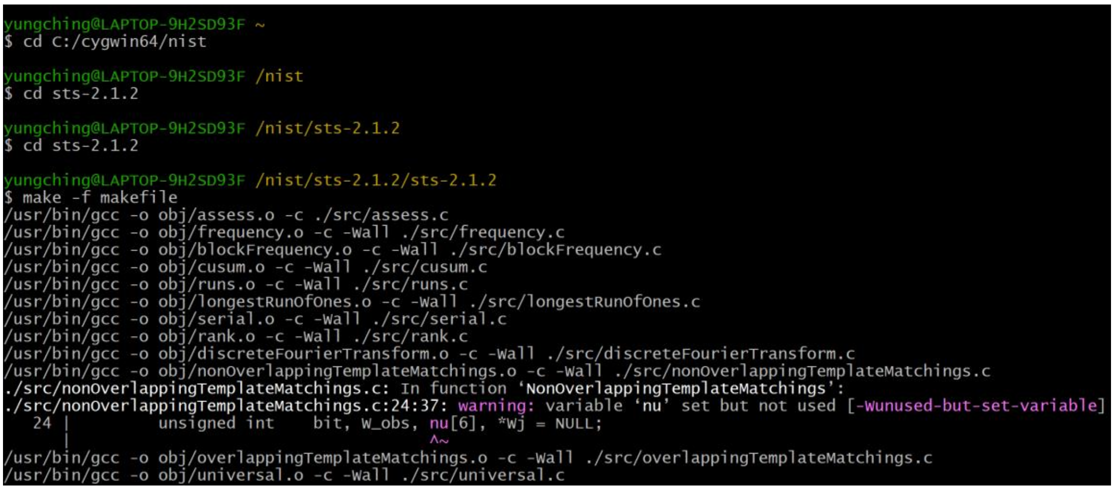


```python
import random

sys_random = random.SystemRandom()
num_bits = 8388608  # 1024 * 1024 * 8 bits = 8388608 bits

random_bits = [sys_random.randint(0, 1) for _ in range(num_bits)]
byte_lists = [random_bits[i:i+8] for i in range(0, len(random_bits), 8)]
random_bytes = bytes([int(''.join(map(str, byte)), 2) for byte in byte_lists])

with open('random.bin', 'wb') as file:
        file.write(random_bytes)
```

b) Steps for NIST SP 800-22 statistical test:

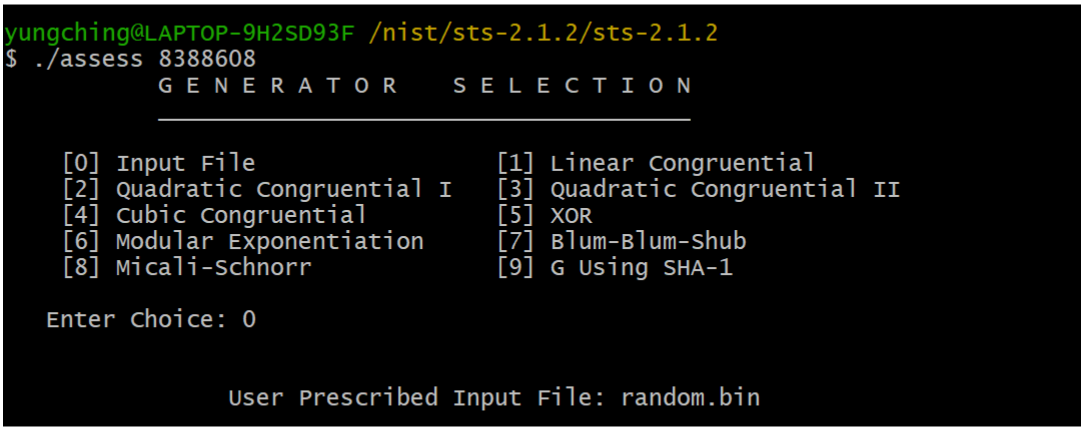
(1) Install NIST and Cygwin. During the installation of Cygwin64, select the following packages: binutils, make, gcc-g++,gdb, mingw-gcc, etc. Next, place the downloaded NIST directory into the Cygwin directory.

(2) Open Cygwin terminal, access the directory ‘sts-2.1.2’ ,and makefile using *‘make -f makefile’* command.



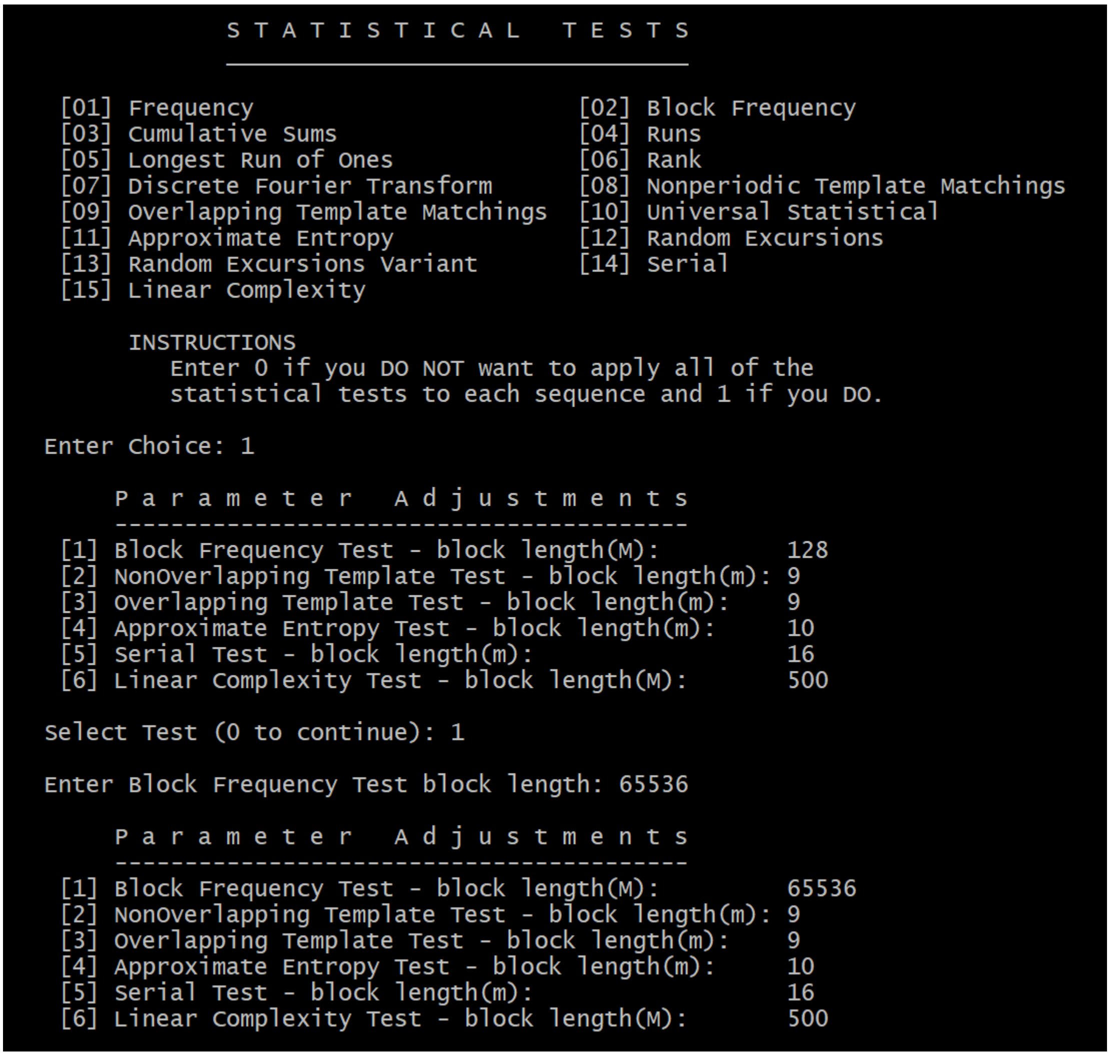
```
yungching@LAPTOP-9H2SD93F ~
$ cd C:/cygwin64/nist

yungching@LAPTOP-9H2SD93F /nist
$ cd sts-2.1.2

yungching@LAPTOP-9H2SD93F /nist/sts-2.1.2
$ cd sts-2.1.2

yungching@LAPTOP-9H2SD93F /nist/sts-2.1.2/sts-2.1.2
$ make -f makefile
/usr/bin/gcc -o obj/assess.o -c ./src/assess.c
/usr/bin/gcc -o obj/frequency.o -c -Wall ./src/frequency.c
/usr/bin/gcc -o obj/blockFrequency.o -c -Wall ./src/blockFrequency.c
/usr/bin/gcc -o obj/cusum.o -c -Wall ./src/cusum.c
/usr/bin/gcc -o obj/runs.o -c -Wall ./src/runs.c
/usr/bin/gcc -o obj/longestRunOfOnes.o -c -Wall ./src/longestRunOfOnes.c
/usr/bin/gcc -o obj/serial.o -c -Wall ./src/serial.c
/usr/bin/gcc -o obj/rank.o -c -Wall ./src/rank.c
/usr/bin/gcc -o obj/discreteFourierTransform.o -c -Wall ./src/discreteFourierTransform.c
/usr/bin/gcc -o obj/nonOverlappingTemplateMatchings.o -c -Wall ./src/nonOverlappingTemplateMatchings.c
./src/nonOverlappingTemplateMatchings.c: In function ‘NonOverlappingTemplateMatchings’:
./src/nonOverlappingTemplateMatchings.c:24:37: warning: variable ‘nu’ set but not used [-Wunused-but-set-variable]
   24 |         unsigned int    bit, W_obs, nu[6], *Wj = NULL;
      |                                     ^~
/usr/bin/gcc -o obj/overlappingTemplateMatchings.o -c -Wall ./src/overlappingTemplateMatchings.c
/usr/bin/gcc -o obj/universal.o -c -Wall ./src/universal.c
```

(3) ./assess 8388608(1024*1024*8) and input file ‘random.bin’



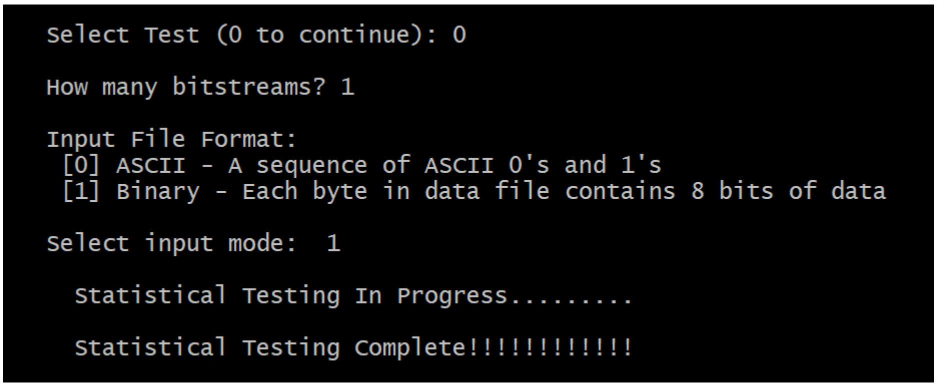
```
yungching@LAPTOP-9H2SD93F /nist/sts-2.1.2/sts-2.1.2
$ ./assess 8388608
           G E N E R A T O R    S E L E C T I O N
           ______________________________________

    [0] Input File                  [1] Linear Congruential
    [2] Quadratic Congruential I    [3] Quadratic Congruential II
    [4] Cubic Congruential          [5] XOR
    [6] Modular Exponentiation      [7] Blum-Blum-Shub
    [8] Micali-Schnorr              [9] G Using SHA-1

   Enter Choice: 0


                User Prescribed Input File: random.bin
```

(4) Adjust block lengths from 128 to 65536.



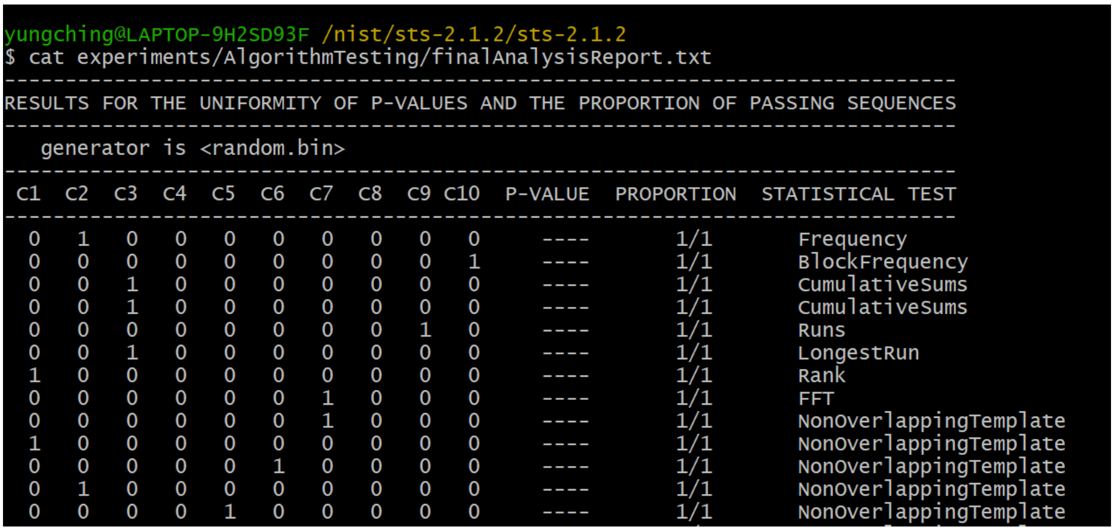
```
                S T A T I S T I C A L   T E S T S
                _________________________________

    [01] Frequency                       [02] Block Frequency
    [03] Cumulative Sums                 [04] Runs
    [05] Longest Run of Ones             [06] Rank
    [07] Discrete Fourier Transform      [08] Nonperiodic Template Matchings
    [09] Overlapping Template Matchings  [10] Universal Statistical
    [11] Approximate Entropy             [12] Random Excursions
    [13] Random Excursions Variant       [14] Serial
    [15] Linear Complexity

         INSTRUCTIONS
            Enter 0 if you DO NOT want to apply all of the
            statistical tests to each sequence and 1 if you DO.

   Enter Choice: 1

        P a r a m e t e r   A d j u s t m e n t s
        -----------------------------------------
    [1] Block Frequency Test - block length(M):         128
    [2] NonOverlapping Template Test - block length(m): 9
    [3] Overlapping Template Test - block length(m):    9
    [4] Approximate Entropy Test - block length(m):     10
    [5] Serial Test - block length(m):                  16
    [6] Linear Complexity Test - block length(M):       500

   Select Test (0 to continue): 1

   Enter Block Frequency Test block length: 65536

        P a r a m e t e r   A d j u s t m e n t s
        -----------------------------------------
    [1] Block Frequency Test - block length(M):         65536
    [2] NonOverlapping Template Test - block length(m): 9
    [3] Overlapping Template Test - block length(m):    9
    [4] Approximate Entropy Test - block length(m):     10
    [5] Serial Test - block length(m):                  16
    [6] Linear Complexity Test - block length(M):       500
```

(5) Select input mode and start statistical testing.

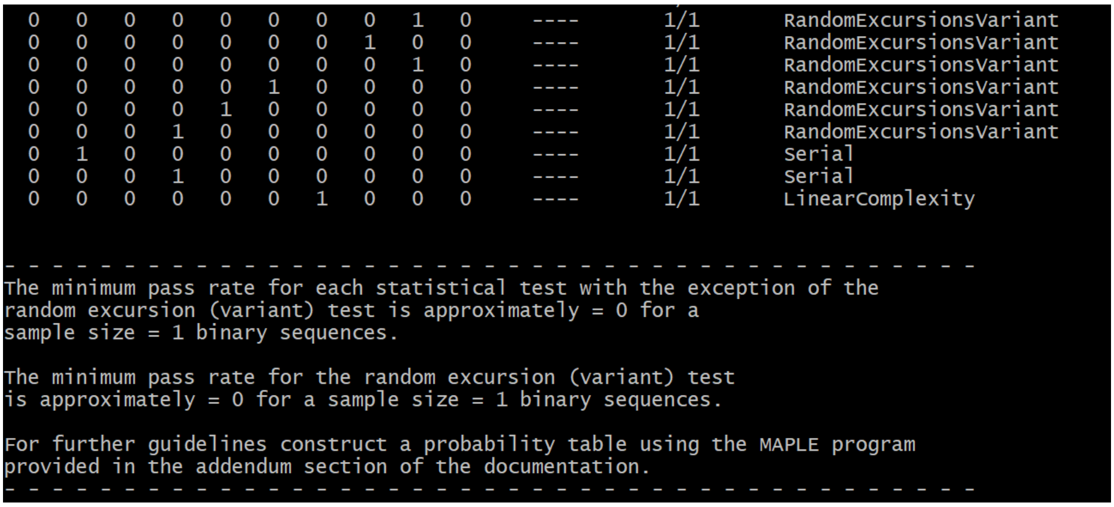


```
Select Test (0 to continue): 0

How many bitstreams? 1

Input File Format:
 [0] ASCII - A sequence of ASCII 0's and 1's
 [1] Binary - Each byte in data file contains 8 bits of data

Select input mode:  1

  Statistical Testing In Progress.........

  Statistical Testing Complete!!!!!!!!!!!!
```

(6) Input *'cat experiments/AlgorithmTesting/finalAnalysisReport.txt'* to see the testing results. This 'random.bin' passed all the tests.



```
yungching@LAPTOP-9H2SD93F /nist/sts-2.1.2/sts-2.1.2
$ cat experiments/AlgorithmTesting/finalAnalysisReport.txt
------------------------------------------------------------------------------
RESULTS FOR THE UNIFORMITY OF P-VALUES AND THE PROPORTION OF PASSING SEQUENCES
------------------------------------------------------------------------------
   generator is <random.bin>
------------------------------------------------------------------------------
 C1  C2  C3  C4  C5  C6  C7  C8  C9 C10  P-VALUE  PROPORTION  STATISTICAL TEST
------------------------------------------------------------------------------
  0   1   0   0   0   0   0   0   0   0    ----       1/1       Frequency
  0   0   0   0   0   0   0   0   0   1    ----       1/1       BlockFrequency
  0   0   1   0   0   0   0   0   0   0    ----       1/1       CumulativeSums
  0   0   1   0   0   0   0   0   0   0    ----       1/1       CumulativeSums
  0   0   0   0   0   0   0   0   1   0    ----       1/1       Runs
  0   0   1   0   0   0   0   0   0   0    ----       1/1       LongestRun
  1   0   0   0   0   0   0   0   0   0    ----       1/1       Rank
  0   0   0   0   0   0   1   0   0   0    ----       1/1       FFT
  0   0   0   0   0   0   1   0   0   0    ----       1/1       NonOverlappingTemplate
  1   0   0   0   0   0   0   0   0   0    ----       1/1       NonOverlappingTemplate
  0   0   0   0   0   1   0   0   0   0    ----       1/1       NonOverlappingTemplate
  0   1   0   0   0   0   0   0   0   0    ----       1/1       NonOverlappingTemplate
  0   0   0   0   1   0   0   0   0   0    ----       1/1       NonOverlappingTemplate
```



```
  0   0   0   0   0   0   0   0   1   0    ----       1/1       RandomExcursionsVariant
  0   0   0   0   0   0   0   1   0   0    ----       1/1       RandomExcursionsVariant
  0   0   0   0   0   0   0   0   1   0    ----       1/1       RandomExcursionsVariant
  0   0   0   0   0   1   0   0   0   0    ----       1/1       RandomExcursionsVariant
  0   0   0   0   1   0   0   0   0   0    ----       1/1       RandomExcursionsVariant
  0   0   0   1   0   0   0   0   0   0    ----       1/1       RandomExcursionsVariant
  0   1   0   0   0   0   0   0   0   0    ----       1/1       Serial
  0   0   0   1   0   0   0   0   0   0    ----       1/1       Serial
  0   0   0   0   0   0   1   0   0   0    ----       1/1       LinearComplexity


- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
The minimum pass rate for each statistical test with the exception of the
random excursion (variant) test is approximately = 0 for a
sample size = 1 binary sequences.

The minimum pass rate for the random excursion (variant) test
is approximately = 0 for a sample size = 1 binary sequences.

For further guidelines construct a probability table using the MAPLE program
provided in the addendum section of the documentation.
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```