作业二：

1. 请解释 SP800-22 中近似熵测试与 SP800-90B 的最小熵估计的异同，标准请自行去 nist 官方网站下载。

   本人查阅了 90B 的文档 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90B.pdf 和 22 的文档 https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-22r1a.pdf 。

   可以发现，22 的近似熵测试只是 15 个检测项中的一项检测，在标准的文档中我们也可以看到其具体的测试过程。

   而对比 90B 的最小熵估计，则是一个整体的策略，针对是否是独立同分布的数据采取不同的策略进行一系列的熵估计，从其中选择最小的熵值作为熵估计值。

   这两者比较的意义我认为不是很大。一个是具体的测试，另一个则是策略。

   但是我们仍然可以从两者的原理、熵值是否存在高估低估等方面进行对比。

   在原理方面，大致可以这么理解近似熵的计算过程。选择一个参数 m，然后计算 m 长的序列的一个评价指标，这个评价指标实际上是把原始序列作为一个 m 长序列的空间，然后计算 m 长序列在该空间的信息熵。然后增加一比特，计算 m+1 长序列在该空间的熵。不难推断，在最理想的情况下，m+1 长序列的熵应该比 m 长序列的熵多 1bit（由于标准中选用的是 ln 函数，所以应是 ln2 nat）。根据我粗浅的理解，我认为或许可以将 m+1 长的熵值减去 m 长的熵值，乘以 n 即可得出熵估计值（当然没有任何的理论依据）。不过标准中构造了一个统计量，对此统计量我暂时也不太理解是如何构造出来的，就不进行分析了。

   而最小熵估计的原理，可以从最简单浅显的例子来理解。这个例子不仅在课件中有，在标准的文档中也出现了。简单的解释一下，就是只考虑 k 个样本中出现概率最大的样本，将其出现的信息熵作为最终的熵值。

   另外，在文档中，最小熵估计还提到对于独立同分布的数据也存在熵被低估的情况。而根据最小熵估计这个名字我们也不难想到，该方法由于总是取用熵值最小的作为结果，所以它极有可能会对大部分数据的熵造成低估。而近似熵估计由于没有这种特点，可能在低估和高估熵值方面没有偏好。

2. 请下载 sp800-22 测试套件，学习使用指南，安装在 linux 或 mac os 机器上，按以下要求测试数据，生成最终报告，并对测试结果的报告作出解释，撰写实验报告。

套件和相关说明链接：

https://csrc.nist.gov/projects/random-bit-generation/documentation-and-software

参数设置：分组长度 100 万比特，分组数目 1000.

测试数据一：测试程序自带的 LCG 随机数生成器

测试数据二：链接：https://pan.baidu.com/s/1rhTVw4TEOprSvqX63UIKZw
提取码：b3fh （注意，测试程序选择的文件格式需要与被测文件一致）

本题目要求我们对测试结果进行解释并撰写实验报告。由于对测试结果的解释将体现在实验报告中，故下文只提供实验报告。

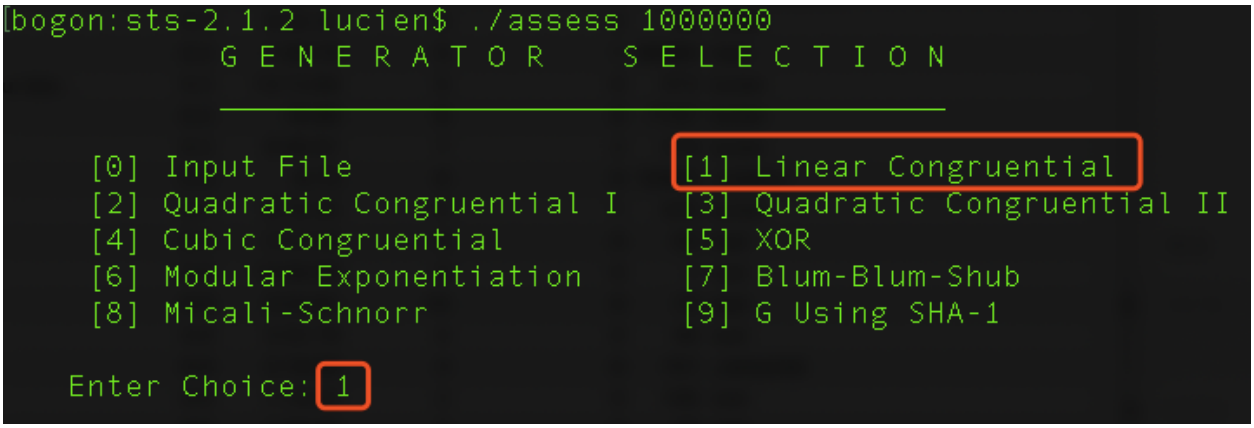
实验报告

实验目的

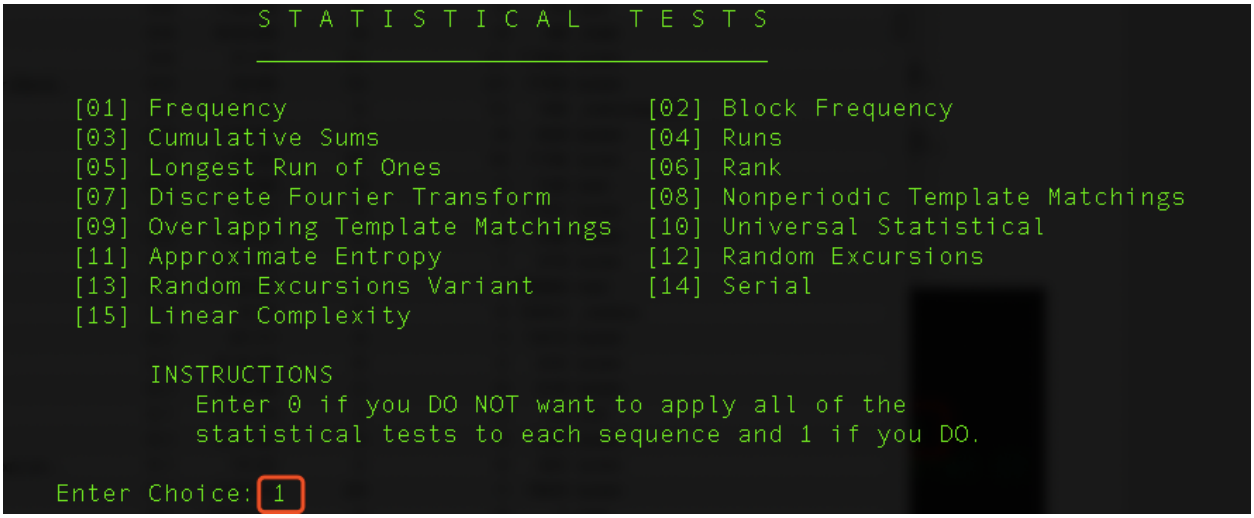
1. 学习使用 sp800-22 测试套件。
2. 学会阅读分析该测试套件生成的测试结果。
3. 撰写实验报告。

实验步骤

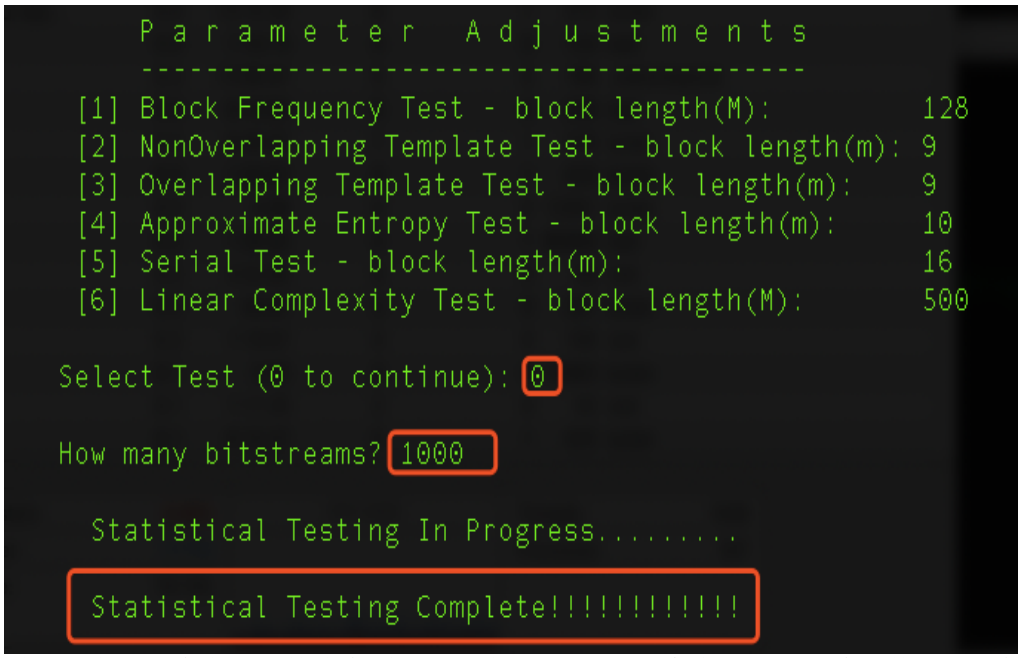
1. 下载测试套件及文档。
2. 阅读文档中第五章 user guide，对软件进行编译。对于 mac os 来说，实际上的操作很简单，只需要在项目的根目录下执行 make 命令，就可以生成可执行文件 access。
3. 对 LCG 随机数发生器生成的随机数以及老师提供的一个测试文件进行测试，得到最终的报告。第 3 步具体言之，如下：
   3.1. 先测试 LCG 随机数发生器，具体如下
   3.2. 执行命令./access 1000000 表示选择分组长度为一百万比特
   3.3. 接着根据提示选择测试的数据是 LCG 产生的随机数



3.4. 接着根据提示选择进行所有 15 项统计测试



3.5. 接着选择不调整参数，并输入分组长度 1000，等待测试完成，大约需要两三个小时



说明，在这一步中，软件将会利用 LCG 生成 1000 组长度为一百万比特的数据，对每组数据进行上述统计测试。

3.6. 对老师提供的文件进行测试，具体如下

3.7. 大体步骤和之前一致。其中有几个需要注意的地方。

首先，由于测试文件有 130 多兆，据说可能会造成内存不足的情况。所以需要对文件进行分组测试。在本次测试当中，我选择只测试前 128 兆的数据，分 128 次测试，及一组为 1 兆字节（为 8388608 比特），分组数为 128.那么需要执行的命令是./access 8388608，并且在程序提示“How many bitstreams?”时输入 128。

其次，需要将文件格式选为 Binary，因为当尝试用 vim 打开该文件时，发现其内容并不是 ascii 文本，而是一堆乱码。

3.8. 测试完成之后，便可以在文件夹中找到生成的报告。关于报告，不仅有每项测试的具体报告，最后还会有一个总的报告。

4. 实验结果分析

4.1. 首先对 LCG 的实验结果进行分析。

打开 finalAnalysisReport.txt。我们可以看到如下内容（开头几行）。

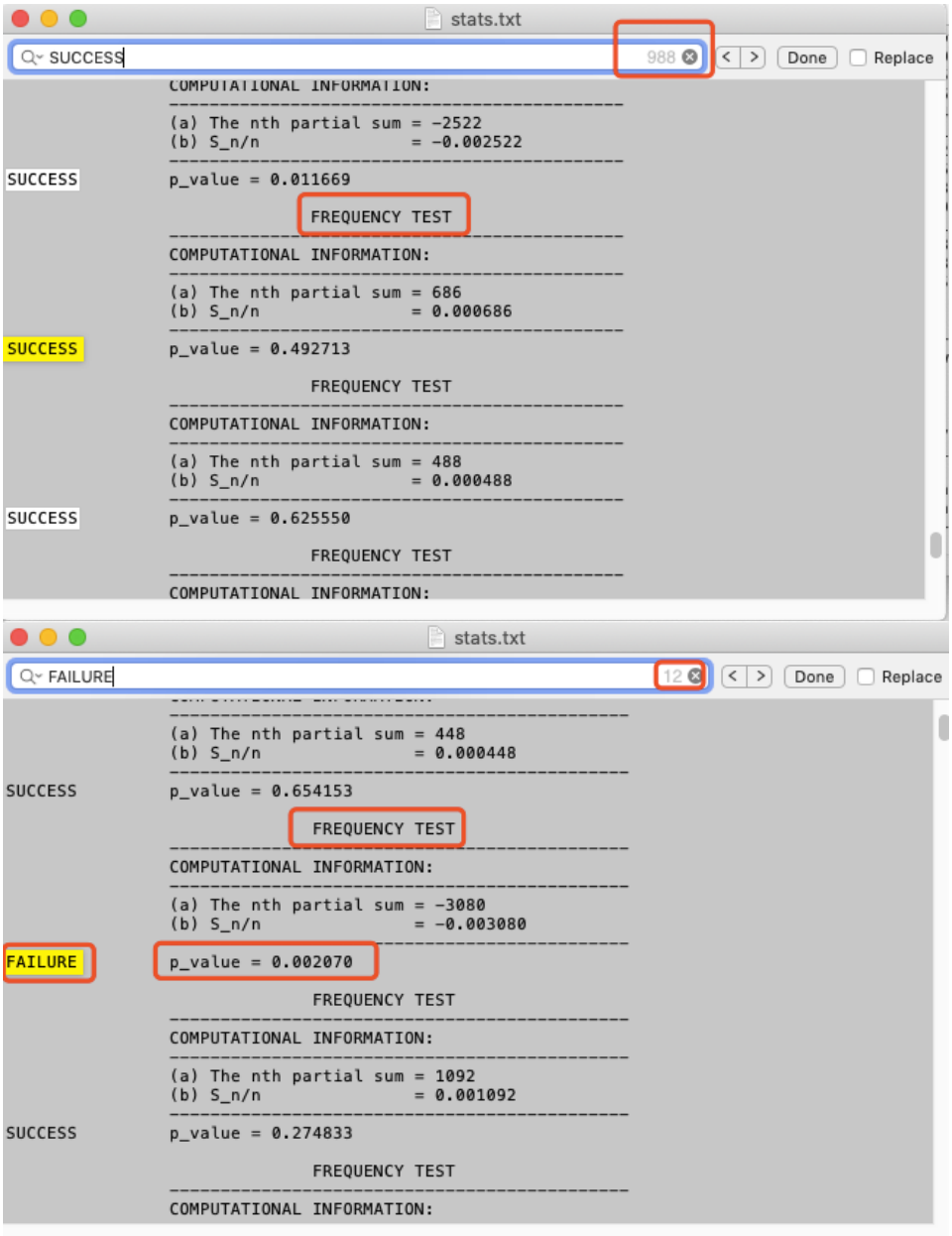
```
------------------------------------------------------------------------------
RESULTS FOR THE UNIFORMITY OF P-VALUES AND THE PROPORTION OF PASSING SEQUENCES
------------------------------------------------------------------------------
   generator is <Linear-Congruential>
------------------------------------------------------------------------------
 C1  C2  C3  C4  C5  C6  C7  C8  C9 C10  P-VALUE  PROPORTION  STATISTICAL TEST
------------------------------------------------------------------------------
 98  89 105 111 103  89 106  96 104  99  0.859637   988/1000    Frequency
 96  92  96  90 105  98 102 125  91 105  0.383827   986/1000    BlockFrequency
 93 116  96  88  99  91 109 112 107  89  0.399442   991/1000    CumulativeSums
```

可以看到第二行文字会提示我们测试的随机数来源。

之后便是一个表格，表格的前十列分别是 C1 到 C10，那么这十个数据分别代表什么呢？我们知道，P-VALUE 的取值范围为区间 0 到 1。将这个区间进行 10 等分，分别统计 1000 组测试中计算得到的 P-VALUE 落在各个区间的组数，就是 C1 到 C10 的含义。我们将 C1 到 C10 的数据加起来，刚好等于 1000。

那么第 11 列的 P-VALUE 又是什么含义呢？它其实是对 C1-C10 这十个数的均匀性的衡量，是利用卡方分布进行统计的。而 P-VALUE 的值大于 0.01 即可认为其实均匀分布的，并且这个值越大越好。

接下来的一列是 PROPORTION,它代表着 1000 组数据中 P-VALUE 值大于 0.01 的组数与 1000 的比值。其中 P-VALUE 值小于 0.01 就被认为是未通过该次测试。这一点我们可以到具体某项测试的结果里证实。例如第一行进行的是频率测试，我们找到频率测试的结果，搜索字符串 SUCCESS 和 FAILURE 出现的次数。会发现刚好是 988 次和 12 次.如下图所示，同时我们也可以看到显示为 FAILURE 的测试中 P-VALUE 值也确实是低于 0.01 的。



第 13 列则提示了每一行对应的是什么测试。

而 15 项测试中，大部分的测试只占了统计结果的一行，而 NonOverlappingTemplate，RandomExcursions，RandomExcursionsVariant，Serial 这四项测试占据了多行。这又是为什么呢？

其中 NonOverlappingTemplate 这项测试可能因为包含了 148 个 Template 所以有 148 行结果。

而对于 RandomExcursions 这类 test 还有一个异常的地方，就是它们的通过比例的分母并不是 1000，而是比 1000 更小的数，通过在网上查阅资料（https://crypto.stackexchange.com/questions/71635/nist-random-excursion-results），发现一个说法是其需要的数据量更大，进行一次测试需要的数据为一千万比特，而每组一兆比特显然还没到一千万比特的量级，所以分母会小于 1000.

同时因为这个测试与其他测试的区别，导致了最后的文字总结中也出现了相应的分类讨论。如下：

```
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
    The minimum pass rate for each statistical test with the exception of the
    random excursion (variant) test is approximately = 980 for a
    sample size = 1000 binary sequences.

    The minimum pass rate for the random excursion (variant) test
    is approximately = 593 for a sample size = 607 binary sequences.

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

在这段文字总结中，它提到出了 random excursion (variant) test 之外，1000 组数据通过测试的次数至少为 980，而单独考虑 random excursion (variant) test，607 组数据的通过次数至少为 593。通过简单的分析，我们可以发现这两个数值略均低于在实际测试中出现的最小值。

而对于老师提供的文件也可以进行上述分析，其中有两个异常的地方是

C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	P-VALUE		PROPORTION		STATISTICAL TEST
14	16	12	12	15	6	15	13	12	13	0.706149		127/128		Frequency
13	12	9	10	13	13	10	16	13	19	0.619772		128/128		BlockFrequency
15	11	13	15	6	16	7	14	10	21	0.078086		127/128		CumulativeSums
12	16	15	10	7	14	21	8	12	13	0.162606		127/128		CumulativeSums
14	14	11	12	14	7	14	16	15	11	0.772760		127/128		Runs
13	19	8	17	7	7	24	10	9	14	0.002792		126/128		LongestRun
34	18	12	15	9	11	5	10	9	5	0.000000	*	121/128	*	Rank

120 7 0 1 0 0 0 0 0 0 0.000000 * 30/128 * LinearComplexity

也就是说测试数据二没有通过两个测试，分别是 Rank 测试和 LinearComplexity 测试。