Cryptography - Part 3

March 18, 2025

Recap question:

March 18, 2025

You want to encrypt the message

`GoTigers`

using a one-time pad. Explain why either of the following keys would be a bad choice:

a) 7,8,4
b) 1,1,1,1,1,1,1,1

Recap question:

March 18, 2025

Recall that a one-time pad encrypts a message by shifting each letter of the message by the corresponding key value.

The key in a) is too short, so not every letter gets a shift. The key in b) shifts every letter by one, so it becomes a Caesar cipher with shift 1, which is easy to break (for instance by frequency analysis).

Cryptography - Part 3

March 18, 2025

By the end of this lecture, you will be able to:

1. Define and give examples of pseudo-random number generators
2. Define and give examples of true random number generators
3. Give examples of tests used to determine if a sequence of numbers is truly random

Random numbers

One-time pad encryption is unbreakable if the key:

1. is at least as long as the message ✓
2. is generated randomly
3. is kept secret ✓
4. is never reused ✓

Random numbers

One-time pad encryption is unbreakable if the key:

1. is at least as long as the message ✓
2. is generated randomly
3. is kept secret ✓
4. is never reused ✓

Random numbers

Why is this important?

Example 1: ANC used one-time pads when building a resistance network against the apartheid regime in South Africa, with one-time pads smuggled by a Belgian airline stewardess

Random numbers

Why is this important?

Example 1: ANC used one-time pads when building a resistance network against the apartheid regime in South Africa, with one-time pads smuggled by a Belgian airline stewardess

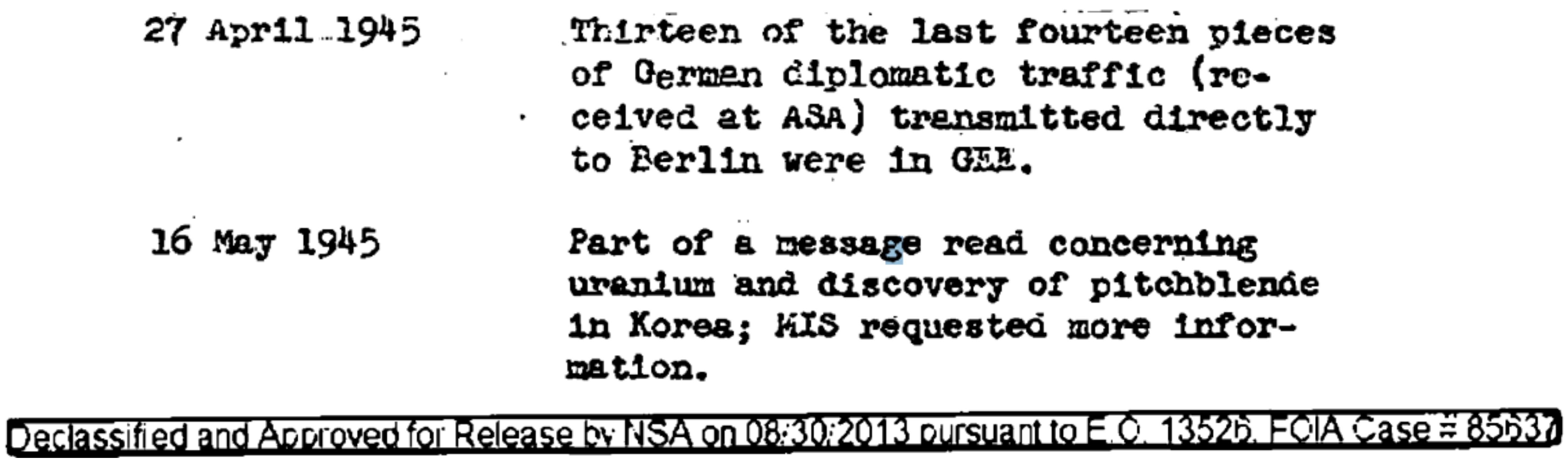
Example 2: Many programs can suggest randomly generated passwords, e.g., Google chrome. How do we make sure they are secure?

Random numbers

Example 3: In WW2, German one-time pads were generated by an Enigma-like machine ("GEE") that produced keys that were not completely random.

Random numbers

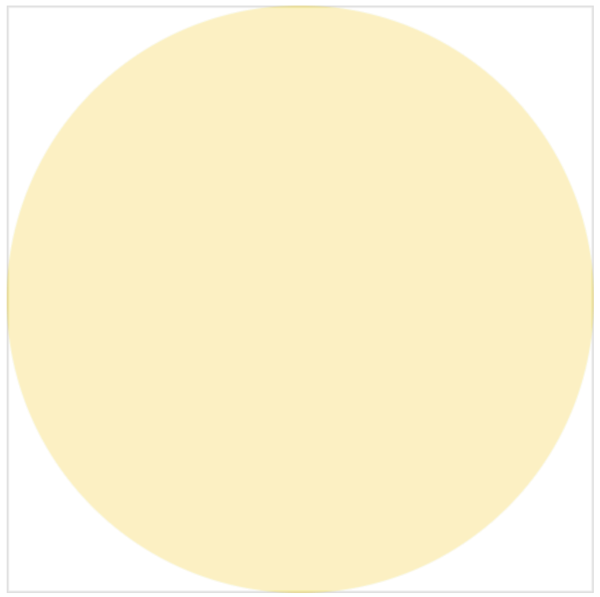
Example 3: In WW2, German one-time pads were generated by an Enigma-like machine (“GEE”) that produced keys that were not completely random. This produced patterns in the encrypted text which allowed US cryptographers to read most diplomatic traffic between Berlin and Tokyo.



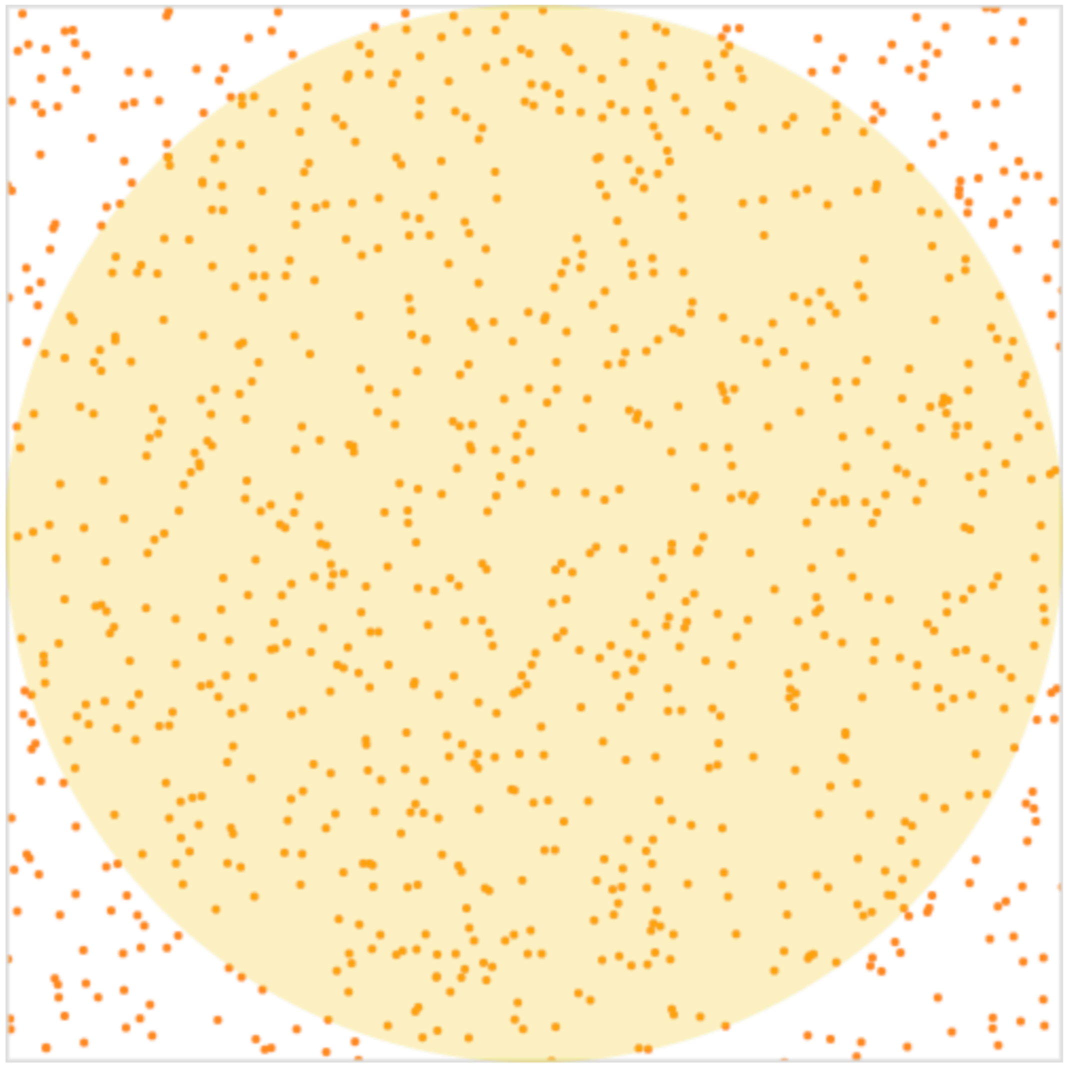
27 April 1945	Thirteen of the last fourteen pieces of German diplomatic traffic (received at ASA) transmitted directly to Berlin were in GEE.
16 May 1945	Part of a message read concerning uranium and discovery of pitchblende in Korea; MIS requested more information.

Example 4: Simulation of real-world phenomena. A bank that operates a network of ATM machines may wish to stress-test its software by simulating the actions of customers accessing their accounts at random times through random machines.

Example 5: Approximations to problems that may be too difficult to solve exactly. Estimate the value of π by uniformly throwing darts on a square containing an inscribed circle.



$$S_{\text{circle}} = \pi r^2$$
$$S_{\text{square}} = 4r^2$$
$$\pi = 4S_{\text{circle}}/S_{\text{square}}$$



m: # of samples in the circle

n: # of samples dropped

m = 798, n = 1000

$$\hat{\pi} = \frac{m}{n} = 3.192$$

The generation of random numbers is too important to be left to chance.

—Robert Coveyou

Random numbers

What does “random” mean?

1. Uniform distribution: each possibility is equally likely

Example:

{2, 3, 5, 1, 4, 4, 1, 1, 6, 6, …}

Random numbers

What does “random” mean?

1. Uniform distribution: each possibility is equally likely

2. Random numbers should be unpredictable.
   1,2,3,4,5,6 is not a good sequence, just like p,a,s,s,w,o,r,d is not a good sequence when generating passwords

How to generate random numbers?

1. Use a complex machine like GEE?
2. Throw dice (or similar)?
3. Ask people to come up with a number between 0 and 25?
4. Use a computer?

How to generate random numbers?

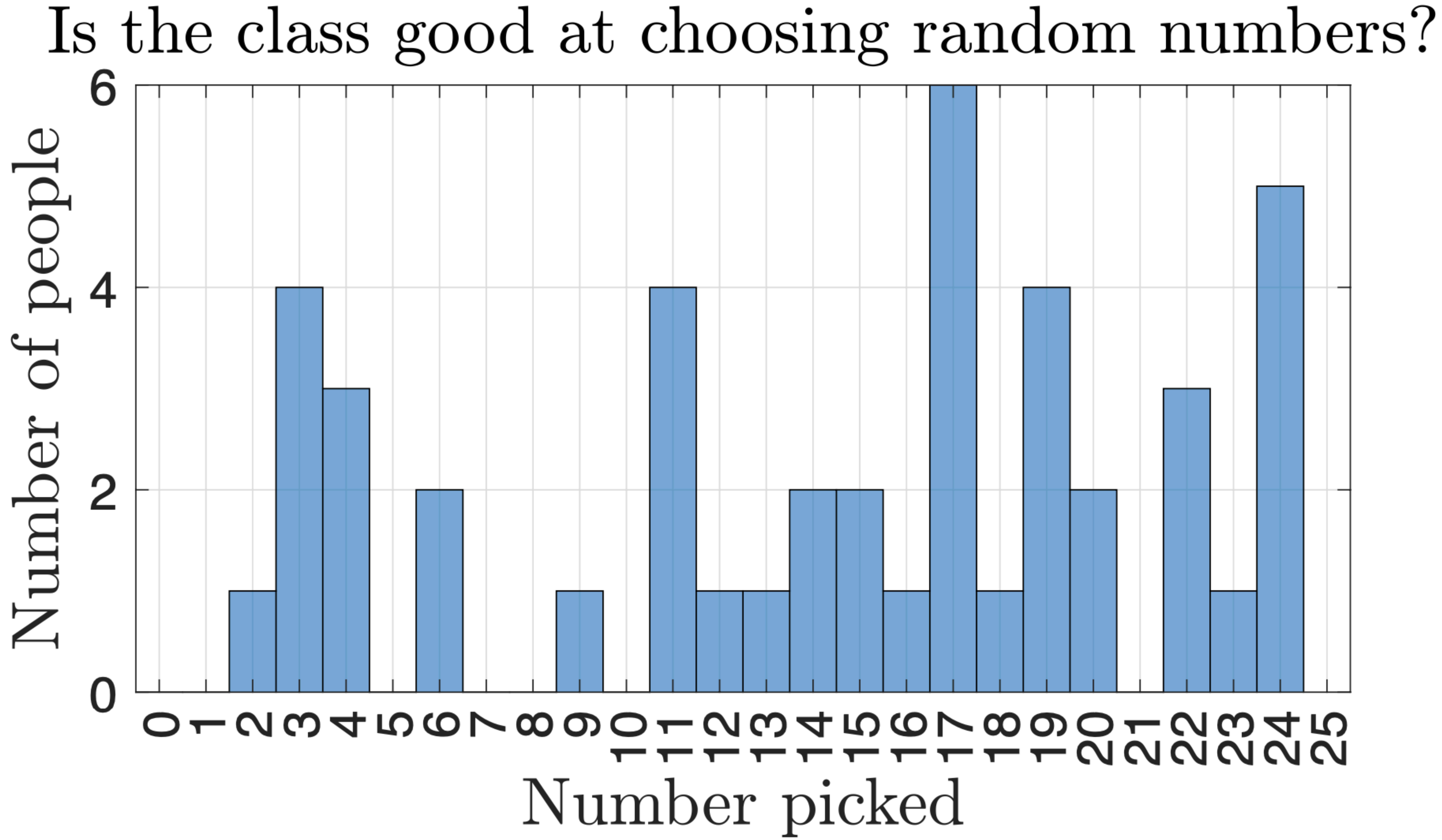
creates patterns $\longrightarrow$ ✗

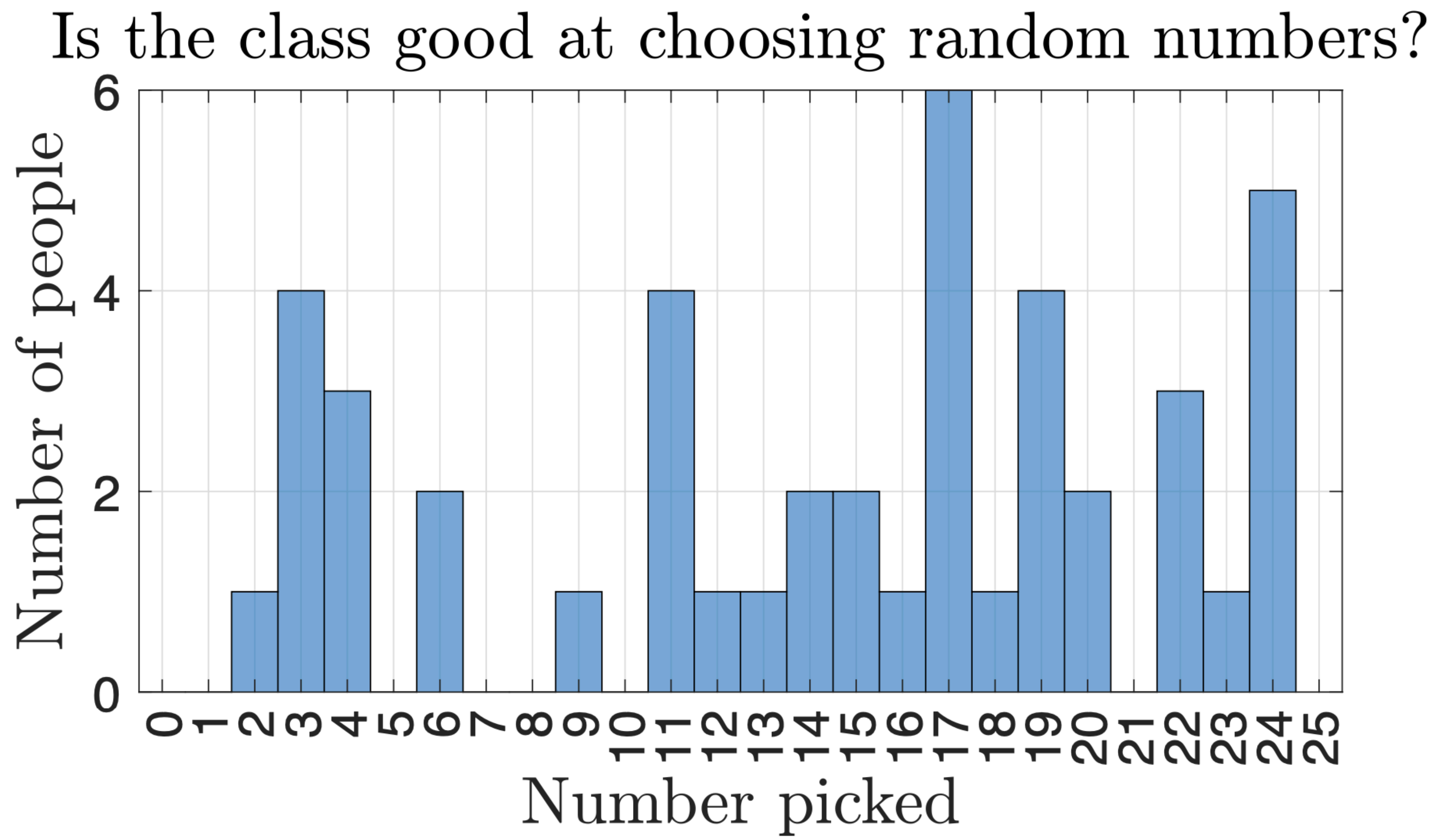
1. Use a complex machine like GEE? ✗
2. Throw dice (or similar)? slow $\longrightarrow$ ✗
3. Ask people to come up with a number between 0 and 25?
4. Use a computer?

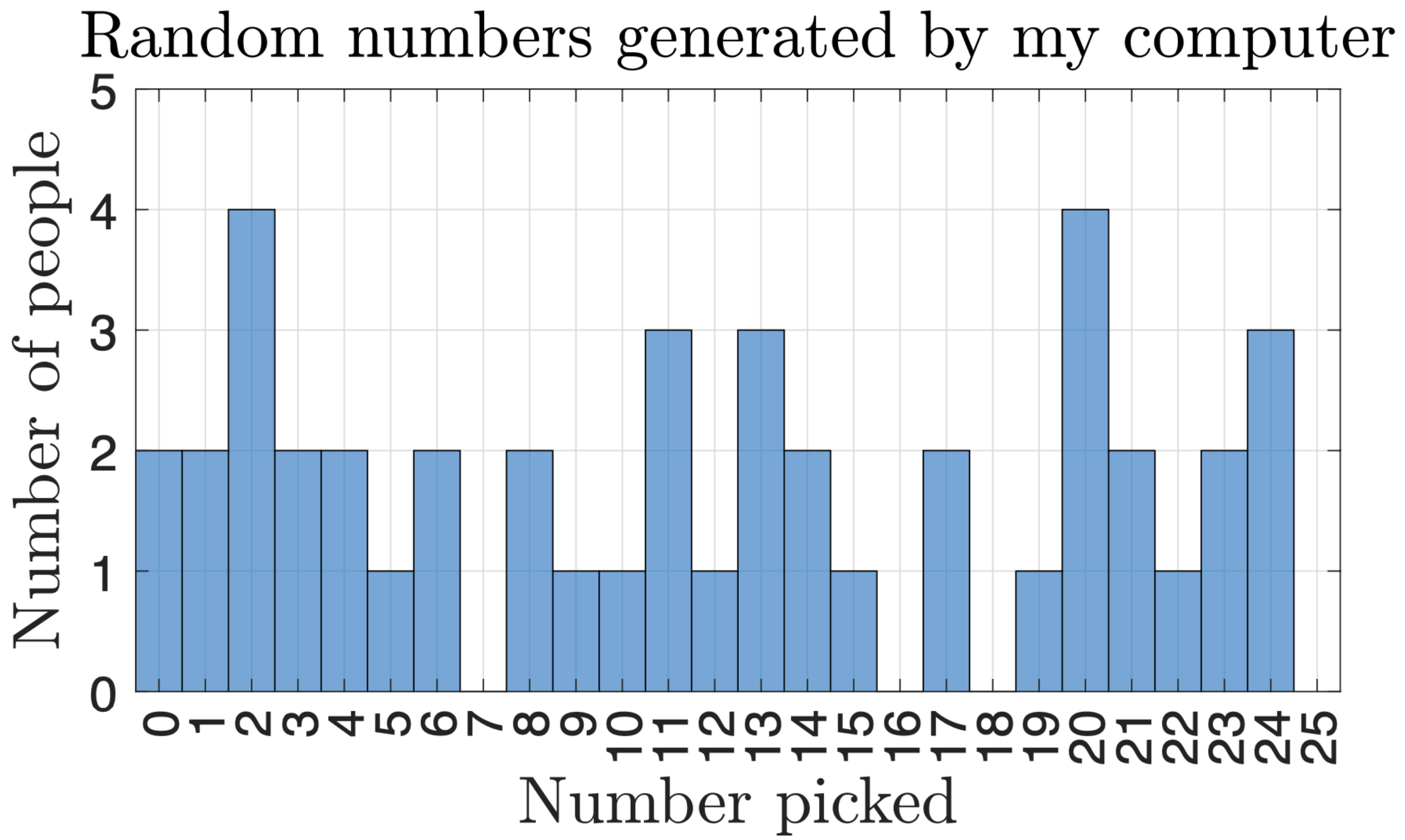
How to generate random numbers?

Is the class good at choosing random numbers?

How to generate random numbers?




Is the class good at choosing random numbers?
Number of people
Number picked
0
2
4
6
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25


Random numbers generated by my computer
Number of people
Number picked
0
1
2
3
4
5
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

How to generate random numbers?

How do we test if this is a random sequence or not?

How to generate random numbers?

How do we test if this is a random sequence or not?

Hypothesis testing!

How to generate random numbers?

How do we test if this is a random sequence or not?

Hypothesis testing!

Example: If the numbers are random, there should be 50% chance of getting a number smaller than 12.5.

How to generate random numbers?

How do we test if this is a random sequence or not?

Hypothesis testing!

Example: If the numbers are random, there should be 50% chance of getting a number smaller than 12.5.

H_0 : the chance of getting a number < 12.5 is 0.5

How to generate random numbers?

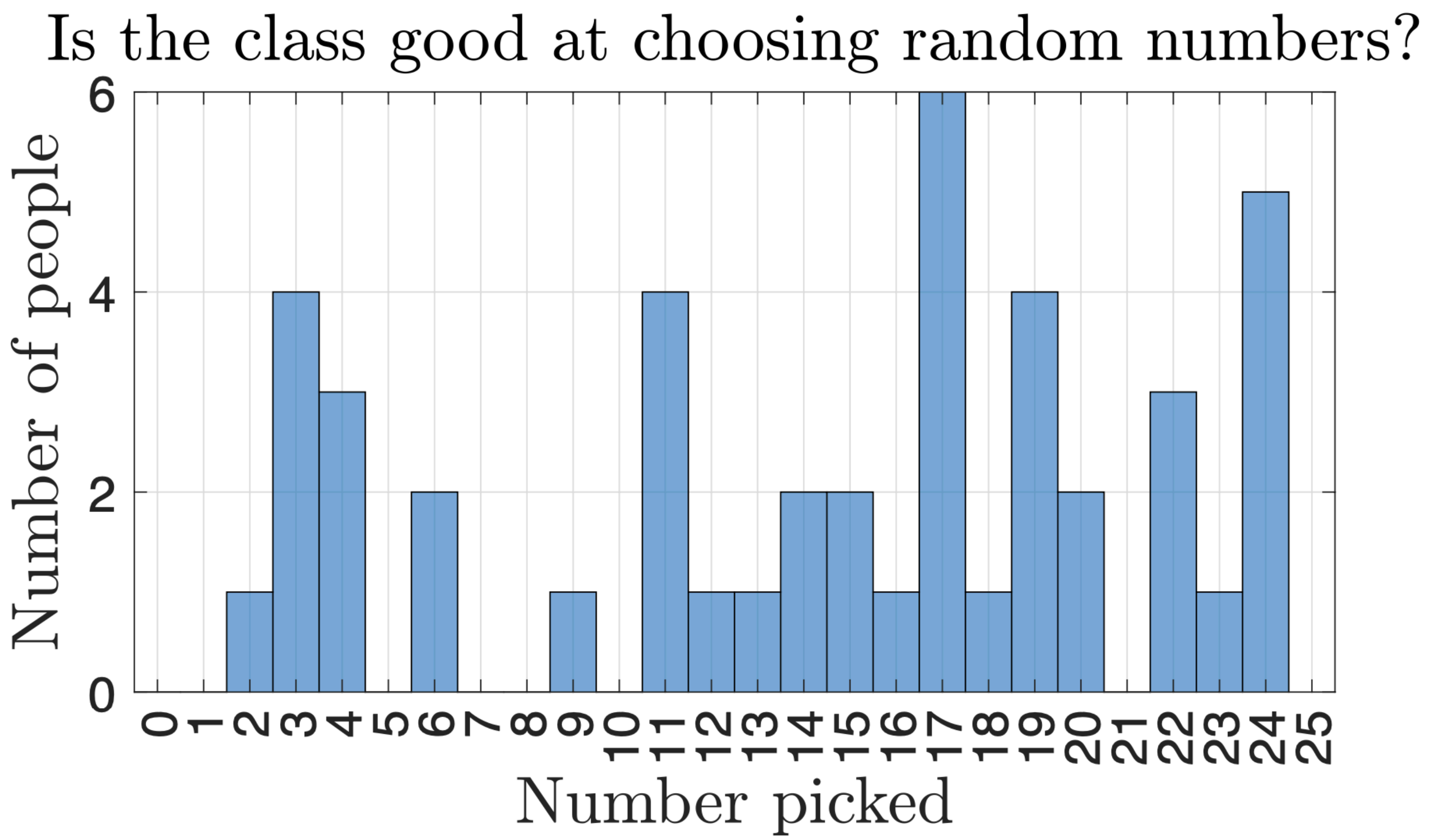
How do we test if this is a random sequence or not?

Hypothesis testing!

Example: If the numbers are random, there should be 50% chance of getting a number smaller than 12.5.

H_0 : the chance of getting a number < 12.5 is 0.5

Compute the 95% confidence interval and check if the data falls inside it!

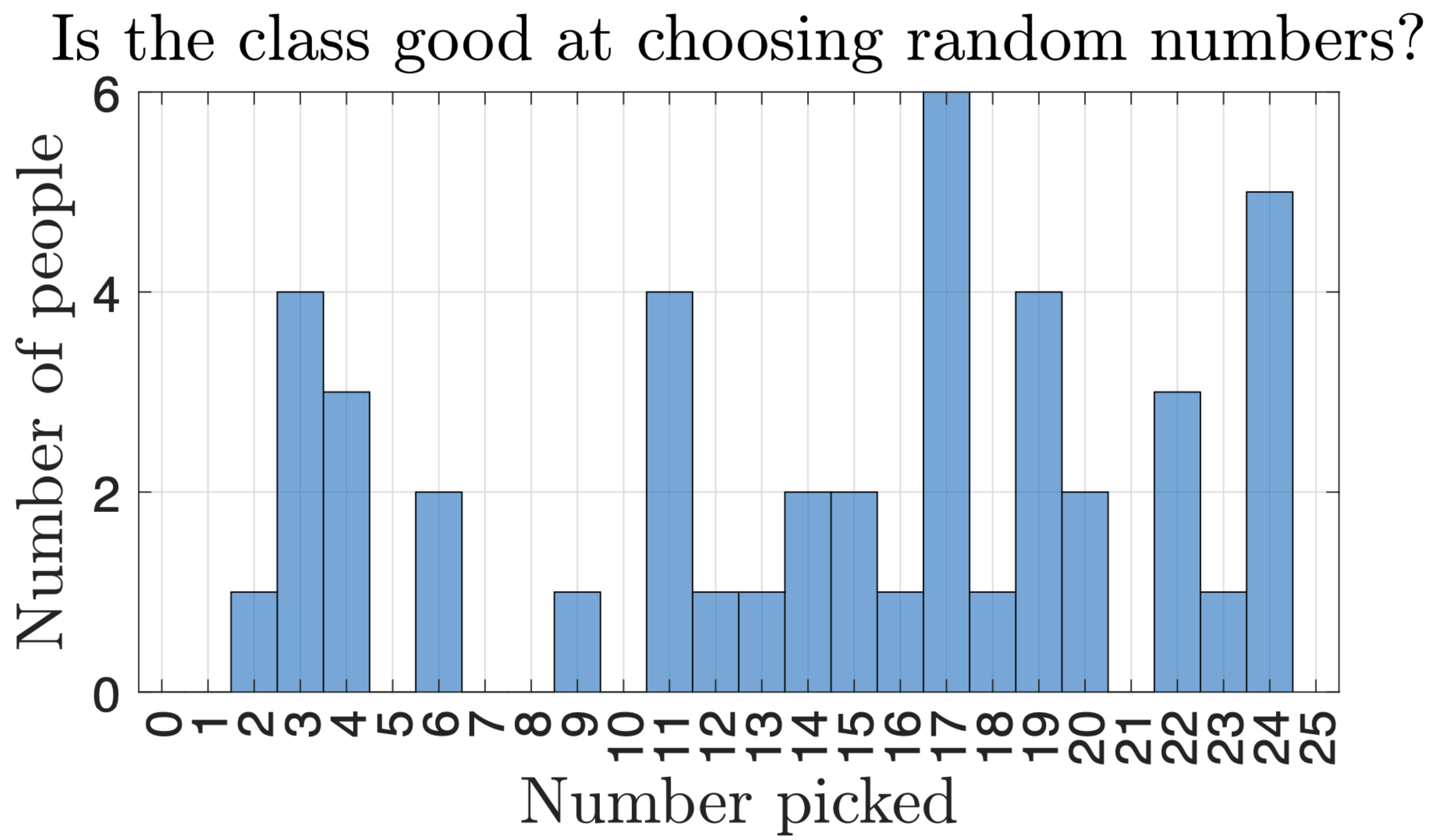


H_0 : the chance of getting a number < 12.5 is 0.5

$s =$ ______________________________

The confidence interval is ______________________________

The data $\hat{p} =$ ___________

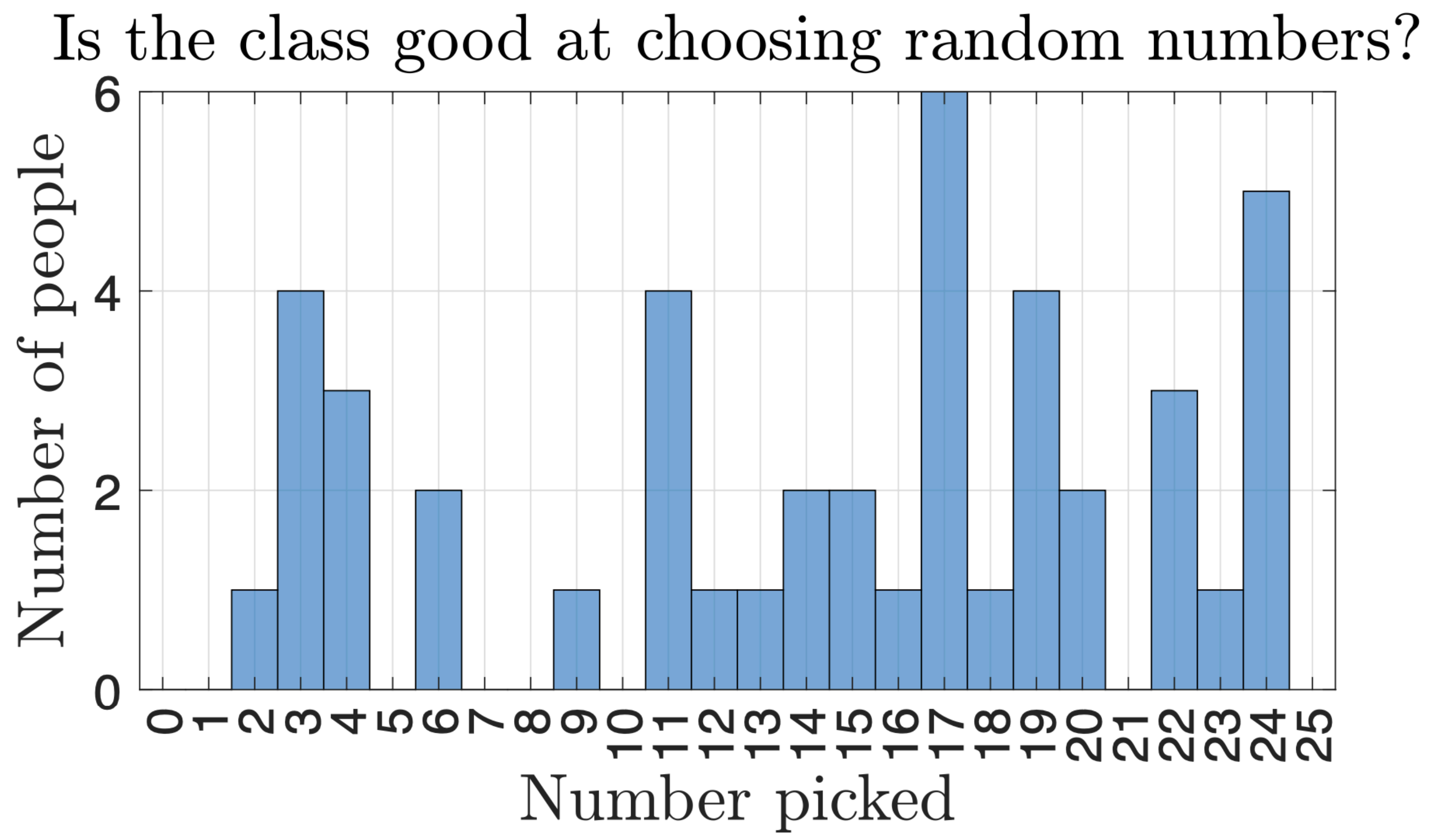


H_0 : the chance of getting a number < 12.5 is 0.5

$$s = \sqrt{p \times (1-p)/n} =$$

The confidence interval is ______________________

The data $\hat{p} =$ __________

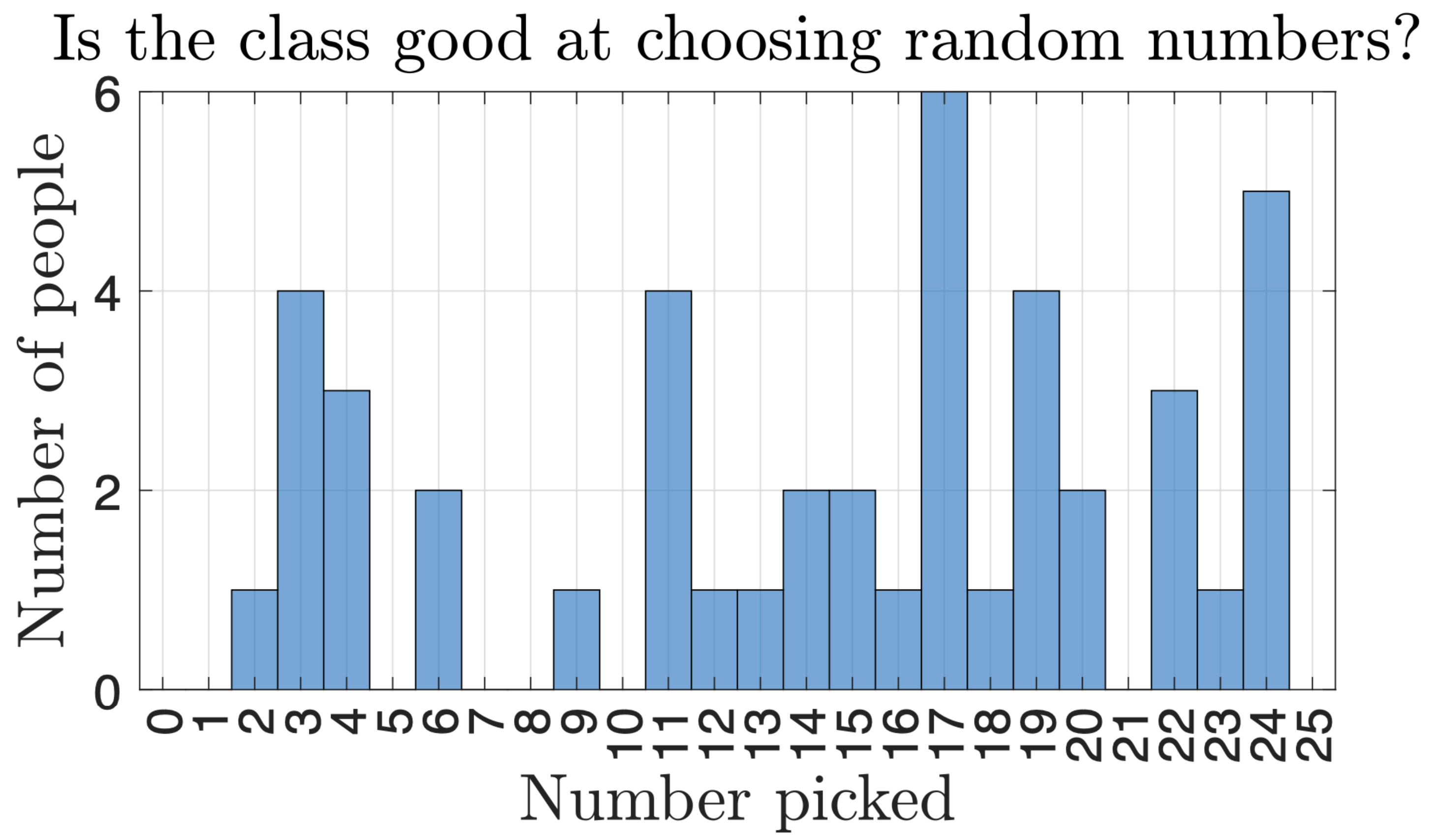


H_0 : the chance of getting a number < 12.5 is 0.5

$$s = \sqrt{p(1-p)/n} = \sqrt{0.5 \times 0.5/44} \approx 0.075$$

The confidence interval is ______________________________

The data $\hat{p} =$ __________

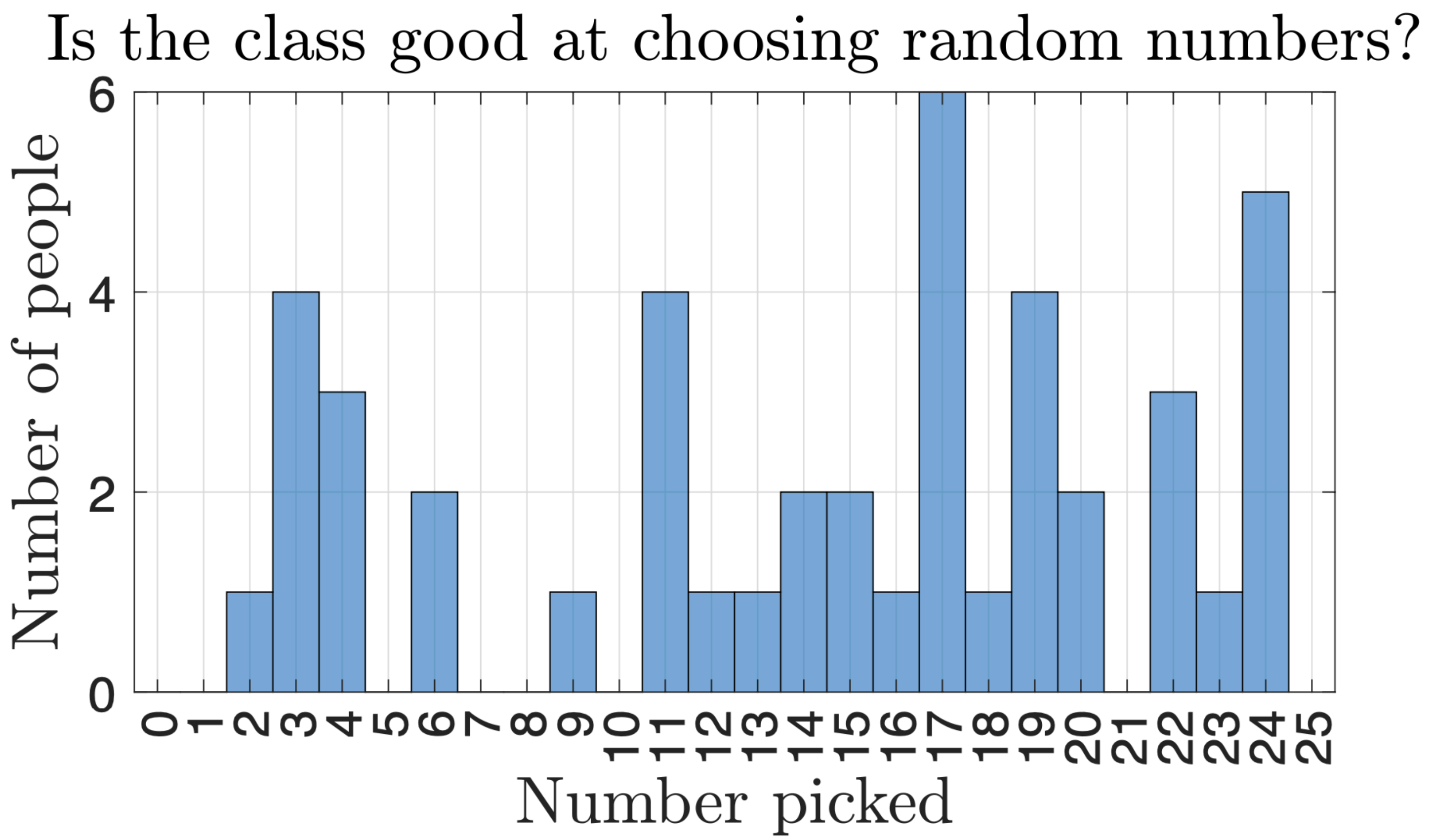


H_0 : the chance of getting a number < 12.5 is 0.5

$$s = \sqrt{p(1-p)/n} = \sqrt{0.5 \times 0.5/44} \approx 0.075$$

The confidence interval is $(0.5 - 2s, 0.5 + 2s) = (0.35, 0.65)$

The data $\hat{p} =$ ________

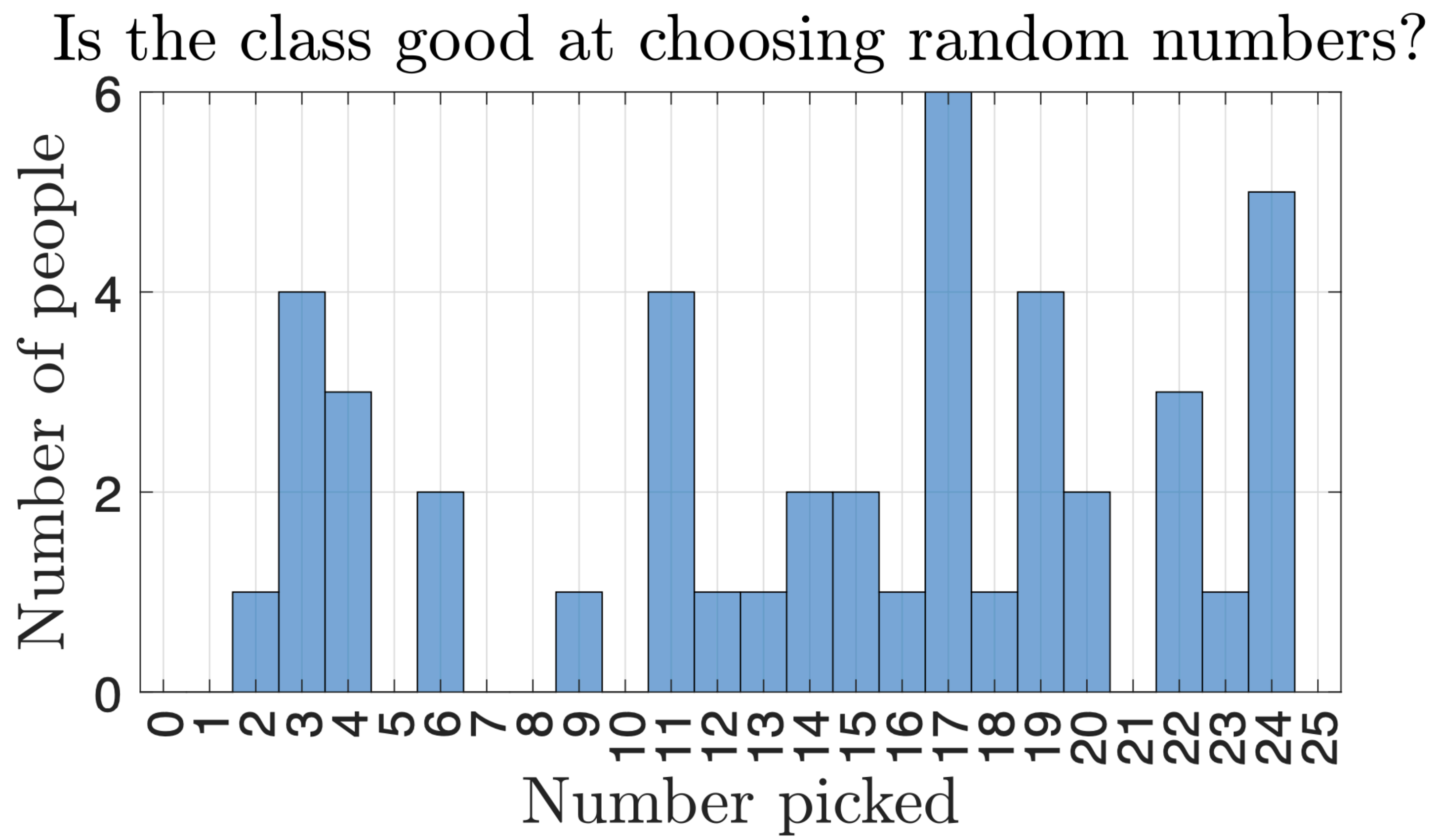


H_0 : the chance of getting a number < 12.5 is 0.5

$$s = \sqrt{p(1-p)/n} = \sqrt{0.5 \times 0.5/44} \approx 0.075$$

The confidence interval is $(0.5 - 2s, 0.5 + 2s) = (0.35, 0.65)$

The data $\hat{p} = 16/44 \approx 0.36$



H_0 : the chance of getting a number < 12.5 is 0.5

$$s = \sqrt{p(1-p)/n} = \sqrt{0.5 \times 0.5/44} \approx 0.075$$

The confidence interval is $(0.5 - 2s, 0.5 + 2s) = (0.35, 0.65)$

The data $\hat{p} = 16/44 \approx 0.36$

falls inside the confidence interval so we **cannot** reject the null hypothesis, i.e., we **cannot** from this test conclude that the numbers are not random.

How to generate random numbers?

We could do more tests than seeing if 50% of picked numbers are lower than the midpoint, for instance test if 50% of picked numbers are even.

How to generate random numbers?

We could do more tests than seeing if 50% of picked numbers are lower than the midpoint, for instance test if 50% of picked numbers are even.

If a sequence of numbers passes a number of these tests, then it can be said to behave like random numbers.

How to generate random numbers?

We could do more tests than seeing if 50% of picked numbers are lower than the midpoint, for instance test if 50% of picked numbers are even.

If a sequence of numbers passes a number of these tests, then it can be said to behave like random numbers.

The mathematical community has a number of collections of tests that can be used, see for instance

https://en.wikipedia.org/wiki/Diehard_tests

Pseudo-random Number Generators (PRNGs)

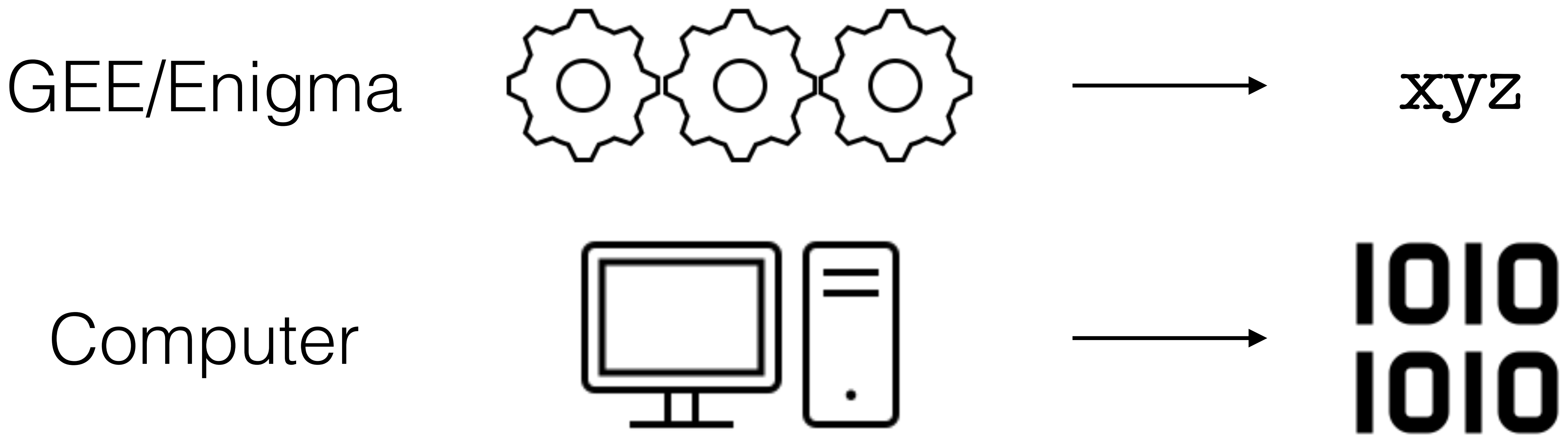
Many applications require generating random numbers efficiently on a computer.

Pseudo-random Number Generators (PRNGs)

Many applications require generating random numbers efficiently on a computer. This is not truly random, because the settings of a computer determine the output deterministically (i.e., not randomly).

Pseudo-random Number Generators (PRNGs)

Many applications require generating random numbers efficiently on a computer. This is not truly random, because the settings of a computer determine the output deterministically (i.e., not randomly).



Pseudo-random Number Generators (PRNGs)

Still, there are algorithms (PRNGs) that generate numbers that behave very close to being random. These numbers should be

Pseudo-random Number Generators (PRNGs)

Still, there are algorithms (PRNGs) that generate numbers that behave very close to being random. These numbers should be

1. Approximately uniform
2. Hard to predict: All PRNGs will repeat eventually, while a good one does not for a very long time

Pseudo-random Number Generators (PRNGs)

Still, there are algorithms (PRNGs) that generate numbers that behave very close to being random. These numbers should be

1. Approximately uniform
2. Hard to predict: All PRNGs will repeat eventually, while a good one does not for a very long time

If someone could predict the output of the PRNG, they could predict the key and break in!

If someone could predict the output of the PRNG, they could predict the key and break in!

If someone could predict the output of the PRNG, they could predict the key and break in!

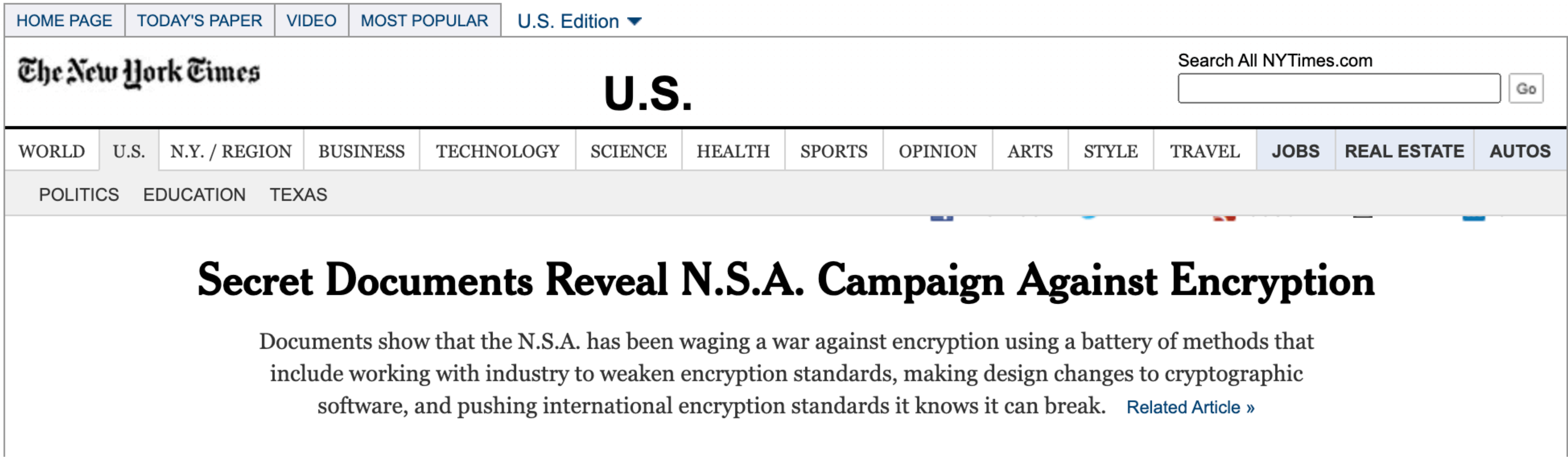
Example 1: In 2010, a U.S. lottery draw was rigged by the information security director of the Multi-State Lottery Association. He programmed a flaw into the PRNG used by the lottery which made the numbers not truly random.

If someone could predict the output of the PRNG, they could predict the key and break in!

Example 1: In 2010, a U.S. lottery draw was rigged by the information security director of the Multi-State Lottery Association. He programmed a flaw into the PRNG used by the lottery which made the numbers not truly random. He was then able to predict the correct numbers a few times in a year. He won a total of $16,500,000.

If someone could predict the output of the PRNG, they could predict the key and break in!

Example 2: NSA funds covert program to build secret flaws into PRNGs that they (and only they) can use to predict the output and therefore the key



Secret Documents Reveal N.S.A. Campaign Against Encryption

Documents show that the N.S.A. has been waging a war against encryption using a battery of methods that include working with industry to weaken encryption standards, making design changes to cryptographic software, and pushing international encryption standards it knows it can break. Related Article »

New York Times, Sept. 5, 2013

Example of a PRNG

Say that we want to randomly generate an 8-digit pseudo-random number. We can do the following procedure:

1. Start with an 8-digit number, say $X_0 = 35385906$
2. Square it to get $X_0^2 = 1252162343440836$
3. Let X_1 be the 8 middle digits of X_0^2 , i.e.,

$$X_1 = 16234344$$

4. Continue in the same way by squaring X_1 and keeping the 8 middle digits

Example of a PRNG

Continuing in the same way by squaring and keeping the 8 middle digits gives the numbers:

$$X_0 = 35385906$$
$$X_1 = 16234344$$
$$X_2 = 55392511$$
$$X_3 = 33027488$$
$$X_4 = 81496359$$

$$X_5 = 65653025$$
$$X_6 = 31969165$$
$$X_7 = 02751079$$
$$X_8 = 56843566$$
$$X_9 = 19099559$$
$$X_{10} = 79315399$$

Example of a PRNG

$$X_0 = 35385906$$
$$X_1 = 16234344$$
$$X_2 = 55392511$$
$$X_3 = 33027488$$
$$X_4 = 81496359$$
$$X_5 = 65653025$$
$$X_6 = 31969165$$
$$X_7 = 02751079$$
$$X_8 = 56843566$$
$$X_9 = 19099559$$
$$X_{10} = 79315399$$

The numbers look pretty random, but the PRNG is flawed. For some starting numbers X_0 the sequence of numbers will start repeating after only 100 numbers, so we can only use this to generate 100 keys/passwords.

Example of a PRNG

$$X_0 = 35385906$$
$$X_1 = 16234344$$
$$X_2 = 55392511$$
$$X_3 = 33027488$$
$$X_4 = 81496359$$

$$X_5 = 65653025$$
$$X_6 = 31969165$$
$$X_7 = 02751079$$
$$X_8 = 56843566$$
$$X_9 = 19099559$$
$$X_{10} = 79315399$$

The numbers look pretty random, but the PRNG is flawed. For some starting numbers X_0 the sequence of numbers will start repeating after only 100 numbers, so we can only use this to generate 100 keys/passwords.

The terminology for this is that this PRNG has period 100.

Example of a PRNG

In 1997, Makoto Matsumoto and Takuji Nishimura found a new random number generator, which they named the Mersenne Twister, with the phenomenally large period

$$2^{19937} - 1 \approx 10^{6001}$$

Seeding the PRNGs

Sometimes it's useful to be able to repeat the program exactly, with the same sequence of random numbers.

1. Reproducible simulations (like when throwing darts for π)
2. Cryptography: sender and receiver might need the same numbers

Even when two seeds are only slightly different, a good PRNG would produce two completely different random number sequences.

Example: Generating 10 random integers in (0, 20)

Seed = 42:

{7, 15, 11, 8, 7, 19, 11, 11, 4, 8}

Seed = 43:

{5, 1, 18, 17, 18, 3, 15, 1, 4, 19}

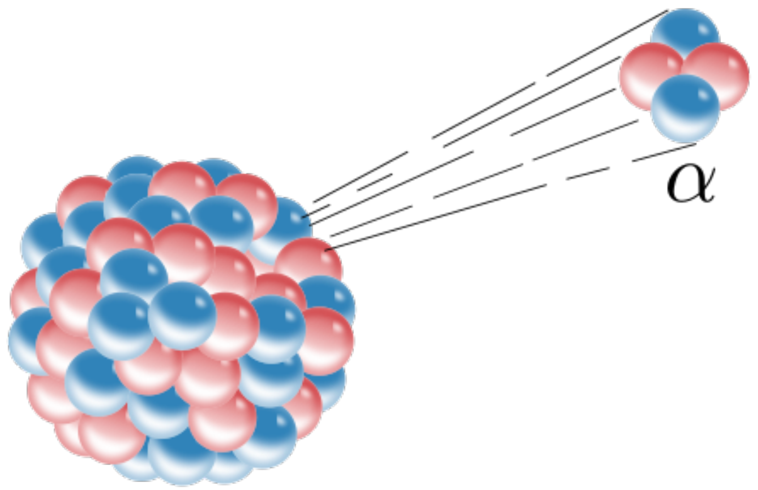
Seed = 42:

{7, 15, 11, 8, 7, 19, 11, 11, 4, 8}

True Random Number Generators (TRNGs)

A TRNG measures some physical phenomenon that is truly random.

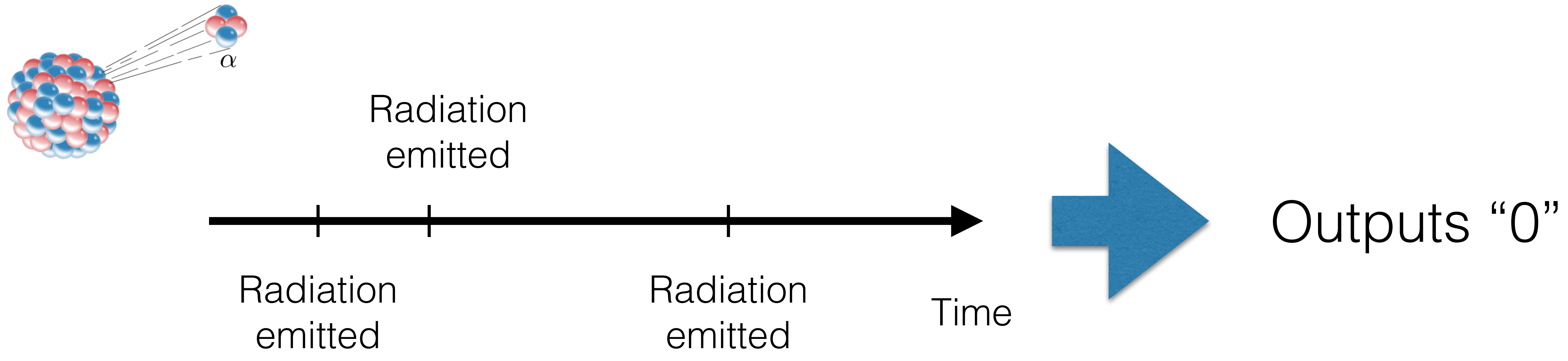
Example 1:



Radioactive decays are completely unpredictable, and they can be detected quite easily and fed into a computer

Record the time between two decays and compare to the time between the next two. If the first time is shorter than the second, output a “0”. If the first time is longer than the second, output a “1”. This produces a sequence of random bits (0’s and 1’s), which can be concatenated to form random integers.

Record the time between two decays and compare to the time between the next two. If the first time is shorter than the second, output a “0”. If the first time is longer than the second, output a “1”. This produces a sequence of random bits (0’s and 1’s), which can be concatenated to form random integers.



The HotBits service (http://www.fourmilab.ch/hotbits/)

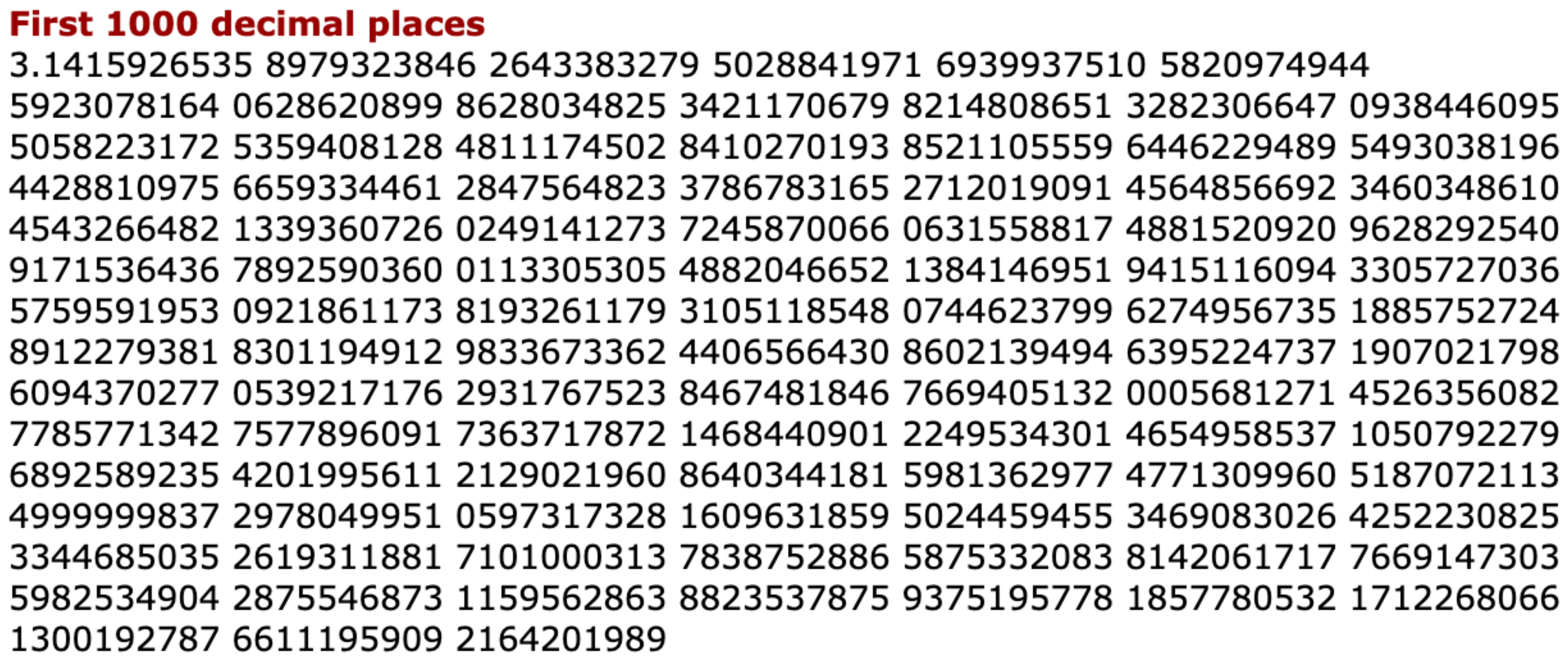
“HotBits is an Internet resource that brings genuine random numbers generated by timing successive pairs of radioactive decays. You order by specifying how many random bytes you want. Once the random bytes are delivered to you, they are immediately discarded — the same data will never be sent to any other user and no records are kept of the data at this or any other site.”

Example 2:

RANDOM.ORG offers true random numbers to anyone on the Internet. The randomness comes from **atmospheric noise**, which is radio noise caused by natural atmospheric processes, primarily lightning discharges in thunderstorms. People use RANDOM.ORG for holding drawings or lotteries, to drive online games, for scientific applications and for art and music.

Example 3:

Numerical evidence, based on looking at trillions of digits of π, suggests that they form a random sequence, but not enough is known at this time to say definitively.



First 1000 decimal places

```
3.1415926535 8979323846 2643383279 5028841971 6939937510 5820974944
5923078164 0628620899 8628034825 3421170679 8214808651 3282306647 0938446095
5058223172 5359408128 4811174502 8410270193 8521105559 6446229489 5493038196
4428810975 6659334461 2847564823 3786783165 2712019091 4564856692 3460348610
4543266482 1339360726 0249141273 7245870066 0631558817 4881520920 9628292540
9171536436 7892590360 0113305305 4882046652 1384146951 9415116094 3305727036
5759591953 0921861173 8193261179 3105118548 0744623799 6274956735 1885752724
8912279381 8301194912 9833673362 4406566430 8602139494 6395224737 1907021798
6094370277 0539217176 2931767523 8467481846 7669405132 0005681271 4526356082
7785771342 7577896091 7363717872 1468440901 2249534301 4654958537 1050792279
6892589235 4201995611 2129021960 8640344181 5981362977 4771309960 5187072113
4999999837 2978049951 0597317328 1609631859 5024459455 3469083026 4252230825
3344685035 2619311881 7101000313 7838752886 5875332083 8142061717 7669147303
5982534904 2875546873 1159562863 8823537875 9375195778 1857780532 1712268066
1300192787 6611195909 2164201989
```

	PRNGs	**TRNGs**
Efficiency	Good	Poor
Determination	Deterministic	Non-deterministic
Periodicity	Periodic	Aperiodic

References

1. http://www.ams.org/publicoutreach/feature-column/fcarc-random
2. https://www.random.org/randomness