K.J.Somaiya College of Engineering, Mumbai

December 14, 2018

A lucid introduction to error correcting codes

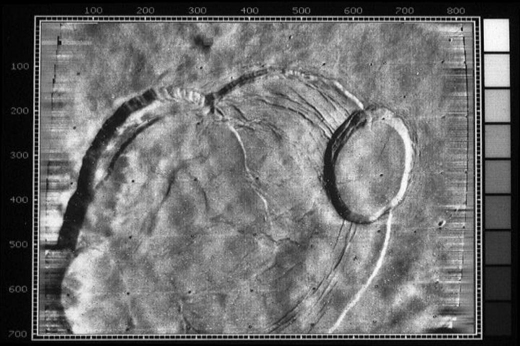
Michel Waldschmidt

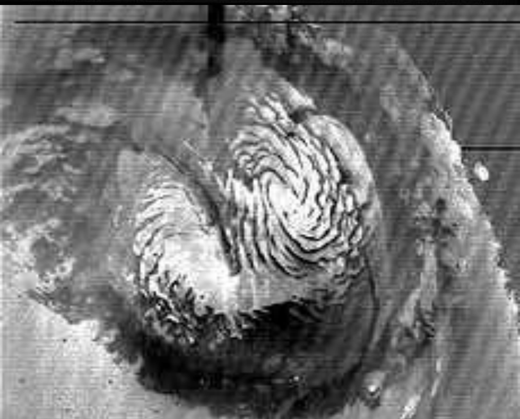
Sorbonne Université

http://webusers.imj-prg.fr/~michel.waldschmidt//

error correcting codes and data transmission

- Transmissions by satellites
- CD’s & DVD’s
- Cellular phones





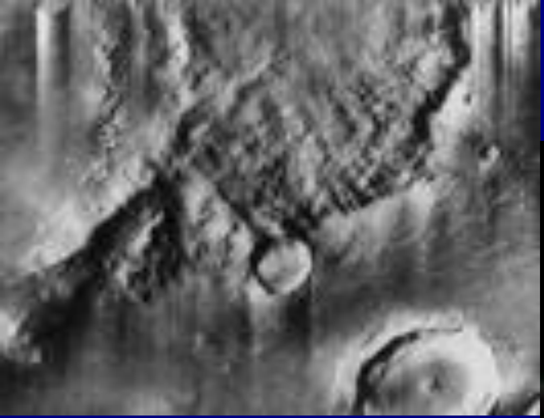
Mariner 2 (1971) and 9 (*1972*)

Olympus Month on Mars planet

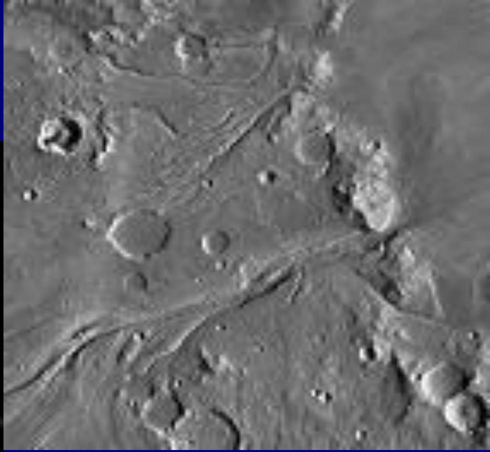
The North polar cap of Mars

Voyager 1 and 2 (*1977*)

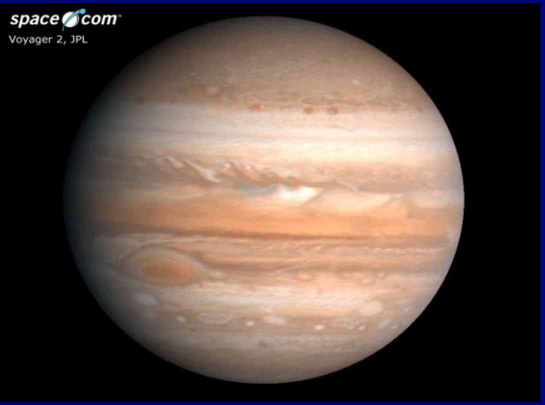
Journey: Cape Canaveral, Jupiter, Saturn, Uranus, Neptune.



Mariner spacecraft 9 (1979)



Black and white photographs of Mars



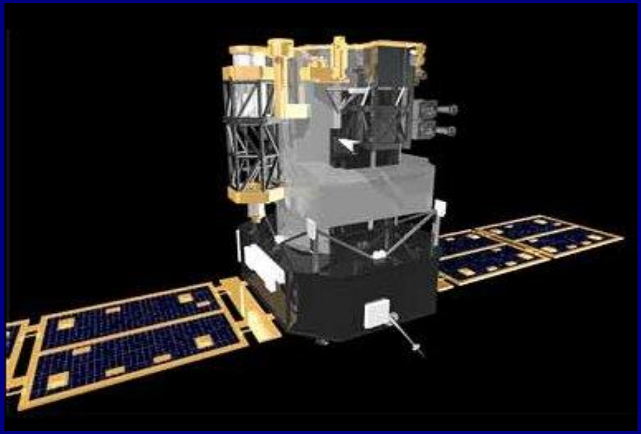
Voyager (1979-81)

Jupiter Saturn

NASA's Pathfinder mission on Mars (1997)

with sojourner rover

- *1998*: lost of control of Soho satellite recovered thanks to double correction by turbo code.



The power of the radio transmitters on these crafts is only a few watts, yet this information is reliably transmitted across hundreds of millions of miles without being completely swamped by noise.

A CD of high quality may have more than *500 000* errors!

- After processing the signals in the CD player, these errors do not lead to any disturbing noise.
- Without error-correcting codes, there would be no CD.

1 second of audio signal = *1 411 200* bits

- 1980's, agreement between Sony and Philips: norm for storage of data on audio CD's.
- 44 100 times per second, 16 bits in each of the two stereo channels

Finite fields and coding theory

- Solving algebraic equations by radicals: Finite fields theory ***Evariste Galois*** *(1811-1832)*

- Construction of regular polygons with rule and compass
- Group theory

Codes and Mathematics

- Algebra
  (discrete mathematics finite fields, linear algebra,…)

- Geometry

- Probability and statistics

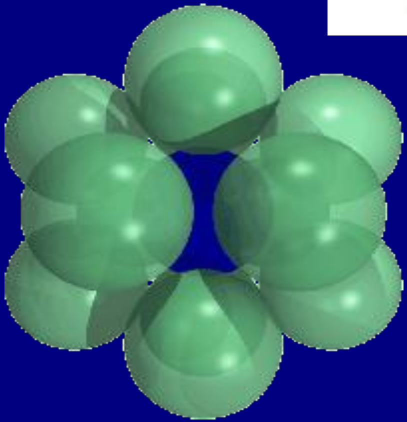
Codes and Geometry

- *1949*: Marcel Golay (specialist of radars): produced two remarkably efficient codes.
- Eruptions on Io (Jupiter's volcanic moon)
- *1963* John Leech uses Golay's ideas for sphere packing in dimension *24 - classification of finite simple groups*
- *1971*: no other *perfect* code than the two found by Golay.

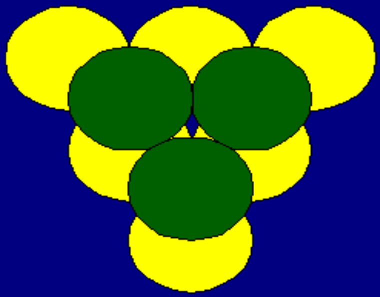
Sphere Packing

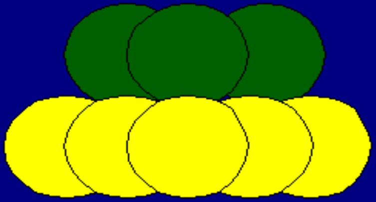
- While Shannon and Hamming were working on information transmission in the States, John Leech invented similar codes while working on Group Theory at Cambridge. This research included work on the sphere packing problem and culminated in the remarkable 24-dimensional Leech lattice, the study of which was a key element in the programme to understand and classify finite symmetry groups.

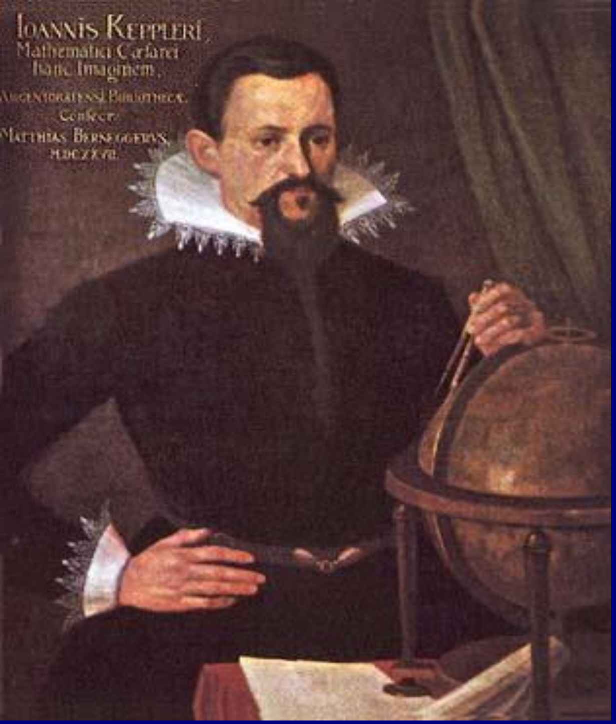
Sphere packing



The kissing number is *12*







Sphere Packing

Kepler Problem: maximal density of a packing of identical sphères :
$\pi / \sqrt{18} = 0.740\,480\,49\ldots$

Conjectured in *1611*.

Proved in *1999* by *Thomas Hales*.

- Connections with crystallography.

Some useful codes

- *1955:* Convolutional codes.
- *1959:* Bose Chaudhuri Hocquenghem codes (BCH codes).
- *1960*: Reed Solomon codes.
- *1970*: Goppa codes.
- *1981*: Algebraic geometry codes.

Current trends

In the past years, the goal of finding explicit codes which reach the limits predicted by Shannon's original work has been achieved. The constructions require techniques from a surprisingly wide range of pure mathematics: linear algebra, the theory of fields and algebraic geometry all play a vital role. Not only has coding theory helped to solve problems of vital importance in the world outside mathematics, it has enriched other branches of mathematics, with new problems as well as new solutions.

Directions of research

- Theoretical questions of existence of specific codes
- connection with cryptography
- lattices and combinatoric designs
- algebraic geometry over finite fields
- equations over finite fields

https://www.ias.ac.in/listing/articles/reso/023

Error Correcting Codes by *Priti Shankar*

Resonance journal of science education

October 1996 Volume 1 Number 10

- How Numbers Protect Themselves
- The Hamming Codes Volume 2 Number 1
- Reed Solomon Codes Volume 2 Number 3

The Hat Problem

The Hat Problem

- Three people are in a room, each has a hat on his head, the colour of which is black or white. Hat colours are chosen randomly. Everybody sees the colour of the hat of everyone else, but not on one's own. People do not communicate with each other.
- Everyone tries to guess (by writing on a piece of paper) the colour of their hat. They may write: Black/White/Abstention.

Rules of the game

- The people in the room win together or lose together as a team.
- The team wins if at least one of the three persons does not abstain, and everyone who did not abstain guessed the colour of their hat correctly.
- What could be the strategy of the team to get the highest probability of winning?

Strategy

- ***A weak strategy:*** anyone guesses randomly.
- Probability of winning: ***$1/2^3 = 1/8$***.

- ***Slightly better strategy:*** they agree that two of them abstain and the other guesses randomly.
- Probability of winning: ***$1/2$***.
- Is it possible to do better?

Information is the key

- ***Hint:***

  Improve the odds by using the available information: everybody sees the colour of the hat on everyone's head except on one's own head.

Solution of the Hat Problem

- ***Better strategy***: anyone who sees two different colours abstains. Anyone who sees the same colour twice guesses that one's hat has the other colour.

The two people with white hats see one white hat and one black hat, so they abstain.

The one with a black hat sees two white hats, so he writes black.

The team wins!



The two people with black hats see one white hat and one black hat, so they abstain.

The one with a white hat sees two black hats, so he writes **white**.

The team wins!

Everybody sees two white hats, and therefore writes **black** on the paper.

The team looses!

Everybody sees two black hats, and therefore writes **white** on the paper.

The team looses!

Winning team:

two white
or
two black

Loosing team:

three white
or
three black

Probability of winning: ***3/4.***

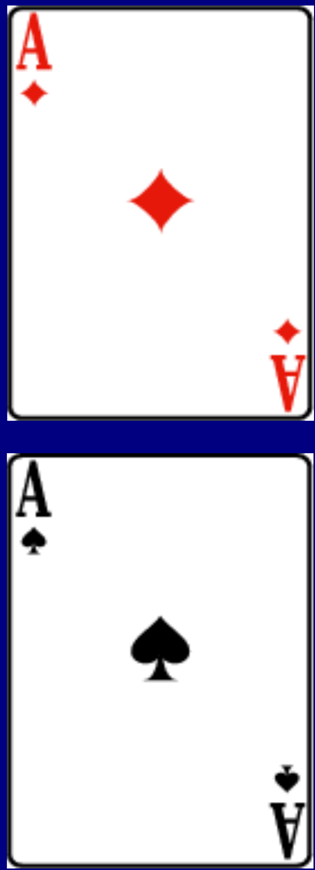
Playing cards:
easy game

I know which card you selected

- Among a collection of playing cards, you select one without telling me which one it is.
- I ask you some questions and you answer yes or no.
- Then I am able to tell you which card you selected.

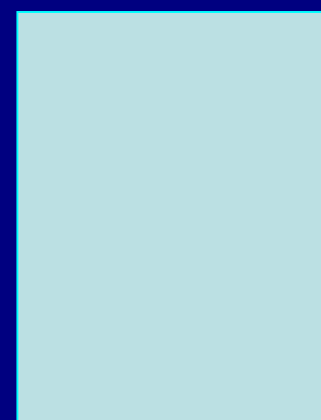
2 cards

- You select one of these two cards
- I ask you one question and you answer *yes* or *no*.
- I am able to tell you which card you selected.

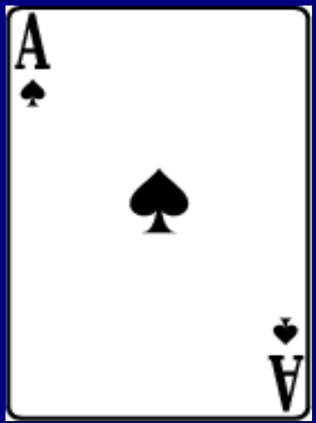


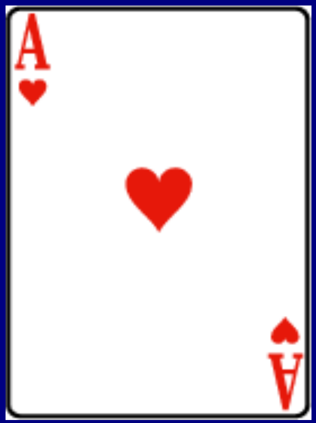
2 cards: one question suffices

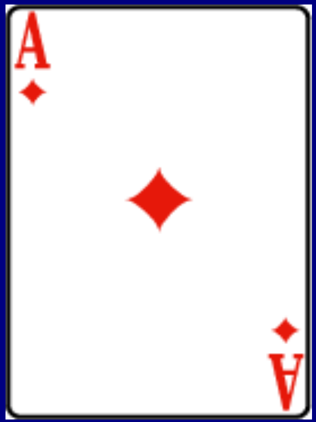
- Question: is it this one?



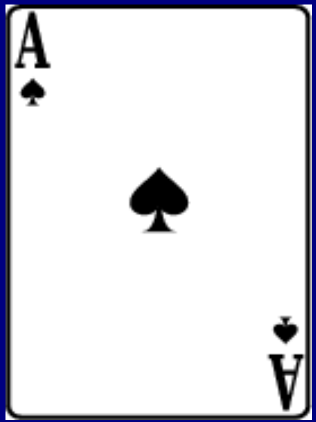
4 cards

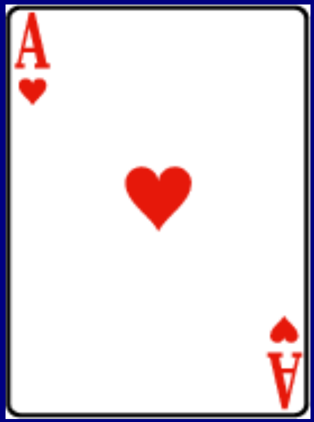




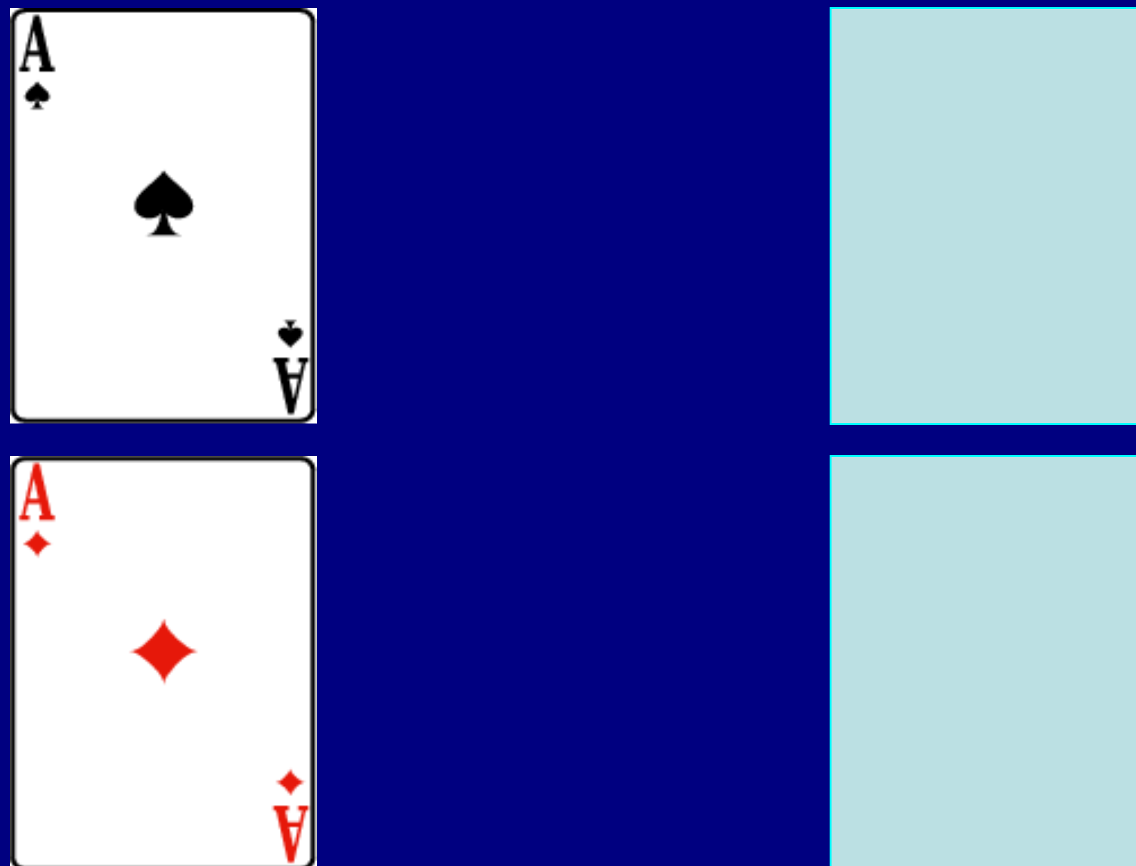


First question: is it one of these two?



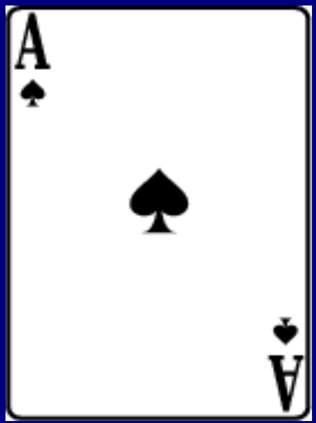


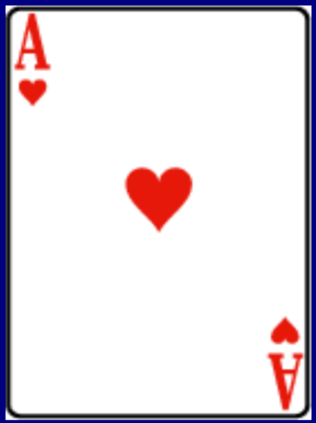
Second question:
is it one of these two ?



4 cards: *2* questions suffice

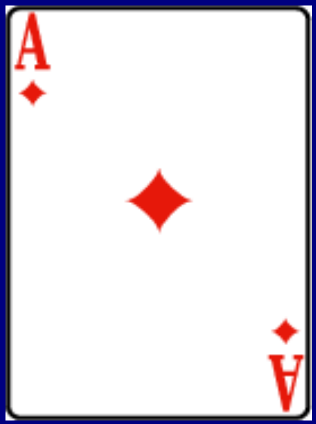
Y Y





Y N

N Y



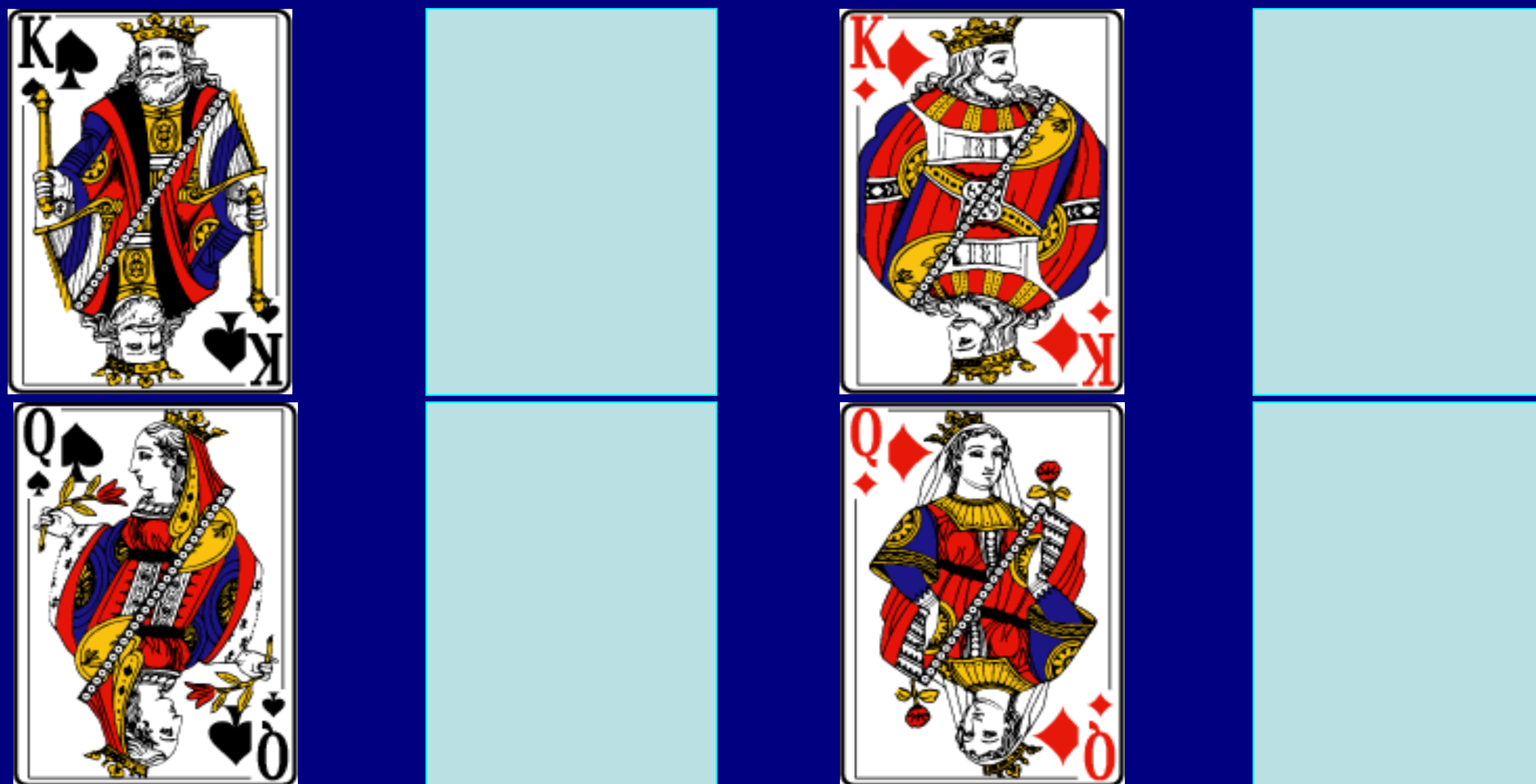
N N

8 Cards

First question: is it one of these?

Second question: is it one of these?

Third question: is it one of these?



8 Cards: *3* questions

YYY	YYN	YNY	YNN
NYY	NYN	NNY	NNN

陰 [阴]　　陽 [阳]

Yes / No

- *0* / *1*
- Yin —— / Yang - -
- True / False
- White / Black
- \+ / -
- Head / Tails (*tossing* or *flipping* a coin)

8 Cards: *3* questions

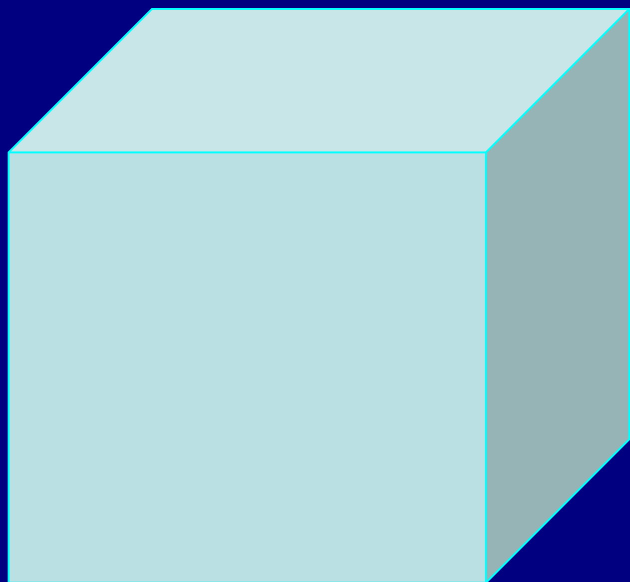
YYY	YYN	YNY	YNN
NYY	NYN	NNY	NNN

Replace Y by 0 and N by 1

3 questions, *8* solutions

0 0 0	*0 0 1*	*0 1 0*	*0 1 1*
0	*1*	*2*	*3*
1 0 0	*1 0 1*	*1 1 0*	*1 1 1*
4	*5*	*6*	*7*

$$8 = 2 \times 2 \times 2 = 2^3$$



One could also display the eight cards on the corners of a cube rather than in two rows of four entries.

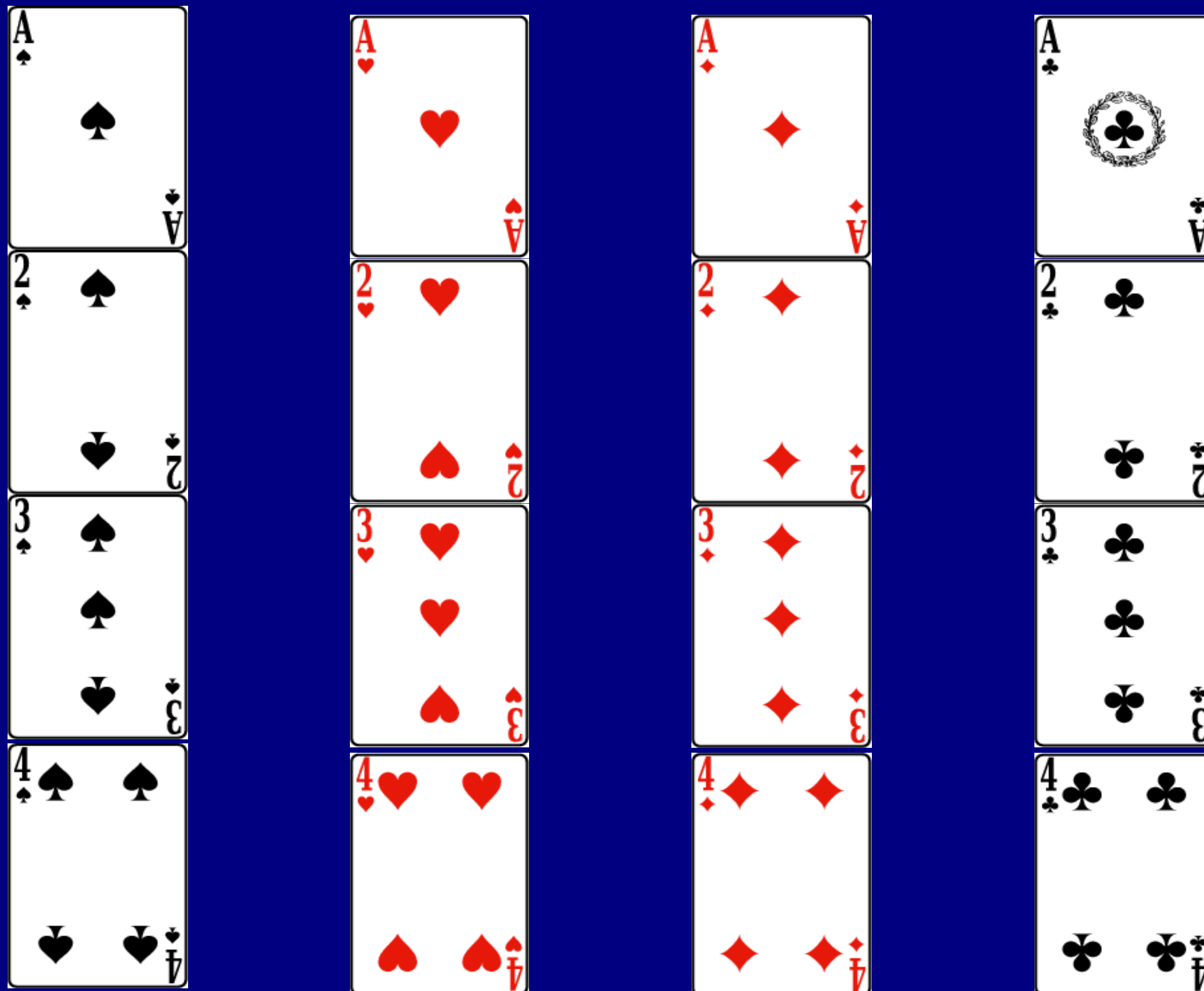
Exponential law

n questions for 2^n cards

Add one question =
multiply the number of cards by *2*

Economy:
Growth rate of *4%* for *25* years = multiply by 2.7

16 Cards *4* questions



Label the *16* cards

0	*1*	*2*	*3*
4	*5*	*6*	*7*
8	*9*	*10*	*11*
12	*13*	*14*	*15*

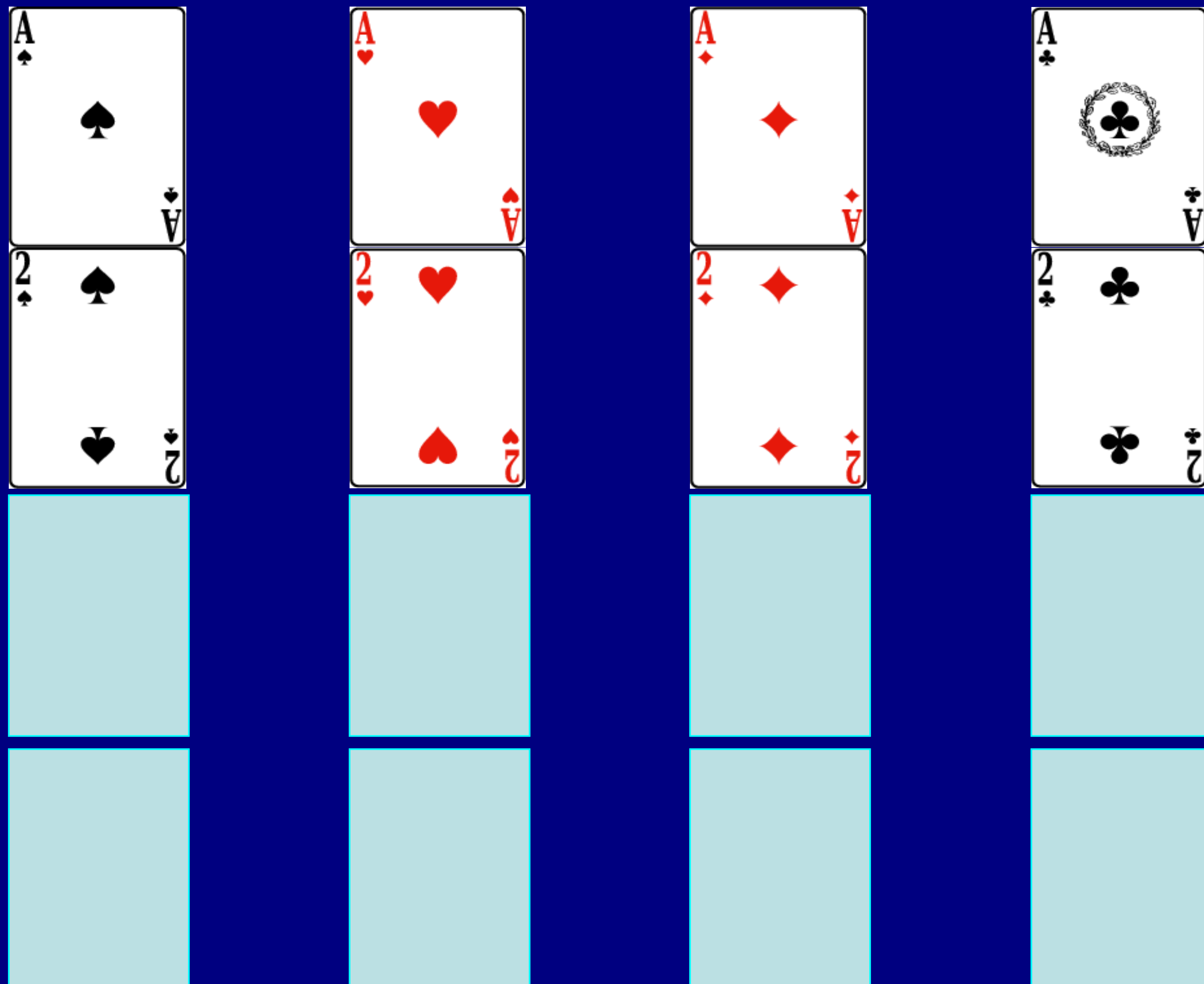
Binary representation:

0 0 0 0	*0 0 0 1*	*0 0 1 0*	*0 0 1 1*
0 1 0 0	*0 1 0 1*	*0 1 1 0*	*0 1 1 1*
1 0 0 0	*1 0 0 1*	*1 0 1 0*	*1 0 1 1*
1 1 0 0	*1 1 0 1*	*1 1 1 0*	*1 1 1 1*

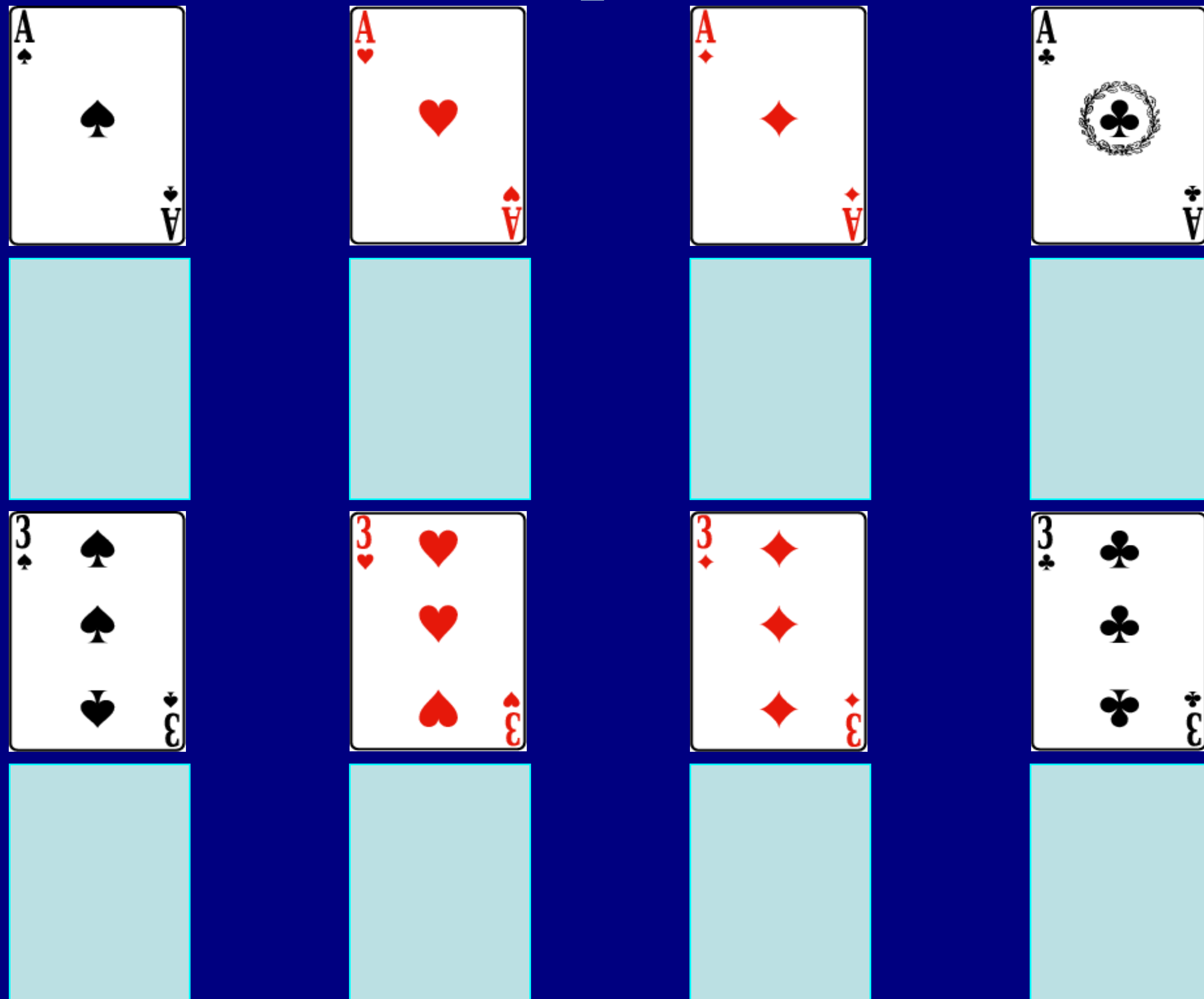
Ask the questions so that the answers are:

Y Y Y Y	Y Y Y N	Y Y N Y	Y Y N N
Y N Y Y	Y N Y N	Y N N Y	Y N N N
N Y Y Y	N Y Y N	N Y N Y	N Y N N
N N Y Y	N N Y N	N N N Y	N N N N

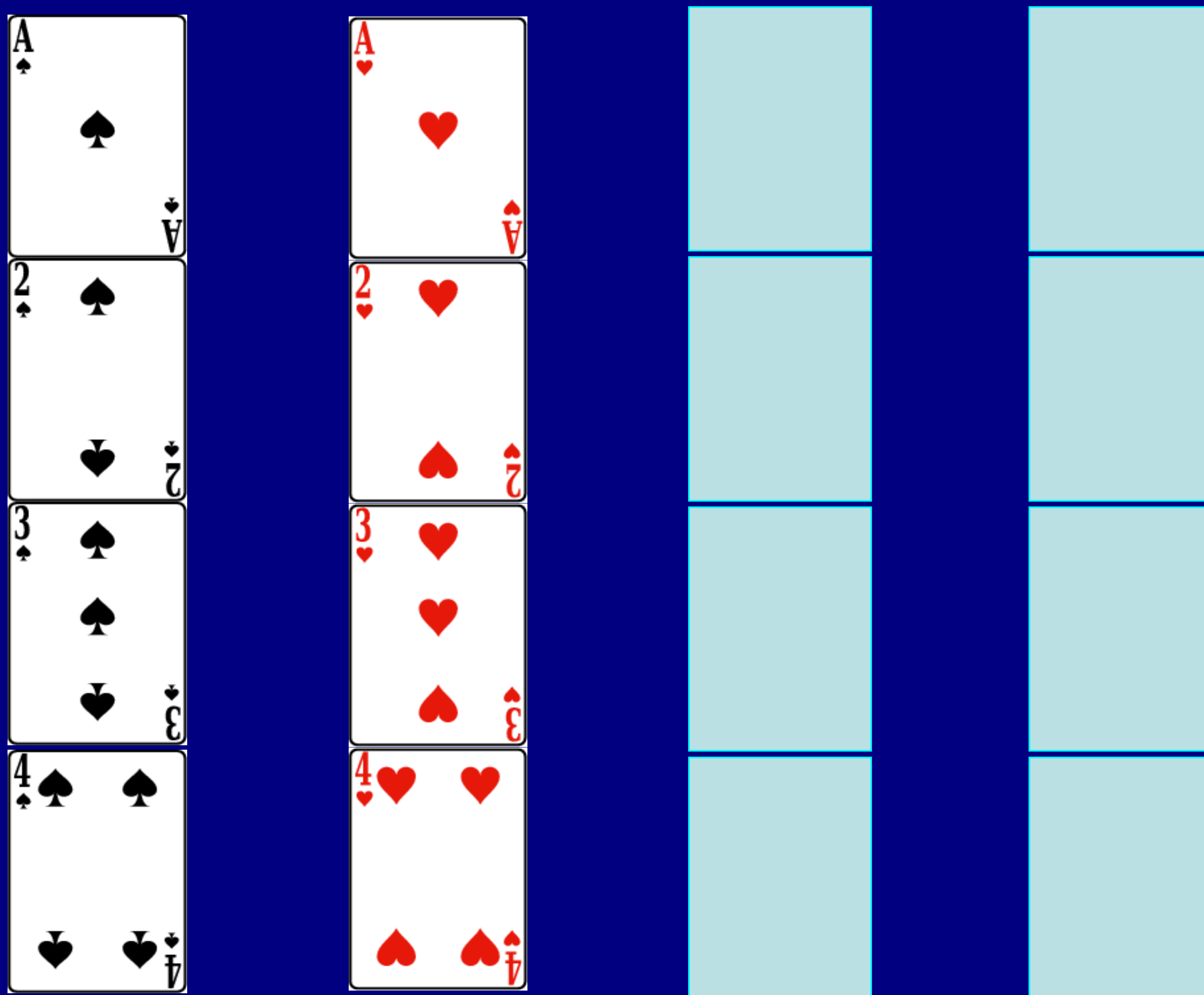
First question:



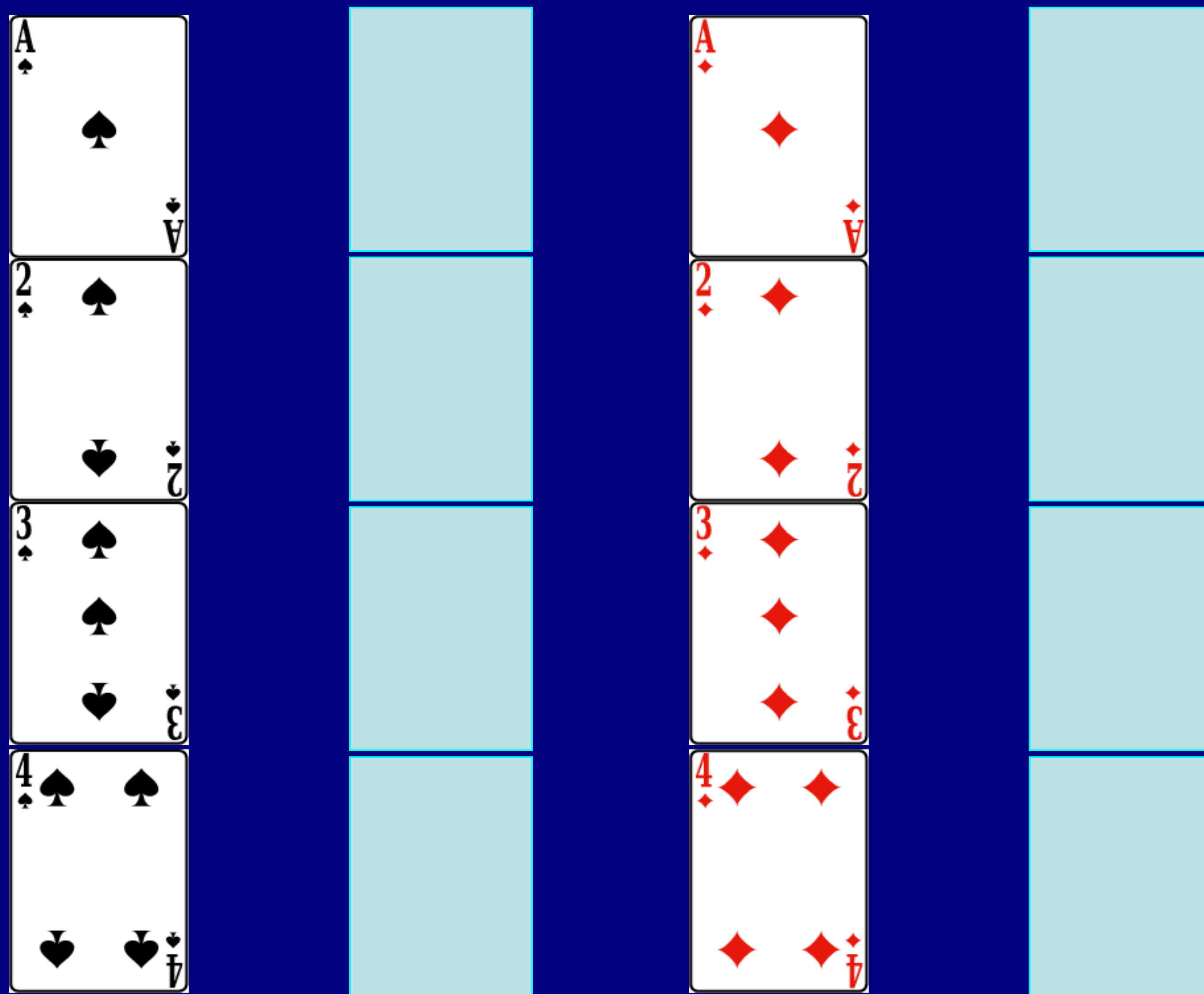
Second question:



Third question:



Fourth question:

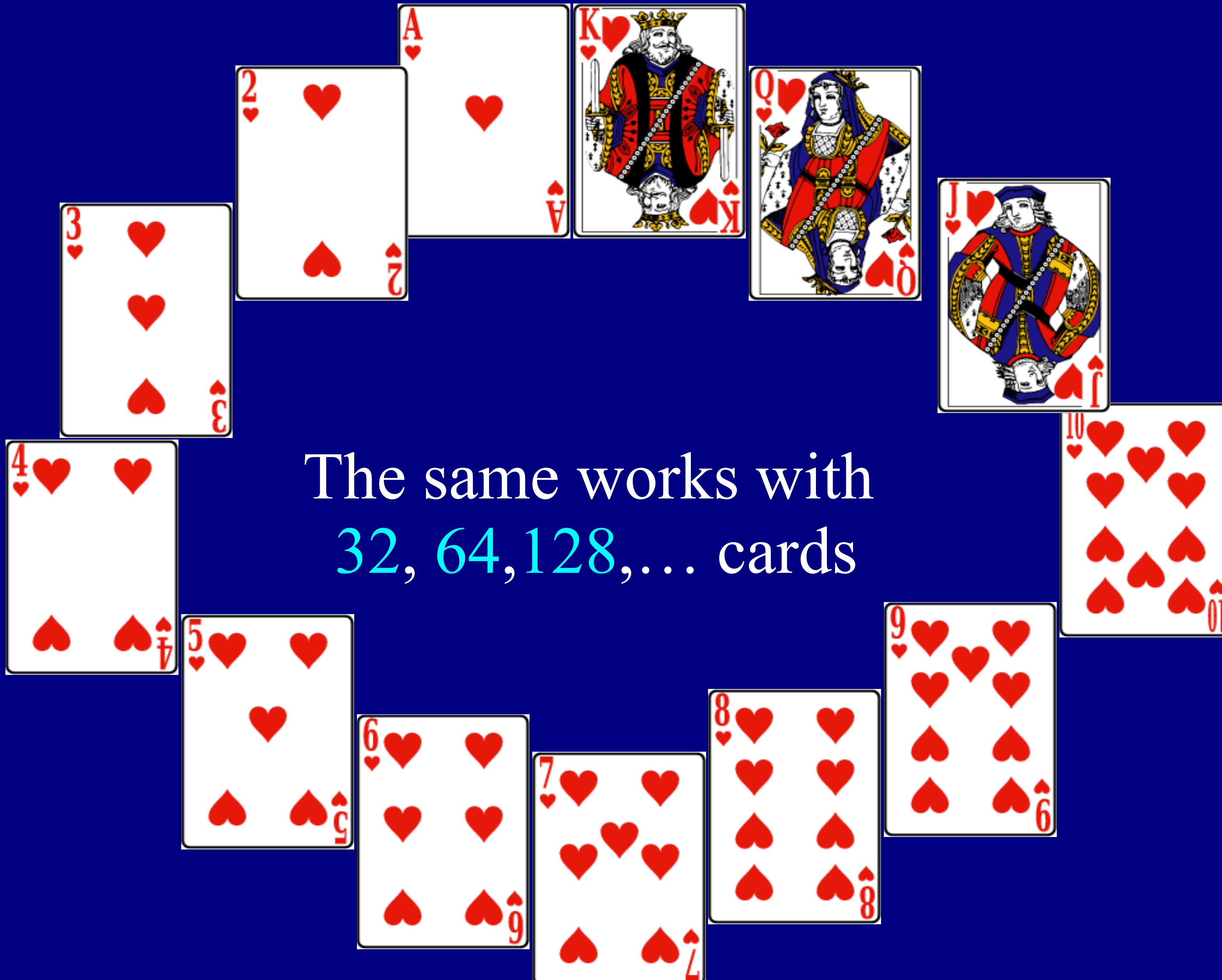


Example with 16 cards

If you selected the card with label 7, in basis 2 it is 0111, you answer

yes, no, no, no.

Your answers give the binary development of the label of the card you selected.



The same works with

32, 64,128,… cards

More difficult:

One answer may be wrong!

One answer may be wrong

- Consider the same problem, but you are allowed to give (at most) one wrong answer.
- How many questions are required so that I am able to know whether your answers are all right or not? And if they are all right, to know the card you selected?

Detecting one mistake

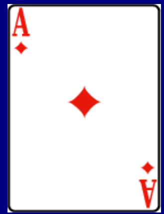
- If I ask one more question, I will be able to detect if one of your answers is not compatible with the other answers.
- And if you made no mistake, I will tell you which is the card you selected.

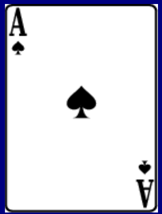
Detecting one mistake with *2* cards

- With two cards I just repeat twice the same question.
- If both your answers are the same, you did not lie and I know which card you selected
- If your answers are not the same, I know that one answer is right and one answer is wrong (but I don't know which one is correct!).

0 0

Y Y





1 1

N N

Principle of coding theory

Only certain words are allowed ***(code*** = *dictionary of valid words)*.

The « useful » letters (*data bits*) carry the information, the other ones (*control bits* or *check bits*) allow detecting errors and sometimes correcting errors.

Detecting one error by sending twice the message

Send twice each bit

2 codewords among $4=2^2$ possible words

(1 data bit, 1 check bit)

Codewords (length two)

$0\ 0$

and

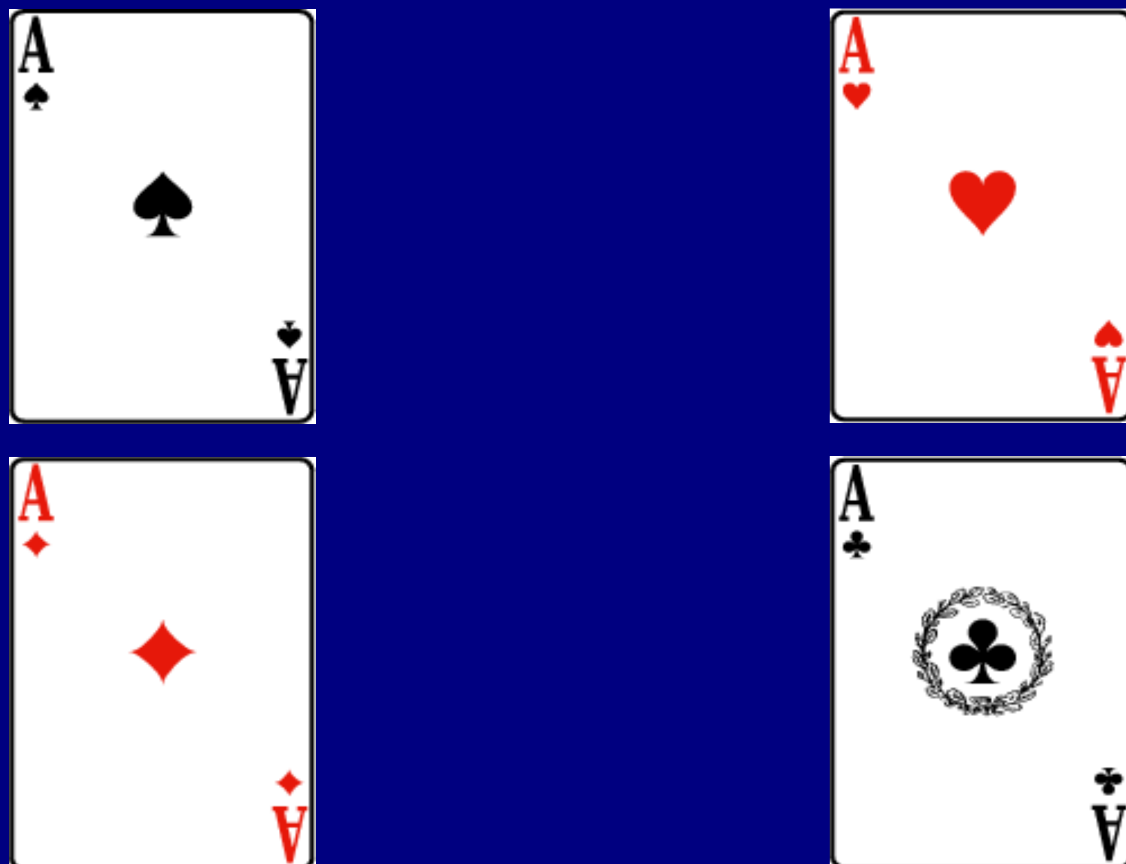
$1\ 1$

Rate: $1/2$

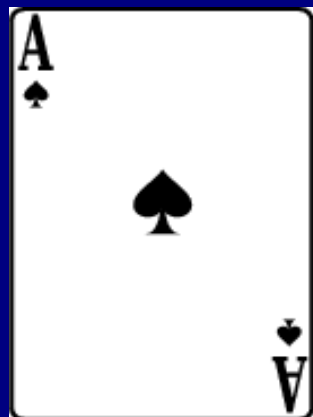
Principle of codes detecting one error:

Two distinct codewords
have at least two distinct letters

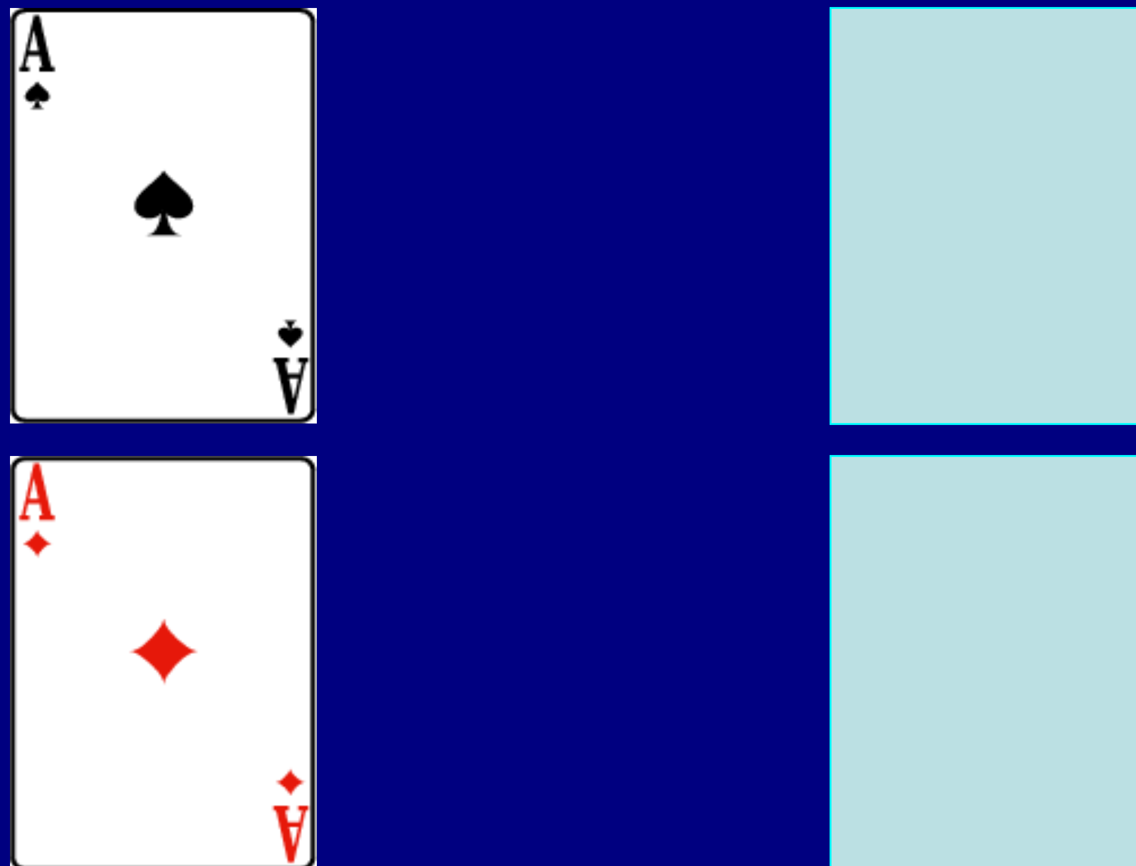
4 cards



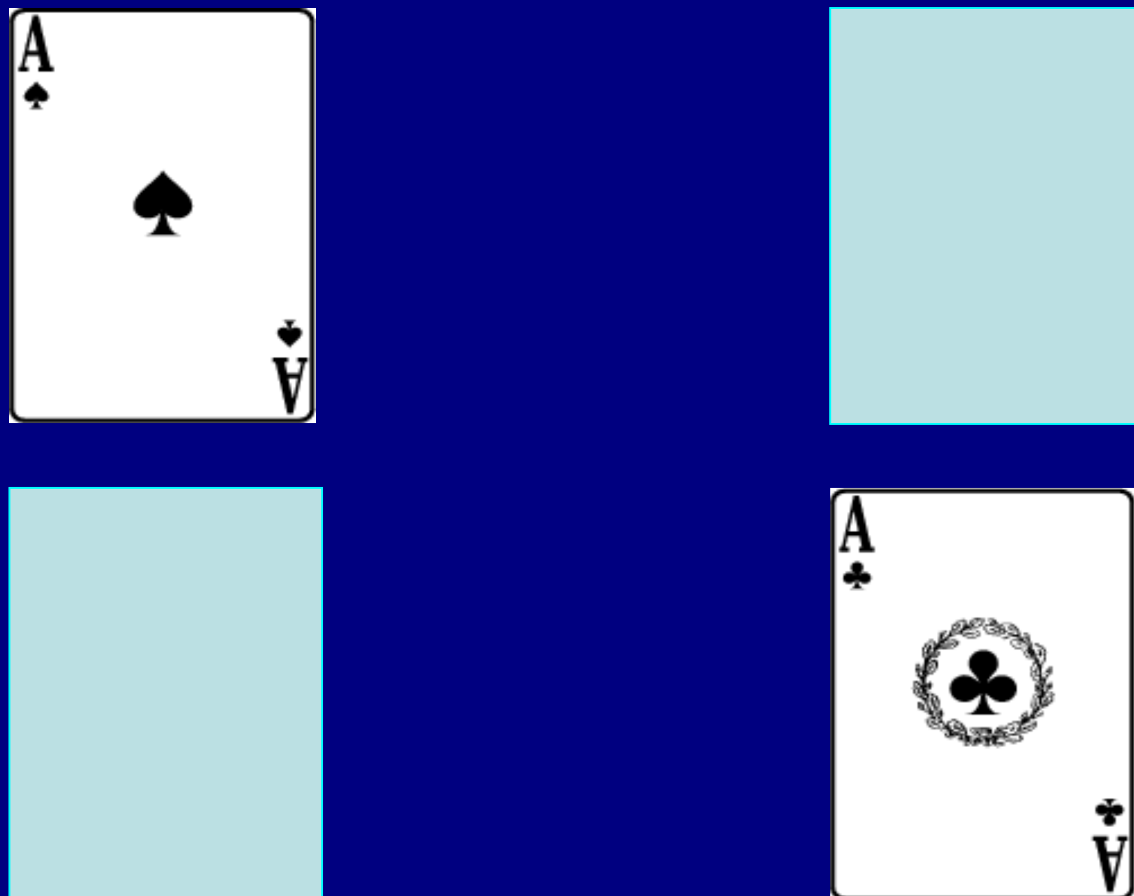
First question: is it one of these two?



Second question: is it one of these two?

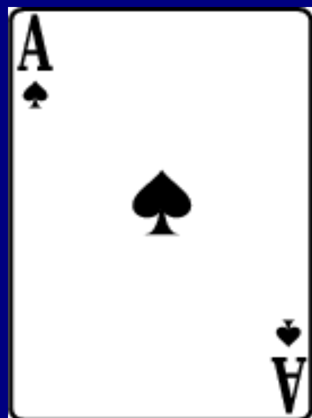


Third question: is it one of these two?



4 cards: *3* questions

Y Y Y



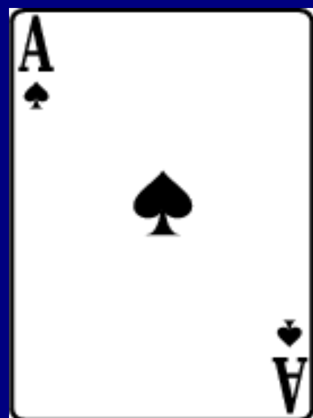
Y N N

N Y N

N N Y

4 cards: *3* questions

0 0 0



0 1 1

1 0 1

1 1 0

Correct triples of answers:

0 0 0 *0 1 1* *1 0 1* *1 1 0*

Wrong triples of answers

0 0 1 *0 1 0* *1 0 0* *1 1 1*

One change in a correct triple of answers yields a wrong triple of answers

In a correct triple of answers, the number of *1*'s is even, in a wrong triple of answers, the number of *1*'s is odd.

Boolean addition

- $0 + 0 = 0$
- $0 + 1 = 1$
- $1 + 0 = 1$
- $1 + 1 = 0$

- even + even = even
- even + odd = odd
- odd + even = odd
- odd + odd = even

Parity bit *or* Check bit

- Use one extra bit defined to be the Boolean sum of the previous ones.
- Now for a correct answer the Boolean sum of the bits should be *0* (the number of *1*'s is even).
- If there is exactly one error, the parity bit will detect it: the Boolean sum of the bits will be *1* instead of *0* (since the number of *1*'s is odd).
- *Remark:* also it corrects one missing bit.

Parity bit *or* Check bit

- In the International Standard Book Number (ISBN) system used to identify books, the last of the ten-digit number is a check bit.
- The Chemical Abstracts Service (CAS) method of identifying chemical compounds, the United States Postal Service (USPS) use check digits.
- Modems, computer memory chips compute checksums.
- One or more check digits are commonly embedded in credit card numbers.

Detecting one error with the parity bit

Codewords (of length *3*):

$$0\ 0\ 0$$
$$0\ 1\ 1$$
$$1\ 0\ 1$$
$$1\ 1\ 0$$

Parity bit : $(x\ y\ z)$ with $z = x + y$.

4 codewords (among *8* words of length *3*),

2 data bits, *1* check bit.

Rate: *2/3*

Codewords Non Codewords

Codewords	Non Codewords
0 0 0	*0 0 1*
0 1 1	*0 1 0*
1 0 1	*1 0 0*
1 1 0	*1 1 1*

Two distinct codewords have at least two distinct letters.

8 Cards

4 questions for *8* cards

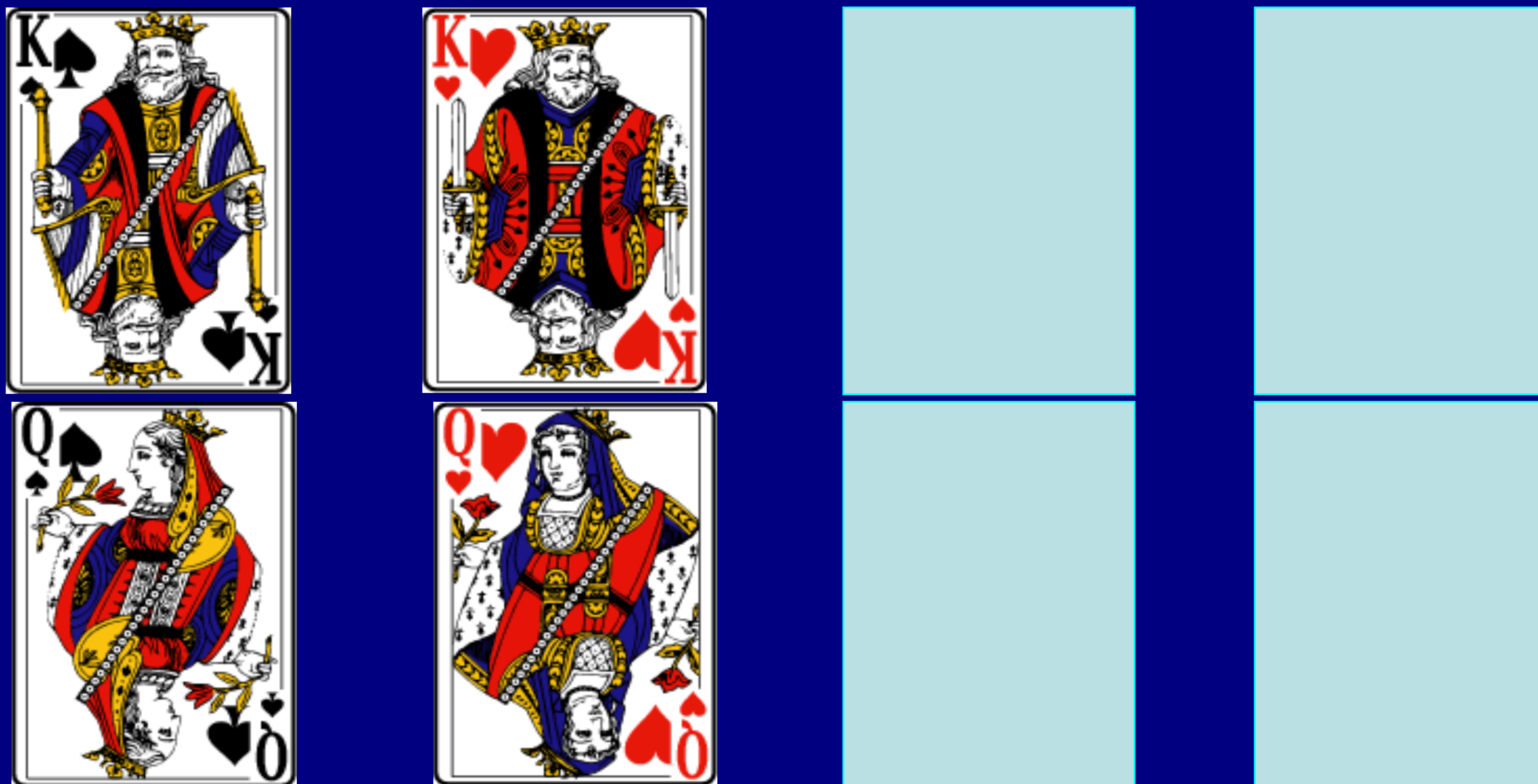
Use the *3* previous questions
plus the parity bit question
(the number of N's should be even).

*000**0***	*001**1***	*010**1***	*011**0***
YYY**Y**	YYN**N**	YNY**N**	YNN**Y**
*100**1***	*101**0***	*110**0***	*111**1***
NYY**N**	NYN**Y**	NNY**Y**	NNN**N**

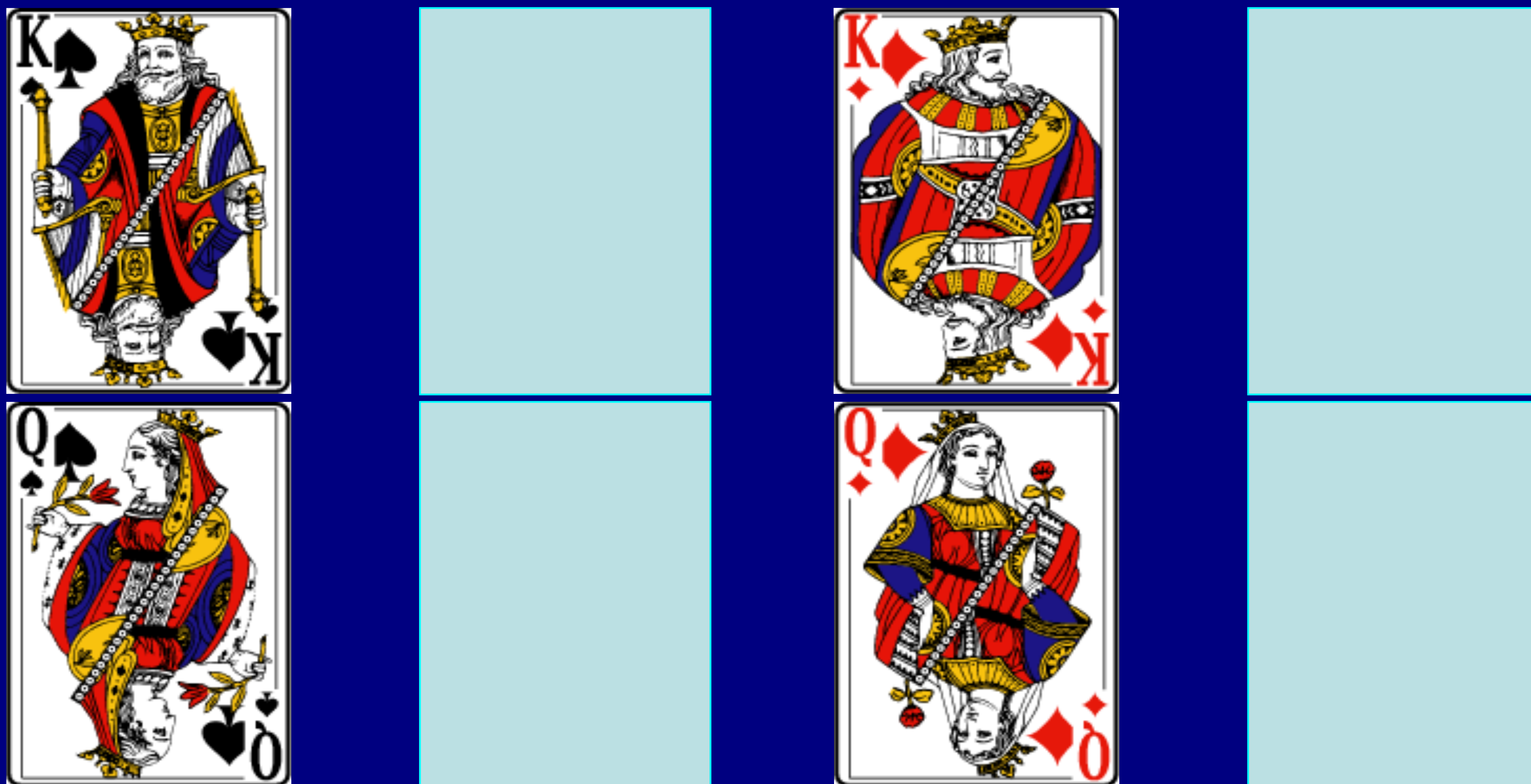
First question: is it one of these?



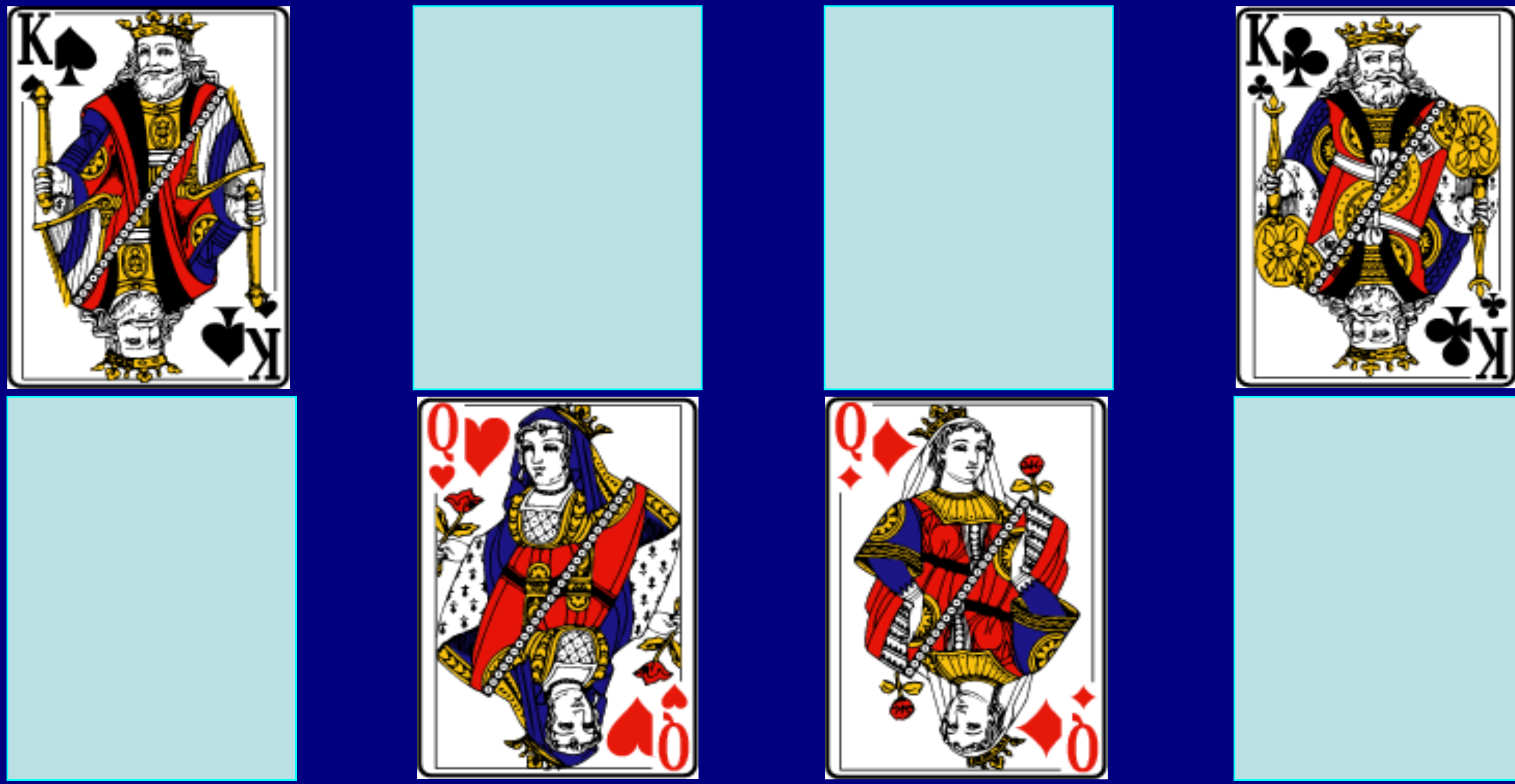

Second question: is it one of these?



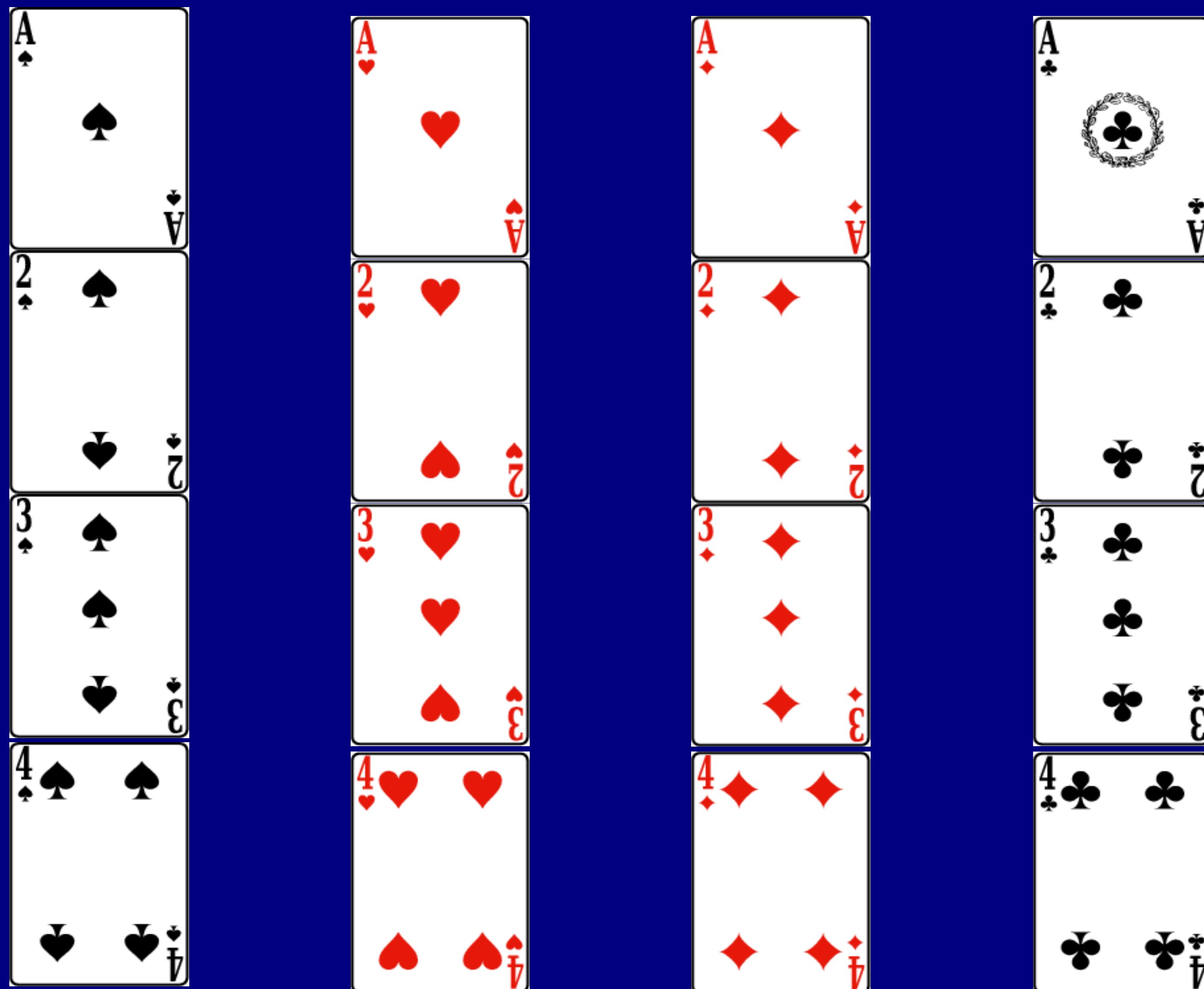
Third question: is it one of these?



Fourth question: is it one of these?



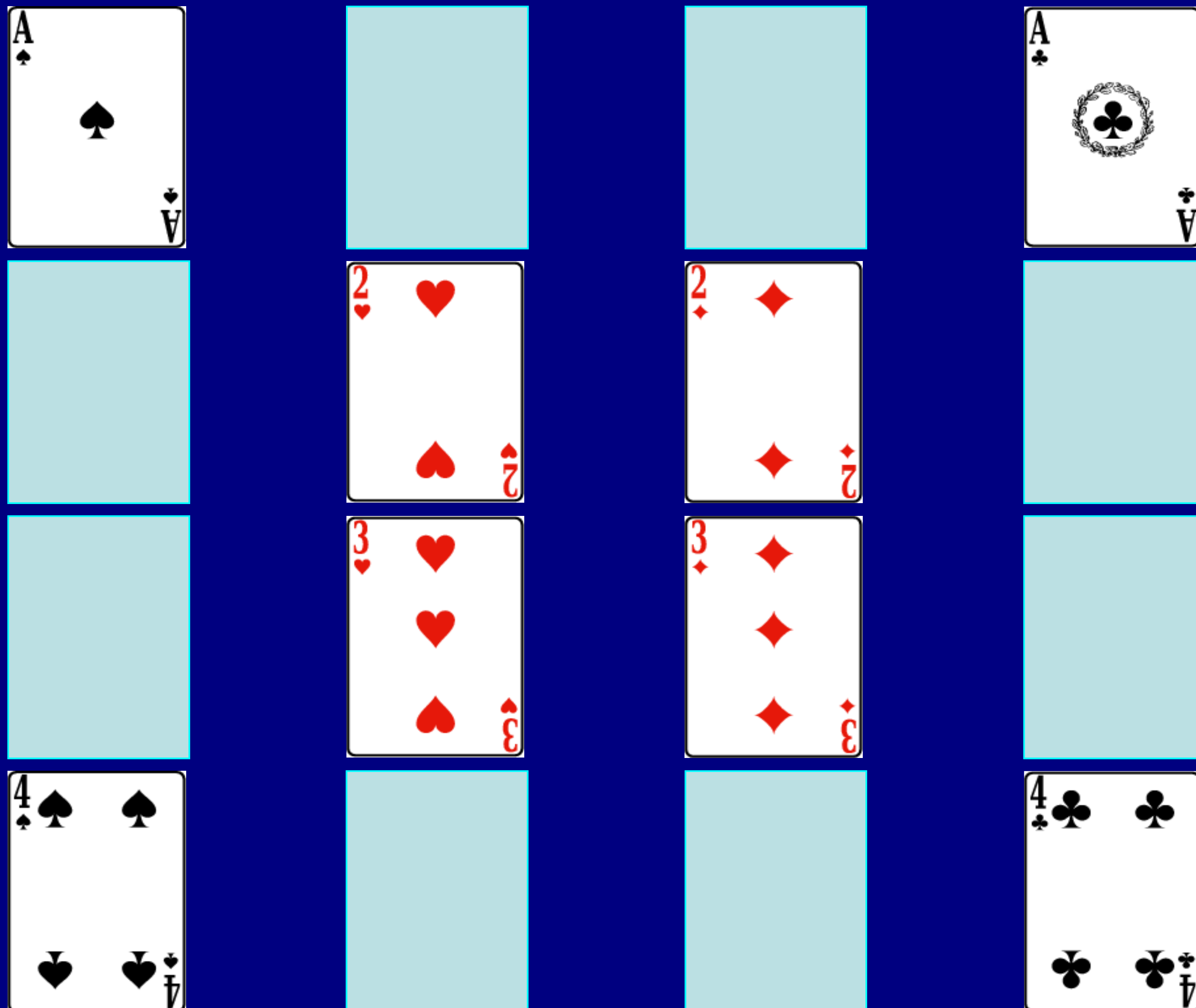
16 cards, at most one wrong answer: *5* questions to detect the mistake

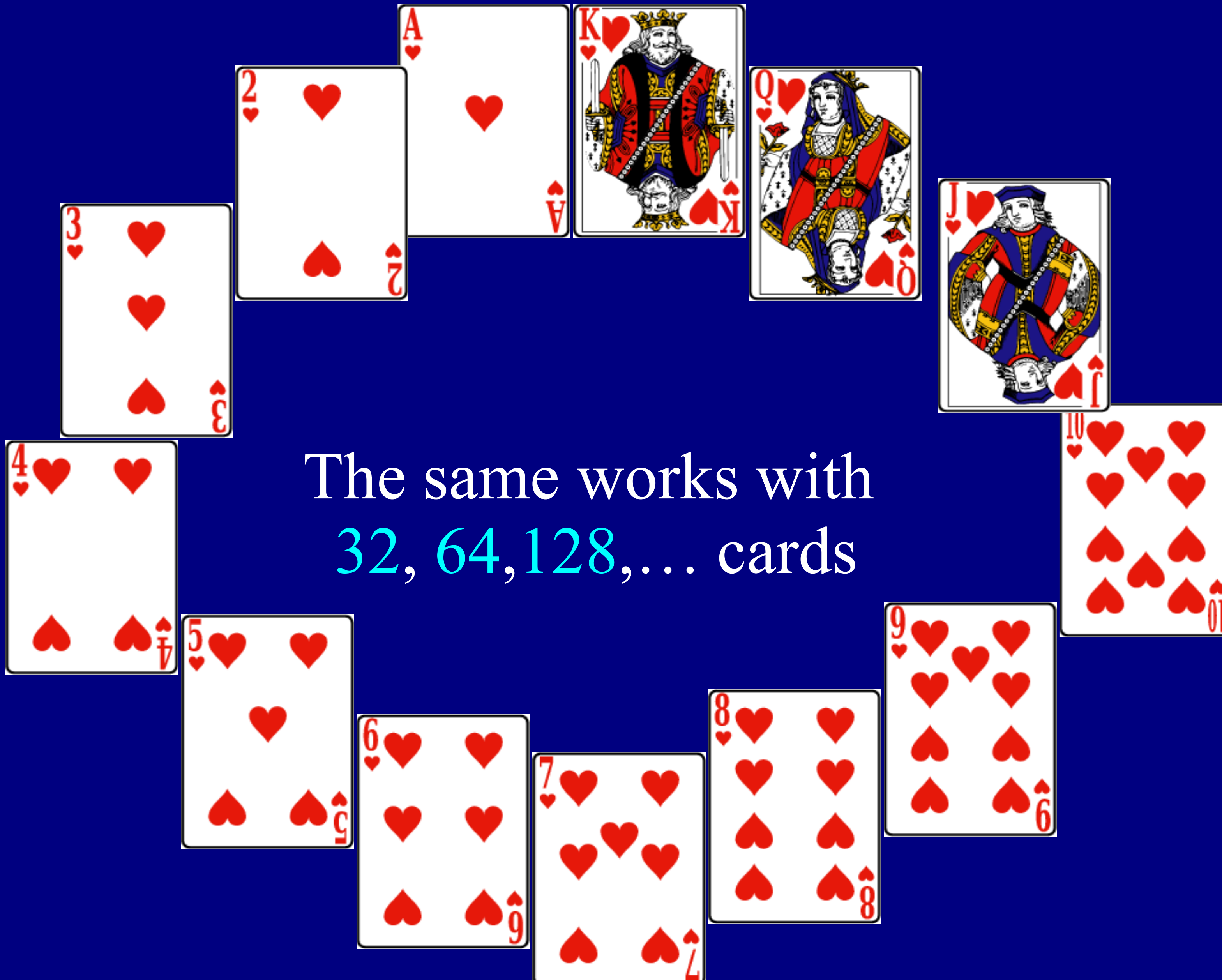


Ask the *5* questions so that the answers are:

YYYYY	YYYNN	YYNYN	YYNNY
YNYYN	YNYNY	YNNYY	YNNNN
NYYYN	NYYNY	NYNYY	NYNNN
NNYYY	NNYNN	NNNYN	NNNNY

Fifth question:





The same works with 32, 64,128,… cards

Correcting one mistake

- Again I ask you questions to each of which your answer is yes or no, again you are allowed to give at most one wrong answer, but now I want to be able to know which card you selected - and also to tell you whether or not you lied and when you eventually lied.

With *2* cards

- I repeat the same question three times.
- The most frequent answer is the right one: *vote with the majority*.
- *2* cards, *3* questions, corrects *1* error.
- Right answers: *000* and *111*

Correcting one error by repeating three times

- Send each bit three times

Codewords (length three)

2 codewords
among *8* possible ones
(*1* data bit, *2* check bits)

0 0 0
1 1 1

Rate: ***1/3***

- Correct *0 0 1* as *0 0 0*
- Correct *0 1 0* as *0 0 0*
- Correct *1 0 0* as *0 0 0*

and

- Correct *1 1 0* as *1 1 1*
- Correct *1 0 1* as *1 1 1*
- Correct *0 1 1* as *1 1 1*

Principle of codes correcting one error:

Two distinct codewords have at least three distinct letters

Hamming Distance between two words:

= number of places in which the two words differ

Examples

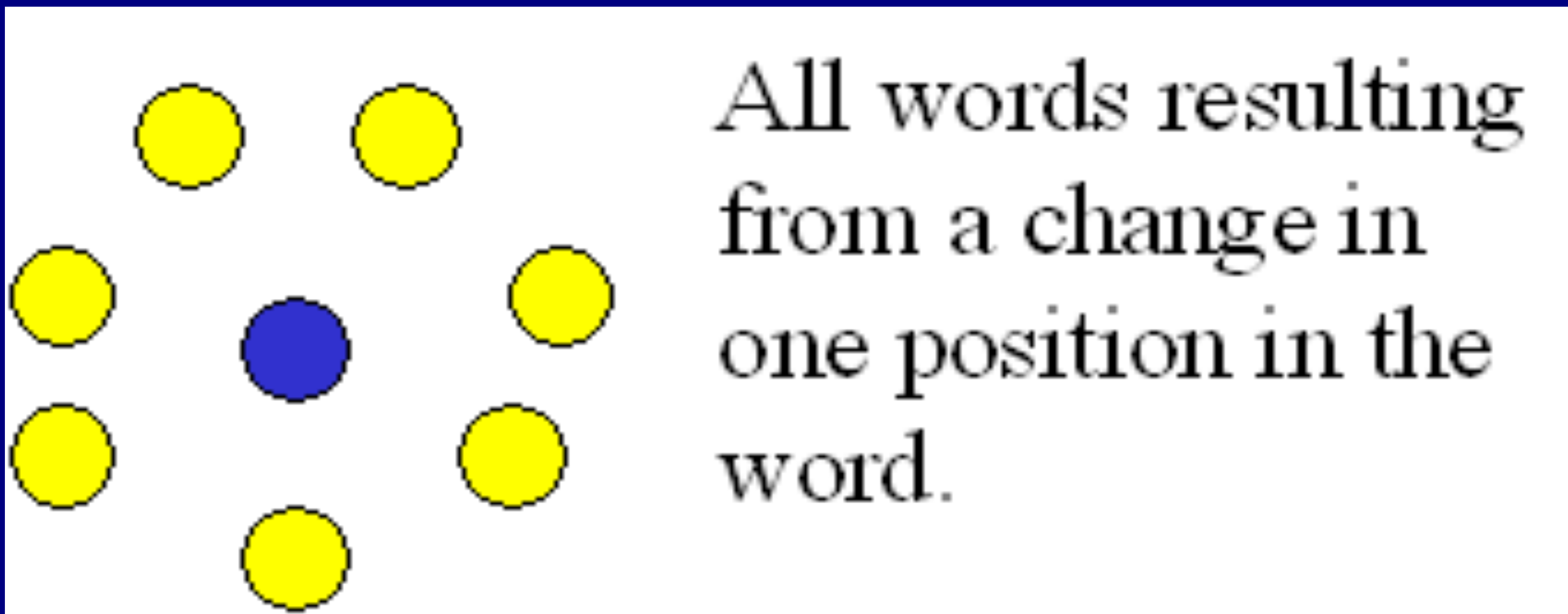
(0,0,1) and *(0,0,0)* have distance *1*

(1,0,1) and *(1,1,0)* have distance *2*

(0,0,1) and *(1,1,0)* have distance *3*

Richard W. Hamming (1915-1998)

Hamming distance 1



Two or three *0*'s

(0,0,1)

(0,1,0)

(0,0,0)

(1,0,0)

Two or three *1*'s

(1,0,1)

(1,1,0)

(1,1,1)

(0,1,1)

The code (0 0 0) (1 1 1)

- The set of words of length *3* (eight elements) splits into two spheres (balls)

- The centers are respectively ***(0,0,0)*** and ***(1,1,1)***

- Each of the two balls consists of elements at distance at most *1* from the center

Back to the Hat Problem

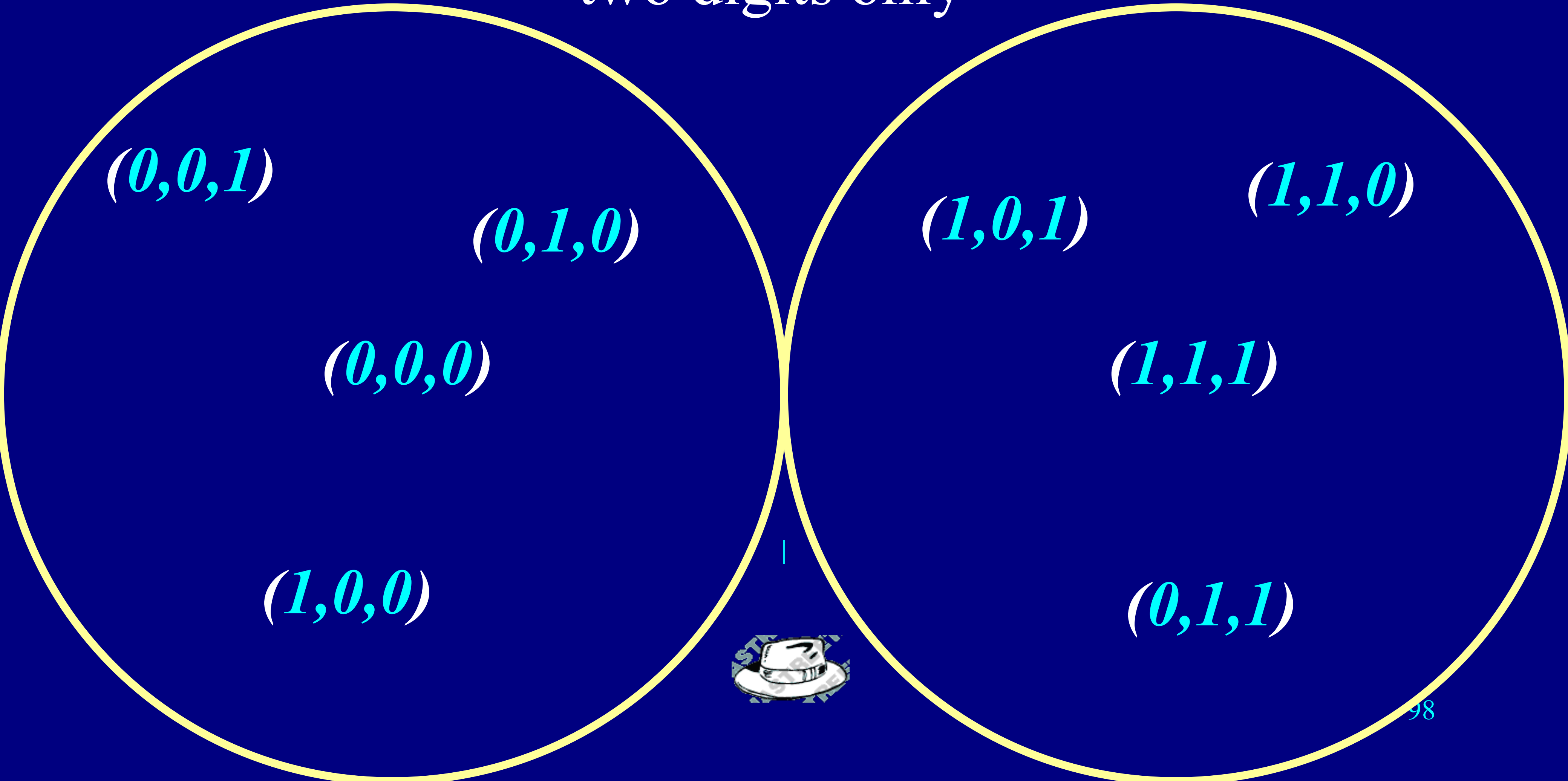
Connection with error detecting codes

- Replace white by *0* and black by *1;*
  hence the distribution of colours becomes a word of length *3* on the alphabet *{0 , 1}*
- Consider the centers of the balls *(0,0,0)* and *(1,1,1).*
- The team bets that the distribution of colours is not one of the two centers.

If a player sees two ***0***, the center of the ball is ***(0,0,0)***

If a player sees two ***1***, the center of the ball is ***(1,1,1)***

Each player knows two digits only



If a player sees one ***0*** and one ***1***,
he does not know the center

(0,0,1)

(0,1,0)

(0,0,0)

(1,0,0)

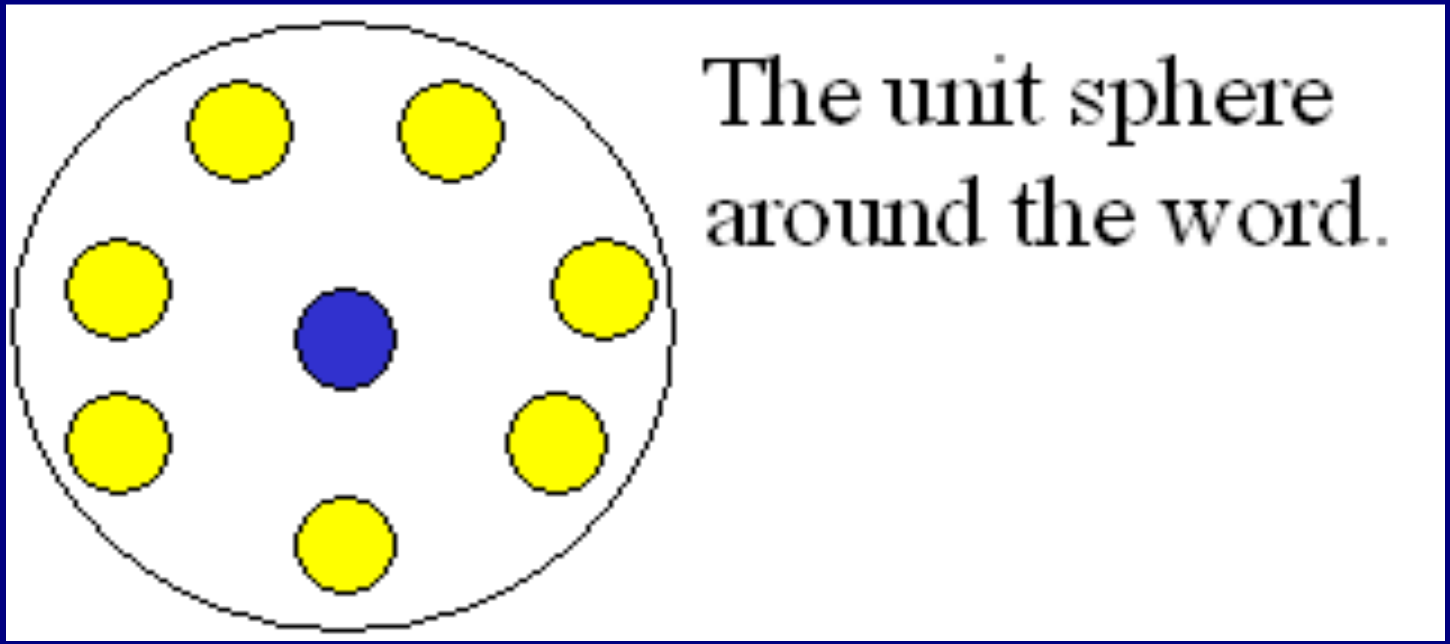
(1,1,0)

(1,0,1)

(1,1,1)

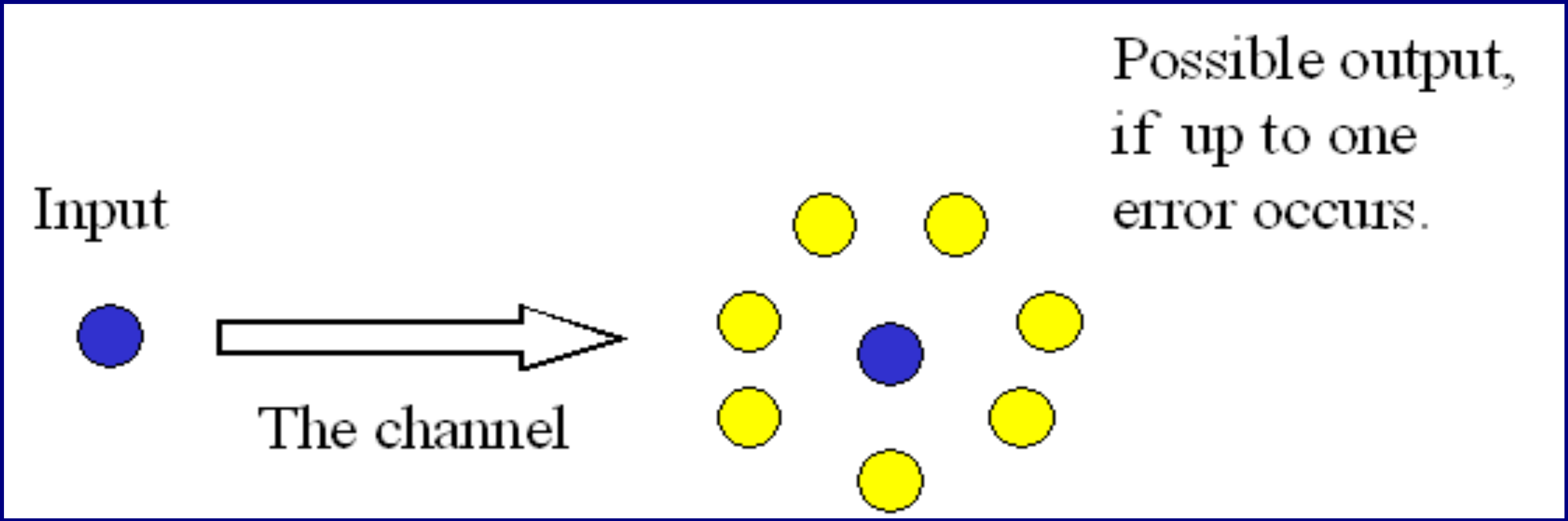
(0,1,1)

Hamming's unit sphere

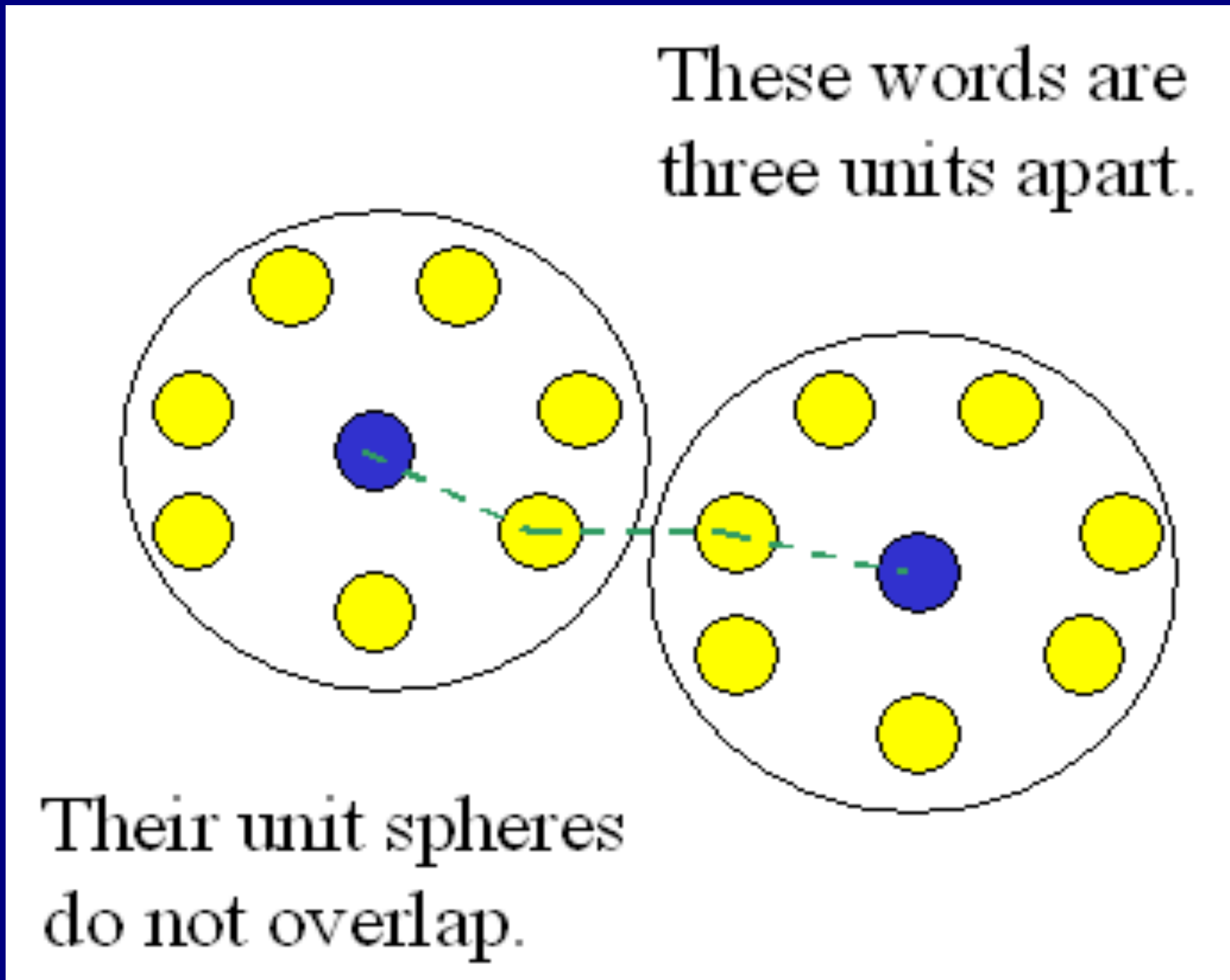


- The unit sphere around a word includes the words at distance at most *1*

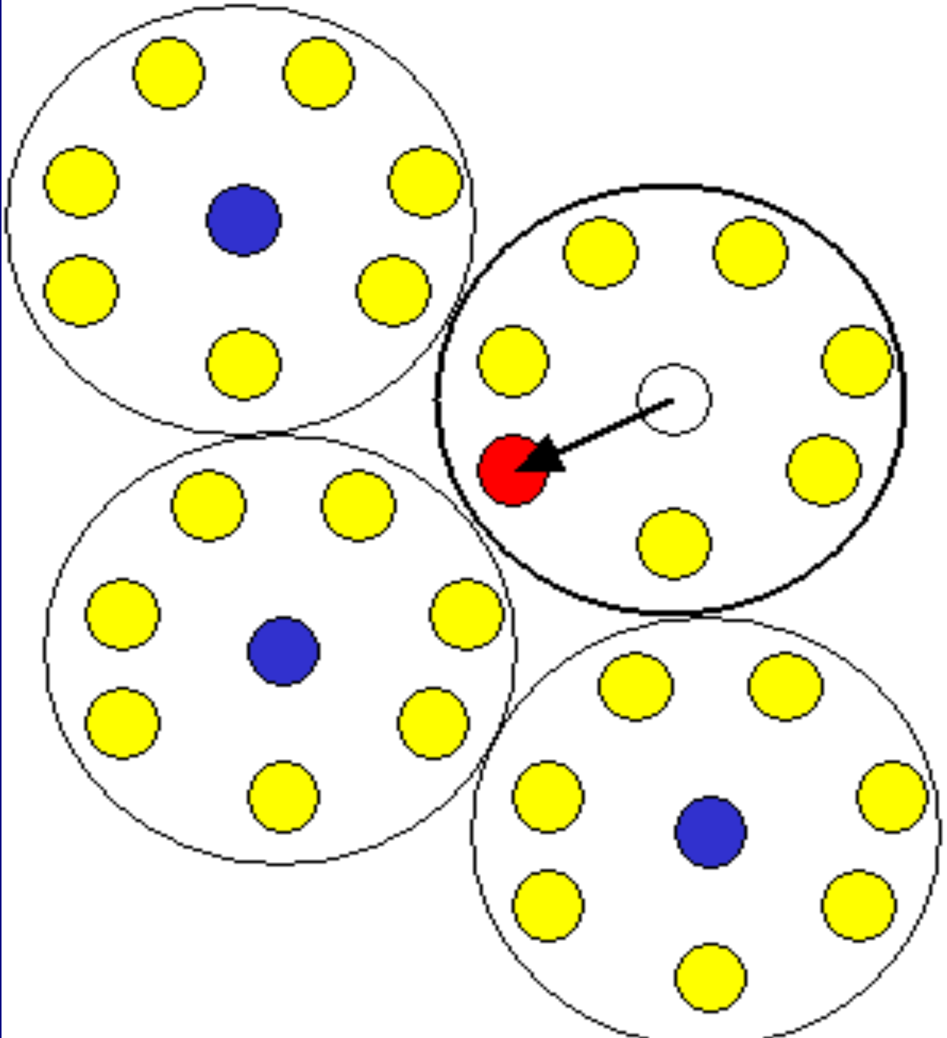
At most one error



Words at distance at least *3*



Decoding



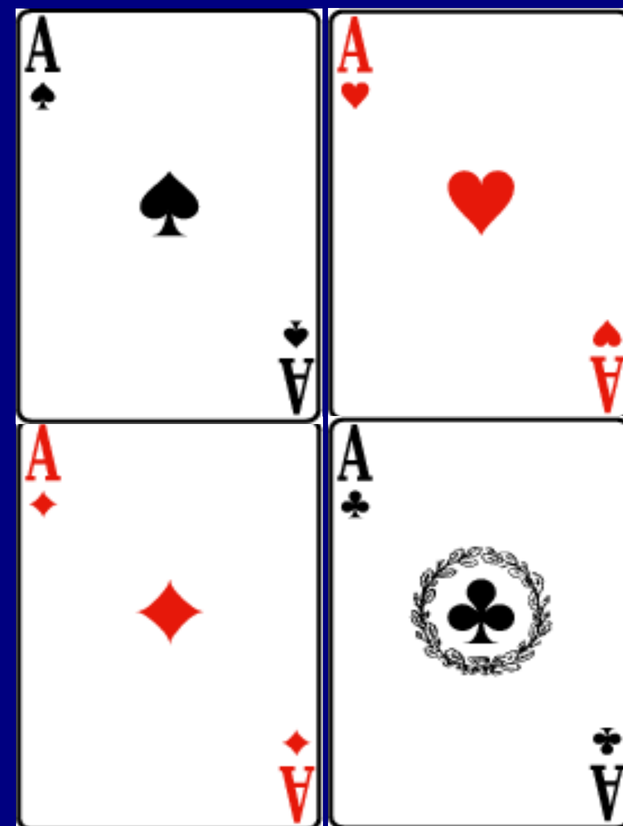
The corrupted word still lies in its original unit sphere. The center of this sphere is the corrected word.

With *4* cards

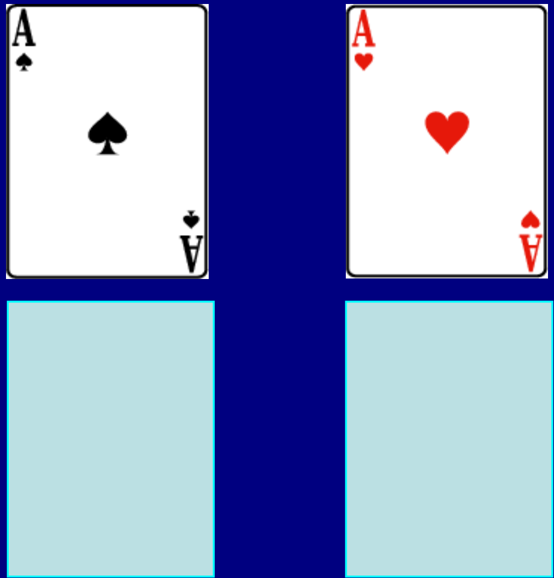
If I repeat my two questions three times each, I need *6* questions

Better way:
5 questions suffice

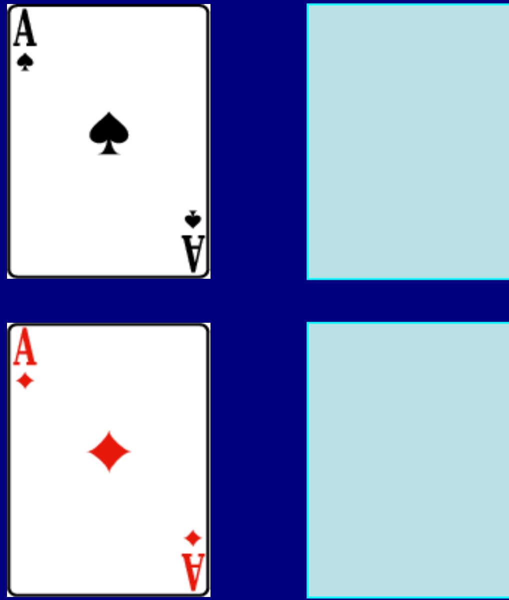
Repeat each of the two previous questions twice and use the parity check bit.



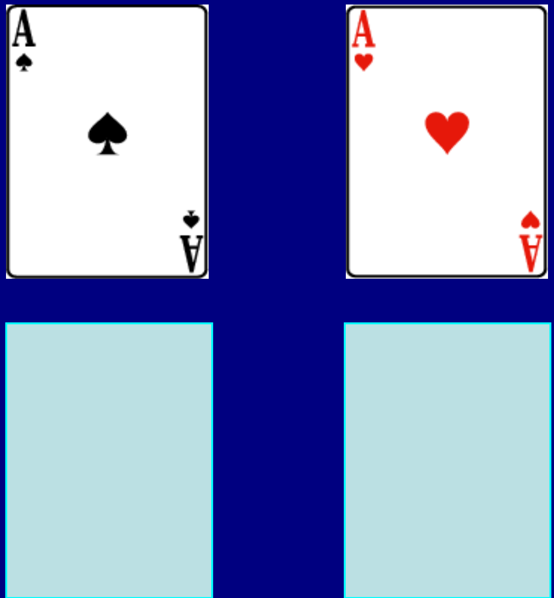
First question:



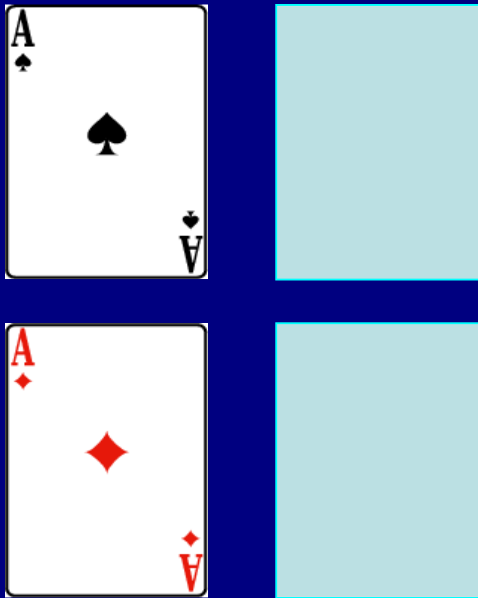
Second question:



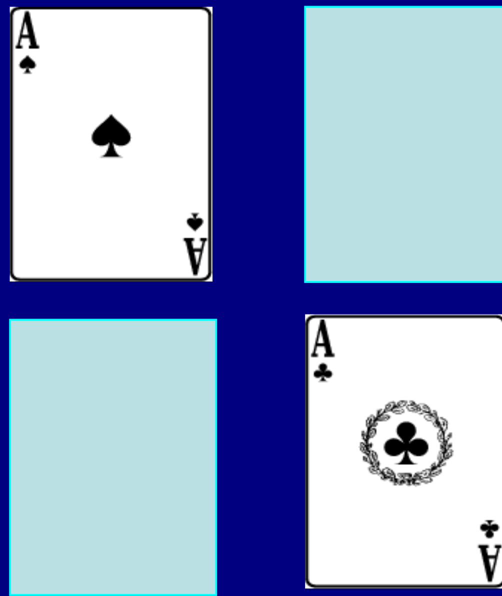
Third question:



Fourth question:



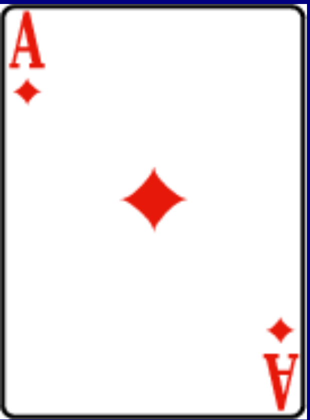
Fifth question:



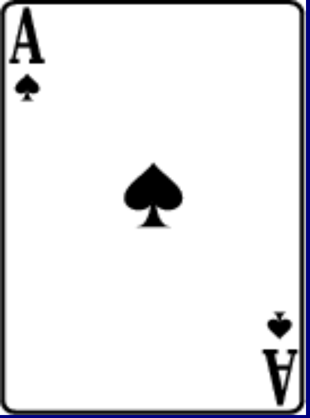
4 cards, *5* questions
it corrects *1* error

4 correct answers: $\boxed{a}$ $\boxed{b}$ $\boxed{a}$ $\boxed{b}$ $\boxed{a+b}$

At most one mistake: you know at least one of a , b



If you know (a or b) and $a+b$
then you know a and b



Length *5*

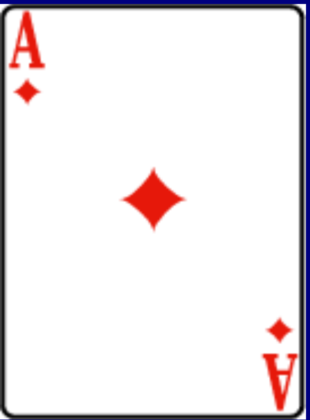
- *2* data bits, *3* check bits

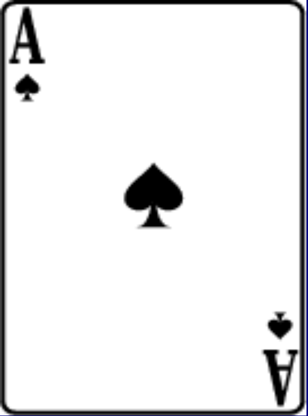
- *4* codewords: $a, b, a, b, a+b$

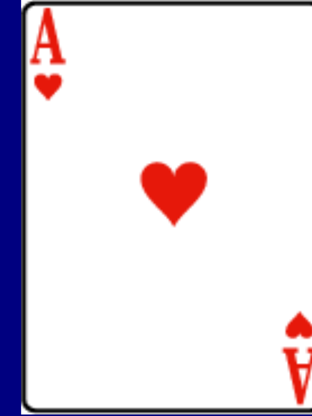
0	0	0	0	0
0	1	0	1	1
1	0	1	0	1
1	1	1	1	0

- Two codewords have distance at least *3*

Rate : *2/5*.







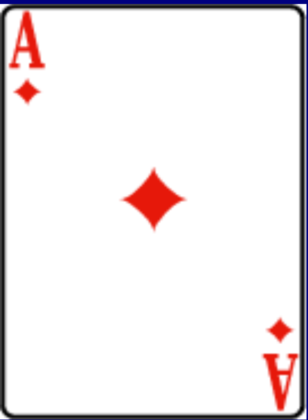
Length *5*

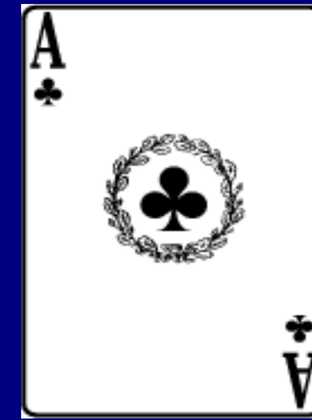
•Number of words $2^5 = 32$

- *4* codewords: *a*, *b*, *a*, *b*, *a+b*
- Each has *5* neighbours
- Each of the *4* balls of radius *1* has *6* elements
- There are ***24*** possible answers containing at most *1* mistake
- ***8*** answers are not possible:

a, *b*, *a+1*, *b+1*, *c*

(at distance ≥ 2 of each codeword)





With *8* Cards

With *8* cards and
6 questions
I can correct
one error

8 cards, *6* questions, corrects *1* error

- Ask the three questions giving the right answer if there is no error, then use the parity check for questions (1,2), (1,3) and (2,3).
- Right answers :

(a, b, c, a+b, a+c, b+c)

with a, b, c replaced by 0 or 1

First question

a

Second question

b

Third question

c

Fourth question

a+b

Fifth question

a+c

Sixth question

b+c


8 cards, *6* questions
Corrects *1* error


- *8* correct answers: a, b, c, $a+b$, $a+c$, $b+c$
  - from a, b, $a+b$ you know whether a and b are correct
- If you know a and b then among c, $a+c$, $b+c$ there is at most one mistake, hence you know c




8 cards, *6* questions
Corrects *1* error

3 data bits,
3 check bits

- *8* codewords: a, b, c, $a+b$, $a+c$, $b+c$

0 0 0	0 0 0	1 0 0	1 1 0
0 0 1	0 1 1	1 0 1	1 0 1
0 1 0	1 0 1	1 1 0	0 1 1
0 1 1	1 1 0	1 1 1	0 0 0

Two codewords
have distance
at least *3*

Rate : *1/2*.

Length 6

•Number of words $2^6 = \mathbf{64}$

- 8 codewords: a, b, c, $a+b$, $a+c$, $b+c$
- Each has 6 neighbours
- Each of the 8 balls of radius 1 has 7 elements
- There are $\mathbf{56}$ possible answers containing at most 1 mistake
- $\mathbf{8}$ answers are not possible:

$$a,\ b,\ c,\ a+b+1,\ a+c+1,\ b+c+1$$

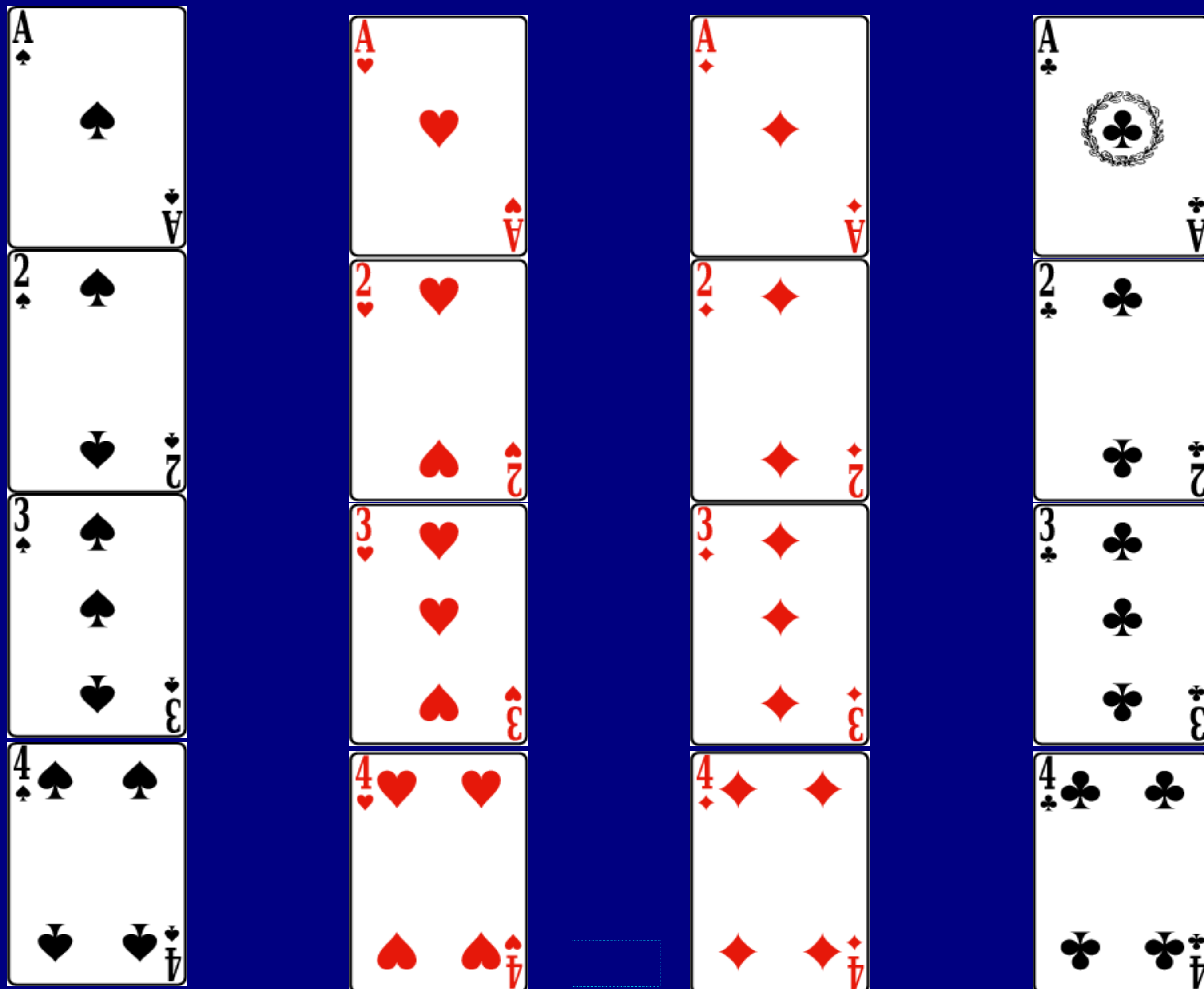
Number of questions

	No error	Detects *1* error	Corrects *1* error
2 cards	*1*	*2*	*3*
4 cards	*2*	*3*	*5*
8 cards	*3*	*4*	*6*
16 cards	*4*	*5*	*?*

Number of questions

	No error	Detects *1* error	Correct *1* error
2 cards	*1*	*2*	*3*
4 cards	*2*	*3*	*5*
8 cards	*3*	*4*	*6*
16 cards	*4*	*5*	*7*

With *16* cards, *7* questions suffice to correct one mistake



Claude Shannon

- In 1948, Claude Shannon, working at Bell Laboratories in the USA, inaugurated the whole subject of coding theory by showing that it was possible to encode messages in such a way that the number of extra bits transmitted was as small as possible. Unfortunately his proof did not give any explicit recipes for these optimal codes.

Richard Hamming

Around the same time, Richard Hamming, also at Bell Labs, was using machines with lamps and relays having an error detecting code. The digits from *1* to *9* were sent on ramps of *5* lamps with two lamps on and three out. There were very frequent errors which were easy to detect and then one had to restart the process.

The first correcting codes

- For his researches, Hamming was allowed to have the machine working during the weekend only, and they were on the automatic mode. At each error the machine stopped until the next Monday morning.
- "If it can detect the error," complained Hamming, "why can't it correct some of them! "

The origin of Hamming's code

- He decided to find a device so that the machine not only would detect the errors but also would correct them.
- In 1950, he published details of his work on explicit error-correcting codes with information transmission rates more efficient than simple repetition.
- His first attempt produced a code in which four data bits were followed by three check bits which allowed not only the detection, but also the correction of a single error.

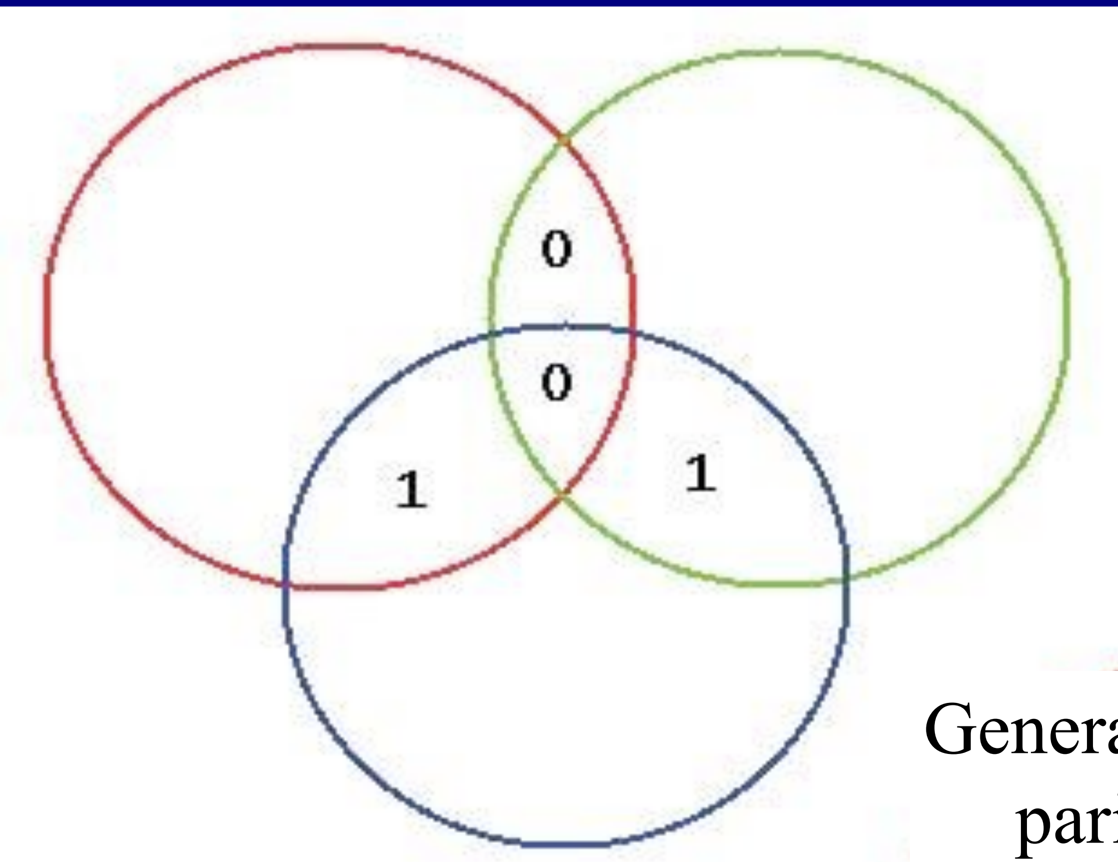
The Bell System Technical Journal

Vol. XXVI April, 1950 No. 2

Error Detecting and Error Correcting Codes

By R. W. HAMMING

The binary code of Hamming (1950)

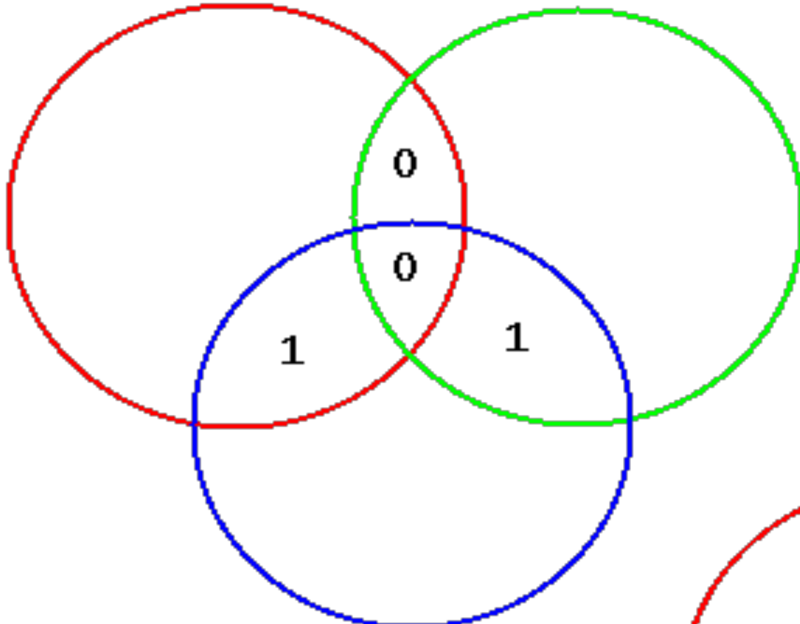


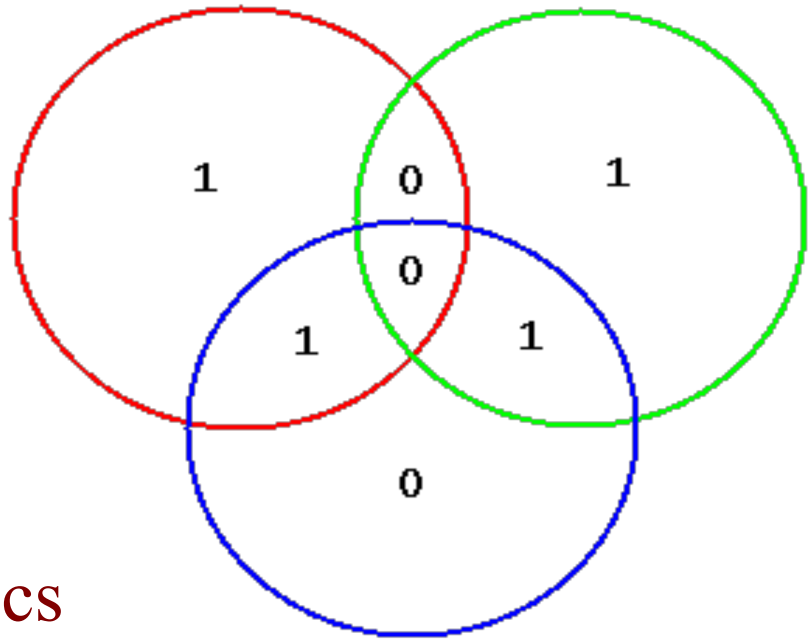
4 previous questions,
3 new ones,
corrects *1* error

Parity check
in each of the three discs

Generalization of the parity check bit

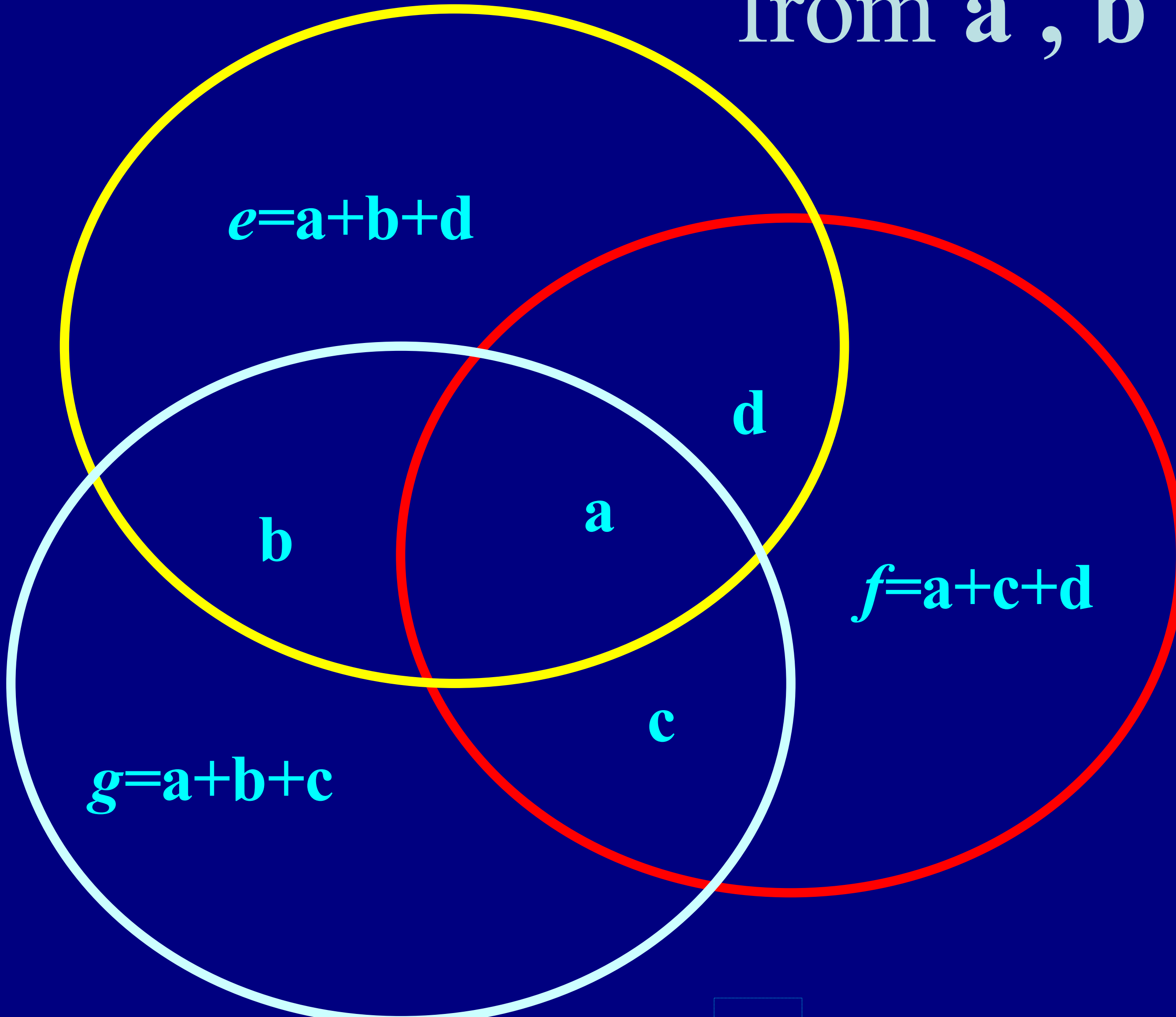
16 cards, *7* questions, corrects *1* error





Parity check
in each of the three discs

How to compute *e* , *f* , *g* from **a , b , c , d**



Hamming code

Words of length *7*

Codewords: ($16=2^4$ among $128=2^7$)

(a, b, c, d, e, f, g)

with

$$e=a+b+d$$

$$f=a+c+d$$

$$g=a+b+c$$

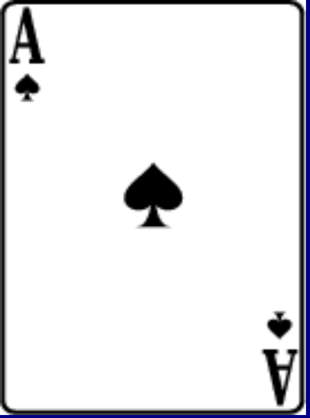
4 data bits, *3* check bits

Rate: 4/7

16 codewords of length *7*

0 0 0 0 *0 0 0*	1 0 0 0 *1 1 1*
0 0 0 1 *1 1 0*	1 0 0 1 *0 0 1*
0 0 1 0 *0 1 1*	1 0 1 0 *1 0 0*
0 0 1 1 *1 0 1*	1 0 1 1 *0 1 0*
0 1 0 0 *1 0 1*	1 1 0 0 *0 1 0*
0 1 0 1 *0 1 1*	1 1 0 1 *1 0 0*
0 1 1 0 *1 1 0*	1 1 1 0 *0 0 1*
0 1 1 1 *0 0 0*	1 1 1 1 *1 1 1*

Two distinct codewords have at least three distinct letters

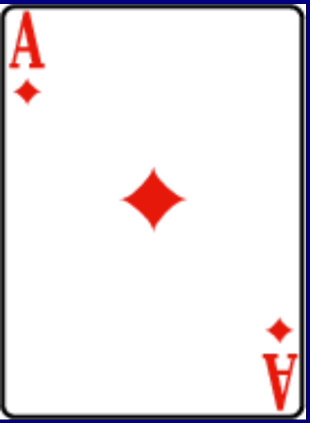



Words of length *7*

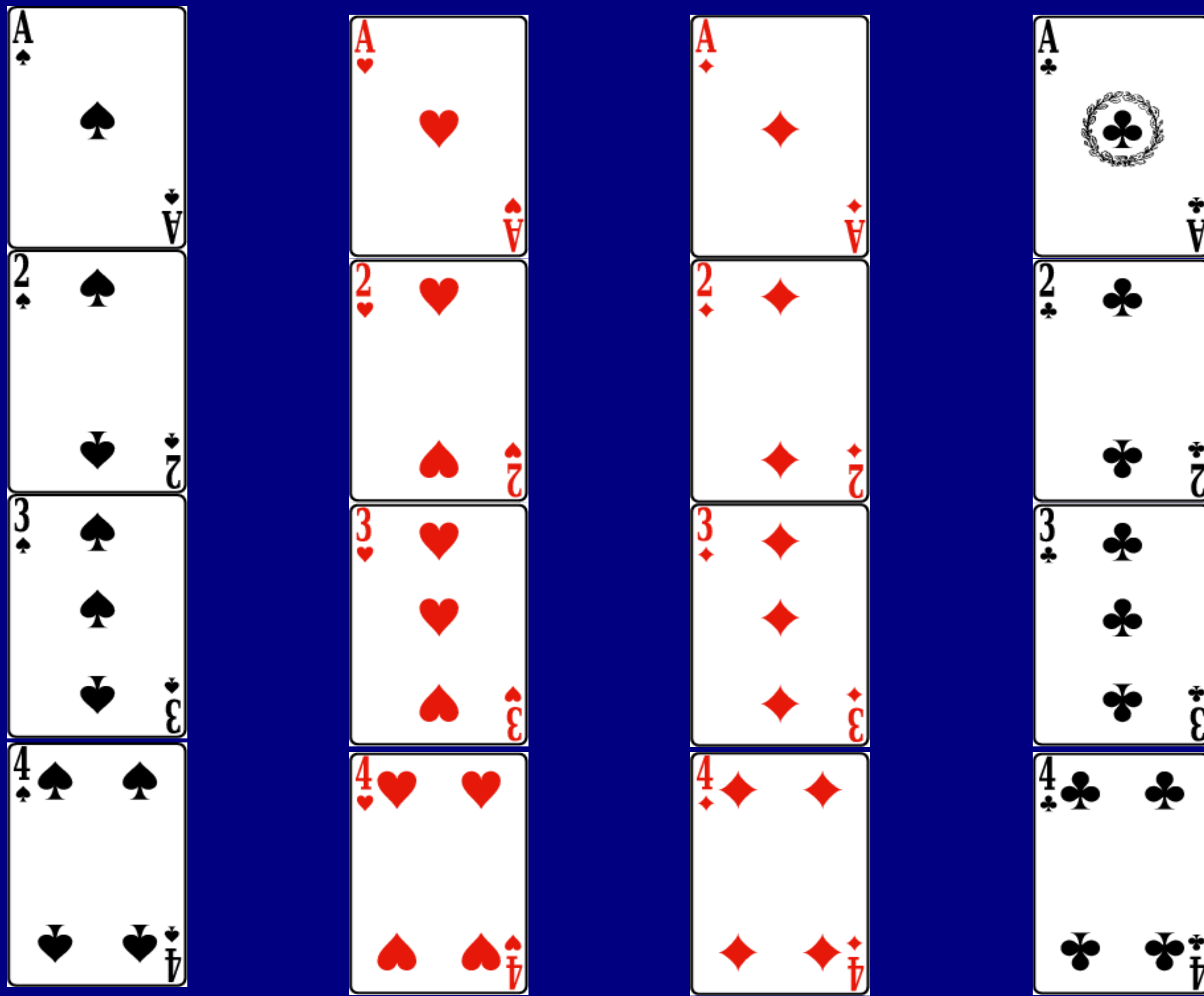
•Number of words: $2^7 = \mathbf{128}$

Hamming code (1950):

- There are $16 = 2^4$ codewords
- Each has *7* neighbours
- Each of the *16* balls of radius *1* has $8 = 2^3$ elements
- Any of the $8 \times 16 = \mathbf{128}$ words is in exactly one ball (perfect packing)




16 cards , *7* questions, corrects one mistake



Replace the cards by labels from *0* to *15* and write the binary expansions of these:

0000, 0001, 0010, 0011

0100, 0101, 0110, 0111

1000, 1001, 1010, 1011

1100, 1101, 1110, 1111

Using the Hamming code, get 7 digits.

Select the questions so that Yes=0 and No=1

7 questions to find the selected number in $\{0,1,2,\ldots,15\}$ with one possible wrong answer

- Is the first binary digit 0?
- Is the second binary digit 0?
- Is the third binary digit 0?
- Is the fourth binary digit 0?
- Is the number in $\{1,2,4,7,9,10,12,15\}$?
- Is the number in $\{1,2,5,6,8,11,12,15\}$?
- Is the number in $\{1,3,4,6,8,10,13,15\}$?

Hat problem with *7* people

For *7* people in the room in place of *3*,
which is the best strategy
and its probability of winning?

Answer:
the best strategy gives a
probability of winning of *7/8*

The Hat Problem with 7 people

- The team bets that the distribution of the hats does not correspond to the *16* elements of the Hamming code
- Loses in *16* cases (they all fail)
- Wins in *128-16=112* cases (one of them bets correctly, the *6* others abstain)
- Probability of winning: *112/128=7/8*

Winning at the lottery

Head or Tails

Toss a coin *7* consecutive times

There are $2^7 = 128$ possible sequences of results

How many bets are required in such a way that you are sure one at least of them has at most one wrong answer?

Tossing a coin *7* times

- Each bet has all correct answers once every *128* cases.
- It has just one wrong answer *7* times: either the first, second, … seventh guess is wrong.
- So it has at most one wrong answer *8* times among *128* possibilities.

Tossing a coin *7* times

- Now $128 = 8 \times 16$.
- Therefore you cannot achieve your goal with less than *16* bets.
- Coding theory tells you how to select your *16* bets, exactly one of them will have at most one wrong answer.

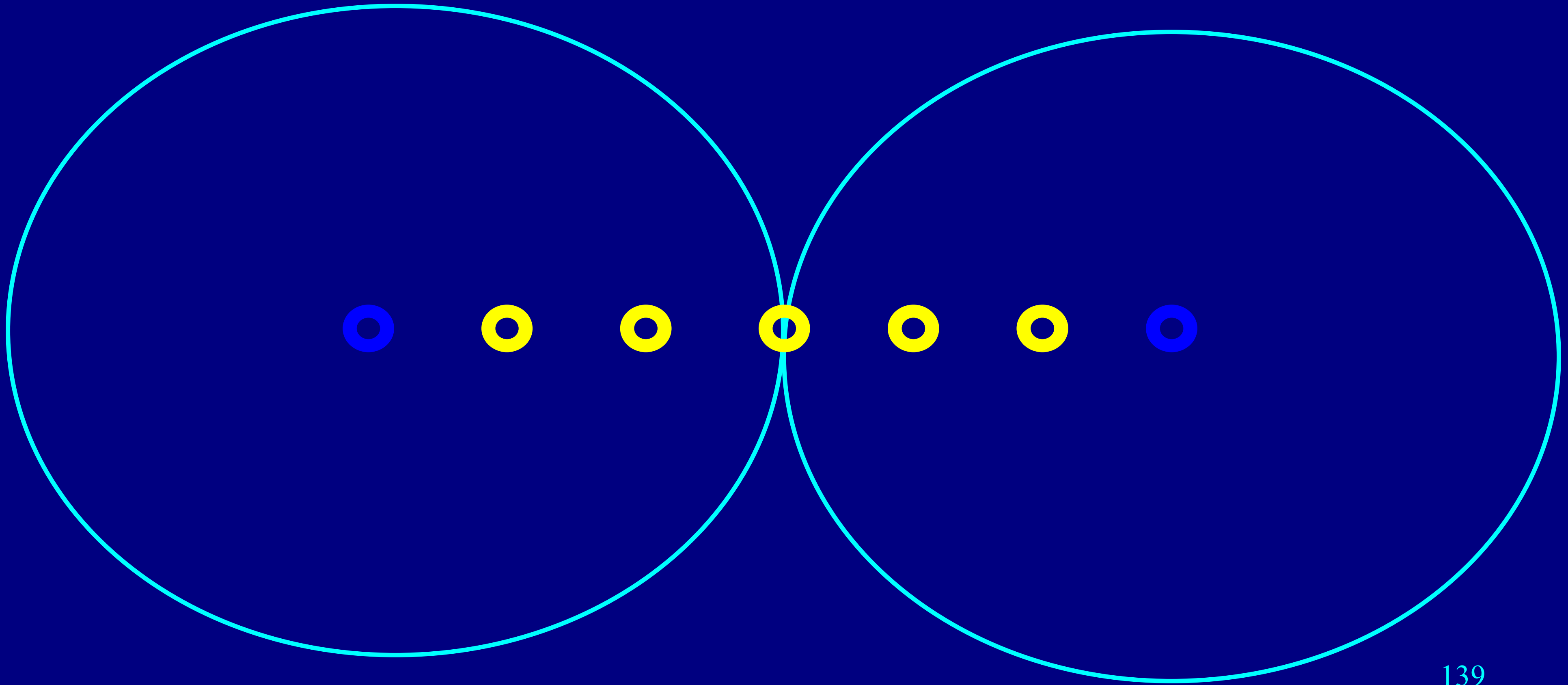
Principle of codes detecting n errors:

Two distinct codewords have
at least $n+1$ distinct letters

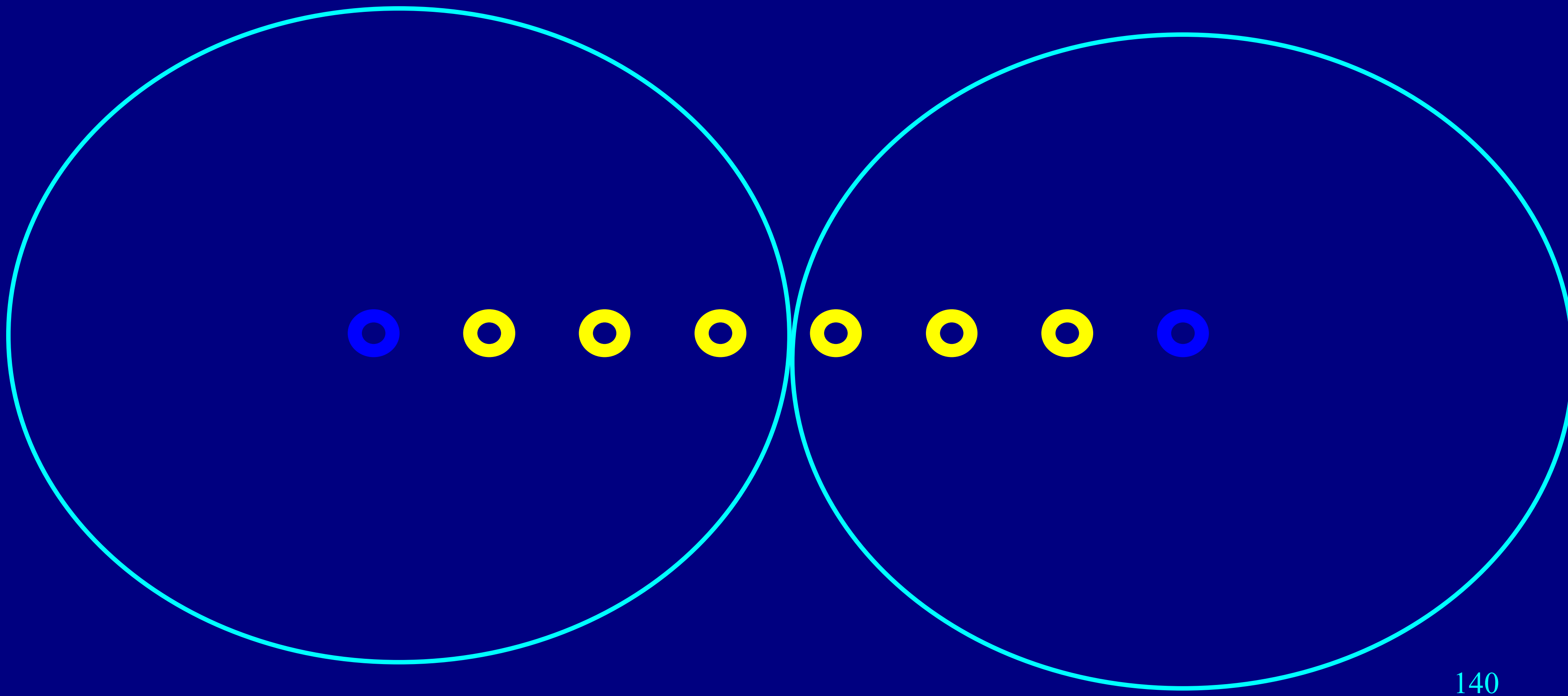
Principle of codes correcting n errors:

Two distinct codewords have
at least $2n+1$ distinct letters

Hamming balls of radius *3*
Distance *6*, detects *5* errors, corrects *2* errors



Hamming balls of radius *3*
Distance *7*, corrects *3* errors



Golay code on $\{0,1\} = \boldsymbol{F}_2$

Words of length *23*, there are 2^{23} words

12 data bits, *11* control bits,

distance *7,* corrects *3* errors

2^{12} codewords, each ball of radius *3* has

$$\binom{23}{0} + \binom{23}{1} + \binom{23}{2} + \binom{23}{3}$$

$$= 1 + 23 + 253 + 1771 = 2048 = 2^{11}$$

elements:

Perfect packing

Golay code on $\{0,1,2\} = \boldsymbol{F}_3$

Words of length 11, there are 3^{11} words

6 data bits, 5 control bits,

distance 5, corrects 2 errors

3^6 codewords, each ball of radius 2 has

$$\binom{11}{0} + 2\binom{11}{1} + 2^2\binom{11}{2}$$

$$=1+22+220=243=3^5$$

elements:

Perfect packing

SPORT TOTO: the oldest error correcting code

- A match between two players (or teams) may give three possible results: either player *1* wins, or player *2* wins, or else there is a draw (write *0*).

- There is a lottery, and a winning ticket needs to have at least *3* correct bets for *4* matches. How many tickets should one buy to be sure to win?

4 matches, *3* correct forecasts

- For *4* matches, there are $3^4 = 81$ possibilities.
- A bet on *4* matches is a sequence of *4* symbols $\{0, 1, 2\}$. Each such ticket has exactly 3 correct answers *8* times.
- Hence each ticket is winning in *9* cases.
- Since $9 \times 9 = 81$, a minimum of *9* tickets is required to be sure to win.

Finnish Sport Journal, 1932

9 tickets

0 0 0 0	*1 0 1 2*	*2 0 2 1*
0 1 1 1	*1 1 2 0*	*2 1 0 2*
0 2 2 2	*1 2 0 1*	*2 2 1 0*

Rule: a, b, $a+b$, $2a+b$ modulo 3

This is an error correcting code on the alphabet $\{0, 1, 2\}$ with rate $1/2$

Perfect packing of $F_3^{\,4}$ with *9* balls radius *1*

(0,0,0,2) *(0,0,1,0)*

(0,0,0,1) *(0,0,2,0)*

(0,0,0,0)

(0,1,0,0) *(1,0,0,0)*

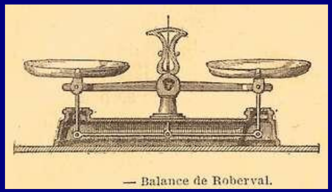
(0,2,0,0) *(2,0,0,0)*

A fake pearl

- Among m pearls all looking the same, there are m-1 genuine identical ones, having the same weight, and a fake one, which is lighter.
- You have a balance which enables you to compare the weight of two objects.
- How many weighings do you need in order to detect the fake pearl?

Each weighing produces three possible results

The fake pearl is not weighed

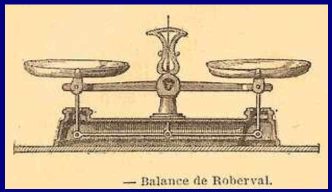


The fake pearl is on the right

The fake pearl is on the left

3 pearls:
put *1* on the left and *1* on the right

The fake pearl is
not weighed

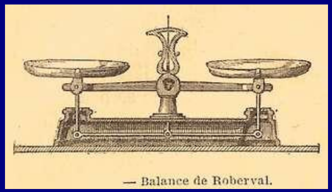


The fake pearl
is on the right

The fake pearl
is on the left

9 pearls:
put *3* on the left and *3* on the right

The fake pearl is not weighed



The fake pearl is on the right

The fake pearl is on the left

Each weighing enables one to select one third of the collection where the fake pearl is

- With *3* pearls, one weighing suffices.
- With *9* pearls, select *6* of them, put *3* on the left and *3* on the right.
- Hence you know a set of *3* pearls including the fake one. One more weighing yields the result.
- Therefore with *9* pearls *2* weighings suffice.

A protocole where each weighing is independent of the previous results

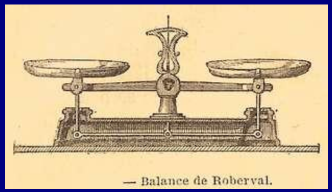
- Label the *9* pearls from *0* to *8*, next replace the labels by their expansion in basis *3*.

0 0	*0 1*	*0 2*
1 0	*1 1*	*1 2*
2 0	*2 1*	*2 2*

- For the first weighing, put on the right the pearls whose label has first digit *1* and on the left those with first digit *2*.

One weighing= one digit 0, 1 or 2

The fake pearl is not weighed



0

The fake pearl is on the right

1

The fake pearl is on the left

2

Result of two weighings

- Each weighing produces one among three possible results: either the fake pearl is not weighed *0*, or it is on the left *1*, or it is on the right *2*.
- The two weighings produce a two digits number in basis *3* which is the label of the fake pearl.

81 pearls including a lighter one

- Assume there are *81* pearls including *80* genuine identical ones, and a fake one which is lighter. Then 4 weighings suffice to detect the fake one.
- For 3^n pearls including a fake one, n weighings are necessary and sufficient.

And if one of the weighings may be erroneous?

- Consider again *9* pearls. If one of the weighings may produce a wrong answer, then 4 four weighings suffice to detect the fake pearl.
- The solution is given by Sport Toto: label the *9* pearls using the *9* tickets.

Labels of the 9 pearls

a, b, $a+b$, $2a+b$ modulo 3

0 0 0 0	*1 0 1 2*	*2 0 2 1*
0 1 1 1	*1 1 2 0*	*2 1 0 2*
0 2 2 2	*1 2 0 1*	*2 2 1 0*

Each weighing corresponds to one
of the four digits. Accordingly,
put on the left the three pearls with digits *1*
And on the right the pearls with digit *2*

THANK YOU !

Michel Waldschmidt

Sorbonne Université

http://webusers.imj-prg.fr/~michel.waldschmidt//