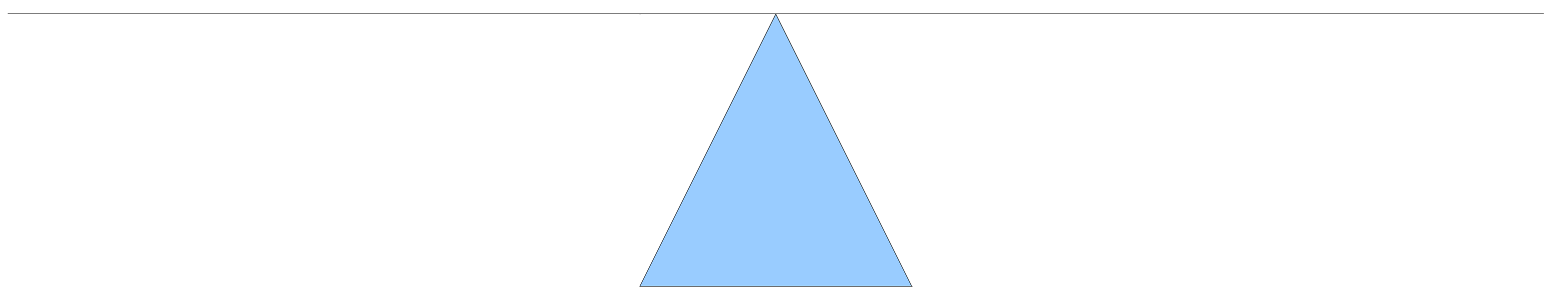
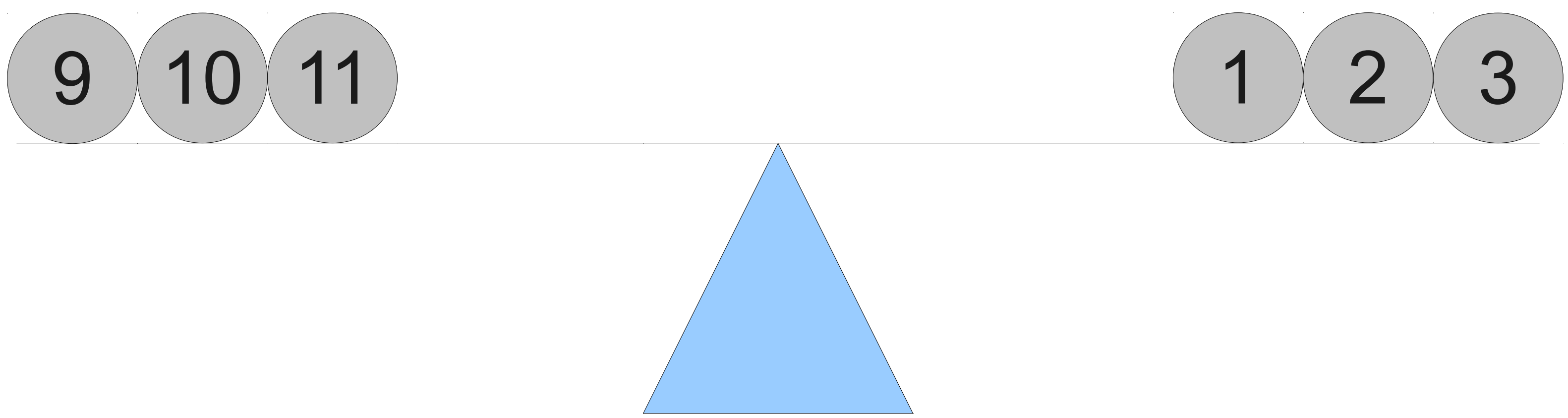
Fun With Number Systems

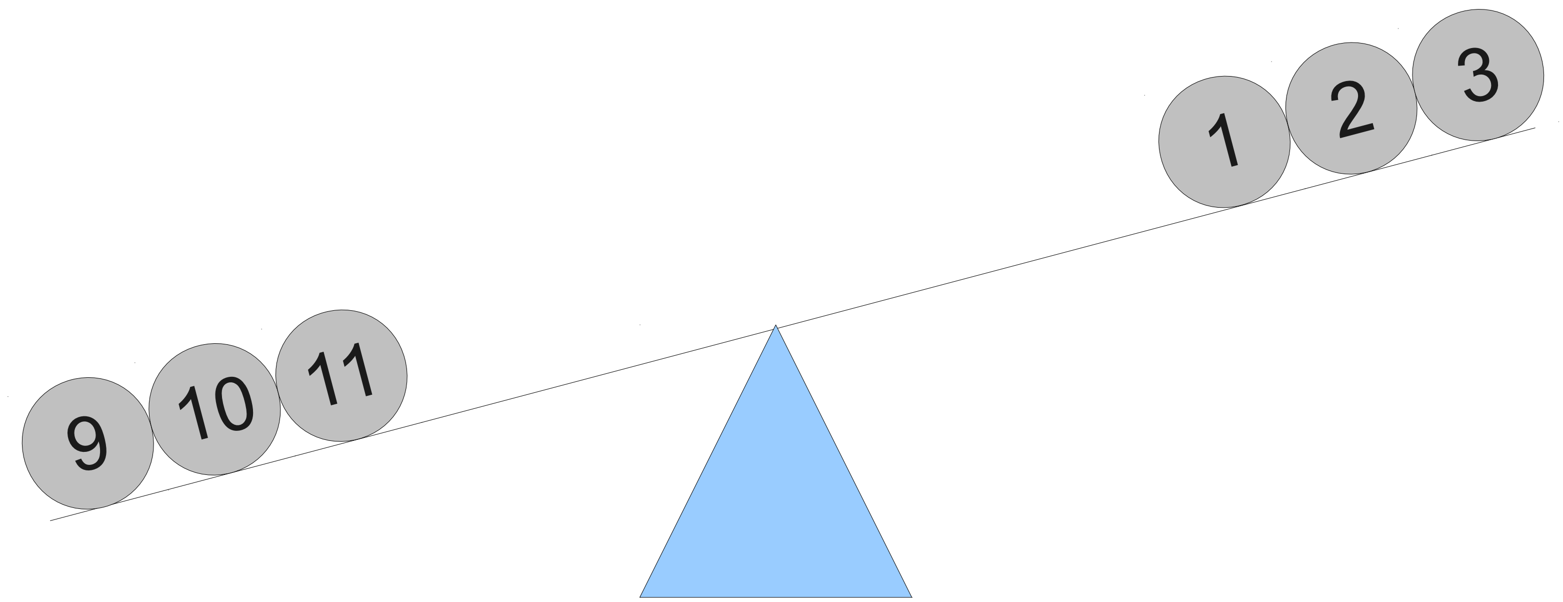
Or: How Knowing Number Systems Can Help You Interview Better

Finding the Odd Ball



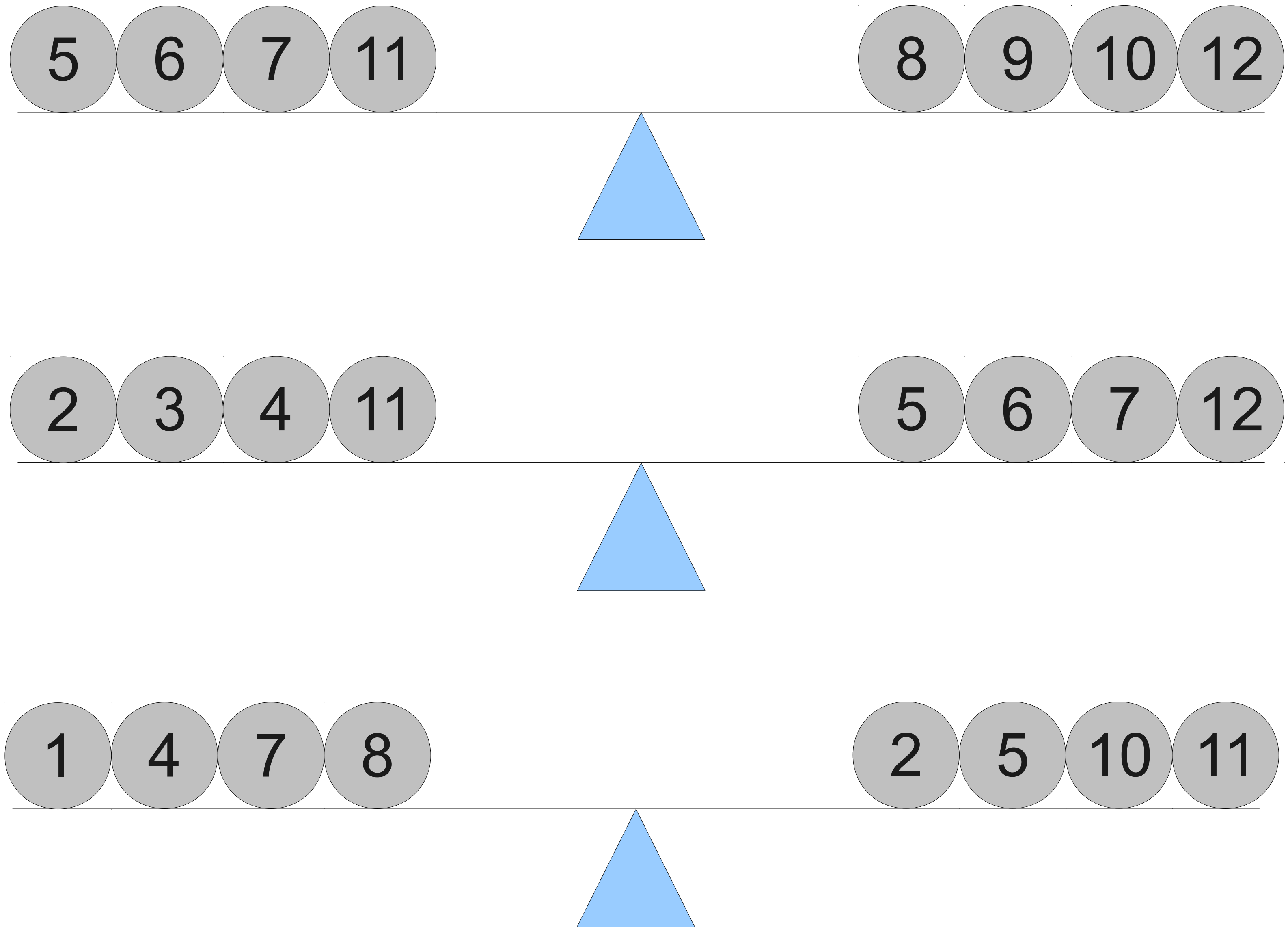


9
10
11
1
2
3


9
10
11
1
2
3

Goal: Find the odd ball and whether it's heavier or lighter in three weighings.

The Solution



Balanced Ternary

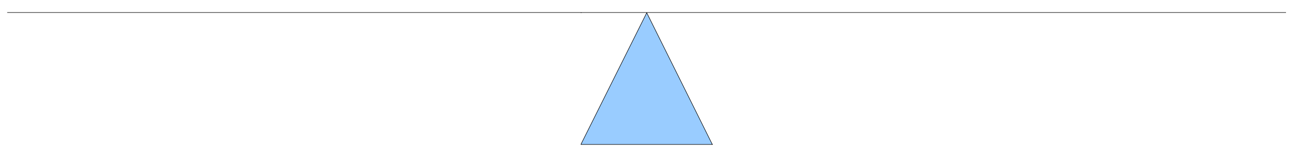
- Number system for encoding three-way comparisons.
- Each digit corresponds to a power of three.
- Digits are -1, 0, +1.
  - For notational simplicity, will use -, 0, +.
- Example: +0–0
  - $1 \times 3^3 + 0 \times 3^2 - 1 \times 3^1 + 0 \times 3^0 = 24$
- Example: ––++
  - $-1 \times 3^3 - 1 \times 3^2 + 1 \times 3^1 + 1 \times 3^0 = -32$

-12 to +12 in Balanced Ternary

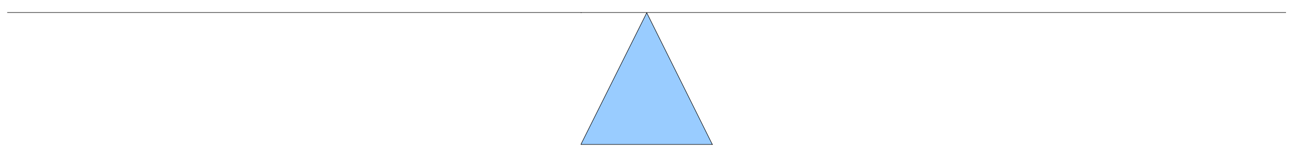
		0	000
-1	00-	1	00+
-2	0-+	2	0+-
-3	0-0	3	0+0
-4	0--	4	0++
-5	-++	5	+--
-6	-+0	6	+-0
-7	-+-	7	+-+
-8	-0+	8	+0-
-9	-00	9	+00
-10	-0-	10	+0+
-11	--+	11	++-
-12	--0	12	++0

1	00+
2	0+−
3	0+0
4	0++
5	+−−
6	+−0
7	+−+
8	+0−
9	+00
10	+0+
11	++−
12	++0

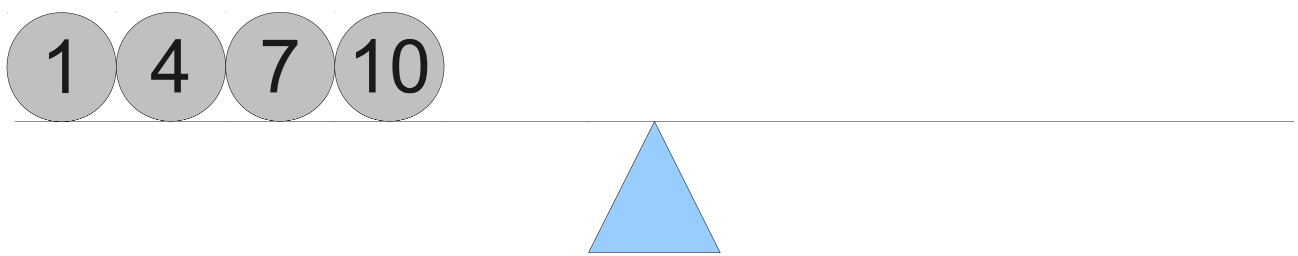
1	00+
2	0+−
3	0+0
4	0++
5	+−−
6	+−0
7	+−+
8	+0−
9	+00
10	+0+
11	++−
12	++0



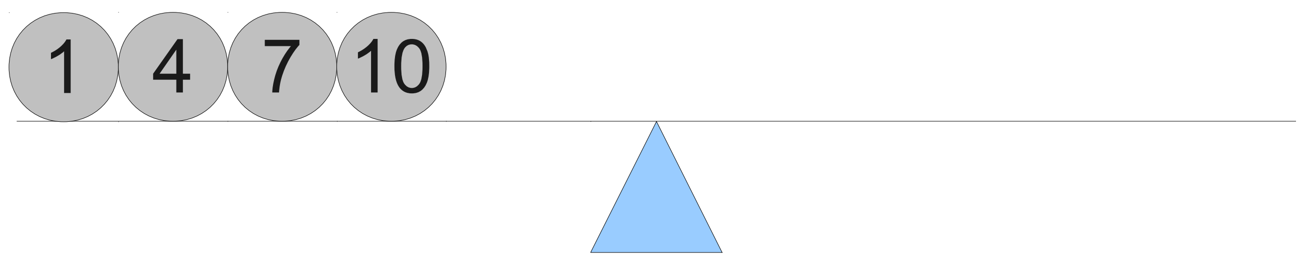
1	00+
2	0+−
3	0+0
4	0++
5	+−−
6	+−0
7	+−+
8	+0−
9	+00
10	+0+
11	++−
12	++0



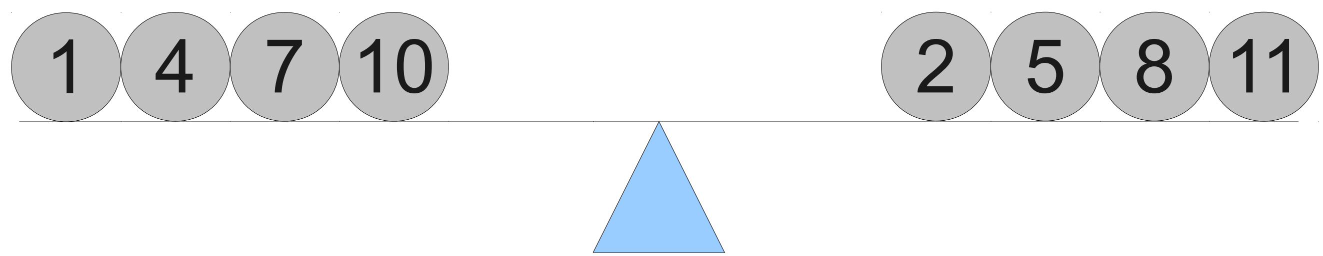
1	00+
2	0+–
3	0+0
4	0++
5	+––
6	+–0
7	+–+
8	+0–
9	+00
10	+0+
11	++–
12	++0



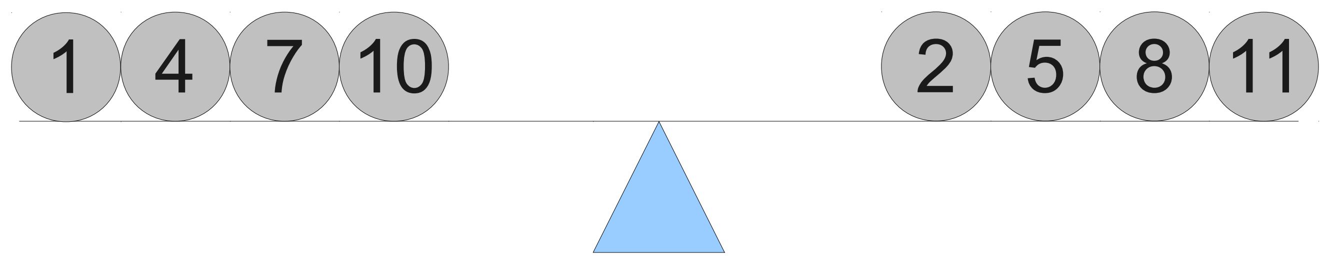
1	00+
2	0+−
3	0+0
4	0++
5	+−−
6	+−0
7	+−+
8	+0−
9	+00
10	+0+
11	++−
12	++0



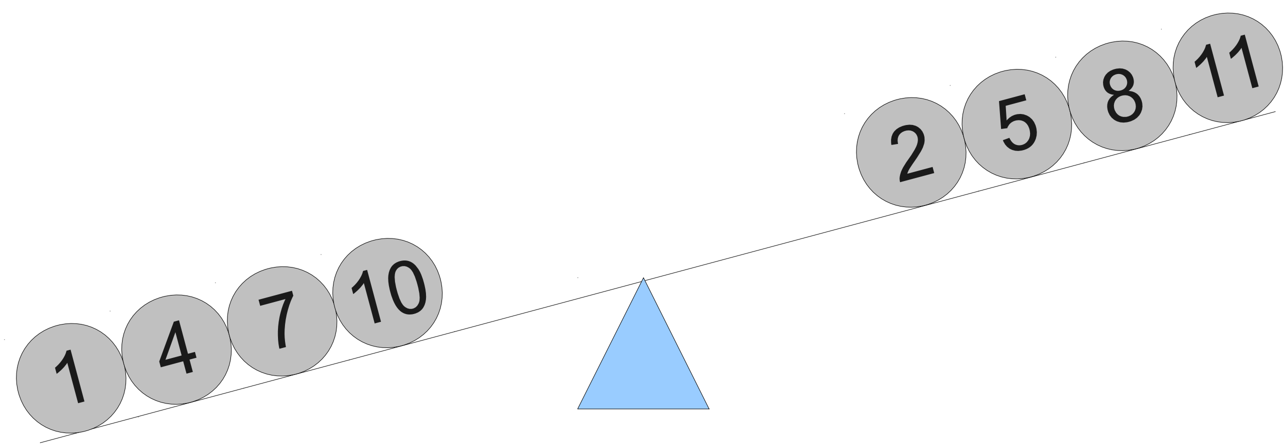
1	00+
2	0+-
3	0+0
4	0++
5	+--
6	+-0
7	+-+
8	+0-
9	+00
10	+0+
11	++-
12	++0



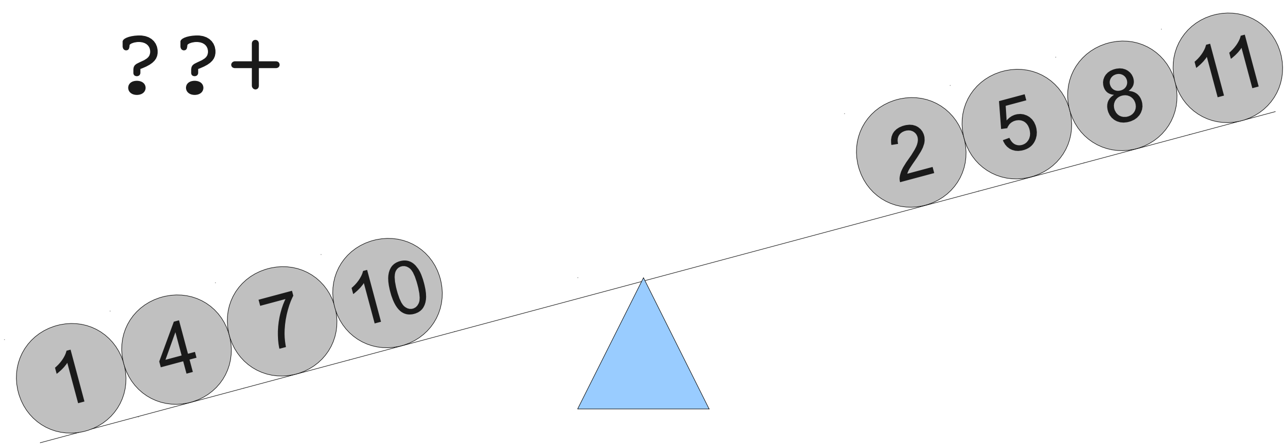
1	00+
2	0+–
3	0+0
4	0++
5	+––
6	+–0
7	+–+
8	+0–
9	+00
10	+0+
11	++–
12	++0



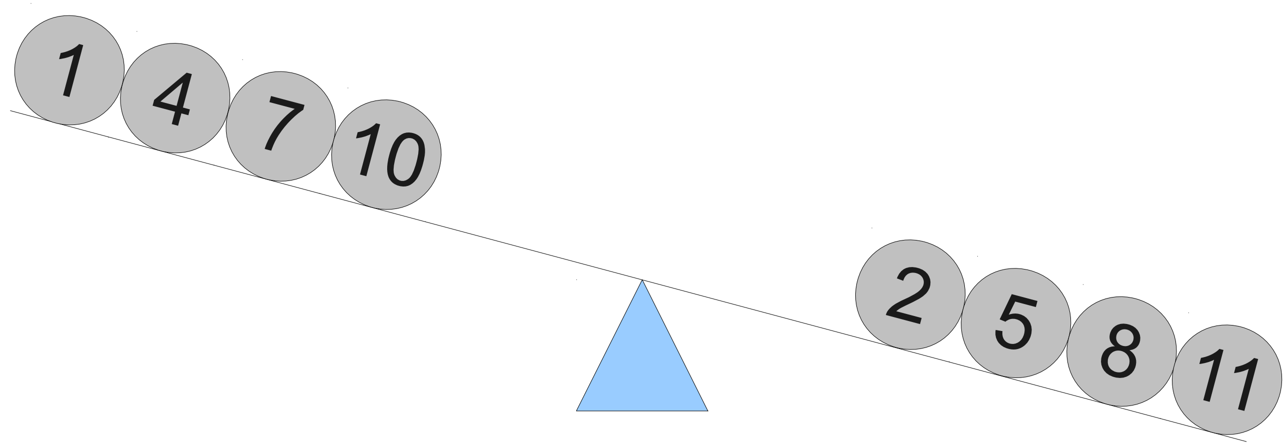
1	00+
2	0+−
3	0+0
4	0++
5	+−−
6	+−0
7	+−+
8	+0−
9	+00
10	+0+
11	++−
12	++0



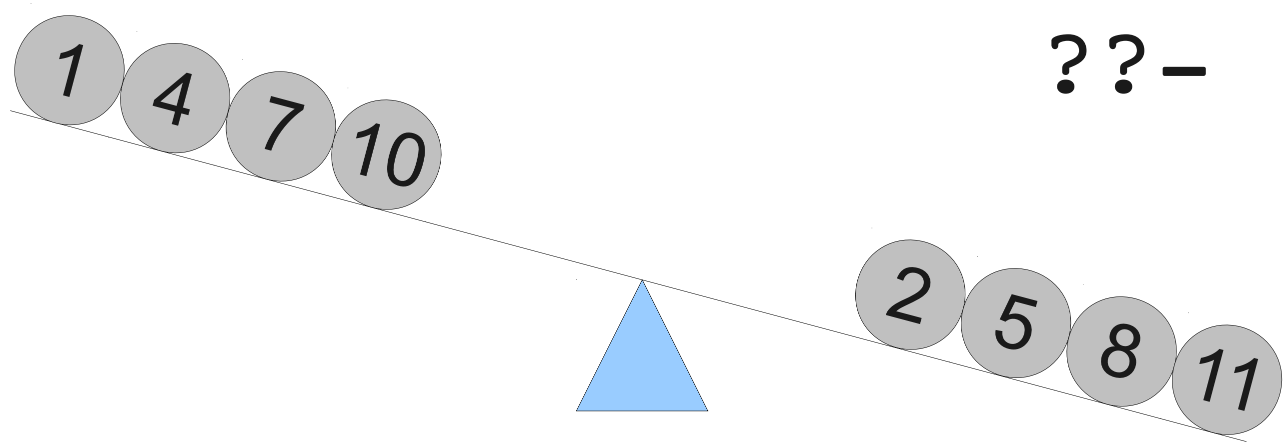
1	00+
2	0+−
3	0+0
4	0++
5	+−−
6	+−0
7	+−+
8	+0−
9	+00
10	+0+
11	++−
12	++0



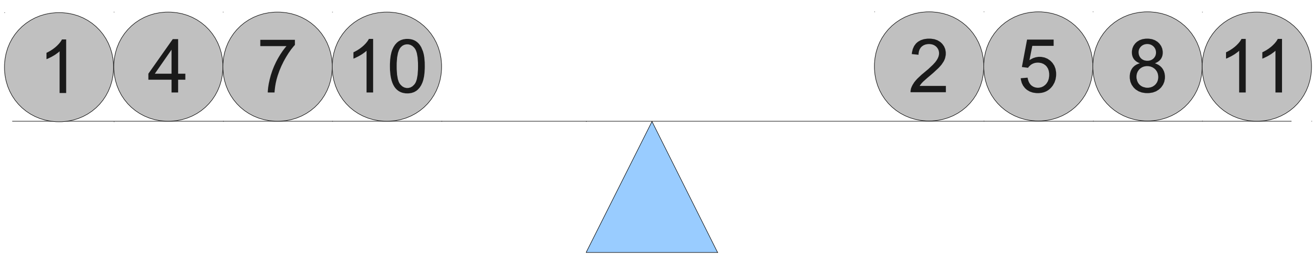
1	00+
2	0+–
3	0+0
4	0++
5	+––
6	+–0
7	+–+
8	+0–
9	+00
10	+0+
11	++–
12	++0



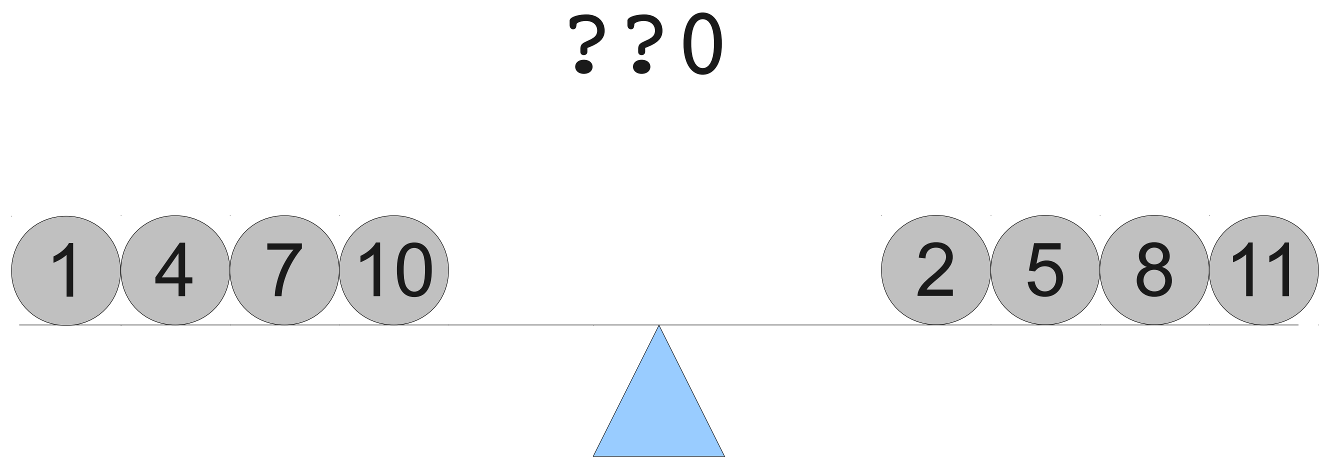
1	00+
2	0+–
3	0+0
4	0++
5	+––
6	+–0
7	+–+
8	+0–
9	+00
10	+0+
11	++–
12	++0



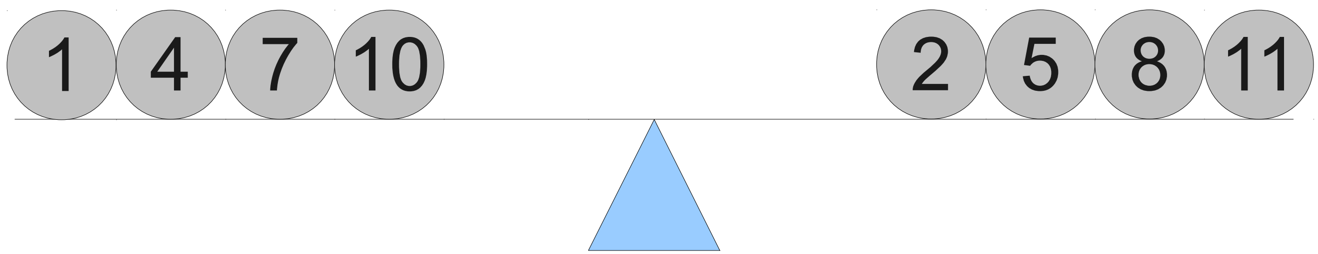
1	00+
2	0+–
3	0+0
4	0++
5	+––
6	+–0
7	+–+
8	+0–
9	+00
10	+0+
11	++–
12	++0



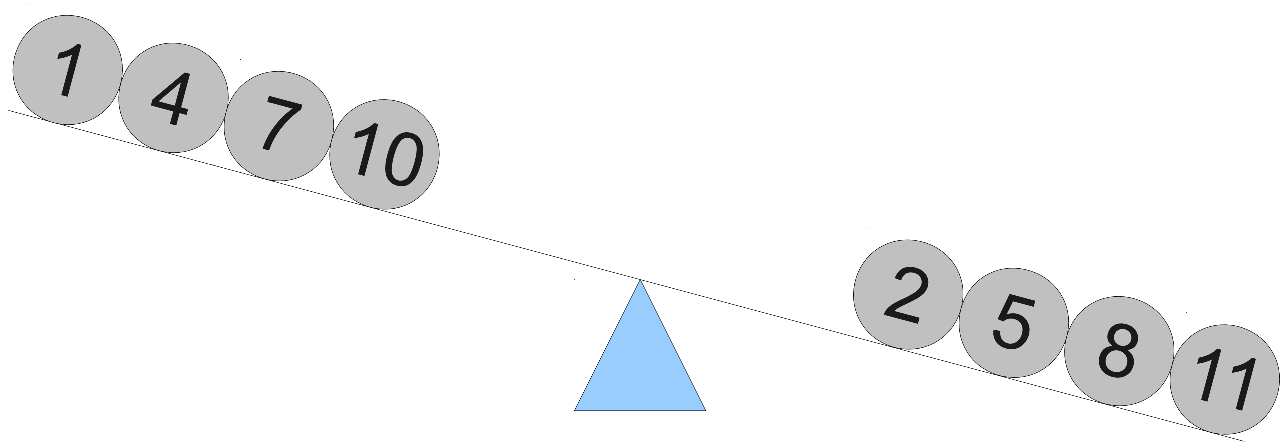
1	00+
2	0+−
3	0+0
4	0++
5	+−−
6	+−0
7	+−+
8	+0−
9	+00
10	+0+
11	++−
12	++0



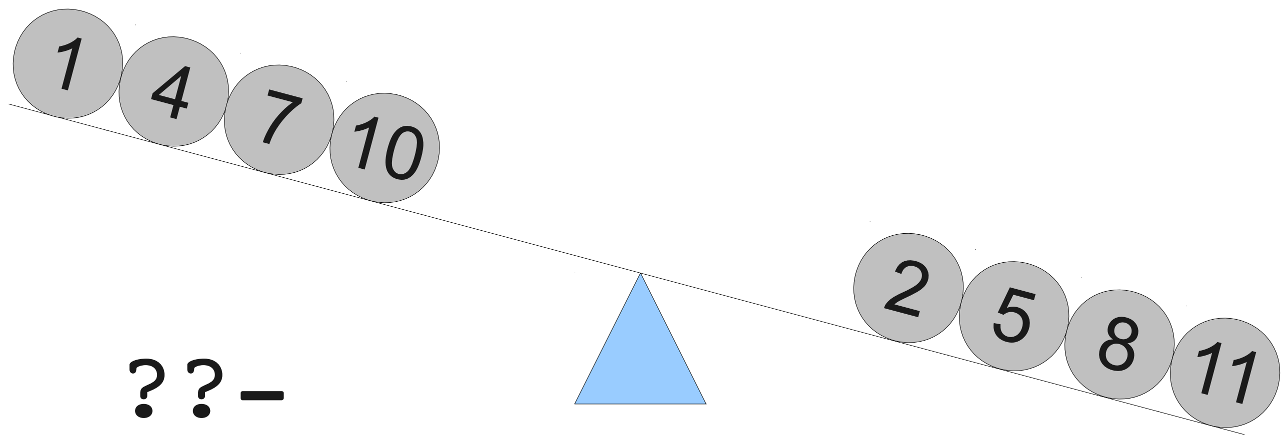
-1	00-
-2	0-+
-3	0-0
-4	0--
-5	-++
-6	-+0
-7	-+-
-8	-0+
-9	-00
-10	-0-
-11	--+
-12	--0



-1	00-
-2	0-+
-3	0-0
-4	0--
-5	-++
-6	-+0
-7	-+-
-8	-0+
-9	-00
-10	-0-
-11	--+
-12	--0

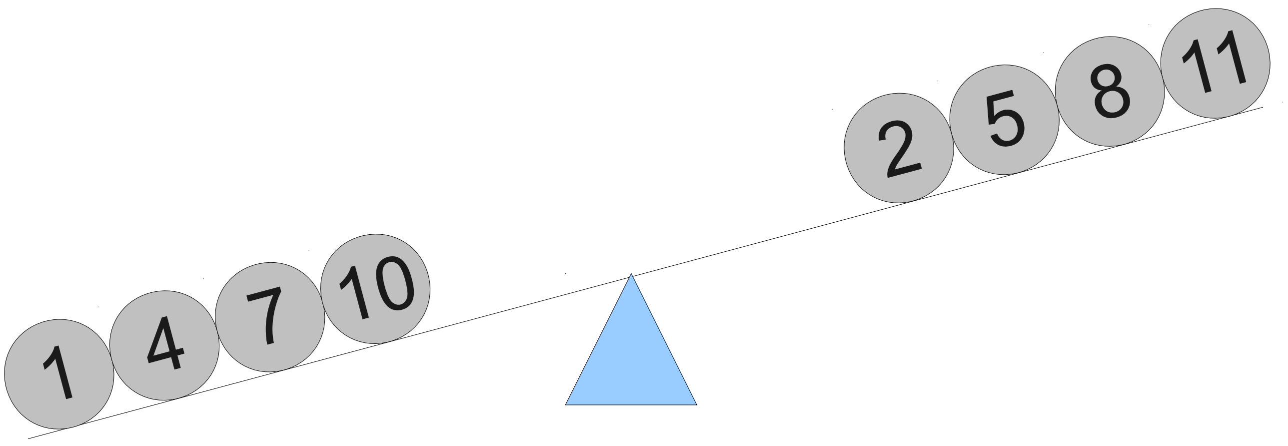


-1	00-
-2	0-+
-3	0-0
-4	0--
-5	-++
-6	-+0
-7	-+-
-8	-0+
-9	-00
-10	-0-
-11	--+
-12	--0

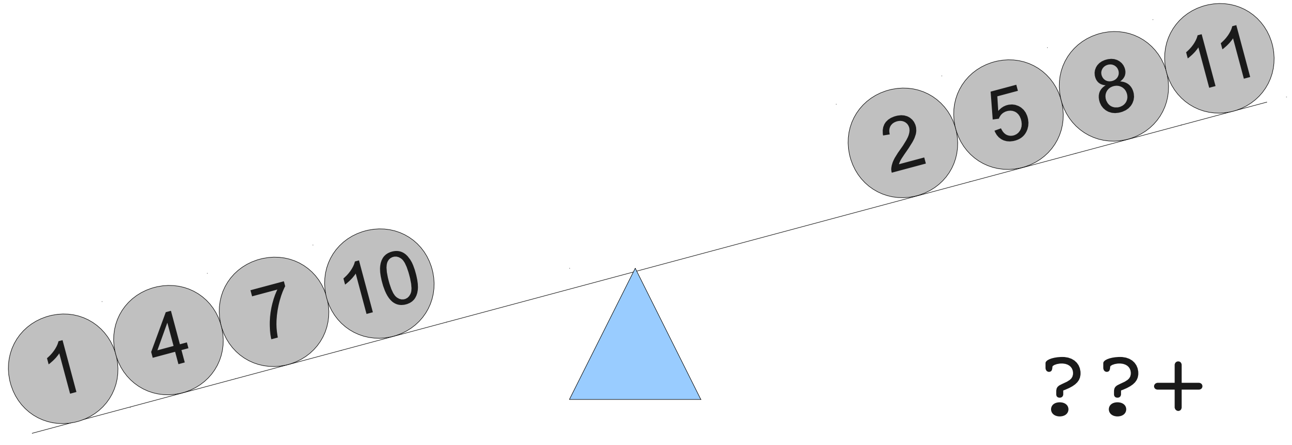


??-

-1	00-
-2	0-+
-3	0-0
-4	0--
-5	-++
-6	-+0
-7	-+-
-8	-0+
-9	-00
-10	-0-
-11	--+
-12	--0

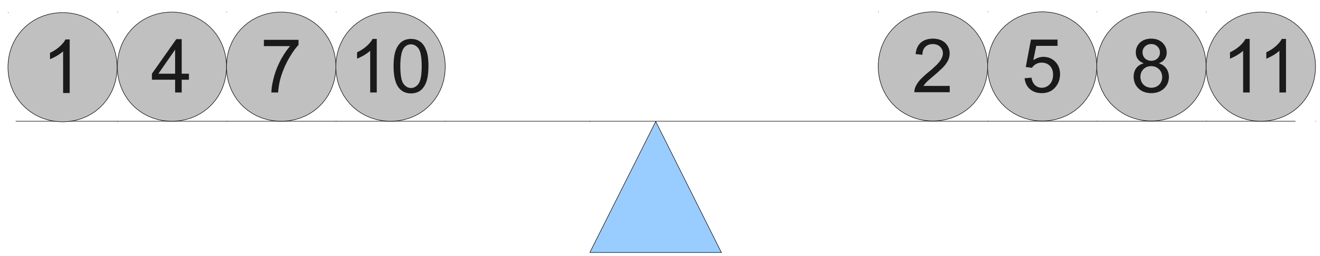


−1	00−
−2	0−+
−3	0−0
−4	0−−
−5	−++
−6	−+0
−7	−+−
−8	−0+
−9	−00
−10	−0−
−11	−−+
−12	−−0

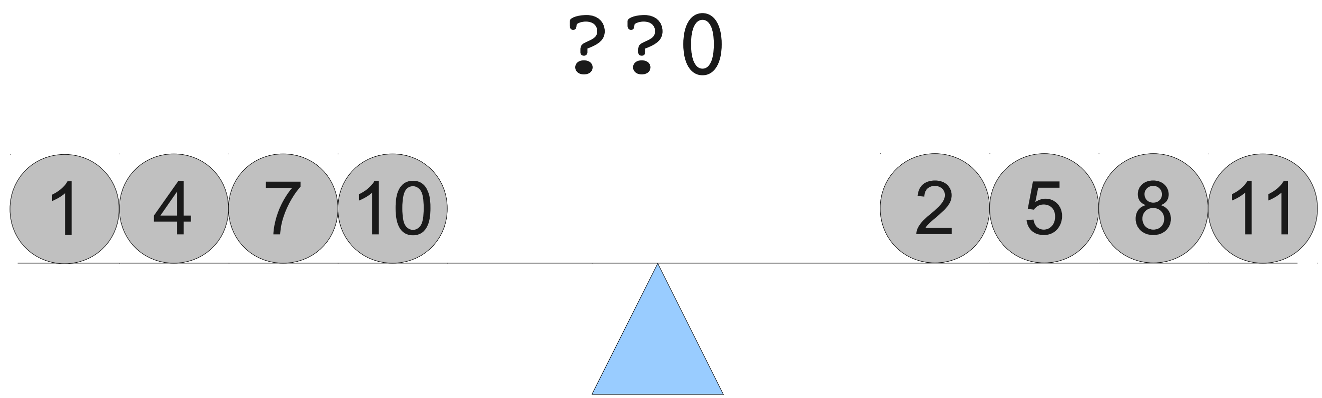


??+

-1	00-
-2	0-+
-3	0-0
-4	0--
-5	-++
-6	-+0
-7	-+-
-8	-0+
-9	-00
-10	-0-
-11	--+
-12	--0



-1	00-
-2	0-+
-3	0-0
-4	0--
-5	-++
-6	-+0
-7	-+-
-8	-0+
-9	-00
-10	-0-
-11	--+
-12	--0

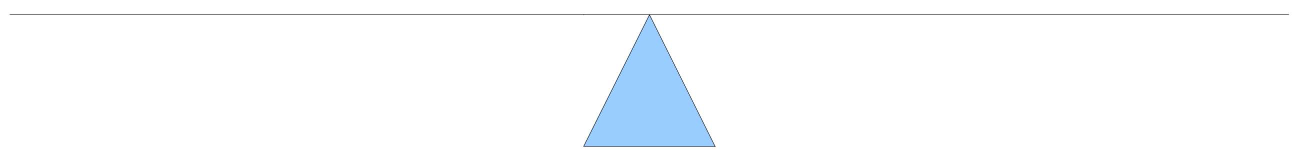


If the left side is heavier, record a +.

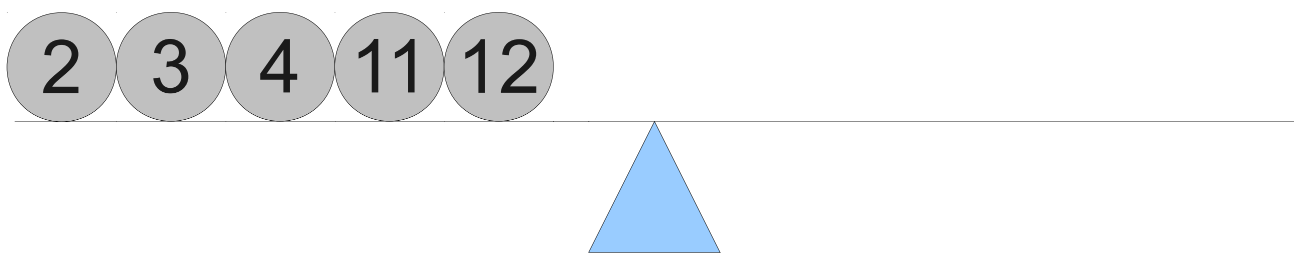
If the right side is heavier, record a -.

If the scale balances, record a 0.

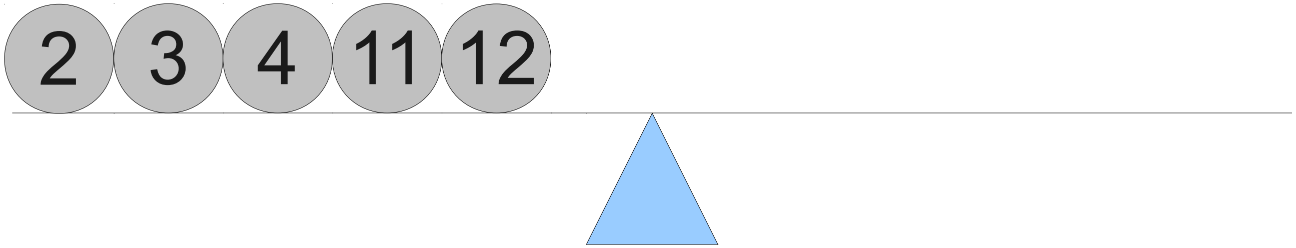
1	00+
2	0+−
3	0+0
4	0++
5	+−−
6	+−0
7	+−+
8	+0−
9	+00
10	+0+
11	++−
12	++0



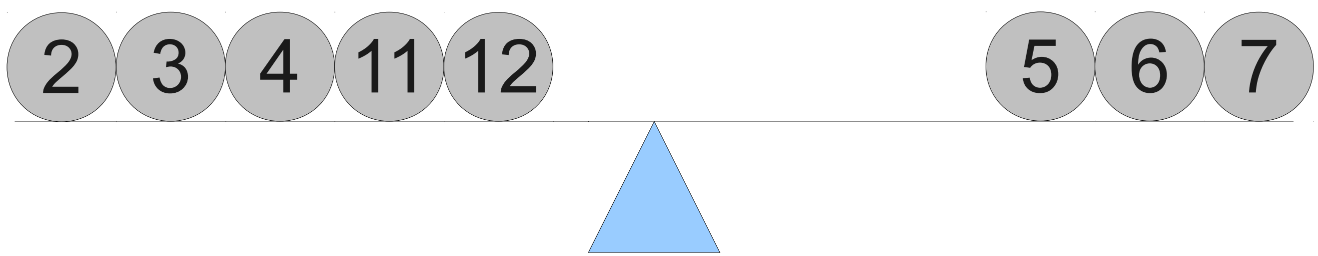
1	00+
2	0+−
3	0+0
4	0++
5	+−−
6	+−0
7	+−+
8	+0−
9	+00
10	+0+
11	++−
12	++0



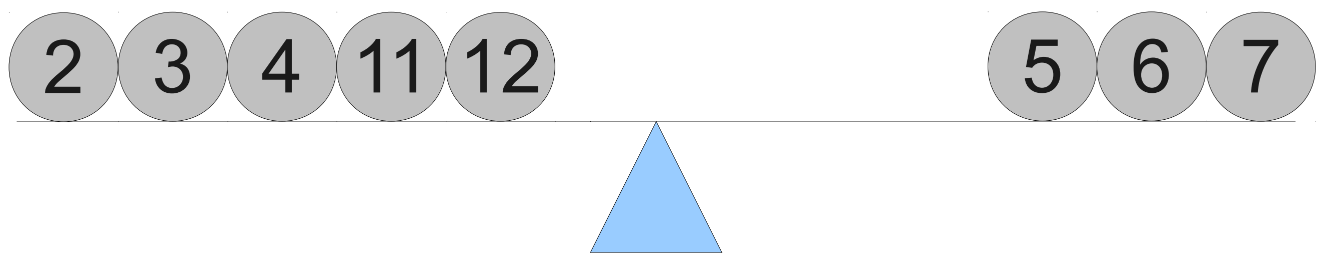
1	00+
2	0+–
3	0+0
4	0++
5	+––
6	+–0
7	+–+
8	+0–
9	+00
10	+0+
11	++–
12	++0



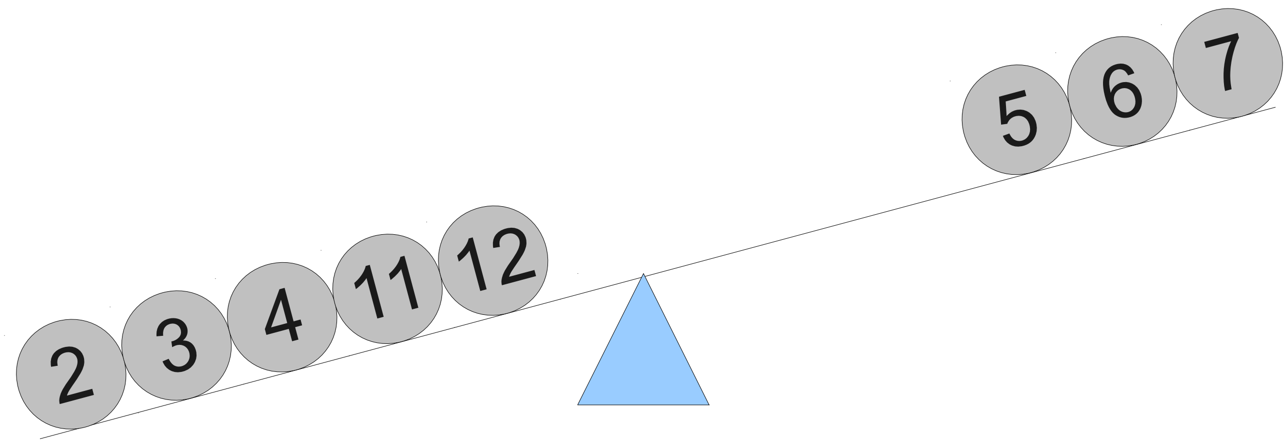
1	00+
2	0+−
3	0+0
4	0++
5	+−−
6	+−0
7	+−+
8	+0−
9	+00
10	+0+
11	++−
12	++0



1	00+
2	0+−
3	0+0
4	0++
5	+−−
6	+−0
7	+−+
8	+0−
9	+00
10	+0+
11	++−
12	++0



1	00+
2	0+–
3	0+0
4	0++
5	+––
6	+–0
7	+–+
8	+0–
9	+00
10	+0+
11	++–
12	++0



Our Encoding Scheme

Lighter		Heavier	
−1	00−	1	00+
−2	0−+	2	0+−
−3	0−0	3	0+0
−4	0−−	4	0++
−5	−++	5	+−−
−6	−+0	6	+−0
−7	−+−	7	+−+
−8	−0+	8	+0−
−9	−00	9	+00
−10	−0−	10	+0+
−11	−−+	11	++−
−12	−−0	12	++0

Our Encoding Scheme

Lighter		Heavier	
−1	00−	1	00+
−2	0−+	2	0+−
−3	0−0	3	0+0
−4	0−−	4	0++
−5	−++	5	+−−
−6	−+0	6	+−0
−7	−+−	7	+−+
−8	−0+	8	+0−
−9	−00	9	+00
−10	−0−	10	+0+
−11	−−+	11	++−
−12	**−−0**	**12**	**++0**

Our Encoding Scheme

Lighter		Heavier	
−1	00−	1	00+
−2	0−+	2	0+−
−3	0−0	3	0+0
−4	0−−	4	0++
−5	−++	5	+−−
−6	−+0	6	+−0
−7	−+−	7	+−+
−8	−0+	8	+0−
−9	−00	9	+00
−10	−0−	10	+0+
−11	−−+	11	++−
12	**++0**	**−12**	**−−0**

Our Encoding Scheme

Lighter		Heavier	
−1	00−	1	00+
−2	0−+	2	0+−
−3	0−0	3	0+0
−4	0−−	4	0++
−5	−++	5	+−−
−6	−+0	6	+−0
−7	−+−	7	+−+
−8	−0+	8	+0−
−9	−00	9	+00
−10	−0−	10	+0+
−11	−−+	11	++−
12	++0	−12	−−0

Our Encoding Scheme

Lighter		Heavier	
−1	00−	1	00+
−2	0−+	2	0+−
−3	0−0	3	0+0
−4	0−−	4	0++
−5	−++	5	+−−
−6	−+0	6	+−0
−7	−+−	7	+−+
−8	**−0+**	**8**	**+0−**
−9	**−00**	**9**	**+00**
−10	**−0−**	**10**	**+0+**
−11	−−+	11	++−
12	++0	−12	−−0

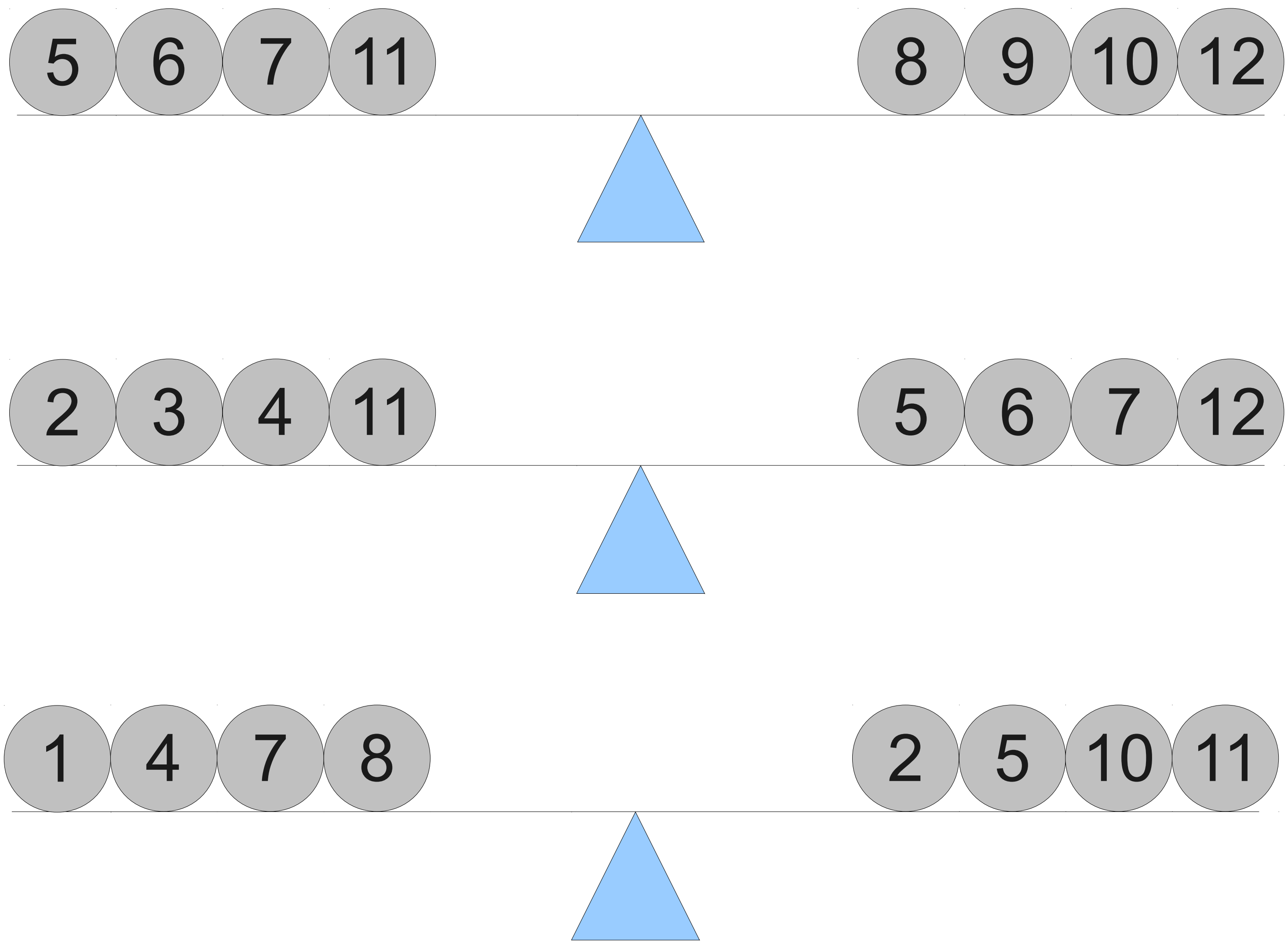
Our Encoding Scheme

Lighter		Heavier	
−1	00−	1	00+
−2	0−+	2	0+−
−3	0−0	3	0+0
−4	0−−	4	0++
−5	−++	5	+−−
−6	−+0	6	+−0
−7	−+−	7	+−+
8	**+0−**	**−8**	**−0+**
9	**+00**	**−9**	**−00**
10	**+0+**	**−10**	**−0−**
−11	−−+	11	++−
12	++0	−12	−−0

Our Encoding Scheme

Lighter		Heavier	
−1	00−	1	00+
−2	0−+	2	0+−
−3	0−0	3	0+0
−4	0−−	4	0++
−5	−++	5	+−−
−6	−+0	6	+−0
−7	−+−	7	+−+
8	+0−	−8	−0+
9	+00	−9	−00
10	+0+	−10	−0−
−11	−−+	11	++−
12	++0	−12	−−0

The Solution



1	00+
2	0+–
3	0+0
4	0++
5	+––
6	+–0
7	+–+
–8	–0+
–9	–00
–10	–0–
11	++–
–12	––0

Generalizing the Result

- Why twelve balls?
  - With three trits, 27 possible combinations. 13 are positive, nine start with +.
  - Must discard one starting with + to ensure number of + and - in each column is the same, leaving 12 positive numbers.
- More generally:
  - With n trits, 3^n possible combinations. $(3^n – 1) / 2$ are positive.
  - 3^{n-1} numbers start with +. To balance + and -, we need to drop one starting with +, leaving **$(3^n – 3) / 2$** positive numbers.
- Can do 3, 12, 39, 120, 363, 1092, ...

Example: Two Weighings

Lighter		Heavier	
-1	0-	1	0+
-2	-+	2	+-
-3	-0	3	+0
-4	--	4	++

Example: Two Weighings

Lighter		Heavier	
−1	0−	1	0+
−2	−+	2	+−
−3	−0	3	+0
−4	**−−**	**4**	**++**

Example: Two Weighings

Lighter		Heavier	
−1	0−	1	0+
−2	−+	2	+−
−3	−0	3	+0

Example: Two Weighings

Lighter		Heavier	
−1	0−	1	0+
−2	−+	2	+−
−3	**−0**	**3**	**+0**

Example: Two Weighings

Lighter		Heavier	
−1	0−	1	0+
−2	−+	2	+−
3	**+0**	**−3**	**−0**

Example: Two Weighings

Lighter		Heavier	
−1	0−	1	0+
−2	−+	2	+−
3	+0	−3	−0

Example: Three Weighings

Lighter		Heavier	
−1	00−	1	00+
−2	0−+	2	0+−
−3	0−0	3	0+0
−4	0−−	4	0++
−5	−++	5	+−−
−6	−+0	6	+−0
−7	−+−	7	+−+
−8	−0+	8	+0−
−9	−00	9	+00
−10	−0−	10	+0+
−11	−−+	11	++−
−12	−−0	12	++0
−13	−−−	13	+++

Example: Three Weighings

Lighter		Heavier	
−1	00−	1	00+
−2	0−+	2	0+−
−3	0−0	3	0+0
−4	0−−	4	0++
−5	−++	5	+−−
−6	−+0	6	+−0
−7	−+−	7	+−+
−8	−0+	8	+0−
−9	−00	9	+00
−10	−0−	10	+0+
−11	−−+	11	++−
−12	−−0	12	++0
−13	**−−−**	**13**	**+++**

Example: Three Weighings

Lighter		Heavier	
−1	00−	1	00+
−2	0−+	2	0+−
−3	0−0	3	0+0
−4	0−−	4	0++
−5	−++	5	+−−
−6	−+0	6	+−0
−7	−+−	7	+−+
−8	−0+	8	+0−
−9	−00	9	+00
−10	−0−	10	+0+
−11	−−+	11	++−
−12	−−0	12	++0

Example: Three Weighings

Lighter		Heavier	
−1	00−	1	00+
−2	0−+	2	0+−
−3	0−0	3	0+0
−4	0−−	4	0++
−5	−++	5	+−−
−6	−+0	6	+−0
−7	−+−	7	+−+
−8	**−0+**	**8**	**+0−**
−9	**−00**	**9**	**+00**
−10	**−0−**	**10**	**+0+**
−11	−−+	11	++−
−12	**−−0**	**12**	**++0**

Example: Three Weighings

Lighter		Heavier	
−1	00−	1	00+
−2	0−+	2	0+−
−3	0−0	3	0+0
−4	0−−	4	0++
−5	−++	5	+−−
−6	−+0	6	+−0
−7	−+−	7	+−+
8	**+0−**	**−8**	**−0+**
9	**+00**	**−9**	**−00**
10	**+0+**	**−10**	**−0−**
−11	−−+	11	++−
12	**++0**	**−12**	**−−0**

Example: Three Weighings

Lighter		Heavier	
−1	00−	1	00+
−2	0−+	2	0+−
−3	0−0	3	0+0
−4	0−−	4	0++
−5	−++	5	+−−
−6	−+0	6	+−0
−7	−+−	7	+−+
8	+0−	−8	−0+
9	+00	−9	−00
10	+0+	−10	−0−
−11	−−+	11	++−
12	++0	−12	−−0

Example: Four Weighings

1	000+	14	+---	27	+000
2	00+-	15	+--0	28	+00+
3	00+0	16	+--+	29	+0+-
4	00++	17	+-0-	30	+0+0
5	0+--	18	+-00	31	+0++
6	0+-0	19	+-0+	32	++--
7	0+-+	20	+-+-	33	++-0
8	0+0-	21	+-+0	34	++-+
9	0+00	22	+-++	35	++0-
10	0+0+	23	+0--	36	++00
11	0++-	24	+0-0	37	++0+
12	0++0	25	+0-+	38	+++-
13	0+++	26	+00-	39	+++0
				40	++++

Example: Four Weighings

1	000+	14	+---	27	+000
2	00+-	15	+--0	28	+00+
3	00+0	16	+--+	29	+0+-
4	00++	17	+-0-	30	+0+0
5	0+--	18	+-00	31	+0++
6	0+-0	19	+-0+	32	++--
7	0+-+	20	+-+-	33	++-0
8	0+0-	21	+-+0	34	++-+
9	0+00	22	+-++	35	++0-
10	0+0+	23	+0--	36	++00
11	0++-	24	+0-0	37	++0+
12	0++0	25	+0-+	38	+++-
13	0+++	26	+00-	39	+++0
				40	**++++**

Example: Four Weighings

1	000+
2	00+-
3	00+0
4	00++
5	0+--
6	0+-0
7	0+-+
8	0+0-
9	0+00
10	0+0+
11	0++-
12	0++0
13	0+++
14	+---
15	+--0
16	+--+
17	+-0-
18	+-00
19	+-0+
20	+-+-
21	+-+0
22	+-++
23	+0--
24	+0-0
25	+0-+
26	+00-
27	+000
28	+00+
29	+0+-
30	+0+0
31	+0++
32	++--
33	++-0
34	++-+
35	++0-
36	++00
37	++0+
38	+++-
39	+++0

Example: Four Weighings

1	000+	14	+---	**27**	**+000**
2	00+-	15	+--0	**28**	**+00+**
3	00+0	16	+--+	**29**	**+0+-**
4	00++	17	+-0-	**30**	**+0+0**
5	0+--	18	+-00	**31**	**+0++**
6	0+-0	19	+-0+	32	++--
7	0+-+	20	+-+-	33	++-0
8	0+0-	21	+-+0	34	++-+
9	0+00	22	+-++	**35**	**++0-**
10	0+0+	**23**	**+0--**	**36**	**++00**
11	0++-	**24**	**+0-0**	**37**	**++0+**
12	0++0	**25**	**+0-+**	38	+++-
13	0+++	**26**	**+00-**	**39**	**+++0**

Example: Four Weighings

1	000+	14	+---	**-27**	**-000**
2	00+-	15	+--0	**-28**	**-00-**
3	00+0	16	+--+	**-29**	**-0-+**
4	00++	17	+-0-	**-30**	**-0-0**
5	0+--	18	+-00	**-31**	**-0--**
6	0+-0	19	+-0+	32	++--
7	0+-+	20	+-+-	33	++-0
8	0+0-	21	+-+0	34	++-+
9	0+00	22	+-++	**-35**	**--0+**
10	0+0+	**-23**	**-0++**	**-36**	**--00**
11	0++-	**-24**	**-0+0**	**-37**	**--0-**
12	0++0	**-25**	**-0+-**	38	+++-
13	0+++	**-26**	**-00+**	**-39**	**---0**

Example: Four Weighings

1	000+	14	+---	-27	-000
2	00+-	15	+--0	-28	-00-
3	00+0	16	+--+	-29	-0-+
4	00++	17	+-0-	-30	-0-0
5	0+--	18	+-00	-31	-0--
6	0+-0	19	+-0+	32	++--
7	0+-+	20	+-+-	33	++-0
8	0+0-	21	+-+0	34	++-+
9	0+00	22	+-++	-35	--0+
10	0+0+	-23	-0++	-36	--00
11	0++-	-24	-0+0	-37	--0-
12	0++0	-25	-0+-	38	+++-
13	0+++	-26	-00+	-39	---0

Some Insights

- What number did we drop?
  - With 2 trits, dropped ++.
  - With 3 trits, dropped +++.
  - With 4 trits, dropped ++++.
  - **Always drop ++…++**
- What numbers did we invert?
  - With two trits: +0
  - With three trits: ++0, +0-, +00, +0-
  - With four trits: +++0, ++0-, ++00, ++0+, +0--, +0-0, +0-+, +00-, +000, +00+, +0+-, +0+0, +0++
  - **Always invert numbers starting with ++…++0.**
- **This always works!**

How Many Extra +s Per Column?

- Answer: The 3^j column has **(3^j – 1) / 2** extra +'s.

0+

+–

+0

How Many Extra +s Per Column?

- Answer: The 3^j column has **$(3^j – 1) / 2$** extra +'s.

0 0 +

0 + –

0 + 0

How Many Extra +s Per Column?

- Answer: The 3^j column has $\mathbf{(3^j - 1) / 2}$ extra +'s.

00+
0+–
0+0
0++
+––
+–0
+–+
+0–
+00
+0+
++–
++0

How Many Extra +s Per Column?

- Answer: The 3^j column has **(3^j – 1) / 2** extra +'s.

0	0+
0	+–
0	+0
0	++
+	––
+	–0
+	–+
+	0–
+	00
+	0+
+	+–
+	+0

How Many +'s Get Flipped?

- Answer: The 3^j column has **$(3^j-1) / 2$** flipped +'s.

How Many +'s Get Flipped?

- Answer: The 3^j column has $\mathbf{(3^j\text{-}1) / 2}$ flipped +'s.

0+

+–

+0

How Many +'s Get Flipped?

- Answer: The 3^j column has **(3^j-1) / 2** flipped +'s.

0	+
+	–
+	0

How Many +'s Get Flipped?

- Answer: The 3^j column has **$(3^j-1) / 2$** flipped +'s.

00+
0+–
0+0
0++
+––
+–0
+–+
+0–
+00
+0+
++–
++0

How Many +'s Get Flipped?

- Answer: The 3^j column has **(3^j-1) / 2** flipped +'s.

00+

0+–

0+0

0++

+––

+–0

+–+

+0–

+00

+0+

++–

++0

How Many +'s Get Flipped?

- Answer: The 3^j column has **(3^j-1) / 2** flipped +'s.

00+
0+–
0+0
0++
+––
+–0
+–+
+0–
+00
+0+
++–
++0

$$\sum_{i=0}^{j-1} 3^i = \frac{3^j - 1}{2}$$

Summary

- A three-way scale lends itself naturally to a **balanced ternary encoding** for each of the balls.
- Given an encoding with the same number of +'s and -'s in each column, we can use the scale to read off one trit of the answer at a time.
- Flipping numbers starting with +0, ++0, etc. guarantees an encoding with this property.

Generating Permutations

“You are given a sorted string S of unique characters. Write a Java-style iterator that traverses all the permutations of S in lexicographical order.”

Example

Example

`abc`

Example

```
abc
acb
bac
bca
cab
cba
```

Example

0	abc
1	acb
2	bac
3	bca
4	cab
5	cba

Lehmer Codes

B	A	E	D	C

Lehmer Codes

B	A	E	D	C

Lehmer Codes

B	A	E	D	C

Lehmer Codes

B	A	E	D	C
1				

Lehmer Codes

B	A	E	D	C
1				

Lehmer Codes

B	A	E	D	C
1				

Lehmer Codes

B	A	E	D	C
1	0			

Lehmer Codes

B	A	E	D	C
1	0			

Lehmer Codes

B	A	E	D	C
1	0			

Lehmer Codes

B	A	E	D	C
1	0			

Lehmer Codes

B	A	E	D	C
1	0	2		

Lehmer Codes

B	A	E	D	C
1	0	2		

Lehmer Codes

B	A	E	D	C
1	0	2		

Lehmer Codes

Lehmer Codes

B	A	E	D	C
1	0	2	1	

Lehmer Codes

B	A	E	D	C
1	0	2	1	

Lehmer Codes

B	A	E	D	C
1	0	2	1	

Lehmer Codes

B	A	E	D	C
1	0	2	1	0

Lehmer Codes

B	A	E	D	C
1	0	2	1	0

Lehmer Codes

B	A	E	D	C
1	0	2	1	0

C	D	A	B	E

Lehmer Codes

B	A	E	D	C
1	0	2	1	0

C	D	A	B	E
2				

Lehmer Codes

B	A	E	D	C
1	0	2	1	0

C	D	A	B	E
2	2			

Lehmer Codes

B	A	E	D	C
1	0	2	1	0

C	D	A	B	E
2	2	0		

Lehmer Codes

B	A	E	D	C
1	0	2	1	0

C	D	A	B	E
2	2	0	0	

Lehmer Codes

B	A	E	D	C
1	0	2	1	0

C	D	A	B	E
2	2	0	0	0

Lehmer Codes

A	B	C	D	E

4 1 0 1 0

Lehmer Codes

A	B	C	D	E

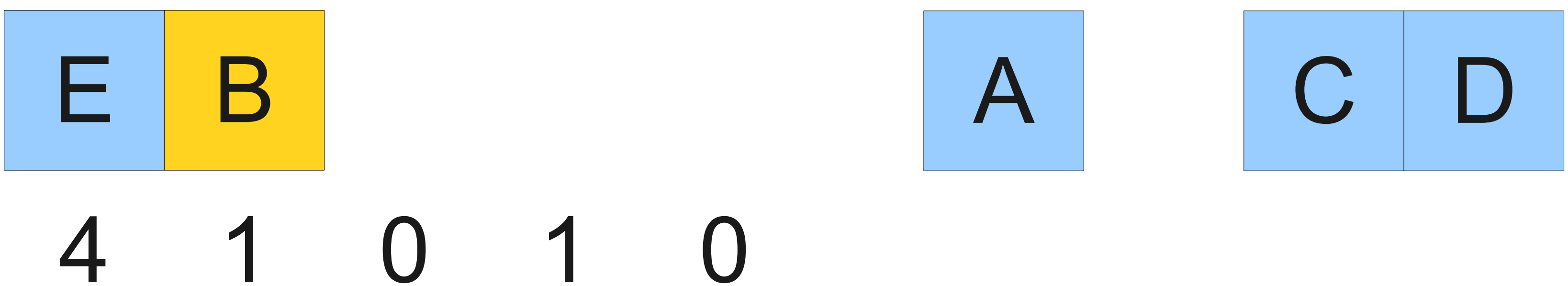
4 1 0 1 0

Lehmer Codes

Lehmer Codes

Lehmer Codes

Lehmer Codes



Lehmer Codes

E B

A C D

4 1 0 1 0

Lehmer Codes

4 1 0 1 0

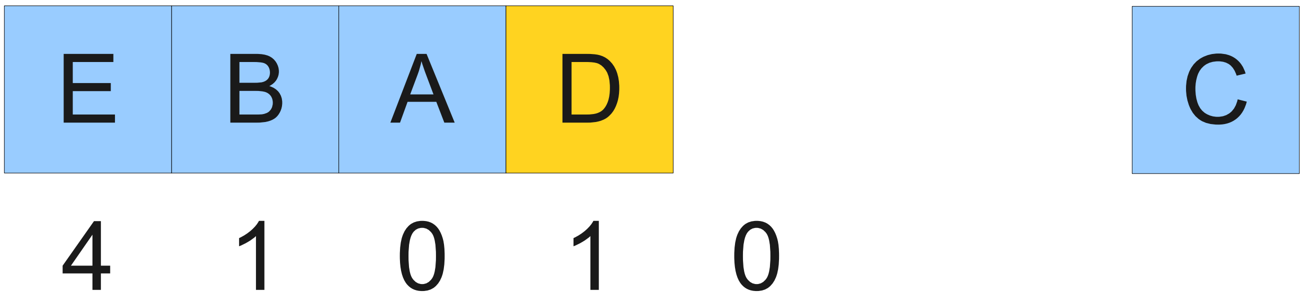
Lehmer Codes

Lehmer Codes

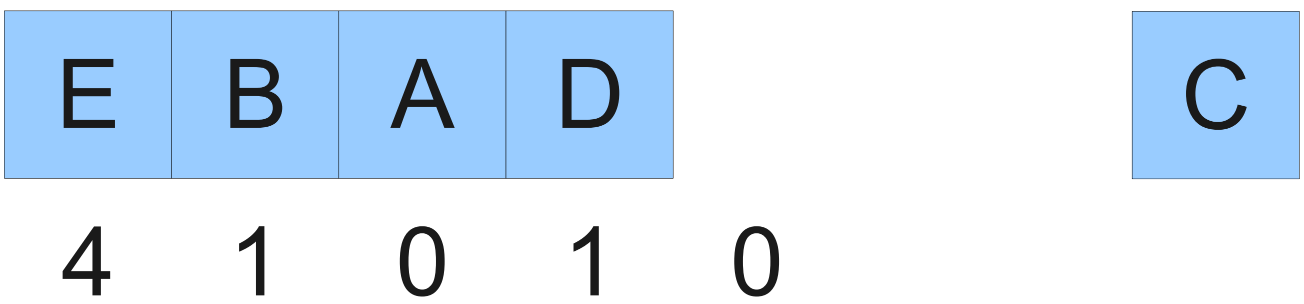
Lehmer Codes

Lehmer Codes

Lehmer Codes



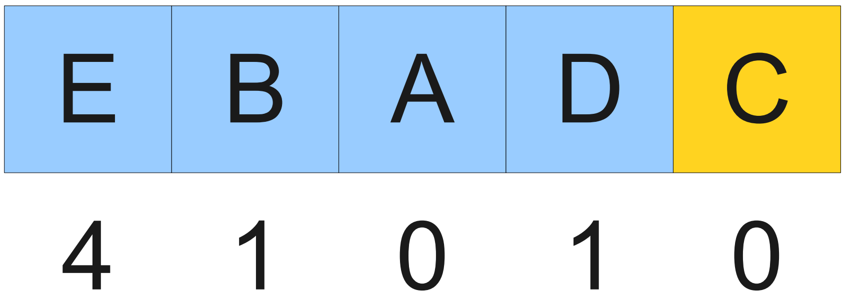
Lehmer Codes



Lehmer Codes

E	B	A	D		C
4	1	0	1	0	

Lehmer Codes



Lehmer Codes

E	B	A	D	C
4	1	0	1	0

Listing Lehmer Codes

0	abcd	0000
1	abdc	0010
2	acbd	0100
3	acdb	0110
4	adbc	0200
5	adcb	0210
6	bacd	1000
7	badc	1010
8	bcad	1100
9	bcda	1110
10	bdac	1200
11	bdac	1210
12	cabd	2000
13	cadb	2010
14	cbad	2100
15	cbda	2110
16	cdab	2200
17	cdba	2210
18	dabc	3000
19	dacb	3010
20	dbac	3100
21	dbca	3110
22	dcab	3200
23	dcba	3210

Factoradic Numbers

- Mixed-radix number system.
- Nth digit in base n!.
- Nth digit can be 0, 1, 2, …, n
- Example: $3110_!$
  - 3 x 3! + 1 x 2! + 1 x 1! + 0 x 0! = 21
- Example: $1210_!$
  - 1 x 3! + 2 x 2! + 1 x 1! + 0 x 0! = 11

Listing Lehmer Codes

0	abcd	0000
1	abdc	0010
2	acbd	0100
3	acdb	0110
4	adbc	0200
5	adcb	0210
6	bacd	1000
7	badc	1010
8	bcad	1100
9	bcda	1110
10	bdac	1200
11	bdac	1210
12	cabd	2000
13	cadb	2010
14	cbad	2100
15	cbda	2110
16	cdab	2200
17	cdba	2210
18	dabc	3000
19	dacb	3010
20	dbac	3100
21	dbca	3110
22	dcab	3200
23	dcba	3210

Listing Lehmer Codes

0	abcd	0000
1	abdc	0010
2	acbd	0100
3	acdb	0110
4	adbc	0200
5	adcb	0210
6	bacd	1000
7	badc	1010
8	bcad	1100
9	bcda	1110
10	bdac	1200
11	**bdac**	**1210**
12	cabd	2000
13	cadb	2010
14	cbad	2100
15	cbda	2110
16	cdab	2200
17	cdba	2210
18	dabc	3000
19	dacb	3010
20	dbac	3100
21	**dbca**	**3110**
22	dcab	3200
23	dcba	3210

Writing n in factoradic gives the nth Lehmer code.

Converting to Factoradic

- Goal: convert k to factoradic.
  - Assume we know n, the number of elements to permute.
- To get the $(n – 1)!$ place, divide k by $(n – 1)!$
  - Quotient is the $(n – 1)!$ place.
- Repeat for the remaining digits using the remainder.
- Identical to converting to any other base, just using factorials instead of powers.

Example: Convert 13 to Factoradic

Example: Convert 13 to Factoradic

`13 = 2 x 3! + 1`

Example: Convert 13 to Factoradic

```
13 = 2 x 3! + 1
 1 = 0 x 2! + 1
```

Example: Convert 13 to Factoradic

```
13 = 2 x 3! + 1

 1 = 0 x 2! + 1

 1 = 1 x 1! + 0
```

Example: Convert 13 to Factoradic

```
13 = 2 x 3! + 1
 1 = 0 x 2! + 1
 1 = 1 x 1! + 0
 0 = 0 x 0! + 0
```

Example: Convert 13 to Factoradic

```
13 = 2 x 3! + 1
 1 = 0 x 2! + 1
 1 = 1 x 1! + 0
 0 = 0 x 0! + 0
```

Example: Convert 13 to Factoradic

```
13 = 2 x 3! + 1
 1 = 0 x 2! + 1
 1 = 1 x 1! + 0
 0 = 0 x 0! + 0
```

Answer: $2010_!$

Generating Permutations

```
public static String kthPermutation(String chars, int k)
{
    String result = "";

    for (int n = chars.length() - 1; n >= 0; --n) {
        int quotient  = k / factorial(n);
        int remainder = k % factorial(n);

        result += chars.charAt(quotient);

        chars = chars.substring(0, quotient) +
                chars.substring(quotient + 1);

        k = remainder;
    }

    return result;
}
```

Analysis of Our Algorithm

- To get the kth permutation of n elements:
  - Converting k to factoradic takes O(n)
  - Building up the permutations takes $O(n^2)$
    - Elements stored in a list; O(n) to remove each.
  - Can reduce to O(n lg n) using **order statistic tree**.
- Easy to build an iterator from this.

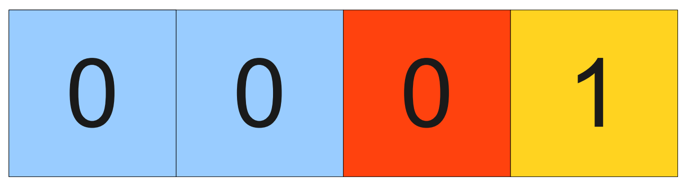
Incrementing Binary Numbers

0	0	0	0

Incrementing Binary Numbers

0	0	0	1

Incrementing Binary Numbers



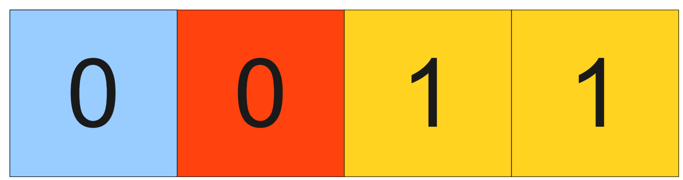
Incrementing Binary Numbers

0	0	1	0

Incrementing Binary Numbers

0	0	1	1

Incrementing Binary Numbers



Incrementing Binary Numbers

0	1	0	0

Incrementing Factoradic Numbers

Incrementing Factoradic Numbers

0	0	0	0

Incrementing Factoradic Numbers

0	0	0	0

Incrementing Factoradic Numbers

0	0	1	0

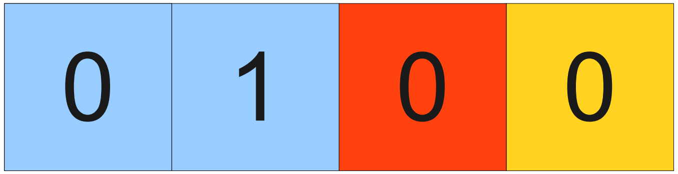
Incrementing Factoradic Numbers

0	0	1	0

Incrementing Factoradic Numbers

0	1	0	0

Incrementing Factoradic Numbers



Incrementing Factoradic Numbers

0	1	1	0

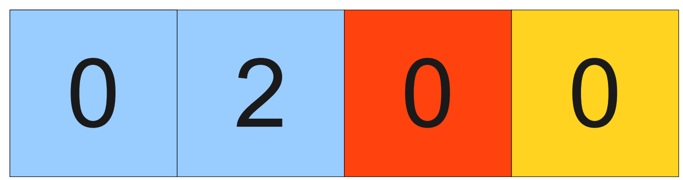
Incrementing Factoradic Numbers

0	1	1	0

Incrementing Factoradic Numbers

0	2	0	0

Incrementing Factoradic Numbers



Incrementing Factoradic Numbers

0	2	1	0

Incrementing Factoradic Numbers

0	2	1	0

Incrementing Factoradic Numbers

1	0	0	0

Incrementing Factoradic Numbers

- Find the digit to increment.
  - Scan backwards from the end to find the first number not at its maximum.
- Increment that digit.
- Set the digits after that to zero.

Incrementing Permutations

Incrementing Permutations

A	B	C	D
0	0	0	0

Incrementing Permutations

A	B	C	D
0	0	0	0

Incrementing Permutations

A	B	C	D
0	0	1	0

Incrementing Permutations

A	B	D	C
0	0	1	0

Incrementing Permutations

A	B	D	C
0	0	1	0

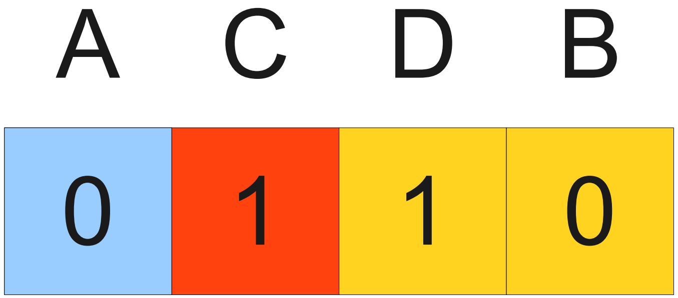
Incrementing Permutations

A	B	D	C
0	0	1	0

Incrementing Permutations

A	B	D	C
0	1	1	0

Incrementing Permutations



Incrementing Permutations

A	C	D	B
0	1	0	0

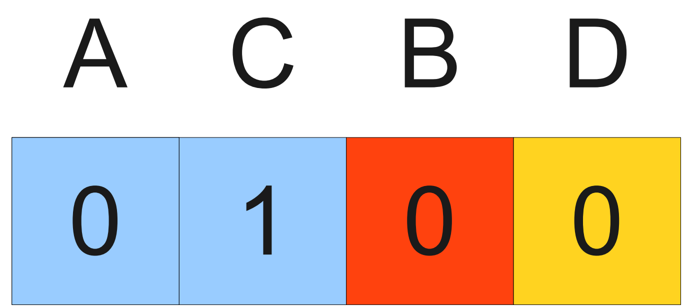
Incrementing Permutations

A	C	B	D
0	1	0	0

Incrementing Permutations

A	C	B	D
0	1	0	0

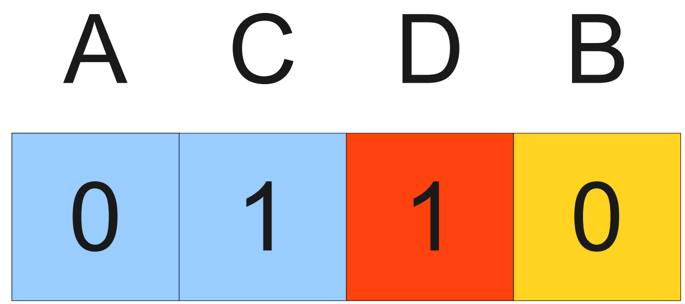
Incrementing Permutations



Incrementing Permutations

A	C	B	D
0	1	1	0

Incrementing Permutations



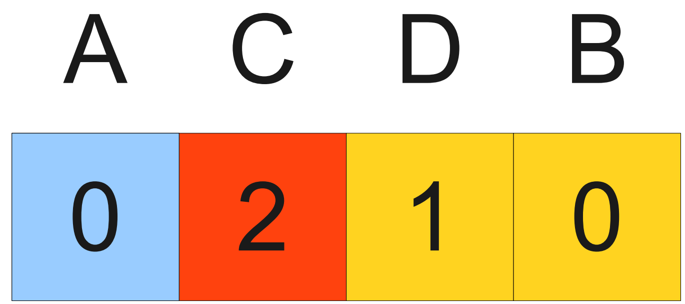
Incrementing Permutations

A	C	D	B
0	1	1	0

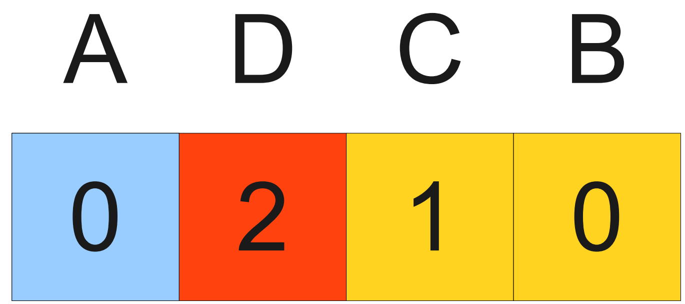
Incrementing Permutations

A	C	D	B
0	1	1	0

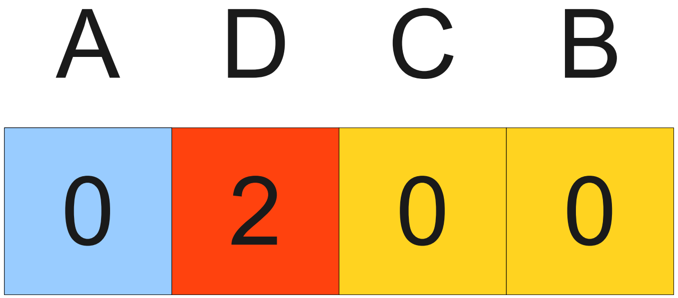
Incrementing Permutations



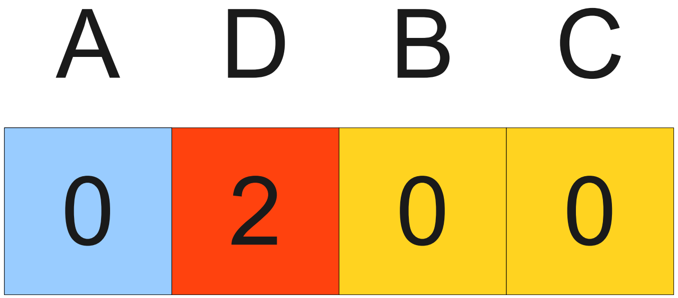
Incrementing Permutations



Incrementing Permutations



Incrementing Permutations



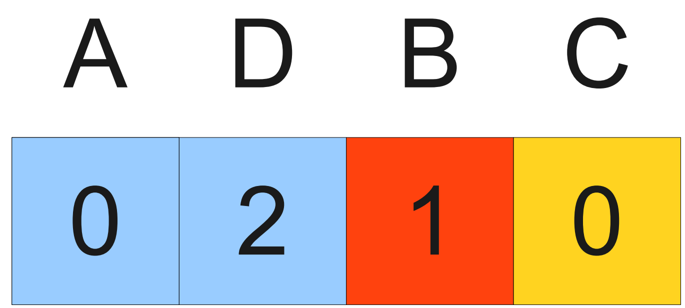
Incrementing Permutations

A	D	B	C
0	2	0	0

Incrementing Permutations

A	D	B	C
0	2	0	0

Incrementing Permutations



Incrementing Permutations

A	D	C	B
0	2	1	0

Incrementing Permutations

A	D	C	B
0	2	1	0

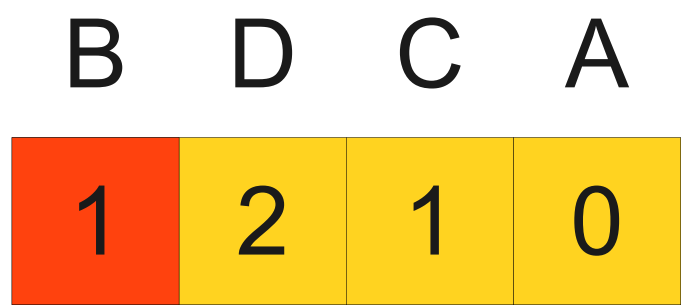
Incrementing Permutations

A	D	C	B
0	2	1	0

Incrementing Permutations

A	D	C	B
1	2	1	0

Incrementing Permutations



Incrementing Permutations

B	D	C	A
1	0	0	0

Incrementing Permutations

B	A	C	D
1	0	0	0

Incrementing Permutations

B	A	C	D
1	0	0	0

Incrementing Permutations

- Find the digit to be incremented.
  - Scan backwards from the end of the sequence to find the longest increasing sequence.
- Increment that digit.
  - Find the smallest element bigger than the element right before that sequence.
  - Swap those two elements.
- Set the digits after that to zero.
  - Reverse the increasing sequence
- Runtime: **O(n) per permutation**.

Summary

- Lehmer codes describe a permutation as a series of elements to choose in order.
- The factoradic number system maps directly onto Lehmer codes, and thus onto permutations.
- By simulating what **would** happen if we wrote out the Lehmer code, we can derive a fast algorithm for generating ordered permutations.

Concluding Thoughts

Number systems are useful in **number recovery** by discovering one component of the number at a time.

Number systems are useful in **enumeration** by revealing structure hidden in the indices.

Fun With Number Systems

My Email Address

htiek@cs.stanford.edu