**Goal:** Model a population of bunnies
(so we know how much food to buy)

**Rules:**

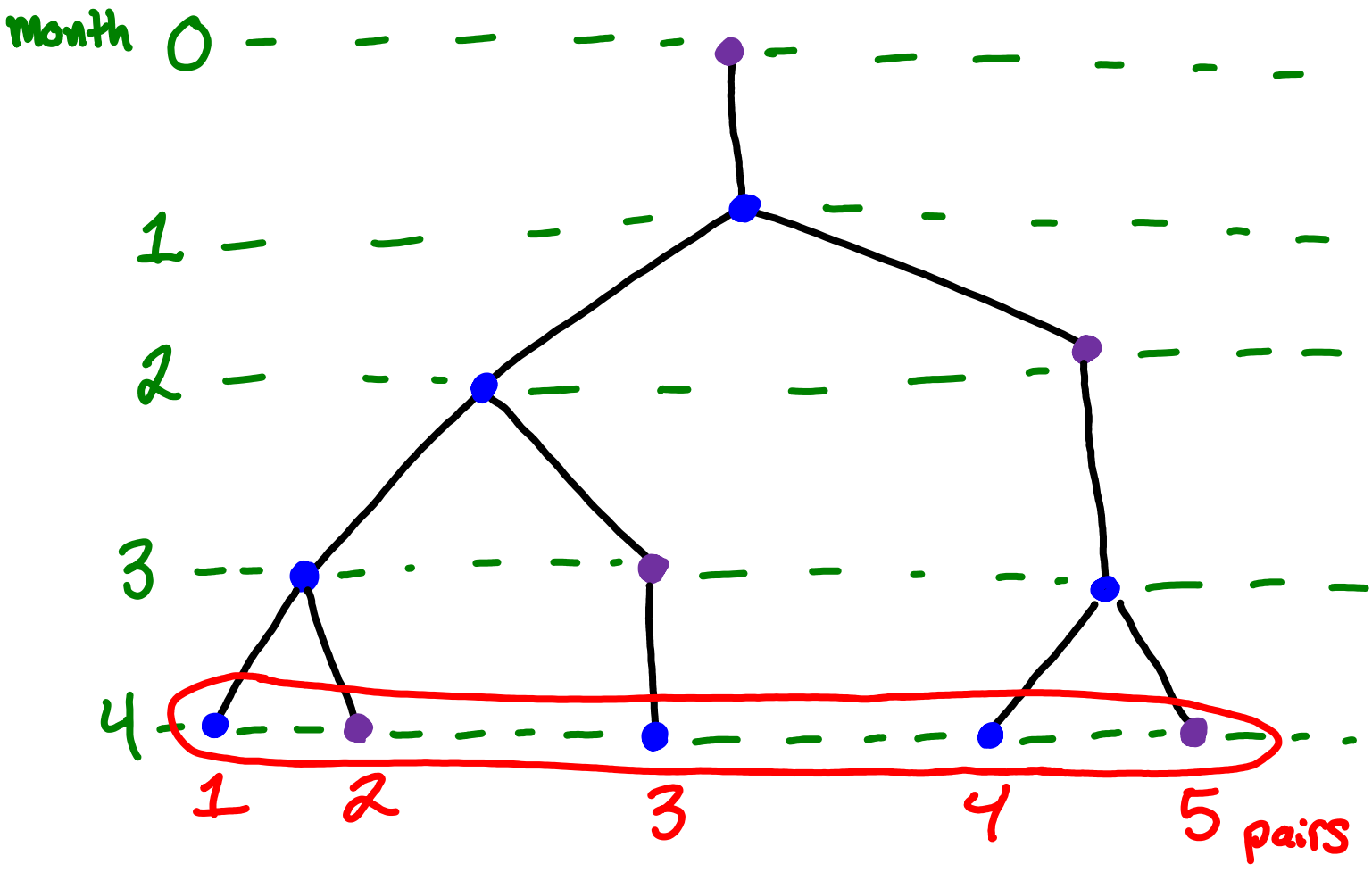
(i) begin w/ 1 pair of baby bunnies

(ii) baby bunnies mature in 1 month

(iii) mature bunnies reproduce once/month

**Q:** How many pairs of bunnies are there after $n$ months?

**e.g.** $n=4$



**Notation:** $F_n := \#$ dots in row $n$

$= \#$ bunnies in month $n$

$F_n := \#$ blue dots in row $n$

$= \#$ mature bunnies in month $n$

$F_n := \#$ purple dots in row $n$

$= \#$ baby bunnies in month $n$

**proposition:**

$$F_n = F_n + F_n$$

$$F_n = F_{n-1}$$

$$F_n = F_{n-1}$$

**proof:** look @ the picture above or try to explain how these formulas encode rules (ii) & (iii) □

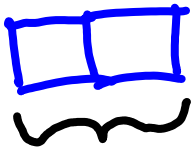
**Corollary:** $F_n = F_{n-1} + F_{n-2}$ □

we now have a recurrence relation for the # of bunnies we will need to feed in month $n$.

The information in rule (i) can be phrased as saying $F_0 = 1$. Solving recurrence relations is not always an easy task, but we will do so eventually. For now this answer suffices

**Goal:** Tile a hallway w/ 2 types of tiles

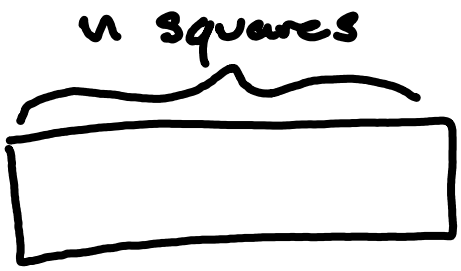
(I) Square 

(II) rectangle 

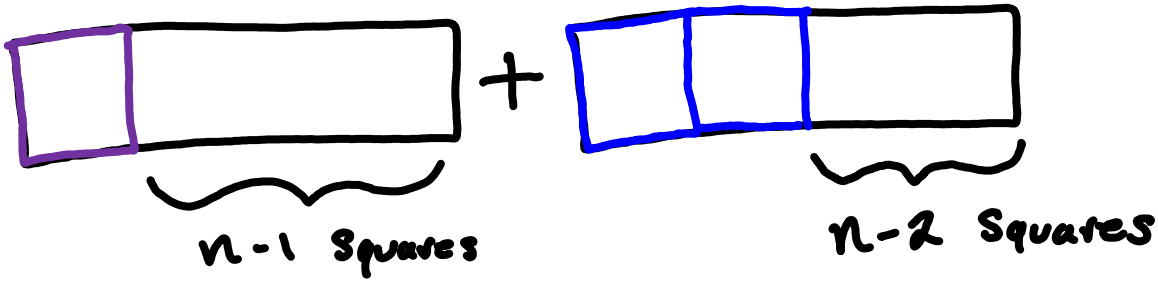
two squares stuck together

**Q:** How many different ways can we tile a hallway of length $n$?

**A:** Either begin w/  or 




=

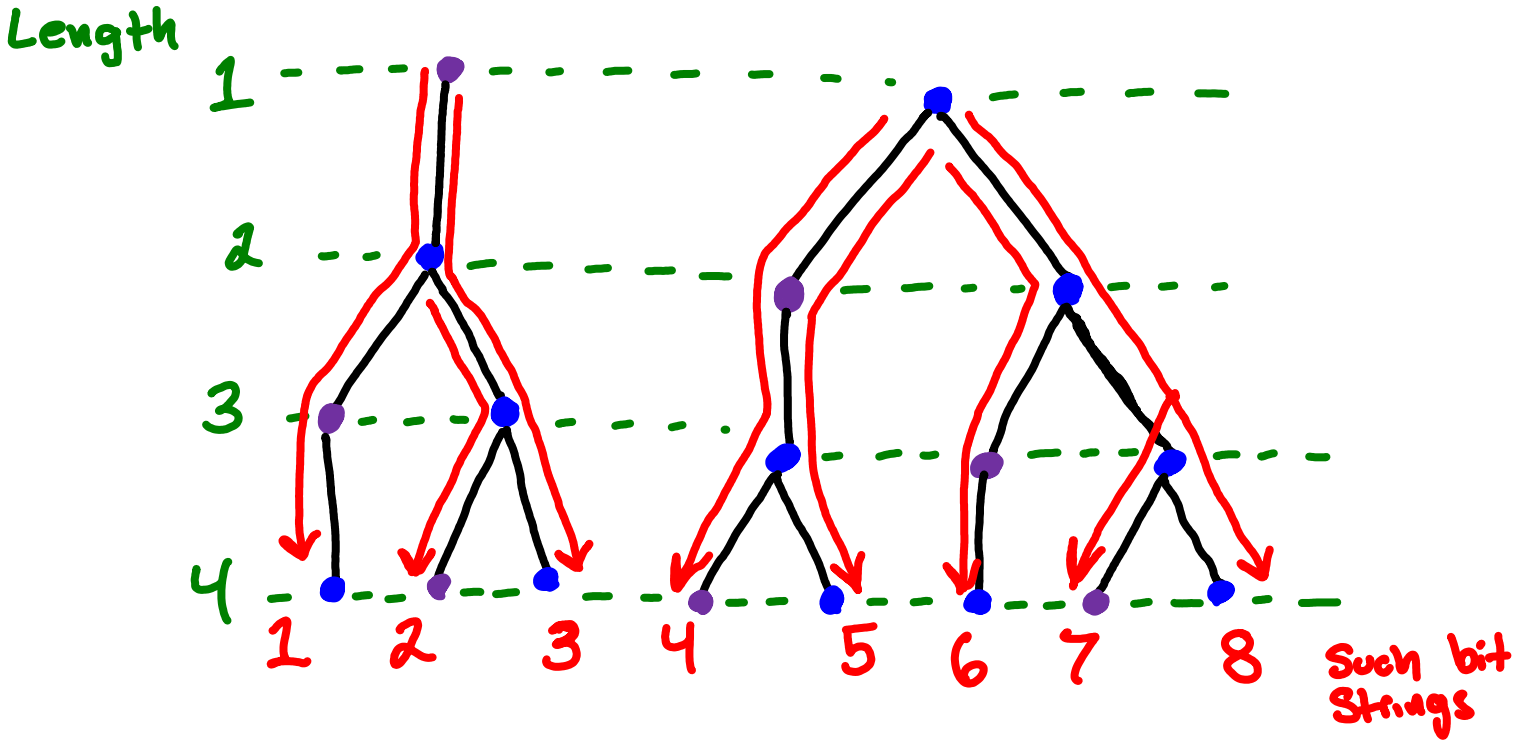



$F_n = F_{n-1} + F_{n-2}$

the same recurrence relation!

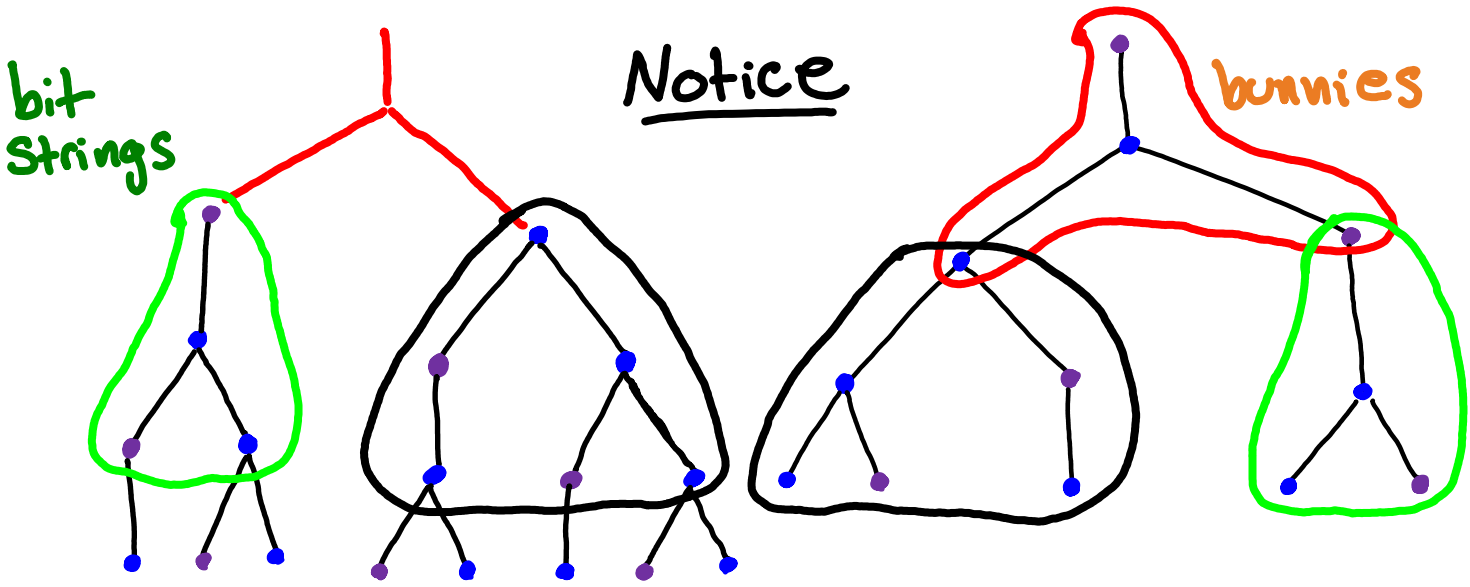
**Goal:** List all binary strings not containing an adjacent pair of 0s

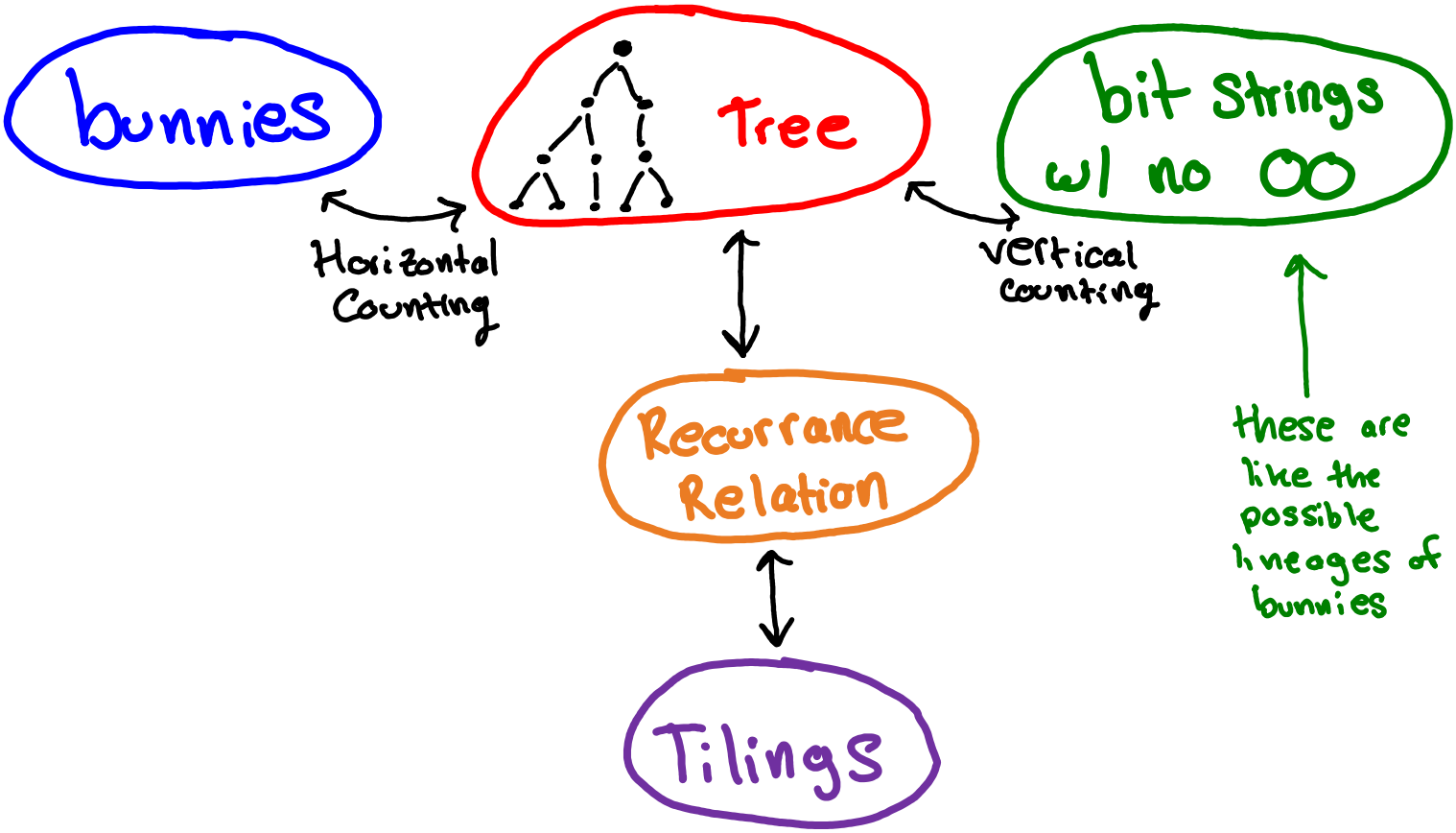
• = 0
• = 1




0101
0110
0111
1010

1011
1101
1110
1111



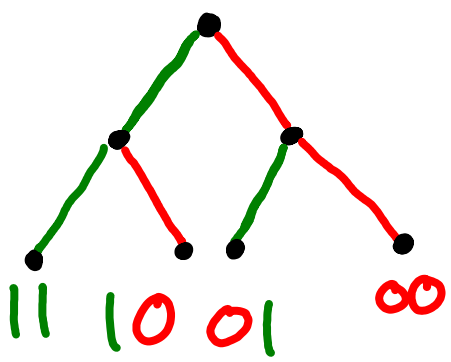



## m-ary rooted trees

**Def:** A rooted tree is (full) m-ary if every parent has exactly $m$ children

**Recall:** Counting in Binary

- to generate all binary numbers with fewer than $n+1$ digits we build a 2-ary tree with $n+1$ rows & label the leaves appropriately.



# Counting in base 3

0, 1, 2, 10, 11, 12, 20, 21, ...

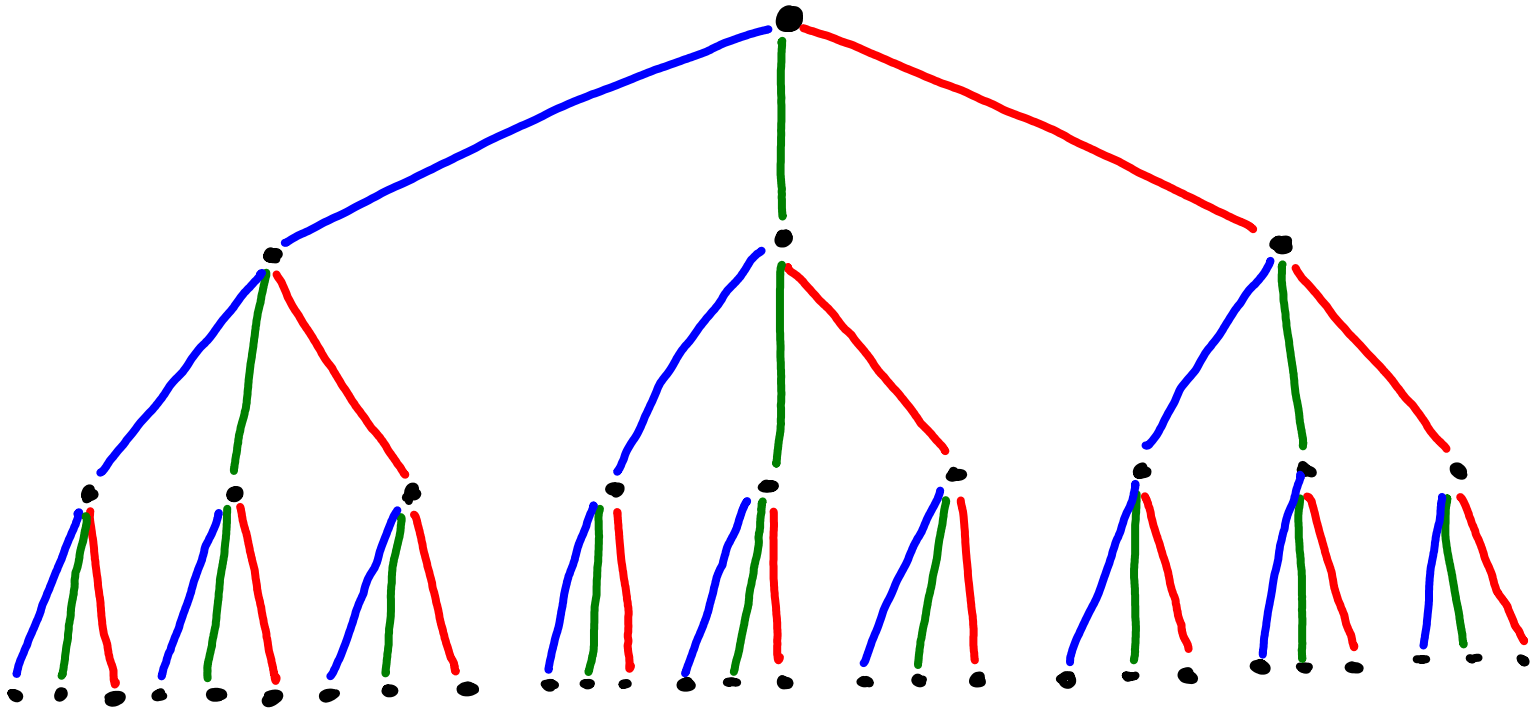
Convert to base 10

$0\cdot 3^0,\ 1\cdot 3^0,\ 2\cdot 3^0,\ 1\cdot 3^1,\ 1\cdot 3^1+1\cdot 3^0,\ 1\cdot 3^1+2\cdot 3^0,\ 2\cdot 3^1,\ 2\cdot 3^1+1\cdot 3^0,\ \ldots$

$=\ 0,\ 1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 7,\ \ldots$

OR

Count leaves in 3-ary tree



Exercise: What is the largest number represented in the above picture?

# Counting in base $\kappa$

totally ordered a set of size $\kappa$

$(X, \leq)$, $|X| = \kappa$

Strings over $X$ are totally ordered

$(X^*$, Dictionary order$)$

can view these as

Numbers in base $\kappa$

$0 < 1 < 2 < 3$

$a < b < c < d$

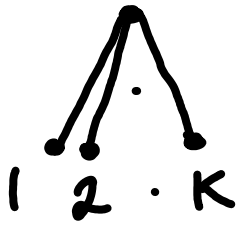
caadb

$2 \cdot \kappa^4 + 0 \cdot \kappa^3 + 0 \cdot \kappa^2 + 3 \cdot \kappa^1 + 1 \cdot \kappa^0$

The leaves of a $\kappa$-ary tree w/ $n+1$ rows encode $\aleph_s$ in base $\kappa$ w/ $\leq n$ digits

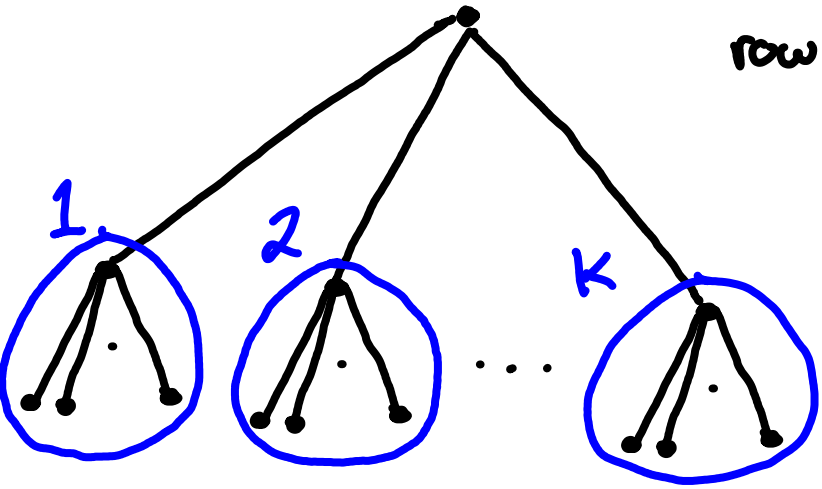
We also see (by example) how the leaves encode these numbers in order (dictionary order)

- row 0 corresponds to $K^0 = 1$




row 1 corresponds to 1 group of K

$K = 1 \cdot K = K^1$




row 2 corresponds to

K groups of K

$K \cdot K = K^2$

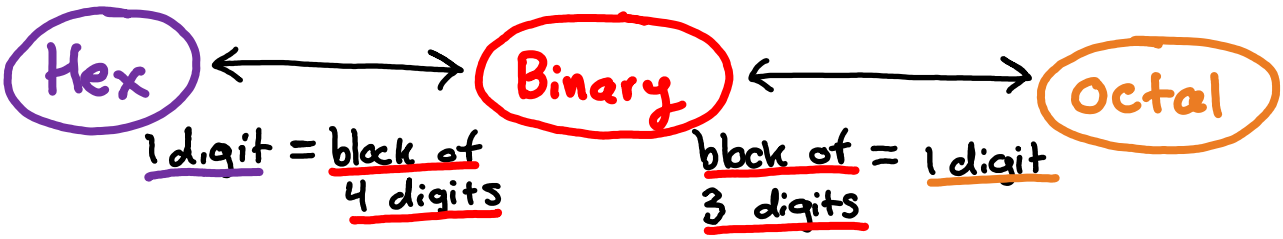
(this is how many different #s are represented but values range from 0 to $K^n - 1$)

particularly useful in CS are

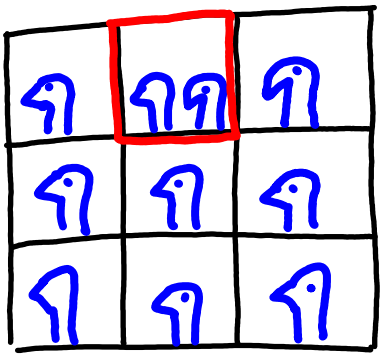
octal & Hexadecimal 8, 16

{0, 1, 2, 3, 4, 5, 6, 7}

{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}



# Pigeonhole Principle



10 pigeons fly into a set of 9 pigeonholes to roost (i.e. rest)

$\#_{pigeons} > \#_{pigeonholes}$

$\Rightarrow$ at least 1 pigeonhole has >1 pigeon in it

(we can try filling the pigeonholes so as to avoid placing 2 pigeons in the same one, but eventually they all fill up & we still have more pigeons to place into holes)

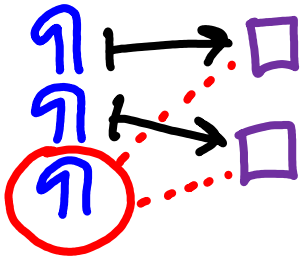
## Restate as fact about functions

$P$ = A set of pigeons

$H$ = A set of pigeonholes

**pigeonhole principle:** $|P| > |H| \Rightarrow \neg\exists f: P \to H$ s.t. $f$ is injective

e.g.



Cant define $f$ on the last pigeon s.t. function remains one-to-one

# Generalized Pigeonhole Principle

When placing $N$ objects into $K$ boxes, there is at least one box with at least $\lceil N/K \rceil$ objects in it.

$\lceil n \rceil$ = "ceiling" of $n$ = round $n$ up to nearest integer

**Idea:** Begin dispersing the objects evenly. You can place $\lfloor N/K \rfloor$ = "floor" of $N/K$ objects this way. Any remaining objects still need to be placed in some box (this is why we take ceilings)

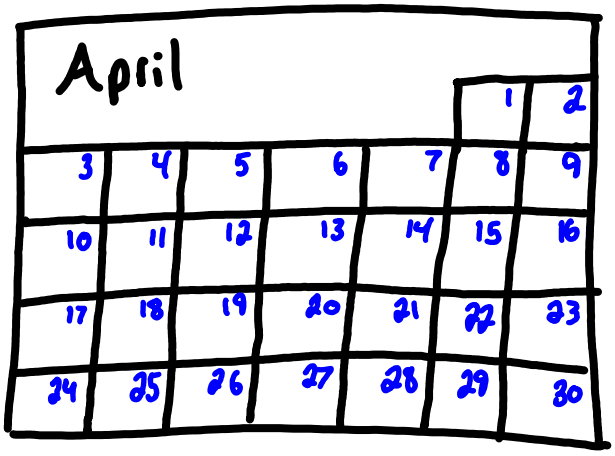
e.g. 100 objects placed in 8 bins

$\Rightarrow \exists$ a bin with $\lceil \frac{100}{8} \rceil = \lceil 12.5 \rceil = 13$ objects in it

---

91 movie tickets sold for 6 different movies $\Rightarrow$ one movie has an audience of at least $\lceil 91/6 \rceil = \lceil 15\ 166 \rceil = 16$

# More Applications




- Each day of a 30 day month a baseball team plays at least 1 game
- Over the course of the the month, the total # of games played does not exceed 45

**Claim:** $\exists$ a consecutive string of days during which the team plays exactly 14 games

**proof:** $a_j :=$ # games played between day 1 & day $j$ (inclusive)

$$a_1 < a_2 < \dots < a_{30} \quad \& \quad 15 \leq a_j + 14 \leq 59 \ \forall j$$

thus, $a_1, \dots, a_{30}, a_1+14, \dots, a_{30}+14$ is a list of 60 positive integers all $\leq 59$

60 pigeons $\sim \{a_i, a_i+14\}$

59 pigeonholes $\sim \{1, 2, \dots, 59\}$ - possible values of above integers

$\Rightarrow$ at least 2 integers are equal.

$a_i \neq a_j$ for $i \neq j$

So we must have $a_i = a_j + 14$ meaning exactly 14 games were played from day $j+1$ to day $i$ □

**Claim:** Among any $n+1$ positive integers not exceeding $2n$ there must be an integer that divides one of the others.

**Idea:** pigeons ~ odd parts of integers

holes ~ $\#$ odds $\leq 2n$

**proof:** Let $\{a_1, a_2, \ldots, a_{n+1}\} \subseteq \mathbb{Z}_{>0}$ be s.t. $a_j \leq 2n$ for all $j$

$\forall j$ write $a_j = 2^{k_j} q_j$ (take out all factors of 2)

$q_1, q_2, \ldots, q_{n+1}$ are odd positive integers $\leq 2n$.

but $\exists n$ odd positive integers $\leq 2n$

$\Rightarrow \exists i, j$ s.t. $q_i = q_j \Rightarrow$ either $a_i$ divides $a_j$ or $a_j$ divides $a_i$ □