ACM CP

Session 4:
Combinatorics

Counting

Counting: Motivation

- Counting problems arise throughout mathematics and computer science. It's important to count things *correctly*

- Outside school, formulas and step-by-step procedures aren't memorised.

- The math that makes a *good programmer* is not what is required in typical math exams from schools.

Sum Rule

- If a task can be done either in one of n_1 ways **or** in one of n_2 ways, where none of the set of n_1 ways is the same as any of the set of n_2 ways, then there are

$$n_1 + n_2$$

ways of doing the overall procedure.

Counting: Warmup

- **Date Night**
  Alicia needs to choose a dress from one of her **three** closets. The closets contain **20**, **21** and **29** possible dresses. All her dresses are unique. How many possible dresses are there for her to choose from?

Product Rule

- Suppose that a procedure can be broken down into two sequential tasks.

- If there are n_1 ways to do the first task and for each one of these ways, there are n_2 ways to do the second task, then there are:

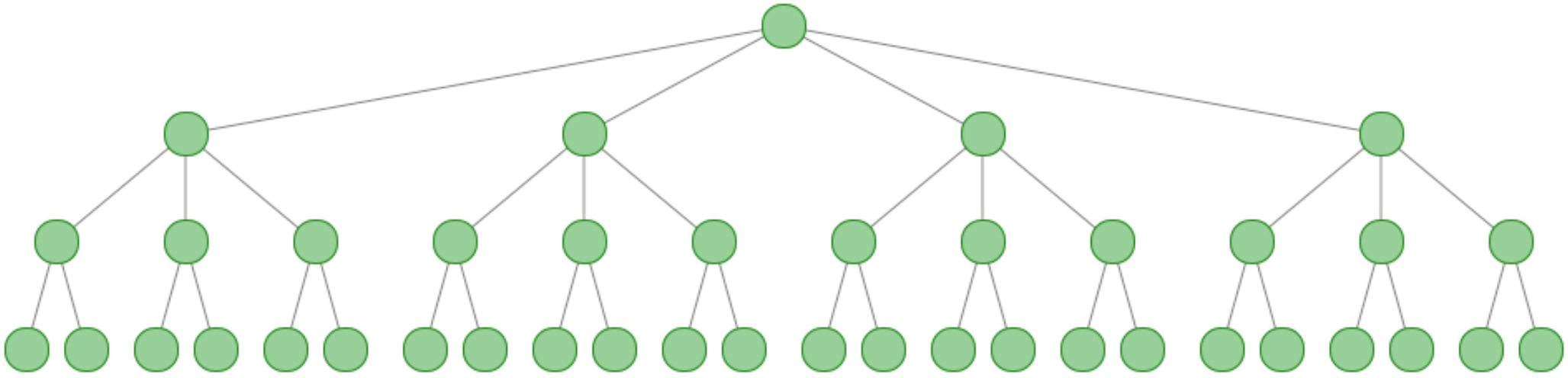
$$n_1 \cdot n_2$$

ways of doing the overall procedure.

A Possibly New Intuition to Counting

As a Computer Scientist

There are 4 connections
between first and second levels of nodes.
There are 3 connection between levels 2 and 3.
2 between levels 3 and 4.



4 x 3 x 2

Now, what does each path represent?

Counting: Warmup

- **Cracking The Code**
  A PIN code is composed of two digits and a letter. It takes one second to try a PIN. In the worst case, how much time do we need to crack a PIN?

- **Team Building**
  There were a total of 14 candidates applying for BPDC ACM council positions. For each candidate, we decided to toss a coin and only hire them if it shows a heads. How many team configurations are possible?

Counting: Warmup

- How many functions are there from a set with **'m'** elements to a set with **'n'** elements?

- How many **one-to-one** maps are there from a set with **'n'** elements to a set with **'m'** elements?
  Hint: What conditions should be satisfied by both sets to **not** support a one-to-one map?

- **Counting the number of onto functions? (5 points)**

Gandalf's Journey

https://www.hackerrank.com/challenges/connecting-towns/problem

Gandalf's Journey Logic

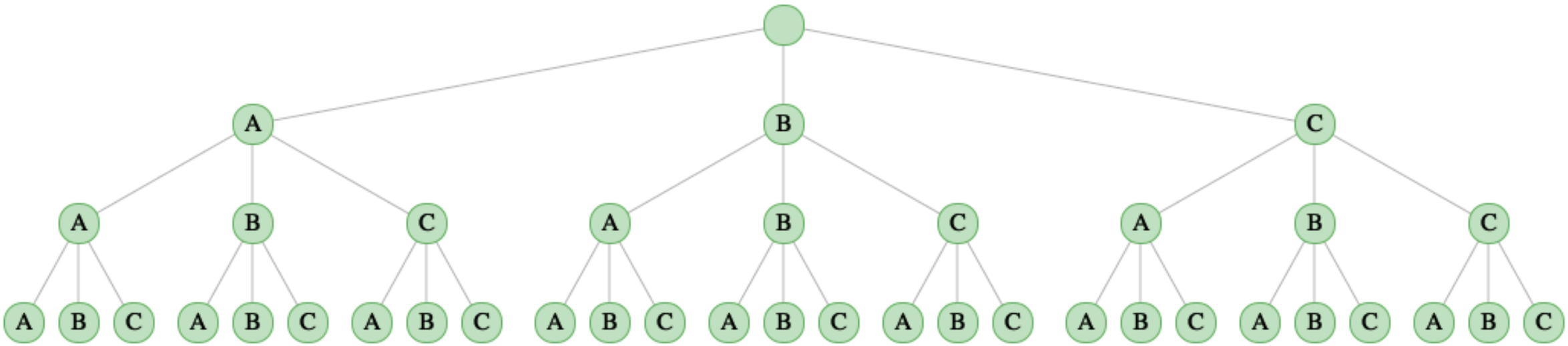
```
for(int i = 0; i < t; i++)
{
    int p = 1;
    cin >> n;

    for(j = 0; j < n-1; j++){

        cin >> x;
        p*= x;
        p%=1234567;
    }

    out[i] = p
}
```

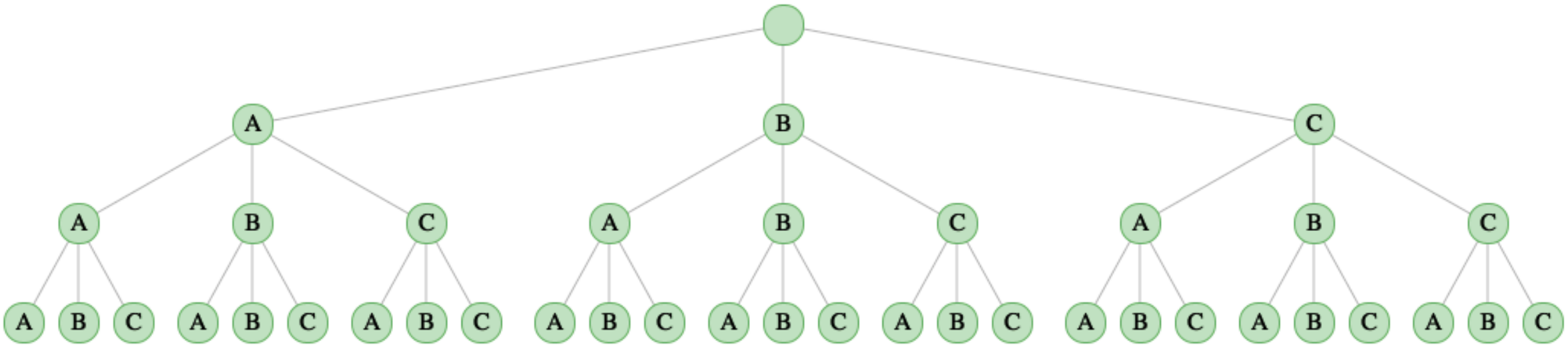
Set {A,B,C} with repetition of letters
A permutation is a function that maps a set of n elements to another set of n elements by swapping objects



You might have come across this as $\mathbf{n^n}$

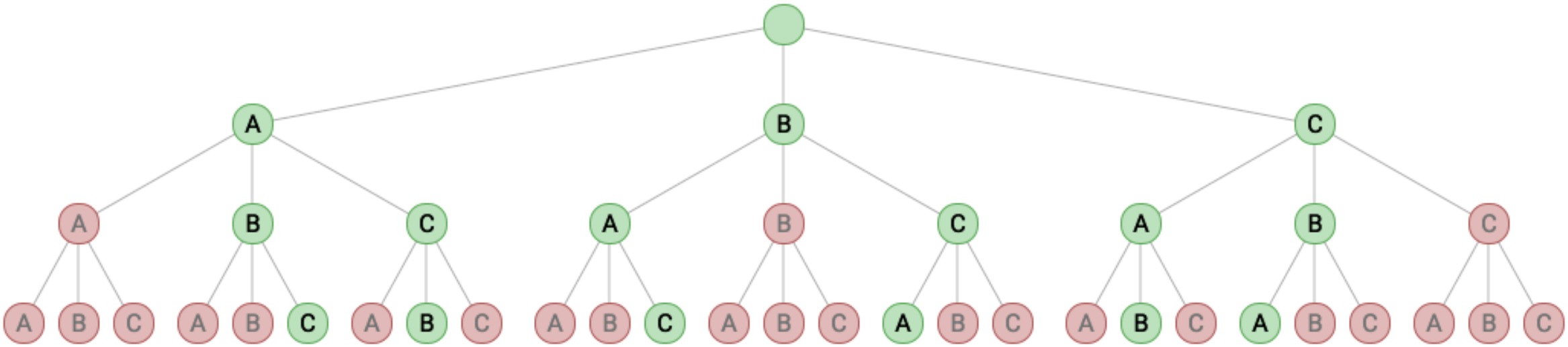
If there are n elements for an $\mathbf{n^n}$ operation,
Every node has **n** children.

Permutations without repeated elements: Start with an n^n tree.



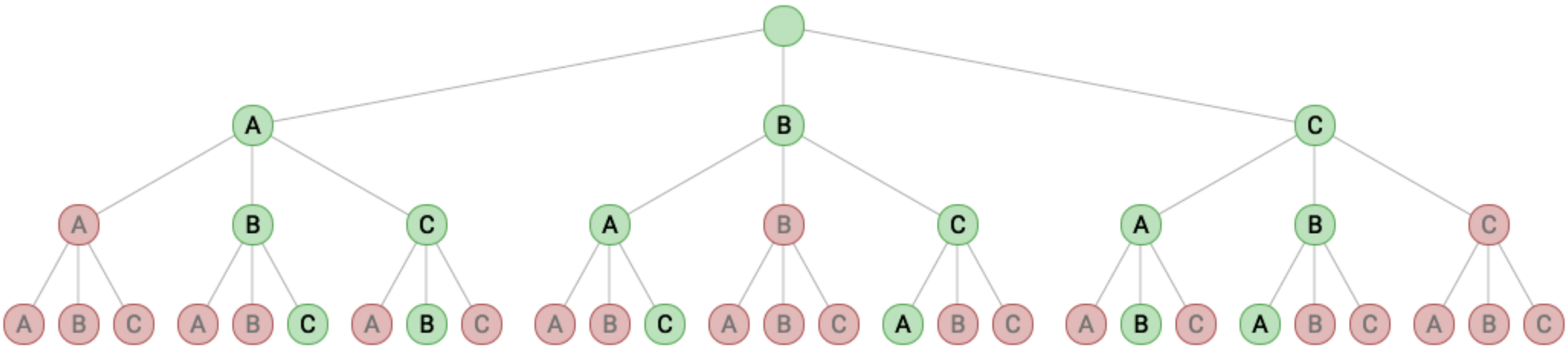
Mark Invalid paths

```
if(node.value has already been visited)
    Mark node as invalid
    Continue until all paths traversed
```

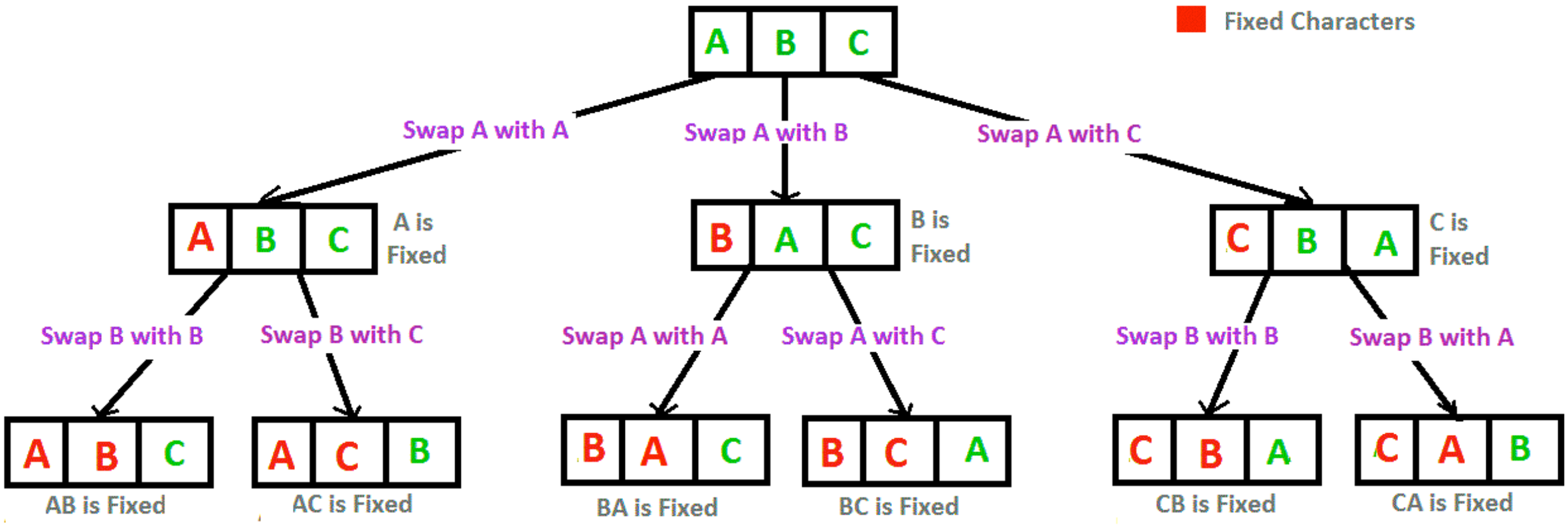


For this ’n’ level tree, the first level node branches ’n’ times, the second level nodes branch ’n - 1’ times, and the third level nodes branch ’n - 2’ times.

```
if(node.value has already been visited)
    Mark node as invalid
    Continue until all paths traversed
```



$$n! = n \times (n-1) \times (n-2) \times \ldots \times 1$$



Recursion Tree for Permutations of String "ABC"

Factorials and Permutations

- Number of ways to arrange *n* objects **(Factorial)**

$$n! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot n$$

- Number of ways to choose *k* objects from a set of *n* objects **(Permutations)**

$$^{n}P_{k} = \frac{n!}{(n - k)!}$$

Basic Counting

- Number of ways to choose *k* objects from a set of *n* objects when order does not matter (Combinations)

$$ {}^{n}C_{k} = \binom{n}{k} = \frac{n!}{k!(n-k)!} $$

Counting with Repetitions

- Number of permutations on *n* objects, where n_i is the number of objects with the '*i*'th value

$$\frac{n!}{n_1!n_2!n_3!\ \ldots\ n_k!}$$

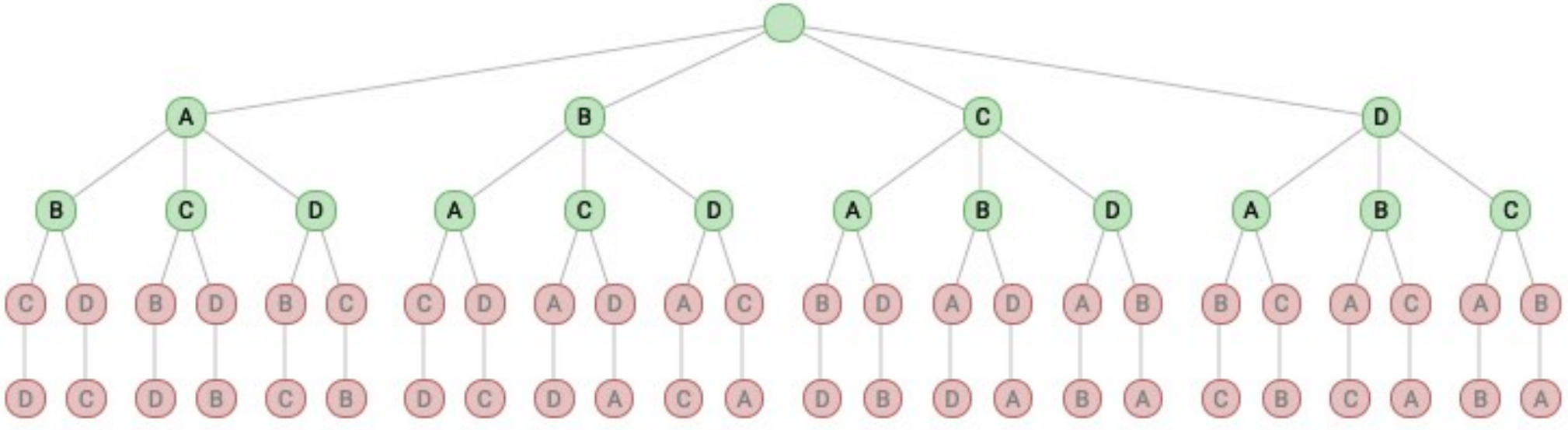
- Number of way to choose *k* objects from a set of *n* objects with, where each value can be chosen more than once

$$\binom{n + k - 1}{k} = \frac{(n + k - 1)!}{k!(n - 1)!}$$

Volunteering time

Explain the partial permutations with the n^n tree.

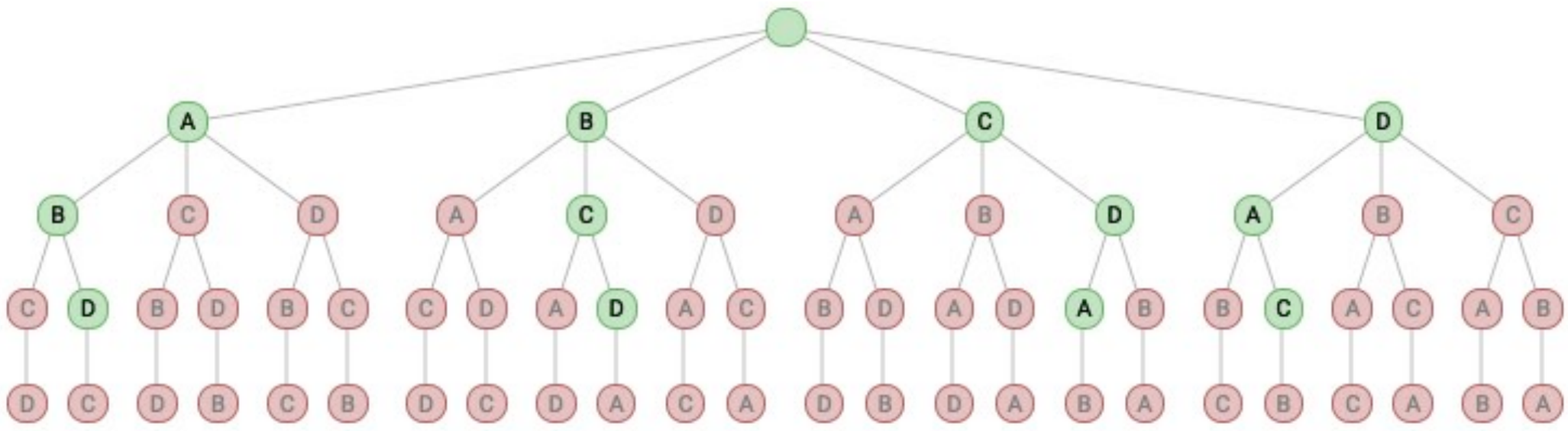
$$ {}^{n}P_{k} = \frac{n!}{(n - k)!} $$


A
B
C
D
B
C
D
A
C
D
A
B
D
A
B
C
C
D
B
D
B
C
C
D
A
D
A
C
B
D
A
D
A
B
B
C
A
C
A
B
D
C
D
B
C
B
D
C
D
A
C
A
D
B
D
A
B
A
C
B
C
A
B
A

Volunteering time

Explain the Combinations with the n^n tree.

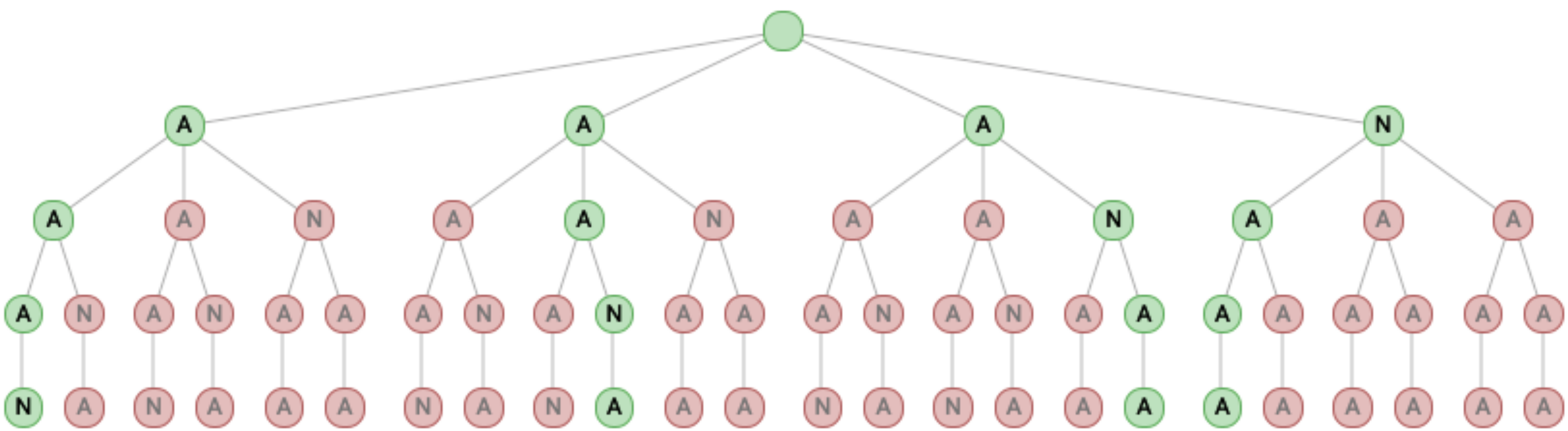
$$ {}^{n}C_{k} = \binom{n}{k} = \frac{n!}{k!(n-k)!} $$



Volunteering time

Explain Multi-Set Permutations with the n^n tree.

$$\frac{n!}{n_1!n_2!n_3! \ldots n_k!}$$



Recursion

- **Recursion** formalises the process of recognising how solutions to smaller cases of a problem can, layer by layer, be built up to solve any case of a problem, no matter how enormous.

- Recursion is particularly important for developing formulas in combinatorics.

Mathematical Thinking vs Computational Thinking

- Mathematical analysis often reveals patterns from a bird's eye point of view. This can be very useful given that computers do not have this ability yet.

- Consider the example of Fibonacci numbers.

- If you were asked to find out the nth fibonacci number given a number 'n', and a computer, pen and piece of paper what would you do?

Fibonacci Program

```
Fibonacci(n)
     a <- 0,
     b <- 1,
     c <- a + b;
     i <- 3;
     while(i < n)
     {
          a <- b;
          b <- c;
          c <- a + b;
          i++;
     }
     return c;
```

When To Use Recursion

- Both the following conditions to hold simultaneously

- The problem asks for the value of a large case, and you can determine small values or they are given.

- There is a clear connection between small cases and large cases.

Recursive Onions

Recursion should be applied as a technique when the problem you are solving is like an onion. Using recursion requires realising that a big, complex problem that's likely to make you cry is really just a slightly smaller problem, plus one super-thin layer. And that problem, in turn, is just a slightly smaller problem, plus one super-thin layer…

Alternative Fibonacci Program

```
Fibonacci(n)
    int c = Plug 'n' into Fibonacci closed form;
    return c;
```

Given:

$$
\begin{aligned}
f(0) &= 0 \\
f(1) &= 1 \\
f(n) &= f(n-1) + f(n-2)
\end{aligned}
$$

Then:

$$
f(n) = \frac{(1+\sqrt{5})^n - (1-\sqrt{5})^n}{2^n \sqrt{5}}.
$$

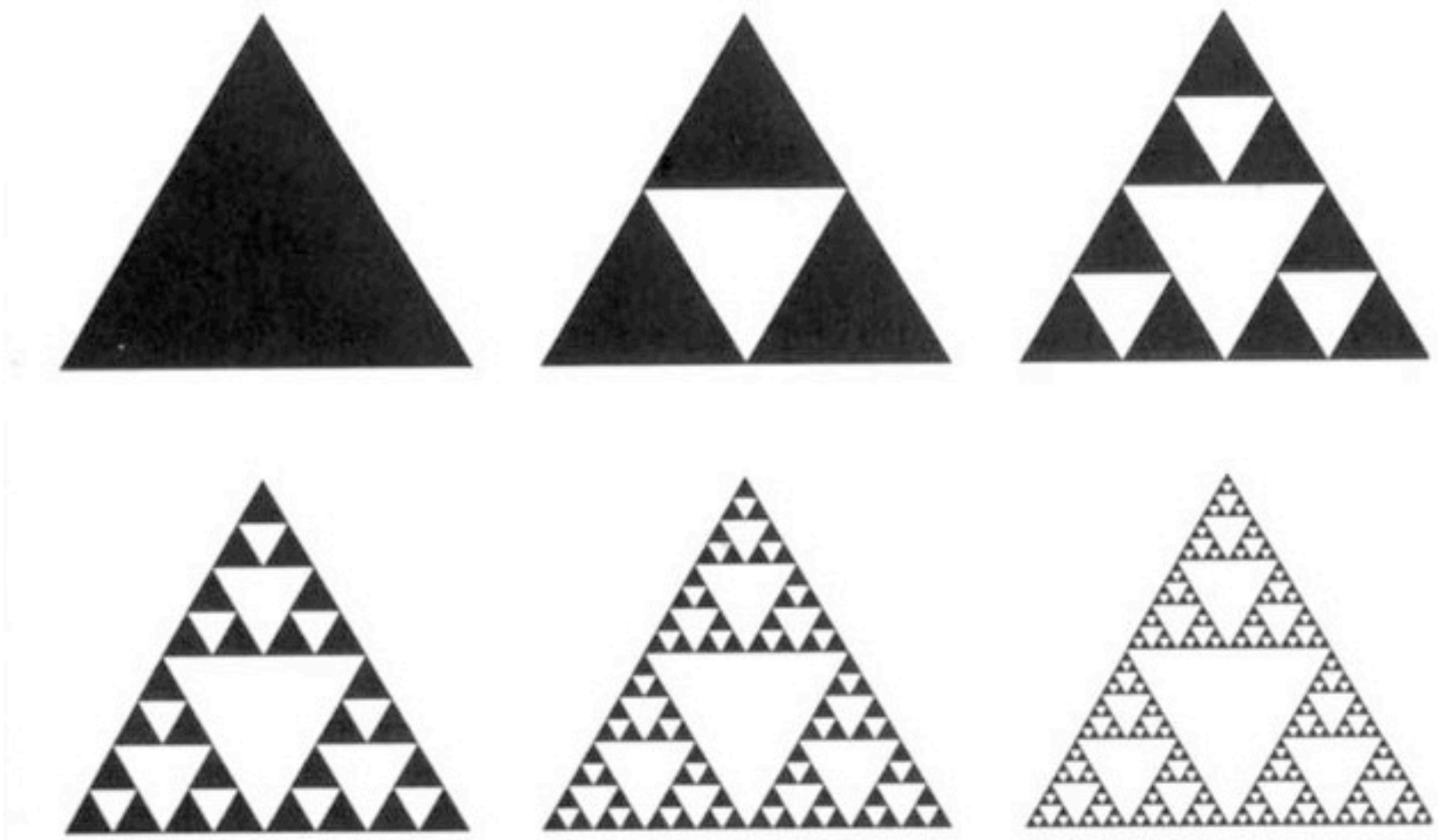
Ancient Wisdom

If Recursion is the operation, Recurrence relations are the structure where this operation can be carried out.

Finding a recurrence relation requires identifying some regular relationship between each case of a problem and the earlier cases. But many times it won't be solely the case just before that's relevant. Sometimes, later cases will evolve from combining two, three, or any number of previous cases together.

Infinitely Recursive Procedures

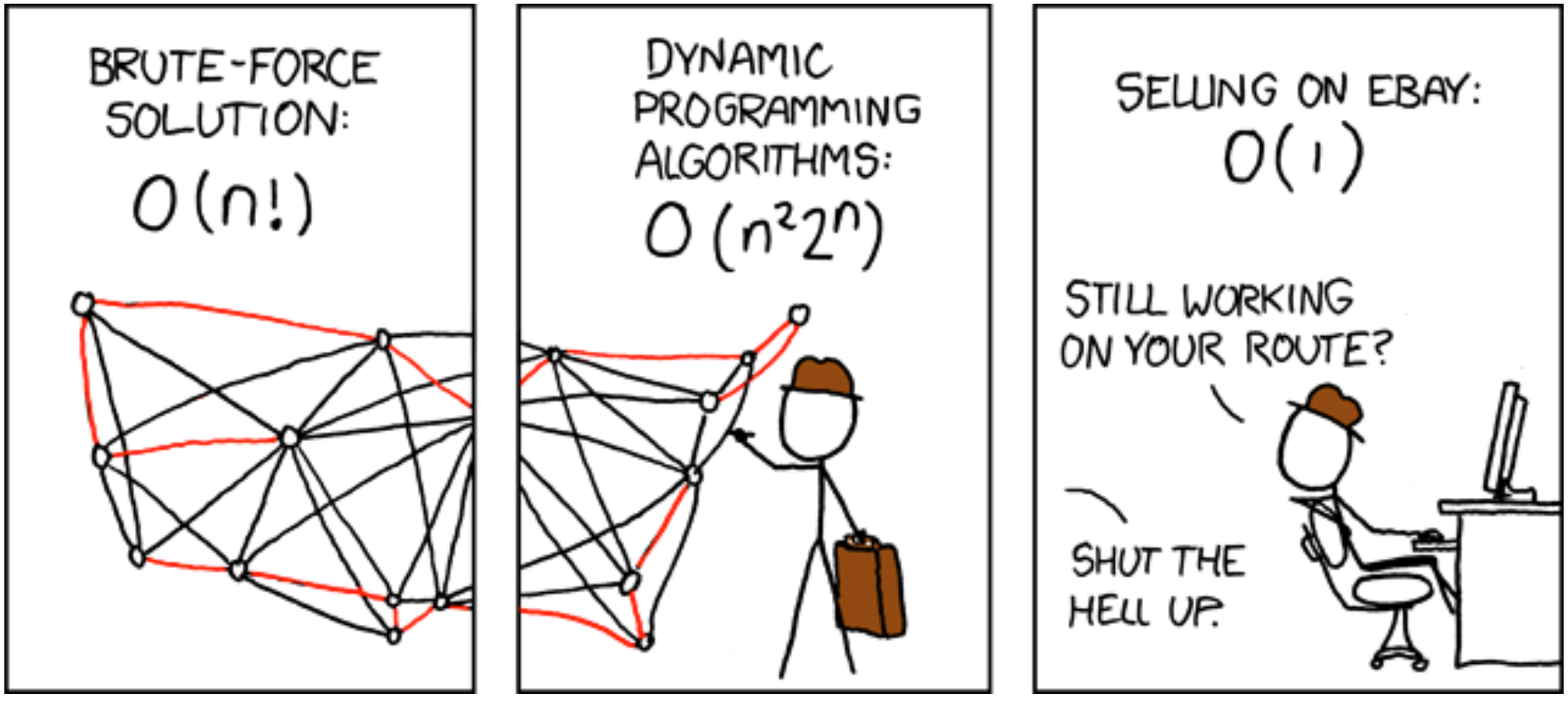
Sierpinski Triangle



https://github.com/relarson/Sierpinski

Back to Factorials

- **Traveling Sailsman**
  You have purchased your own private cruise ship and wish to sail to 20 famous port cities as a millionaire tourist. You ask Ralexa, your personal voice assistant for the best route that minimises your boat's fuel consumption. Ralexa has access to a powerful computer and helps you run the numbers. If it takes 1 microsecond to calculate the distance of one sea-route, how long does it take to compute the distance of all possible sea-routes?


BRUTE-FORCE SOLUTION:
O(n!)
DYNAMIC PROGRAMMING ALGORITHMS:
O(n²2ⁿ)
SELLING ON EBAY:
O(1)
STILL WORKING ON YOUR ROUTE?
SHUT THE HELL UP.

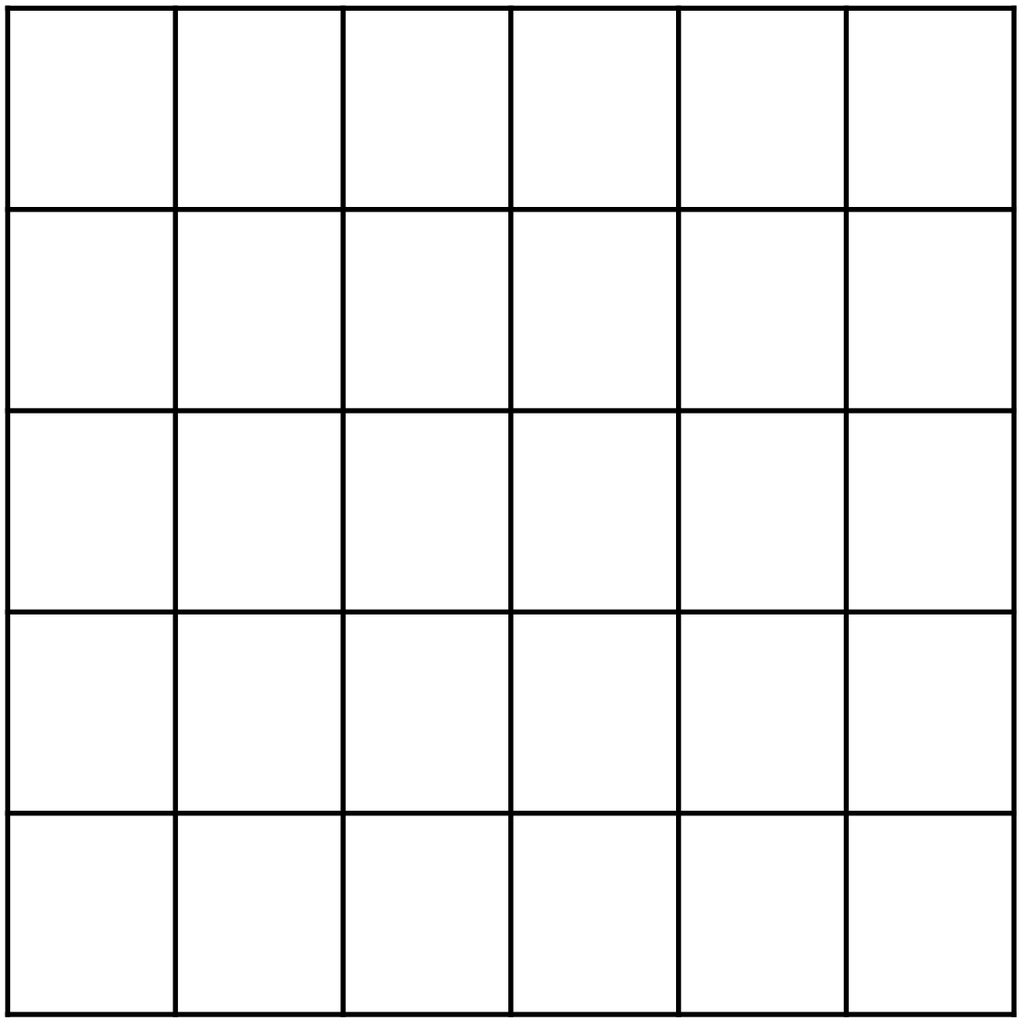
xkcd.com

Any guesses on what this is?

Problem for 3 points

- **Think Different**
  As the new technical head of Apple's Logic X Music Studio Software, you are approached by a famous musician. She is using a scale of 15 different notes in her new composition. She wants Logic X to render all possible melodies that uses 8 notes only. Each note should play once per melody, and each eight-note melody should play for a second. How much audio runtime does Logic X need to support?

How many rectangles in n x m grid?



$$\binom{n}{2} * \binom{m}{2}$$

Properties of Binomial Coefficient

$$\binom{n}{k} = \binom{n}{n-k}$$

$$\binom{n}{0} = \binom{n}{n} = 1$$

$$\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$$

Pascal's Triangle

$$\binom{0}{0}$$

$$\binom{1}{0} \quad \binom{1}{1}$$

$$\binom{2}{0} \quad \binom{2}{1} \quad \binom{2}{2}$$

$$\binom{3}{0} \quad \binom{3}{1} \quad \binom{3}{2} \quad \binom{3}{3}$$

$$\binom{4}{0} \quad \binom{4}{1} \quad \binom{4}{2} \quad \binom{4}{3} \quad \binom{4}{4}$$

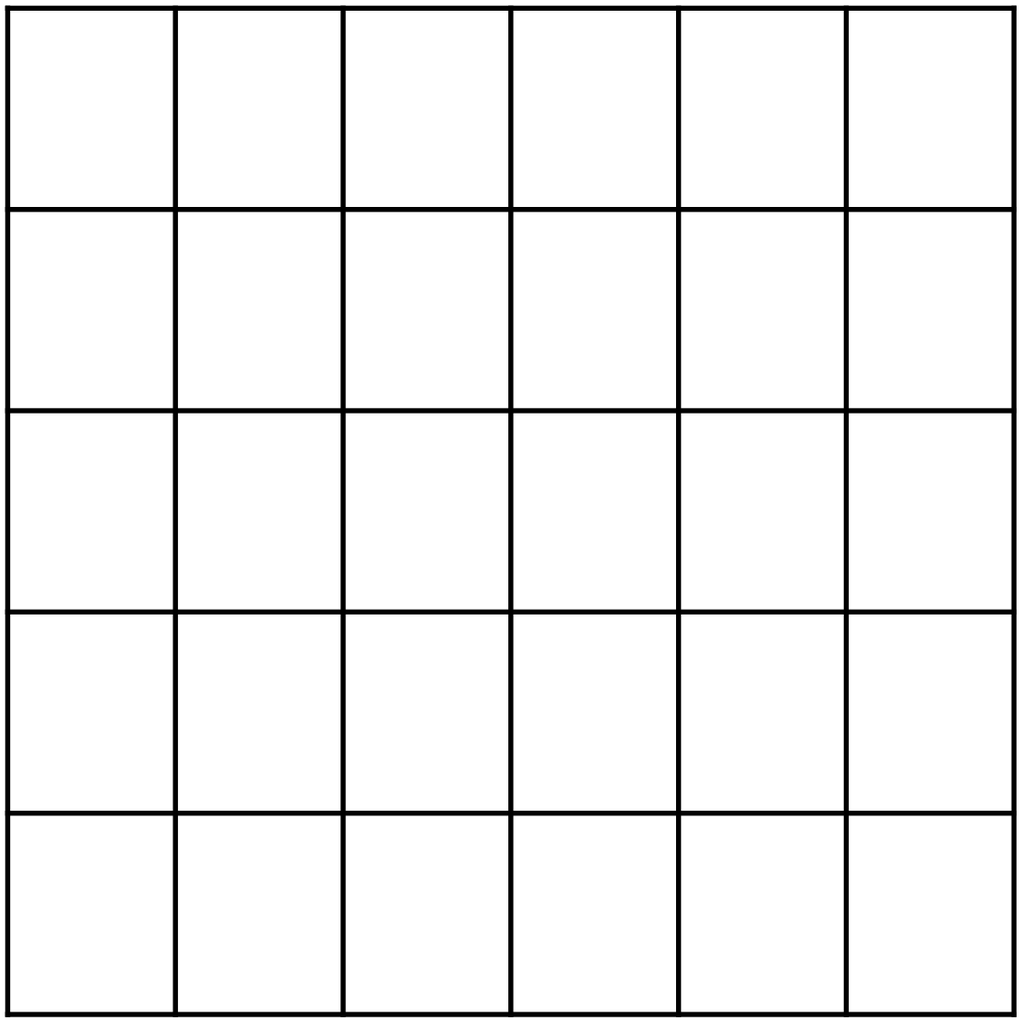
$$\binom{5}{0} \quad \binom{5}{1} \quad \binom{5}{2} \quad \binom{5}{3} \quad \binom{5}{4} \quad \binom{5}{5}$$

Grid Walks

- A **grid walk** is a path that is taken along a graph from a starting node to an ending node.

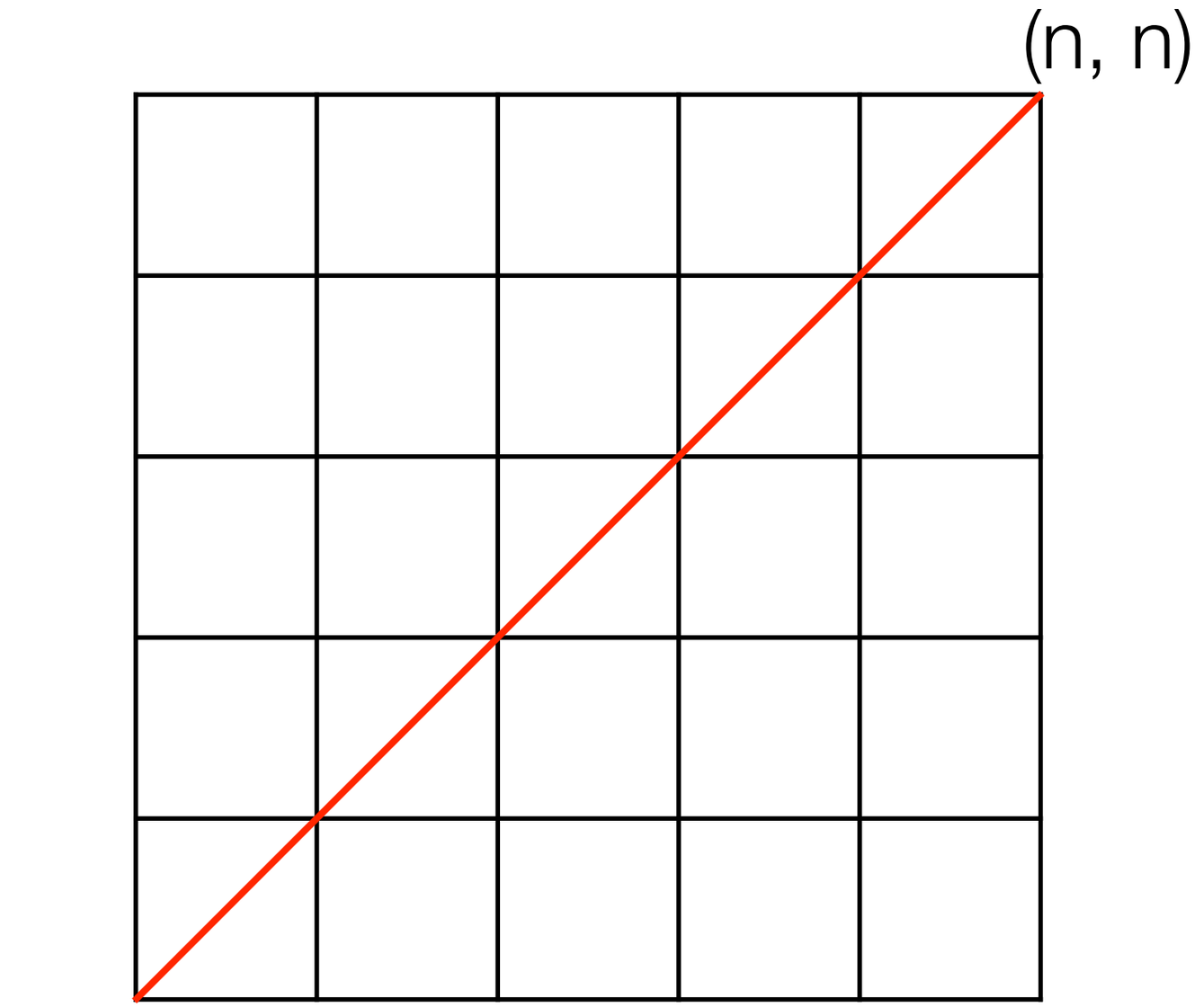
- In a grid walk problem, the goal is to find the number of possible paths.

How many paths from (0, 0) to (n, m)?



$$\binom{m+n}{m}$$

How many paths from (0, 0) to (n, n)?



$$C_n = \frac{1}{n+1}\binom{2n}{n}$$

Partner up!

Coding Time

A Chocolate Fiesta

https://www.hackerrank.com/challenges/a-chocolate-fiesta

Sherlock and Pairs

https://www.hackerrank.com/challenges/sherlock-and-pairs/problem

Random Teams

http://codeforces.com/problemset/problem/478/B

Birthdays

https://www.hackerrank.com/contests/infinitum15/challenges/birthdays

References

- Recursion Reference
- https://buildingvts.com/intuition-behind-permutations-and-combinations-db6ffa5272be
- https://brilliant.org/wiki/combinatorics/
- Computer Science Distilled by Filho
- Discrete Mathematics and Applications by Rosen