Fermat's (Little) Theorem:
If p is a prime and if a is any integer, then $a^p \equiv a \pmod{p}$.
In particular, if p does not divide a, then $a^{p-1} \equiv 1 \pmod{p}$.

MASTER THEOREM Let f be an increasing function that satisfies the recurrence relation

$$f(n) = af(n/b) + cn^d$$

whenever $n = b^k$, where k is a positive integer, $a \geq 1$, b is an integer greater than 1, and c and d are real numbers with c positive and d nonnegative. Then

$$f(n) \text{ is } \begin{cases} O(n^d) & \text{if } a < b^d, \\ O(n^d \log n) & \text{if } a = b^d, \\ O(n^{\log_b a}) & \text{if } a > b^d. \end{cases}$$

Simple: no loops or multiple edges
Vertex: nodes
Edge: lines connecting the nodes
Degree of vertex: how many lines connecting
Degree sequence: degree in a low to high sequence
Direct vs undirect: look for the arrows
Multiple edges: look for the circle around any edge
Loops: look for two lines in between two nodes(lines that start and end at the same node)
Handshake thm: degree of vertex will be twice the num of edges in total

Bipartite Graph: Imagine you have a group of friends and you can divide them into two smaller groups in such a way that no one in the first group is directly friends with anyone else in the same group, and the same goes for the second group. But people from the first group can be friends with people from the second group. That's a bipartite graph. The two smaller groups are V1 and V2, and they make up the whole group of friends (V). Every friendship (edge) is between someone from V1 and someone from V2.

Bipartite cannot contain odd cycles

Bipartite thm:
The graph is bipartite.
The graph is 2-colorable. This means you can give everyone in one group (say, V1) a red shirt and everyone in the other group (V2) a blue shirt, and no two friends will have the same color shirt. In more technical terms, there's a function that assigns a color (red or blue) to each person so that if two people are friends, they have different colors.
The graph does not contain any odd cycles. In other words, you can't start at one person, go through an odd number of friendships, and end up back at the same person without retracing any friendships. In more technical terms, the graph doesn't have any subgraphs that are odd cycles (C2k+1).

Path: Think of a path as a walk through a park, where each step takes you to a new place (vertex). You can go from one place to another as long as they're connected by a path (edge).

Simple Path: This is like taking a walk without visiting the same place twice.

Connected Vertices: Two places (vertices) are connected if you can walk from one to the other, even if they're not directly next to each other.

Connected Component: This is like a group of places in the park that you can get to from any other place in the group.

Connected Graph: A park is connected if you can walk from any place to any other place in the park.

Subgraph: This is like a smaller park within the larger park. All the places (vertices) and paths (edges) in the smaller park are also in the larger park.

Cyclic Graph: This is like a park where you can start walking from a place, take a few steps, and end up back at the same place.

Acyclic Graph: This is like a park where no matter how you walk, you can never end up back at the same place without retracing your steps.

Tree: This is like a park where you can walk from any place to any other place exactly one way, without having to retrace your steps or ending up back at the same place.

Tree Theorems: If you're in a tree (the park described above), there's only one way to get from one place to another. And if there are n places, there are n-1 paths.

Euler Path: This is like taking a walk in the park where you walk on every path exactly once. You can start and end at the same place, or at different places.
Euler path has to have all even degrees of vertex
Euler Circuit: This is like an Euler Path, but you start and end at the same place.

Euler's Theorem: A park has an Euler circuit if and only if you can walk out of any place and come back the same number of times.

Hamiltonian Path: This is like taking a walk in the park where you visit every place exactly once.
Hamiltonian: has to be connected! Common counterexamples: even number of nodes like 0 or 4 with one being disconnected

Hamiltonian Cycle: This is like a Hamiltonian Path, but you start and end at the same place.

Tree: has to be connected and not contain cyclic subgraphs/or cycles

- **Addition:** $(f_1 + f_2)(n) = \Theta(\max(g_1(n), g_2(n)))$.
- **Scalar Multiplication:** $a \cdot f_1(n) = \Theta(f_1(n))$.
- **Product:** $(f_1 \cdot f_2)(n) = \Theta(g_1(n) \cdot g_2(n))$.

Cards: C(13, ranks)*C(suits, target)*repeat how many times to choose

$$P(F_2 \mid E) = \frac{P(E \mid F_2)P(F_2)}{P(E \mid F_1)P(F_1) + P(E \mid F_2)P(F_2) + P(E \mid F_3)P(F_3)}$$

Graph Isomorphism: Two graphs are isomorphic if they are essentially the same graph, just drawn or represented differently. Imagine you have two pictures of the same park, but taken from different angles. The pictures might look different, but they're still of the same park. That's what graph isomorphism is like. The technical definition involves a bijection (a one-to-one correspondence) between the vertices of the two graphs that preserves the connections between vertices (edges).
Look for degrees of vertex!

Graph Invariant: A graph invariant is a property that doesn't change, no matter how you draw or represent the graph. It's like the number of trees in the park – it doesn't matter what angle you take the picture from, the number of trees remains the same. Some examples of graph invariants include:

Number of vertices: The number of vertices in a graph doesn't change, no matter how you draw or represent the graph. It's like the number of places in the park.

Number of edges: The number of edges in a graph also doesn't change. It's like the number of paths in the park.

Degree sequence: The degree of a vertex is the number of edges connected to it. The sequence of all these degrees, sorted in non-increasing order, doesn't change under isomorphism. It's like if each place in the park had a sign showing how many paths lead from it, and you wrote down these numbers in decreasing order.

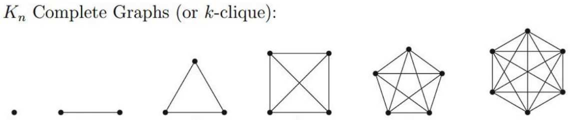
Existence of subgraphs/path properties: Certain properties related to the existence of subgraphs or paths within the graph don't change under isomorphism. It's like the fact that there's a path from the entrance to the playground, no matter how you draw the park.

Cyclic or acyclic: Whether a graph is cyclic (has cycles) or acyclic (doesn't have cycles) doesn't change under isomorphism. It's like whether or not there's a roundabout in the park.

Having paths of certain length: The existence of paths of a certain length between vertices doesn't change under isomorphism. It's like whether or not there's a long path from the entrance to the playground.

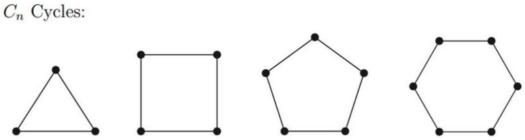
Remember, if two graphs are isomorphic, they share all these invariants, but just because two graphs share these invariants doesn't necessarily mean they're isomorphic. There could be two parks with the same number of places, the same number of paths, and so on, but they're still different parks.

K_n Complete Graphs (or k-clique):



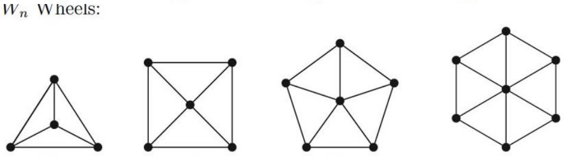
K_1 K_2 K_3 K_4 K_5 K_6

C_n Cycles:



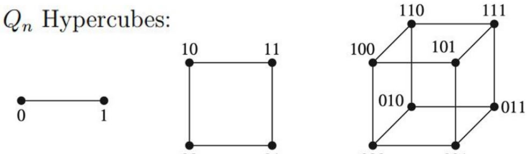
C_3 C_4 C_5 C_6

W_n Wheels:



W_3 W_4 W_5 W_6

Q_n Hypercubes:



Graph Type	Is Bipartite?
K (Complete)	Yes, if $n = 2$. No, if $n > 2$
C (Cycle)	Yes, if n is even. No, if n is odd
Wheels	Yes, if n (excluding the center vertex) is even. No, if n is odd
Q (Hypercube)	Yes, for all n
Tree	Yes, for all n

Bernoulli Trials and the Binomial Distribution

For a set number of trials, examining number of successes.

- **Bernoulli Trials:** Each performance of an experiment with two possible outcomes is called a Bernoulli trial.
  - We typically label these outcomes success and failure
  - If *p = Pr(success)* and *q = Pr(failure)*, then ***p + q = 1***
- **Binomial Distribution:** We call the probability distribution of the number of successes in a sequence of *n* Bernoulli trials the Binomial Distribution
  - The **probability of exactly *k* successes** in *n* independent Bernoulli trials, with *p = Pr(success)* and *q = Pr(failure) = 1 - p*, is

    $$Pr(numSuccesses = k) = C(n,k)\,(p^k)\,(q^{n-k})$$

  - The **expected number of successes** in *n* independent Bernoulli trials where *p = Pr(success)*, is

    $$E(numSuccesses) = np$$

The Geometric Distribution

For examining number of trials to get a success.

- **Bernoulli Trials:** A random variable X has a geometric distribution with parameter p if

  $$Pr(X = k) = (1 - p)^{k-1}p$$

  for k = 1,2,3, ...

  This is saying that the first *k–1* trials were failures (so probability *1–p*) and the k^{th} trial was a success (so probability *p*). So the **parameter *p*** stands for the probability of success and the **random variable X** is representing some number of trials it took to get a success.
  - The **expected value of X** (ie the expected number of trials to get a success) is

$$O(1) < O(\log n) < O(\sqrt{n}) < O(n) < O(n \log n) < O(n^2) < O(n^3) < O(2^n) < O(n!) < O(n^n)$$

$$P(A) = P(A \cap B_1) + P(A \cap B_2) + P(A \cap B_3) + \ldots + P(A \cap B_n)$$

This can also be written as: Law of total probability

$$P(A) = P(A|B_1)P(B_1) + P(A|B_2)P(B_2) + P(A|B_3)P(B_3) + \ldots + P(A|B_n)P(B_n)$$

Proving Graph Isomorphisms
● To prove that two graphs are isomorphic:
○ The only way to prove that two graphs are isomorphic is to provide an example of an isomorphism.
○ An isomorphism is a function from one set of vertices to the other such that ∀u,v [{u,v} ∈ E1 ↔ {f(u),f(v)} ∈ E2], as defined on the previous slide.
■ It is not sufficient to simply list some consistent invariants.
■ The following statement is true: IF two graphs are isomorphic, THEN the invariants are preserved, but NOT the other way around (the converse – not necessarily true).
■ Because of this, however, it is easier to disprove isomorphism.
Disproving Graph Isomorphisms
● To prove that two graphs are NOT isomorphic:
○ If you are trying to disprove that two graphs are isomorphic, you are trying to prove that there does not exist an isomorphism between them.
○ Thus, if a graph invariant is not the same in two graphs, they are NOT isomorphic.
○ As such, it is sufficient to simply list or describe an invariant that is different between the two graphs.

- **Product Rule**: If a task has multiple steps, and each step has a certain number of options, then the total number of ways to complete the task is just the options for each step multiplied together. Like if you're getting dressed and you have 2 shirts and 3 pants to choose from, then you have 2 * 3 = 6 different outfits you can wear.

- **Sum Rule**: If there are different ways to do a task, and each way has a certain number of options, then the total number of ways to do the task is just the options for each way added together. Like if you can get to school by bus or by bike, and there are 2 different bus routes and 3 different bike routes, then you have 2 + 3 = 5 different routes to school.

- **Division Rule**: If there are lots of ways to pick something, but each thing can be picked in the same number of ways, then the total number of things is just the total ways to pick divided by the ways to pick each thing. Like if you have 12 chocolates and each chocolate has 3 different flavors, then you have 12 / 3 = 4 different types of chocolates.

- **Difference Rule**: If you're counting how many ways you can do something, but some of the ways are not allowed, then the total number of ways you can do it is just the total number of ways minus the ones that are not allowed. Like if there are 10 roads to your house but 2 are closed for construction, then there are 10 - 2 = 8 roads you can take.

- **Inclusion-Exclusion Principle**: If a task can be done in different ways and some ways are counted in both, then the total number of ways to do the task is the sum of the ways for each, minus the ways that are counted twice. Like if you can travel to a city by bus or train, and there are 5 bus routes and 4 train routes, but 2 routes are the same for both, then there are 5 + 4 - 2 = 7 different routes to the city.

Permutations: This is like picking players for a sports team. If you have 10 players and you're picking 5 for your team, the order matters because the first player you pick is the captain. The number of different teams you can pick is calculated using the formula P(n,k) = n! / (n−k)!. In our example, it would be P(10,5) = 10! / (10-5)!. The "!" means factorial, which is just multiplying all the numbers from that number down to 1.

Combinations: This is like picking numbers for a lottery ticket. If you're picking 6 numbers out of 49, the order doesn't matter because 1, 2, 3, 4, 5, 6 is the same as 6, 5, 4, 3, 2, 1. The number of different tickets you can pick is calculated using the formula C(n,k) = n! / ((n−k)!k!). In our example, it would be C(49,6) = 49! / ((49-6)!6!).

Distinguishable Objects: These are things that are different from each other, like people or different cards in a deck. If you're picking people for a team or cards for a hand, each choice is unique.

Indistinguishable Objects: These are things that are the same, like cookies or copies of the same book. If you're picking cookies for a snack or books for a shelf, it doesn't matter which ones you pick because they're all the same.

Distributing Objects Into Bins: This is like sorting things into boxes. If the things and the boxes are different (distinguishable), it's like sorting different books into labeled boxes. If the things are the same but the boxes are different, it's like sorting cookies into labeled boxes. If the things are different but the boxes are the same, it's like sorting different books into unlabeled boxes. If the things and the boxes are the same (indistinguishable), it's like sorting cookies into unlabeled boxes. The way you count the number of ways to sort things changes depending on whether the things and the boxes are different or the same.

- **Experiment**: This is just something you do that has different possible outcomes. Like flipping a coin, you don't know if it will be heads or tails.

- **Sample Space**: These are all the possible outcomes of your experiment. If you're flipping a coin, the sample space is {Heads, Tails}.

- **Event**: This is a group of outcomes that you're interested in. If you're flipping a coin and you want it to land on heads, the event is {Heads}.

- **Probability of an Event (Equally Likely Outcomes)**: This is how likely it is that your event will happen. If all the outcomes are equally likely, like flipping a coin or rolling a die, you just divide the number of outcomes in your event by the total number of outcomes. If you're rolling a 6-sided die and you want to roll a 2 or 6, the probability is 2 (the number of outcomes in your event) divided by 6 (the total number of outcomes), which is 2/6 or 1/3.

- **General Probability of Events**: This is how likely it is that your event will happen if the outcomes aren't equally likely. You just add up the probabilities of each outcome in your event. The probabilities of all the outcomes in your sample space always add up to 1.

- **Conditional Probability**: This is the probability that one event happens given that another event has already happened. For example, if you have a bag of red and blue marbles, the probability that you pick a red marble given that you've already picked a blue one would be the conditional probability. It's denoted as P(E1 | E2), which means "the probability of E1 given E2".

- **Independence**: Two events are independent if the outcome of one event doesn't affect the outcome of the other. For example, flipping a coin and rolling a die are independent because the outcome of the coin flip doesn't affect the outcome of the die roll. You can check if two events are independent by seeing if the probability of both events happening is the same as the product of the probabilities of each event happening. This is expressed as P(E ∩ F) = P(E) * P(F).

- **Conditional Probability and Independence**: If the conditional probability of an event is the same as the probability of that event, then the events are independent. This means that the outcome of one event doesn't affect the outcome of the other. For example, the probability of getting a head on a coin flip is the same whether or not you've rolled a die first, so the two events are independent. This is expressed as P(E) = P(E | F).

- **Random Variable**: This is a way to assign numbers to the outcomes of an experiment. For example, if you're flipping a coin, you could assign a 1 to getting heads and a 0 to getting tails. That way, you can do math with the outcomes of your experiment.

- **Expected Value**: This is the average outcome you'd expect if you did the experiment many times. You calculate it by multiplying each possible outcome by its probability and then adding those up. For example, if you're flipping a coin (where heads is 1 and tails is 0), the expected value is 1*0.5 (probability of heads) + 0*0.5 (probability of tails) = 0.5.

- **Linearity of Expectations**: This is a fancy way of saying that you can add up expected values. If you have some random variables and you want to know the expected value of their sum, you can just add up their individual expected values. This is true even if the random variables are dependent (i.e., the outcome of one affects the outcomes of the others).

- **Indicator Random Variable**: This is a simple type of random variable that only takes on the values 0 and 1. It's often used to indicate whether a certain event has happened. For example, you might have an indicator random variable that is 1 if you roll a 6 on a die and 0 otherwise.

- **Big-O Notation (O(g))**: <=

- **Big-Omega Notation (Ω(g))**: >=

- **Big-Theta Notation (Θ(g))**: =

- **Note**: A function f(n) is Θ(g(n)) if and only if f(n) is both O(g(n)) and Ω(g(n)). This means that the growth rate of f(n) is both no faster than and at least as fast as g(n), which implies that f(n) grows at the same rate as g(n).

Counting 101

The first digit should not start with 0
Distinct means different numbers for each digit
Divisible by n in an intvl of a to b means b/n

Examples:
Product: labeling chairs with letters followed by num
License plate with no restrictions
Sets of n to sets of m = n x n m times = n^m

Sum: no duplicates choosing by steps

See or? Inclusion exclusion: always minus the smaller overlap

Division: given n ppl, overcount with n! and divide the overcount (usually this is n for tables, or books that are not labeled) also divide by n if rotation is involved

Distinguishable: different from each other, “labeled”
○ Different people
○ Labelled boxes
○ Different cards in a standard deck of cards
● Indistinguishable: considered identical, “unlabeled”
○ Chocolate chip cookies
○ Occurrences of S in “SUCCESS”
○ Zeros in a bit string
○ Copies of the same book
○ Unlabelled boxes

Expected values: linear, the total expected

Choosing password of n digits with no restrictions from r numbers is r^n because you just choose n time from r numbers

Disjoint = mutually exclusive

$$P(E_1|E_2) = \frac{P(E_1 \cap E_2)}{P(E_2)}$$

Putting in identical bins = 1 way
Both unique = P(a, b)
Uniq bins but same obj = C(a,b)
So, in your example, you divide by (3!) 4 times to account for the indistinguishable objects in each bin (side of the table), and you divide by $4!$ to account for the interchangeable bins (sides of the table).
Remember, this rule of thumb applies to situations where you have indistinguishable objects within bins or interchangeable bins. If all objects are distinguishable and bins are not interchangeable, then you typically do not need to divide by any factorials.

Examples:
Permutation:
winner
No restrictions -> group objects
Choosing numbers:
Total!/dup!

Combination:
Cards
Choosing committee
Choosing letters: C(n,k)*C(n-k,k-1)*C(n-k-(k-1),k-1-1)*

Can add up the ones that match the criteria, or subtract the ones that don’t

Disproving independent: P(E n F) != P(E)*P(F)

1. **Constant Time ($O(1)$)**: indexing

2. **Logarithmic Time ($O(\log n)$)**: The size of the problem is halved. Binary search is a typical example.(halve the iterator)

3. **Linear Time ($O(n)$)**: loops thru every element

4. **Linearithmic Time ($O(n \log n)$)**: Merge sort and quicksort, use recursion.

5. **Quadratic Time ($O(n^2)$)**: nested loops

7. **Exponential Time ($O(2^n)$)**: The number of operations doubles with each addition to the input data set. The classic recursive calculation of Fibonacci numbers is an example.

8. **Factorial Time ($O(n!)$)**: The operations increase factorially with the input data set. This is often seen in algorithms solving the traveling salesman problem using brute force, or generating all permutations of a list.