CS 13

Mathematical Foundations of Computing

Combinatorics

Outline

Baseball Tournaments

Imagine you're designing a tournament for n little-league baseball teams. There are several different ways that they could play each other:

- Each team plays every other team once. (Round Robin)
- Each team plays until they lose. (Single Elimination)
- Each team plays until they lose twice. (Double Elimination)

You have been tasked with figuring out which type of tournament is best for the children to play in. Since each game costs your boss money, he would like them to play a minimal number of games. Which type of tournament should you recommend?

DNA Sequencing

Imagine you're working in bioinformatics, and you've been asked to identify if a strand of DNA could have replicated from from a set of other strands of DNA. Recall that DNA strands are just strings of $\{A, C, T, G\}$.

Your first thought is to write a program to brute force all the possibilities. Is this a reasonable approach?

Poker

You're playing a game of poker and you have a pair of 10's and a pair of queens.

How likely are you to win?

It's about the process

As a Computer Scientist, you will often write algorithms. You'll also need to reason about:

It's about the process

As a Computer Scientist, you will often write algorithms. You'll also need to reason about:

1. **Enumeration.** How **many** solutions are there to a problem?

It's about the process

As a Computer Scientist, you will often write algorithms. You'll also need to reason about:

1. **Enumeration.** How **many** solutions are there to a problem?
   Can we solve Sudoku boards by solving all of them and looking them up in a database?

It's about the process

As a Computer Scientist, you will often write algorithms. You'll also need to reason about:

1. **Enumeration.** How **many** solutions are there to a problem?
   Can we solve Sudoku boards by solving all of them and looking them up in a database?
2. **Existence.** Is it even **possible** to find a solution?

As a Computer Scientist, you will often write algorithms. You'll also need to reason about:

1. **Enumeration.** How **many** solutions are there to a problem? Can we solve Sudoku boards by solving all of them and looking them up in a database?
2. **Existence.** Is it even **possible** to find a solution? Can we draw maps of countries so that no two adjacent ones have the same color with just four colors?

It's about the process

As a Computer Scientist, you will often write algorithms. You'll also need to reason about:

1. **Enumeration.** How **many** solutions are there to a problem? Can we solve Sudoku boards by solving all of them and looking them up in a database?
2. **Existence.** Is it even **possible** to find a solution? Can we draw maps of countries so that no two adjacent ones have the same color with just four colors?
3. **Construction.** Is it possible to transmit data over a **faulty** connection?

As a Computer Scientist, you will often write algorithms. You'll also need to reason about:

1. **Enumeration.** How **many** solutions are there to a problem?
   Can we solve Sudoku boards by solving all of them and looking them up in a database?
2. **Existence.** Is it even **possible** to find a solution?
   Can we draw maps of countries so that no two adjacent ones have the same color with just four colors?
3. **Construction.** Is it possible to transmit data over a **faulty** connection?
   How do computers read CDs that have some scratches on them?

As a Computer Scientist, you will often write algorithms. You'll also need to reason about:

1. **Enumeration.** How **many** solutions are there to a problem?
   Can we solve Sudoku boards by solving all of them and looking them up in a database?
2. **Existence.** Is it even **possible** to find a solution?
   Can we draw maps of countries so that no two adjacent ones have the same color with just four colors?
3. **Construction.** Is it possible to transmit data over a **faulty** connection?
   How do computers read CDs that have some scratches on them?
4. **Optimization.** What is the **best** solution to a problem? Why can't we do better?

It's about the process

As a Computer Scientist, you will often write algorithms. You'll also need to reason about:

1. **Enumeration.** How **many** solutions are there to a problem? Can we solve Sudoku boards by solving all of them and looking them up in a database?
2. **Existence.** Is it even **possible** to find a solution? Can we draw maps of countries so that no two adjacent ones have the same color with just four colors?
3. **Construction.** Is it possible to transmit data over a **faulty** connection? How do computers read CDs that have some scratches on them?
4. **Optimization.** What is the **best** solution to a problem? Why can't we do better? How does a GPS know the best route between any two locations?

It’s about the process

To solve each of these questions, you have to reason about **how many** of something there are. This process is “thinking combinatorially”, and we’re going to talk about it next!

Thinking combinatorially can sometimes make very difficult problems much easier.

Outline

How should you approach a combinatorial problem?

How should you approach a combinatorial problem?
Let's build up a "toolbox" of approaches we can take!

How should you approach a combinatorial problem?
Let's build up a "toolbox" of approaches we can take!

This may seem a little strange, but our three most powerful tools in counting are laws of **sets**!

Rule of Product

Definition (Rule of Product)

If we have sets $X_1, X_2, \ldots X_n$ then

$$|X_1 \times X_2 \times \cdots \times X_n| = |X_1| \times |X_2| \times \cdots \times |X_n|$$

Definition (Rule of Product)

If we have sets $X_1, X_2, \ldots X_n$ then

$$|X_1 \times X_2 \times \cdots \times X_n| = |X_1| \times |X_2| \times \cdots \times |X_n|$$

What does this have to do with counting?

Definition (Rule of Product)

If we have sets $X_1, X_2, \ldots X_n$ then

$$|X_1 \times X_2 \times \cdots \times X_n| = |X_1| \times |X_2| \times \cdots \times |X_n|$$

What does this have to do with counting?

Example

How many ways can I roll two six-sided dice?

Rule of Product

Definition (Rule of Product)

If we have sets $X_1, X_2, \ldots X_n$ then

$$|X_1 \times X_2 \times \cdots \times X_n| = |X_1| \times |X_2| \times \cdots \times |X_n|$$

What does this have to do with counting?

Example

How many ways can I roll two six-sided dice?

Proof.

Let D_6 be the set of outcomes for rolling a die.

Definition (Rule of Product)

If we have sets $X_1, X_2, \ldots X_n$ then

$$|X_1 \times X_2 \times \cdots \times X_n| = |X_1| \times |X_2| \times \cdots \times |X_n|$$

What does this have to do with counting?

Example

How many ways can I roll two six-sided dice?

Proof.

Let D_6 be the set of outcomes for rolling a die.
The outcomes of rolling two six-sided dice are members of $D_6 \times D_6$.

Definition (Rule of Product)

If we have sets $X_1, X_2, \dots X_n$ then

$$|X_1 \times X_2 \times \cdots \times X_n| = |X_1| \times |X_2| \times \cdots \times |X_n|$$

What does this have to do with counting?

Example

How many ways can I roll two six-sided dice?

Proof.

Let D_6 be the set of outcomes for rolling a die.
The outcomes of rolling two six-sided dice are members of $D_6 \times D_6$.
We know $|D_6| = |\{1,2,3,4,5,6\}| = 6$, and $|D_6 \times D_6| = |D_6| \times |D_6|$ by the Rule of Product.

Definition (Rule of Product)

If we have sets $X_1, X_2, \ldots X_n$ then

$$|X_1 \times X_2 \times \cdots \times X_n| = |X_1| \times |X_2| \times \cdots \times |X_n|$$

What does this have to do with counting?

Example

How many ways can I roll two six-sided dice?

Proof.

Let D_6 be the set of outcomes for rolling a die.
The outcomes of rolling two six-sided dice are members of $D_6 \times D_6$.
We know $|D_6| = |\{1,2,3,4,5,6\}| = 6$, and $|D_6 \times D_6| = |D_6| \times |D_6|$ by the Rule of Product.
So, the number of outcomes is $6 \times 6 = 36$. □

Rule of Product

Definition (Rule of Product)

If we have sets $X_1, X_2, \ldots X_n$ then

$$|X_1 \times X_2 \times \cdots \times X_n| = |X_1| \times |X_2| \times \cdots \times |X_n|$$

What does this have to do with counting?

Example

How many ways can I roll two six-sided dice?

Proof.

We know that there are six ways to roll a single die. To roll two dice, we follow this procedure:

- Roll one die.
- Roll one die.

Each step of the procedure has six possibilities; so, multiplying them together by the Rule of Product, we get $6 \times 6 = 36$ outcomes. □

Definition (Disjoint Sets)

$X_1, X_2, \ldots, X_n$ are pairwise disjoint sets iff

$$\forall (i \neq j).\, X_i \cap X_j = \varnothing$$

Definition (Disjoint Sets)

$X_1, X_2, \ldots, X_n$ are pairwise disjoint sets iff

$$\forall (i \neq j).\, X_i \cap X_j = \varnothing$$

Definition (Rule of Sum)

If $X_1, X_2, \ldots X_n$ are pairwise disjoint sets, then

$$|X_1 \cup X_2 \cup \cdots \cup X_n| = |X_1| + |X_2| + \cdots + |X_n|$$

Definition (Disjoint Sets)

$X_1, X_2, \ldots, X_n$ are pairwise disjoint sets iff

$$\forall (i \neq j).\, X_i \cap X_j = \varnothing$$

Definition (Rule of Sum)

If $X_1, X_2, \ldots X_n$ are pairwise disjoint sets, then

$$|X_1 \cup X_2 \cup \cdots \cup X_n| = |X_1| + |X_2| + \cdots + |X_n|$$

Example

How many ways can I roll two six-sided dice to get a sum of 4?

Definition (Rule of Sum)

If $X_1, X_2, \ldots X_n$ are pairwise disjoint sets, then

$$|X_1 \cup X_2 \cup \cdots \cup X_n| = |X_1| + |X_2| + \cdots + |X_n|$$

Example

How many ways can I roll two six-sided dice to get a sum of 4?

Proof.

Note that the first roll could be 1 through 6.

Definition (Rule of Sum)

If $X_1, X_2, \ldots X_n$ are pairwise disjoint sets, then

$$|X_1 \cup X_2 \cup \cdots \cup X_n| = |X_1| + |X_2| + \cdots + |X_n|$$

Example

How many ways can I roll two six-sided dice to get a sum of 4?

Proof.

Note that the first roll could be 1 through 6. We partition on these cases:

Rule of Sum

Definition (Rule of Sum)

If $X_1, X_2, \ldots X_n$ are pairwise disjoint sets, then

$$|X_1 \cup X_2 \cup \cdots \cup X_n| = |X_1| + |X_2| + \cdots + |X_n|$$

Example

How many ways can I roll two six-sided dice to get a sum of 4?

Proof.

Note that the first roll could be 1 through 6. We partition on these cases:

- If the first roll is a 1, then the second roll is 3.

Definition (Rule of Sum)

If $X_1, X_2, \ldots X_n$ are pairwise disjoint sets, then

$$|X_1 \cup X_2 \cup \cdots \cup X_n| = |X_1| + |X_2| + \cdots + |X_n|$$

Example

How many ways can I roll two six-sided dice to get a sum of 4?

Proof.

Note that the first roll could be 1 through 6. We partition on these cases:

- If the first roll is a 1, then the second roll is 3.
- If the first roll is a 2, then the second roll is 2.

Definition (Rule of Sum)

If $X_1, X_2, \ldots X_n$ are pairwise disjoint sets, then

$$|X_1 \cup X_2 \cup \cdots \cup X_n| = |X_1| + |X_2| + \cdots + |X_n|$$

Example

How many ways can I roll two six-sided dice to get a sum of 4?

Proof.

Note that the first roll could be 1 through 6. We partition on these cases:

- If the first roll is a 1, then the second roll is 3.
- If the first roll is a 2, then the second roll is 2.
- If the first roll is a 3, then the second roll is 1.

Definition (Rule of Sum)

If $X_1, X_2, \ldots X_n$ are pairwise disjoint sets, then

$$|X_1 \cup X_2 \cup \cdots \cup X_n| = |X_1| + |X_2| + \cdots + |X_n|$$

Example

How many ways can I roll two six-sided dice to get a sum of 4?

Proof.

Note that the first roll could be 1 through 6. We partition on these cases:

- If the first roll is a 1, then the second roll is 3.
- If the first roll is a 2, then the second roll is 2.
- If the first roll is a 3, then the second roll is 1.
- If the first roll is 4, 5, or 6, then we can never sum to 4.

Definition (Rule of Sum)

If $X_1, X_2, \ldots X_n$ are pairwise disjoint sets, then

$$|X_1 \cup X_2 \cup \cdots \cup X_n| = |X_1| + |X_2| + \cdots + |X_n|$$

Example

How many ways can I roll two six-sided dice to get a sum of 4?

Proof.

Note that the first roll could be 1 through 6. We partition on these cases:

- If the first roll is a 1, then the second roll is 3.
- If the first roll is a 2, then the second roll is 2.
- If the first roll is a 3, then the second roll is 1.
- If the first roll is 4, 5, or 6, then we can never sum to 4.

Note that these cases are mutually exclusive.

Rule of Sum

Definition (Rule of Sum)

If $X_1, X_2, \ldots X_n$ are pairwise disjoint sets, then

$$|X_1 \cup X_2 \cup \cdots \cup X_n| = |X_1| + |X_2| + \cdots + |X_n|$$

Example

How many ways can I roll two six-sided dice to get a sum of 4?

Proof.

Note that the first roll could be 1 through 6. We partition on these cases:

- If the first roll is a 1, then the second roll is 3.
- If the first roll is a 2, then the second roll is 2.
- If the first roll is a 3, then the second roll is 1.
- If the first roll is 4, 5, or 6, then we can never sum to 4.

Note that these cases are mutually exclusive. Furthermore, this covers all the possible cases for the first die.

Definition (Rule of Sum)

If $X_1, X_2, \ldots X_n$ are pairwise disjoint sets, then

$$|X_1 \cup X_2 \cup \cdots \cup X_n| = |X_1| + |X_2| + \cdots + |X_n|$$

Example

How many ways can I roll two six-sided dice to get a sum of 4?

Proof.

Note that the first roll could be 1 through 6. We partition on these cases:

- If the first roll is a 1, then the second roll is 3.
- If the first roll is a 2, then the second roll is 2.
- If the first roll is a 3, then the second roll is 1.
- If the first roll is 4, 5, or 6, then we can never sum to 4.

Note that these cases are mutually exclusive. Furthermore, this covers all the possible cases for the first die. Putting these together, we see that $1+1+1+0+0+0=3$ is our answer by the Rule of Sum. □

Sometimes, instead of counting the things we want, we count the things we **don't** want and remove them.

Sometimes, instead of counting the things we want, we count the things we **don't** want and remove them.

Definition (Counting by Complement)

If $\mathcal{U}$ is the universal set, then

$$A = \mathcal{U} \setminus \overline{A}$$

Sometimes, instead of counting the things we want, we count the things we **don't** want and remove them.

Definition (Counting by Complement)

If $\mathcal{U}$ is the universal set, then

$$A = \mathcal{U} \setminus \overline{A}$$

Example

How many binary strings of length n are there that have at least one 1.

Sometimes, instead of counting the things we want, we count the things we **don't** want and remove them.

Definition (Counting by Complement)

If $\mathcal{U}$ is the universal set, then

$$A = \mathcal{U} \setminus \overline{A}$$

Example

How many binary strings of length n are there that have at least one 1.

Proof.

First, we show that there are 2^n binary strings.

Sometimes, instead of counting the things we want, we count the things we **don't** want and remove them.

Definition (Counting by Complement)

If $\mathcal{U}$ is the universal set, then

$$A = \mathcal{U} \setminus \overline{A}$$

Example

How many binary strings of length n are there that have at least one 1.

Proof.

First, we show that there are 2^n binary strings. To generate a binary string, we use an n-step process:

Sometimes, instead of counting the things we want, we count the things we **don't** want and remove them.

Definition (Counting by Complement)

If $\mathcal{U}$ is the universal set, then

$$A = \mathcal{U} \setminus \overline{A}$$

Example

How many binary strings of length n are there that have at least one 1.

Proof.

First, we show that there are 2^n binary strings. To generate a binary string, we use an n-step process:

- Choose the 1st bit.

Counting by Complement

Sometimes, instead of counting the things we want, we count the things we **don't** want and remove them.

Definition (Counting by Complement)

If $\mathcal{U}$ is the universal set, then

$$A = \mathcal{U} \setminus \overline{A}$$

Example

How many binary strings of length n are there that have at least one 1.

Proof.

First, we show that there are 2^n binary strings. To generate a binary string, we use an n-step process:

- Choose the 1st bit.
- Choose the 2nd bit.

Sometimes, instead of counting the things we want, we count the things we **don't** want and remove them.

Definition (Counting by Complement)

If $\mathcal{U}$ is the universal set, then

$$A = \mathcal{U} \setminus \overline{A}$$

Example

How many binary strings of length n are there that have at least one 1.

Proof.

First, we show that there are 2^n binary strings. To generate a binary string, we use an n-step process:

- Choose the 1st bit.
- Choose the 2nd bit.
- ...

Sometimes, instead of counting the things we want, we count the things we **don't** want and remove them.

Definition (Counting by Complement)

If $\mathcal{U}$ is the universal set, then

$$A = \mathcal{U} \setminus \overline{A}$$

Example

How many binary strings of length n are there that have at least one 1.

Proof.

First, we show that there are 2^n binary strings. To generate a binary string, we use an n-step process:

- Choose the 1st bit.
- Choose the 2nd bit.
- ...
- Profit!

□

Example

How many binary strings of length n are there that have at least one 1.

Proof.

First, we show that there are 2^n binary strings. To generate a binary string, we use an n-step process:

- Choose the 1st bit.
- Choose the 2nd bit.
- ...
- Choose the nth bit.

Example

How many binary strings of length n are there that have at least one 1.

Proof.

First, we show that there are 2^n binary strings. To generate a binary string, we use an n-step process:

- Choose the 1st bit.
- Choose the 2nd bit.
- ...
- Choose the nth bit.

Since each step of this procedure has 2 options, the total number of binary strings of length n is $\underbrace{2 \times 2 \times \cdots \times 2}_{n \text{ times}} = 2^n$ by the Rule of Product.

Example

How many binary strings of length n are there that have at least one 1.

Proof.

First, we show that there are 2^n binary strings. To generate a binary string, we use an n-step process:

- Choose the 1st bit.
- Choose the 2nd bit.
- ...
- Choose the nth bit.

Since each step of this procedure has 2 options, the total number of binary strings of length n is $\underbrace{2 \times 2 \times \cdots \times 2}_{n \text{ times}} = 2^n$ by the Rule of Product.

Now, we count how many binary strings of length n have no 1's.

Example

How many binary strings of length n are there that have at least one 1.

Proof.

First, we show that there are 2^n binary strings. To generate a binary string, we use an n-step process:

- Choose the 1st bit.
- Choose the 2nd bit.
- ...
- Choose the nth bit.

Since each step of this procedure has 2 options, the total number of binary strings of length n is $\underbrace{2 \times 2 \times \cdots \times 2}_{n \text{ times}} = 2^n$ by the Rule of Product.

Now, we count how many binary strings of length n have no 1's. We use the same procedure as before, except, now, we only have 1 choice at each step.

Example

How many binary strings of length n are there that have at least one 1.

Proof.

First, we show that there are 2^n binary strings. To generate a binary string, we use an n-step process:

- Choose the 1st bit.
- Choose the 2nd bit.
- ...
- Choose the nth bit.

Since each step of this procedure has 2 options, the total number of binary strings of length n is $\underbrace{2 \times 2 \times \cdots \times 2}_{n \text{ times}} = 2^n$ by the Rule of Product.

Now, we count how many binary strings of length n have no 1's. We use the same procedure as before, except, now, we only have 1 choice at each step. It follows that there is 1 bad binary string.

Example

How many binary strings of length n are there that have at least one 1.

Proof.

First, we show that there are 2^n binary strings. To generate a binary string, we use an n-step process:

- Choose the 1st bit.
- Choose the 2nd bit.
- ...
- Choose the nth bit.

Since each step of this procedure has 2 options, the total number of binary strings of length n is $\underbrace{2 \times 2 \times \cdots \times 2}_{n \text{ times}} = 2^n$ by the Rule of Product.

Now, we count how many binary strings of length n have no 1's. We use the same procedure as before, except, now, we only have 1 choice at each step. It follows that there is 1 bad binary string.
So, Counting by Complement, we see that there are $2^n - 1$ binary strings with at least one 1. □

Outline

Now that we know what we're trying to do, let's build up the primitives of our language.

Think of these like **if statements** and **for loops** in programming.

We can use these to build up larger, more complicated counting arguments!

Primitive: Arranging $\{x_1, x_2, \ldots, x_n\}$

We would like to arrange n distinct things, $\{x_1, x_2, \ldots, x_n\}$, in a row:

How many places could we put x_1?

Primitive: Arranging $\{x_1, x_2, \ldots, x_n\}$

We would like to arrange n distinct things, $\{x_1, x_2, \ldots, x_n\}$, in a row:

$$\square \quad \square \quad \square \quad \cdots \quad \square \quad \square \quad \square$$

How many places could we put x_1? n

Primitive: Arranging $\{x_1, x_2, \ldots, x_n\}$

We would like to arrange n distinct things, $\{x_1, x_2, \ldots, x_n\}$, in a row:

How many places could we put x_1? n

Primitive: Arranging $\{x_1, x_2, \ldots, x_n\}$

We would like to arrange n distinct things, $\{x_1, x_2, \ldots, x_n\}$, in a row:

How many places could we put x_1? n

How many places could we put x_2?

Primitive: Arranging $\{x_1, x_2, \ldots, x_n\}$

We would like to arrange n distinct things, $\{x_1, x_2, \ldots, x_n\}$, in a row:

How many places could we put x_1? n

How many places could we put x_2? $n - 1$

Primitive: Arranging $\{x_1, x_2, \ldots, x_n\}$

We would like to arrange n distinct things, $\{x_1, x_2, \ldots, x_n\}$, in a row:

How many places could we put x_1? n
How many places could we put x_2? $n-1$

$\ldots$

How many places could we put x_k?

Primitive: Arranging $\{x_1, x_2, \dots, x_n\}$

We would like to arrange n distinct things, $\{x_1, x_2, \dots, x_n\}$, in a row:

$$\boxed{} \quad \boxed{x_2} \quad \boxed{x_3} \quad \cdots \quad \boxed{x_1} \quad \boxed{} \quad \boxed{}$$

How many places could we put x_1? n

How many places could we put x_2? $n-1$

$\dots$

How many places could we put x_k? $n-(k-1)$

Primitive: Arranging $\{x_1, x_2, \ldots, x_n\}$

We would like to arrange n distinct things, $\{x_1, x_2, \ldots, x_n\}$, in a row:

$$\boxed{x_{n-1}} \quad \boxed{x_2} \quad \boxed{x_3} \quad \cdots \quad \boxed{x_1} \quad \boxed{} \quad \boxed{x_k}$$

How many places could we put x_1? n
How many places could we put x_2? $n-1$
…
How many places could we put x_k? $n-(k-1)$
…
How many places could we put x_n?

Primitive: Arranging $\{x_1, x_2, \ldots, x_n\}$

We would like to arrange n distinct things, $\{x_1, x_2, \ldots, x_n\}$, in a row:

$$\boxed{x_{n-1}} \quad \boxed{x_2} \quad \boxed{x_3} \quad \cdots \quad \boxed{x_1} \quad \boxed{} \quad \boxed{x_k}$$

How many places could we put x_1? n

How many places could we put x_2? $n-1$

…

How many places could we put x_k? $n-(k-1)$

…

How many places could we put x_n? 1

Primitive: Arranging $\{x_1, x_2, \ldots, x_n\}$

We would like to arrange n distinct things, $\{x_1, x_2, \ldots, x_n\}$, in a row:

$$\boxed{x_{n-1}} \quad \boxed{x_2} \quad \boxed{x_3} \quad \cdots \quad \boxed{x_1} \quad \boxed{} \quad \boxed{x_k}$$

How many places could we put x_1? n

How many places could we put x_2? $n-1$

$\ldots$

How many places could we put x_k? $n-(k-1)$

$\ldots$

How many places could we put x_n? 1

Proof.

We can arrange $\{x_1, x_2, \ldots, x_n\}$ in an n-step process, where, on step k, we place x_k. There are $n-(k-1)$ ways to do step k, since there are that many spots remaining. It follows that the number of ways to arrange our set is $n(n-1)\cdots 2(1) = n!$ by Rule of Product. □

Primitive: Choosing a subset of k elements of $\{x_1, x_2, \ldots, x_n\}$

We've already seen this!

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

Here's a tempting but **incorrect** argument for how to calculate $\binom{n}{k}$:

Counting Combinations

To generate a subset of k elements, we take the following two steps:

Primitive: Choosing a subset of k elements of $\{x_1, x_2, \ldots, x_n\}$

We've already seen this!

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

Here's a tempting but **incorrect** argument for how to calculate $\binom{n}{k}$:

Counting Combinations

To generate a subset of k elements, we take the following two steps:

(1) Arrange all n elements of the set.

Primitive: Choosing a subset of k elements of $\{x_1, x_2, \ldots, x_n\}$

We've already seen this!

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

Here's a tempting but **incorrect** argument for how to calculate $\binom{n}{k}$:

Counting Combinations

To generate a subset of k elements, we take the following two steps:

(1) Arrange all n elements of the set.

(2) Get rid of the last $n-k$ of them.

Primitive: Choosing a subset of k elements of $\{x_1, x_2, \ldots, x_n\}$

We've already seen this!

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

Here's a tempting but **incorrect** argument for how to calculate $\binom{n}{k}$:

Counting Combinations

To generate a subset of k elements, we take the following two steps:

(1) Arrange all n elements of the set.

(2) Get rid of the last $n-k$ of them.

We know that there are $n!$ ways to do the first step, and only 1 way to do the second step. So, by the Product Rule, we see that $\binom{n}{k} = n!$

Our Procedure

(1) Arrange all n elements of the set.

(2) Get rid of the last $n-k$ of them.

Suppose Adam wants to choose two of their favorite shapes. For reference, Adam's favorite shapes are:

$$\{\triangle, \square, \clubsuit, \diamond\}$$

Our Procedure

(1) Arrange all n elements of the set.

(2) Get rid of the last $n-k$ of them.

Suppose Adam wants to choose two of their favorite shapes. For reference, Adam's favorite shapes are:

$$\{\triangle, \square, \clubsuit, \diamond\}$$

Our argument first generates an ordering of these shapes:

Our Procedure

(1) Arrange all n elements of the set.

(2) Get rid of the last $n-k$ of them.

Suppose Adam wants to choose two of their favorite shapes. For reference, Adam's favorite shapes are:

$$\{\triangle, \Box, \clubsuit, \diamond\}$$

Our argument first generates an ordering of these shapes:

$\triangle$	$\Box$	$\clubsuit$	$\diamond$
$\triangle$	$\Box$	$\diamond$	$\clubsuit$
$\triangle$	$\clubsuit$	$\Box$	$\diamond$
$\triangle$	$\clubsuit$	$\diamond$	$\Box$
$\triangle$	$\diamond$	$\Box$	$\clubsuit$
$\triangle$	$\diamond$	$\clubsuit$	$\Box$
$\Box$	$\triangle$	$\clubsuit$	$\diamond$
$\Box$	$\triangle$	$\diamond$	$\clubsuit$

$\Box$	$\diamond$	$\triangle$	$\clubsuit$
$\Box$	$\diamond$	$\clubsuit$	$\triangle$
$\Box$	$\clubsuit$	$\diamond$	$\triangle$
$\Box$	$\clubsuit$	$\triangle$	$\diamond$
$\clubsuit$	$\Box$	$\triangle$	$\diamond$
$\clubsuit$	$\Box$	$\diamond$	$\triangle$
$\clubsuit$	$\diamond$	$\Box$	$\triangle$
$\clubsuit$	$\diamond$	$\triangle$	$\Box$

$\clubsuit$	$\triangle$	$\Box$	$\diamond$
$\clubsuit$	$\triangle$	$\diamond$	$\Box$
$\diamond$	$\Box$	$\triangle$	$\clubsuit$
$\diamond$	$\Box$	$\clubsuit$	$\triangle$
$\diamond$	$\clubsuit$	$\Box$	$\triangle$
$\diamond$	$\clubsuit$	$\triangle$	$\Box$
$\diamond$	$\triangle$	$\Box$	$\clubsuit$
$\diamond$	$\triangle$	$\clubsuit$	$\Box$

Combinations

Then, it throws away the last $n-k$:

△	□		
△	□		
△	♣		
△	♣		
△	◇		
△	◇		
□	△		
□	△		

□	◇		
□	◇		
□	♣		
□	♣		
♣	□		
♣	□		
♣	◇		
♣	◇		

♣	△		
♣	△		
◇	□		
◇	□		
◇	♣		
◇	♣		
♣	△		
♣	△		

Oops! We've counted each set of favorite shapes multiple times.

Combinations

Then, it throws away the last $n-k$:

△	□		
△	□		
△	♣		
△	♣		
△	◇		
△	◇		
□	△		
□	△		

□	◇		
□	◇		
□	♣		
□	♣		
♣	□		
♣	□		
♣	◇		
♣	◇		

♣	△		
♣	△		
◇	□		
◇	□		
◇	♣		
◇	♣		
♣	△		
♣	△		

Oops! We've counted each set of favorite shapes multiple times.
Can we be more specific?

Combinations

Then, it throws away the last $n-k$:

△	□		
△	□		
△	♣		
△	♣		
△	◇		
△	◇		
□	△		
□	△		

□	◇		
□	◇		
□	♣		
□	♣		
♣	□		
♣	□		
♣	◇		
♣	◇		

♣	△		
♣	△		
◇	□		
◇	□		
◇	♣		
◇	♣		
♣	△		
♣	△		

Oops! We've counted each set of favorite shapes multiple times.
Can we be more specific?

- Our argument ordered the first k shapes when we didn't actually want them ordered.

Combinations

Then, it throws away the last $n-k$:

△	□		
△	□		
△	♣		
△	♣		
△	◇		
△	◇		
□	△		
□	△		

□	◇		
□	◇		
□	♣		
□	♣		
♣	□		
♣	□		
♣	◇		
♣	◇		

♣	△		
♣	△		
◇	□		
◇	□		
◇	♣		
◇	♣		
♣	△		
♣	△		

Oops! We've counted each set of favorite shapes multiple times.
Can we be more specific?

- Our argument ordered the first k shapes when we didn't actually want them ordered. (So, they showed up $k!$ times.)

Then, it throws away the last $n-k$:

△	□		
△	□		
△	♣		
△	♣		
△	◇		
△	◇		
□	△		
□	△		

□	◇		
□	◇		
□	♣		
□	♣		
♣	□		
♣	□		
♣	◇		
♣	◇		

♣	△		
♣	△		
◇	□		
◇	□		
◇	♣		
◇	♣		
♣	△		
♣	△		

Oops! We've counted each set of favorite shapes multiple times.
Can we be more specific?

- Our argument ordered the first k shapes when we didn't actually want them ordered. (So, they showed up $k!$ times.)
- Our argument also ordered the remaining $n-k$ shapes when we didn't want them ordered.

Then, it throws away the last $n-k$:

△	□			□	◇			♣	△		
△	□			□	◇			♣	△		
△	♣			□	♣			◇	□		
△	♣			□	♣			◇	□		
△	◇			♣	□			◇	♣		
△	◇			♣	□			◇	♣		
□	△			♣	◇			♣	△		
□	△			♣	◇			♣	△		

Oops! We've counted each set of favorite shapes multiple times.
Can we be more specific?

- Our argument ordered the first k shapes when we didn't actually want them ordered. (So, they showed up $k!$ times.)
- Our argument also ordered the remaining $n-k$ shapes when we didn't want them ordered. (So, they showed up $(n-k)!$ times.)

Let S_k be the set of size k subsets of $\{x_1, x_2, \ldots, x_n\}$.
Here's another way of looking at the argument we just made. We claim that:

$$n! = |S_k| k! (n-k)!$$

Let S_k be the set of size k subsets of $\{x_1, x_2, \ldots, x_n\}$.
Here's another way of looking at the argument we just made. We claim that:

$$n! = |S_k| k! (n-k)!$$

The right side is a three step procedure:

- Choose the first k elements of the sequence.

Let S_k be the set of size k subsets of $\{x_1, x_2, \ldots, x_n\}$.
Here's another way of looking at the argument we just made. We claim that:

$$n! = |S_k| k! (n-k)!$$

The right side is a three step procedure:

- Choose the first k elements of the sequence.
- Arrange the first k elements of the sequence.

Let S_k be the set of size k subsets of $\{x_1, x_2, \ldots, x_n\}$.
Here's another way of looking at the argument we just made. We claim that:

$$n! = |S_k| k! (n-k)!$$

The right side is a three step procedure:

- Choose the first k elements of the sequence.
- Arrange the first k elements of the sequence.
- By choosing the first k elements of the sequence, we left behind $n-k$ to be the rest. Arrange those.

Let S_k be the set of size k subsets of $\{x_1, x_2, \ldots, x_n\}$.
Here's another way of looking at the argument we just made. We claim that:

$$n! = |S_k| k! (n-k)!$$

The right side is a three step procedure:

- Choose the first k elements of the sequence.
- Arrange the first k elements of the sequence.
- By choosing the first k elements of the sequence, we left behind $n-k$ to be the rest. Arrange those.

The result of our procedure is that we've arranged the elements of $\{x_1, x_2, \ldots, x_n\}$, and we know there are $n!$ ways to do that.

Let S_k be the set of size k subsets of $\{x_1, x_2, \ldots, x_n\}$.
Here's another way of looking at the argument we just made. We claim that:

$$n! = |S_k| k! (n-k)!$$

The right side is a three step procedure:

- Choose the first k elements of the sequence.
- Arrange the first k elements of the sequence.
- By choosing the first k elements of the sequence, we left behind $n-k$ to be the rest. Arrange those.

The result of our procedure is that we've arranged the elements of $\{x_1, x_2, \ldots, x_n\}$, and we know there are $n!$ ways to do that.
It follows that the equality holds and $|S_k| = \frac{n!}{k!(n-k)!}$.

Let S_k be the set of size k subsets of $\{x_1, x_2, \ldots, x_n\}$.
Here's another way of looking at the argument we just made. We claim that:

$$n! = |S_k| k! (n-k)!$$

The right side is a three step procedure:

- Choose the first k elements of the sequence.
- Arrange the first k elements of the sequence.
- By choosing the first k elements of the sequence, we left behind $n-k$ to be the rest. Arrange those.

The result of our procedure is that we've arranged the elements of $\{x_1, x_2, \ldots, x_n\}$, and we know there are $n!$ ways to do that.
It follows that the equality holds and $|S_k| = \frac{n!}{k!(n-k)!}$.

By convention, we call $|S_k| = \binom{n}{k}$, and pronounce it "n choose k".

Outline

DNA is made up of $\{A, C, T, G\}$. How many strands of DNA of length n are there with exactly 4 C's?

DNA is made up of $\{A, C, T, G\}$. How many strands of DNA of length n are there with exactly 4 C's?

Proof.

We count this via the following process:

DNA is made up of $\{A,C,T,G\}$. How many strands of DNA of length n are there with exactly 4 C's?

Proof.

We count this via the following process:

- Choose which 4 of the n spots to put C's in.

DNA is made up of $\{A, C, T, G\}$. How many strands of DNA of length n are there with exactly 4 C's?

Proof.

We count this via the following process:

- Choose which 4 of the n spots to put C's in.
- For each of the remaining spots, choose between A, T, and G.

DNA is made up of $\{A, C, T, G\}$. How many strands of DNA of length n are there with exactly 4 C's?

Proof.

We count this via the following process:

- Choose which 4 of the n spots to put C's in.
- For each of the remaining spots, choose between A, T, and G.

The number of ways to do the first step is $\binom{n}{4}$, and the number of ways to do the other $n-4$ steps is 3.

DNA is made up of $\{A, C, T, G\}$. How many strands of DNA of length n are there with exactly 4 C's?

Proof.

We count this via the following process:

- Choose which 4 of the n spots to put C's in.
- For each of the remaining spots, choose between A, T, and G.

The number of ways to do the first step is $\binom{n}{4}$, and the number of ways to do the other $n-4$ steps is 3. Using the Rule of Product, we get that there are $\binom{n}{4}3^{n-4}$ possible strands of DNA with 4 C's. □

How many five card hands are there with three or four Aces?

Counting Cards

How many five card hands are there with three or four Aces?

Proof.

We partition on if there are three Aces or four.

How many five card hands are there with three or four Aces?

Proof.

We partition on if there are three Aces or four.

- If there are three Aces, choose which Aces there are, and then choose two non-Aces.

How many five card hands are there with three or four Aces?

Proof.

We partition on if there are three Aces or four.

- If there are three Aces, choose which Aces there are, and then choose two non-Aces. By Rule of Product, this works out to $\binom{4}{3}\binom{48}{2}$.

Counting Cards

How many five card hands are there with three or four Aces?

Proof.

We partition on if there are three Aces or four.

- If there are three Aces, choose which Aces there are, and then choose two non-Aces. By Rule of Product, this works out to $\binom{4}{3}\binom{48}{2}$.
- If there are four Aces, choose all four Aces, and then choose the remaining card.

How many five card hands are there with three or four Aces?

Proof.

We partition on if there are three Aces or four.

- If there are three Aces, choose which Aces there are, and then choose two non-Aces. By Rule of Product, this works out to $\binom{4}{3}\binom{48}{2}$.
- If there are four Aces, choose all four Aces, and then choose the remaining card. By Rule of Product, this works out to $\binom{4}{4}\binom{48}{1}$.

How many five card hands are there with three or four Aces?

Proof.

We partition on if there are three Aces or four.

- If there are three Aces, choose which Aces there are, and then choose two non-Aces. By Rule of Product, this works out to $\binom{4}{3}\binom{48}{2}$.
- If there are four Aces, choose all four Aces, and then choose the remaining card. By Rule of Product, this works out to $\binom{4}{4}\binom{48}{1}$.

Note that every hand with 3 or 4 Aces must either have 3 or 4 Aces, and that no hand can have both 3 and 4 Aces; so, these cases form a partition.

How many five card hands are there with three or four Aces?

Proof.

We partition on if there are three Aces or four.

- If there are three Aces, choose which Aces there are, and then choose two non-Aces. By Rule of Product, this works out to $\binom{4}{3}\binom{48}{2}$.
- If there are four Aces, choose all four Aces, and then choose the remaining card. By Rule of Product, this works out to $\binom{4}{4}\binom{48}{1}$.

Note that every hand with 3 or 4 Aces must either have 3 or 4 Aces, and that no hand can have both 3 and 4 Aces; so, these cases form a partition.It follows, by Rule of Sum, that the number of five card hands with three or four Aces is $\binom{4}{3}\binom{48}{2} + \binom{4}{4}\binom{48}{1}$. □

How many five card hands are there with three or four Aces?

How many five card hands are there with three or four Aces?

"Proof."

We count the hands with the following process:

- Choose three of the four Aces.
- Out of the remaining 49 cards, choose 2 of them.

By the Rule of Product, the number of five card hands with three or four Aces is $\binom{4}{3}\binom{49}{2}$.

This argument gives us the number 4704.

How many five card hands are there with three or four Aces?

"Proof."

We count the hands with the following process:

- Choose three of the four Aces.
- Out of the remaining 49 cards, choose 2 of them.

By the Rule of Product, the number of five card hands with three or four Aces is $\binom{4}{3}\binom{49}{2}$.

This argument gives us the number 4704. Our previous (correct) argument gave us the number 4560.

How many five card hands are there with three or four Aces?

"Proof."

We count the hands with the following process:

- Choose three of the four Aces.
- Out of the remaining 49 cards, choose 2 of them.

By the Rule of Product, the number of five card hands with three or four Aces is $\binom{4}{3}\binom{49}{2}$.

This argument gives us the number 4704. Our previous (correct) argument gave us the number 4560. This means we must be **overcounting** (getting the same output more than once).

How many five card hands are there with three or four Aces?

"Proof."

We count the hands with the following process:

- Choose three of the four Aces.
- Out of the remaining 49 cards, choose 2 of them.

By the Rule of Product, the number of five card hands with three or four Aces is $\binom{4}{3}\binom{49}{2}$.

This argument gives us the number 4704. Our previous (correct) argument gave us the number 4560. This means we must be **overcounting** (getting the same output more than once).

Consider $\{A\spadesuit, A\heartsuit, A\clubsuit, A\diamondsuit, 4\clubsuit\}$

How many five card hands are there with three or four Aces?

"Proof."

We count the hands with the following process:

- Choose three of the four Aces.
- Out of the remaining 49 cards, choose 2 of them.

By the Rule of Product, the number of five card hands with three or four Aces is $\binom{4}{3}\binom{49}{2}$.

This argument gives us the number 4704. Our previous (correct) argument gave us the number 4560. This means we must be **overcounting** (getting the same output more than once).

$$\text{Consider } \{A\spadesuit, A\heartsuit, A\clubsuit, A\diamondsuit, 4\clubsuit\}$$

We could have gotten this set by. . .

How many five card hands are there with three or four Aces?

"Proof."

We count the hands with the following process:

- Choose three of the four Aces.
- Out of the remaining 49 cards, choose 2 of them.

By the Rule of Product, the number of five card hands with three or four Aces is $\binom{4}{3}\binom{49}{2}$.

This argument gives us the number 4704. Our previous (correct) argument gave us the number 4560. This means we must be **overcounting** (getting the same output more than once).

$$\text{Consider } \{A\spadesuit, A\heartsuit, A\clubsuit, A\diamondsuit, 4\clubsuit\}$$

We could have gotten this set by...

- Choosing $A\spadesuit, A\heartsuit, A\clubsuit$, and then choosing $A\diamondsuit, 4\clubsuit$.

How many five card hands are there with three or four Aces?

"Proof."

We count the hands with the following process:

- Choose three of the four Aces.
- Out of the remaining 49 cards, choose 2 of them.

By the Rule of Product, the number of five card hands with three or four Aces is $\binom{4}{3}\binom{49}{2}$.

This argument gives us the number 4704. Our previous (correct) argument gave us the number 4560. This means we must be **overcounting** (getting the same output more than once).

$$\text{Consider } \{A\spadesuit, A\heartsuit, A\clubsuit, A\diamondsuit, 4\clubsuit\}$$

We could have gotten this set by...

- Choosing $A\spadesuit, A\heartsuit, A\clubsuit$, and then choosing $A\diamondsuit, 4\clubsuit$.
- Choosing $A\diamondsuit, A\heartsuit, A\clubsuit$, and then choosing $A\spadesuit, 4\clubsuit$.

How many five card hands are there with three or four Aces?

"Proof."

We count the hands with the following process:

- Choose three of the four Aces.
- Out of the remaining 49 cards, choose 2 of them.

By the Rule of Product, the number of five card hands with three or four Aces is $\binom{4}{3}\binom{49}{2}$.

This argument gives us the number 4704. Our previous (correct) argument gave us the number 4560. This means we must be **overcounting** (getting the same output more than once).

$$\text{Consider } \{A\spadesuit, A\heartsuit, A\clubsuit, A\diamondsuit, 4\clubsuit\}$$

We could have gotten this set by...

- Choosing $A\spadesuit, A\heartsuit, A\clubsuit$, and then choosing $A\diamondsuit, 4\clubsuit$.
- Choosing $A\diamondsuit, A\heartsuit, A\clubsuit$, and then choosing $A\spadesuit, 4\clubsuit$.

If a counting argument is correct, we must be able to take an output and trace it to a particular choice pattern.