Some Basic Principles of Combinatorics

Daniel H. Luecking

August 21, 2023

The science of counting

Don't we already know how to count?

The science of counting

Don't we already know how to count?

Yes, we can all point to the objects we are counting, saying "1, 2, 3,. . . ". But imagine counting 11,881,376 things this way: it could take most of a year.

The science of counting

Don't we already know how to count?

Yes, we can all point to the objects we are counting, saying "1, 2, 3,. . . ". But imagine counting 11,881,376 things this way: it could take most of a year. An example of this many things we might want to count is the collection of all possible 5-letter strings.

The science of counting

Don't we already know how to count?

Yes, we can all point to the objects we are counting, saying "1, 2, 3,. . . ". But imagine counting 11,881,376 things this way: it could take most of a year. An example of this many things we might want to count is the collection of all possible 5-letter strings. But I got that number without counting anything.

The science of counting

Don't we already know how to count?

Yes, we can all point to the objects we are counting, saying "1, 2, 3,. . . ". But imagine counting 11,881,376 things this way: it could take most of a year. An example of this many things we might want to count is the collection of all possible 5-letter strings. But I got that number without counting anything.

To actually count them one by one, we would need to have a list of them. So we might ask a computer to generated that list. . . .

The science of counting

Don't we already know how to count?

Yes, we can all point to the objects we are counting, saying "1, 2, 3,. . . ". But imagine counting 11,881,376 things this way: it could take most of a year. An example of this many things we might want to count is the collection of all possible 5-letter strings. But I got that number without counting anything.

To actually count them one by one, we would need to have a list of them. So we might ask a computer to generated that list. . . . But we might as well ask it to count them for us. This would take a fraction of a second (plus the time it took us to write the program).

The science of counting

Don't we already know how to count?

Yes, we can all point to the objects we are counting, saying "1, 2, 3,. . . ". But imagine counting 11,881,376 things this way: it could take most of a year. An example of this many things we might want to count is the collection of all possible 5-letter strings. But I got that number without counting anything.

To actually count them one by one, we would need to have a list of them. So we might ask a computer to generated that list. . . . But we might as well ask it to count them for us. This would take a fraction of a second (plus the time it took us to write the program). But the same task for 12-letter strings would take a computer at least a year. . .

The science of counting

Don't we already know how to count?

Yes, we can all point to the objects we are counting, saying "1, 2, 3,. . . ". But imagine counting 11,881,376 things this way: it could take most of a year. An example of this many things we might want to count is the collection of all possible 5-letter strings. But I got that number without counting anything.

To actually count them one by one, we would need to have a list of them. So we might ask a computer to generated that list. . . . But we might as well ask it to count them for us. This would take a fraction of a second (plus the time it took us to write the program). But the same task for 12-letter strings would take a computer at least a year. . . but there are ways to compute the number of them (95,428,956,661,682,176) in a fraction of a second.

The science of counting

Don't we already know how to count?

Yes, we can all point to the objects we are counting, saying "1, 2, 3,. . . ". But imagine counting 11,881,376 things this way: it could take most of a year. An example of this many things we might want to count is the collection of all possible 5-letter strings. But I got that number without counting anything.

To actually count them one by one, we would need to have a list of them. So we might ask a computer to generated that list. . . . But we might as well ask it to count them for us. This would take a fraction of a second (plus the time it took us to write the program). But the same task for 12-letter strings would take a computer at least a year. . . but there are ways to compute the number of them (95,428,956,661,682,176) in a fraction of a second.

Combinatorics gives us tools to count these large amounts in a short time.

What combinatorics helps us count

What we count are sets. You will be expected to know the rudiments of set theory (some of which is reviewed below).

What combinatorics helps us count

What we count are sets. You will be expected to know the rudiments of set theory (some of which is reviewed below).

We usually use a capital letter like A or S to represent a set. We use "$x \in A$" to express the statement that x is an element in A.

What combinatorics helps us count

What we count are sets. You will be expected to know the rudiments of set theory (some of which is reviewed below).

We usually use a capital letter like A or S to represent a set. We use "$x \in A$" to express the statement that x is an element in A. We use "$B \subseteq A$" to express the statement that B is a subset of A. This means that every element of B is also in A.

What combinatorics helps us count

What we count are sets. You will be expected to know the rudiments of set theory (some of which is reviewed below).

We usually use a capital letter like A or S to represent a set. We use "$x \in A$" to express the statement that x is an element in A. We use "$B \subseteq A$" to express the statement that B is a subset of A. This means that every element of B is also in A.

We use $|A|$ to represent the number of elements in A.

What combinatorics helps us count

What we count are sets. You will be expected to know the rudiments of set theory (some of which is reviewed below).

We usually use a capital letter like A or S to represent a set. We use "$x \in A$" to express the statement that x is an element in A. We use "$B \subseteq A$" to express the statement that B is a subset of A. This means that every element of B is also in A.

We use $|A|$ to represent the number of elements in A. We use curly braces around a list of objects to represent the set with those objects as elements: $B = \{a, b, c\}$ is a set with 3 elements so $|B| = 3$.

What combinatorics helps us count

What we count are sets. You will be expected to know the rudiments of set theory (some of which is reviewed below).

We usually use a capital letter like A or S to represent a set. We use "$x \in A$" to express the statement that x is an element in A. We use "$B \subseteq A$" to express the statement that B is a subset of A. This means that every element of B is also in A.

We use $|A|$ to represent the number of elements in A. We use curly braces around a list of objects to represent the set with those objects as elements: $B = \{a, b, c\}$ is a set with 3 elements so $|B| = 3$.

Sets can themselves be elements in another set. For example $S = \{\{a, b\}, \{a, c\}, \{b, c\}\}$ contains all the two-element subsets of B.

What combinatorics helps us count

What we count are sets. You will be expected to know the rudiments of set theory (some of which is reviewed below).

We usually use a capital letter like A or S to represent a set. We use "$x \in A$" to express the statement that x is an element in A. We use "$B \subseteq A$" to express the statement that B is a subset of A. This means that every element of B is also in A.

We use $|A|$ to represent the number of elements in A. We use curly braces around a list of objects to represent the set with those objects as elements: $B = \{a, b, c\}$ is a set with 3 elements so $|B| = 3$.

Sets can themselves be elements in another set. For example $S = \{\{a, b\}, \{a, c\}, \{b, c\}\}$ contains all the two-element subsets of B. A set with k elements is called a k-set, so S is a 3-set and the elements of S are 2-sets.

Sets with structure

If a set is given by a short list, we can just count its elements. If a large set has some structure, we can often make use of that structure to count it.

Sets with structure

If a set is given by a short list, we can just count its elements. If a large set has some structure, we can often make use of that structure to count it.

For example, if a large set can be generated by a short algorithm, we might be able to use that algorithm to compute the number of elements. We'll see an example of this shortly.

Sets with structure

If a set is given by a short list, we can just count its elements. If a large set has some structure, we can often make use of that structure to count it.

For example, if a large set can be generated by a short algorithm, we might be able to use that algorithm to compute the number of elements. We'll see an example of this shortly.

If a large set is defined by a short description, we might be able to use that description to compute the number of elements. We'll see an example shortly.

Sets with structure

If a set is given by a short list, we can just count its elements. If a large set has some structure, we can often make use of that structure to count it.

For example, if a large set can be generated by a short algorithm, we might be able to use that algorithm to compute the number of elements. We'll see an example of this shortly.

If a large set is defined by a short description, we might be able to use that description to compute the number of elements. We'll see an example shortly.

We use the special symbol $\mathbb{N}$ for the counting numbers (also known as the "natural numbers"). That is $\mathbb{N} = \{1, 2, 3, 4, \dots\}$. This is an infinite set.

Two examples

If $P(x)$ denotes some statement about an element x, then the notation $C = \{x \in E : P(x)\}$ represents the set of objects x that belong to E for which the statement $P(x)$ is true. This is part of what I meant by 'a set given by description'.

Two examples

If $P(x)$ denotes some statement about an element x, then the notation $C = \{x \in E : P(x)\}$ represents the set of objects x that belong to E for which the statement $P(x)$ is true. This is part of what I meant by 'a set given by description'.

The set $D = \{x \in \mathbb{N} : x \leq 100\}$ has 100 elements. We get this from the description: D contains all the counting numbers from 1 to 100. It essentially counts itself.

Two examples

If $P(x)$ denotes some statement about an element x, then the notation $C = \{x \in E : P(x)\}$ represents the set of objects x that belong to E for which the statement $P(x)$ is true. This is part of what I meant by 'a set given by description'.

The set $D = \{x \in \mathbb{N} : x \leq 100\}$ has 100 elements. We get this from the description: D contains all the counting numbers from 1 to 100. It essentially counts itself.

The set of 5-letter strings (lets say they contain only uppercase English letters) can be generated by an algorithm using loops of length 26 nested 5 levels deep. A loop of length 26 inside a loop of length 26 has it contents executed 26×26 times. Extending this to 5 levels deep gives us 26^5,which is the number 11,881,376 we saw before.

The rules of 1-to-1 correspondence and containment

Rule

If two sets A and B are in 1-to-1 correspondence, then $|A| = |B|$.

The rules of 1-to-1 correspondence and containment

Rule

If two sets A and B are in 1-to-1 correspondence, then $|A| = |B|$.

This is basic to counting. Counting by ones: "1, 2, 3,. . ." is nothing more than matching the set we are counting with a set of counting numbers. When the set runs out, the number we have reached is the number of its elements.

The rules of 1-to-1 correspondence and containment

Rule

If two sets A and B are in 1-to-1 correspondence, then $|A| = |B|$.

This is basic to counting. Counting by ones: "1, 2, 3,. . ." is nothing more than matching the set we are counting with a set of counting numbers. When the set runs out, the number we have reached is the number of its elements.

Nearly as basic is the following

Rule

If $B \subseteq A$ then $|B| \leq |A|$. If there are elements of A that are not in B then $|B| < |A|$.

The rules of 1-to-1 correspondence and containment

Rule

If two sets A and B are in 1-to-1 correspondence, then $|A| = |B|$.

This is basic to counting. Counting by ones: "1, 2, 3,..." is nothing more than matching the set we are counting with a set of counting numbers. When the set runs out, the number we have reached is the number of its elements.

Nearly as basic is the following

Rule

If $B \subseteq A$ then $|B| \leq |A|$. If there are elements of A that are not in B then $|B| < |A|$.

If we put these together we get: if B is in 1-to-1 correspondence with a subset of A then $|B| \leq |A|$.

Rule of sum

Let me first state this in terms of sets. The *intersection* of sets A and B is the collection of elements that are in both A and B. We denote it $A \cap B$.

Rule of sum

Let me first state this in terms of sets. The *intersection* of sets A and B is the collection of elements that are in both A and B. We denote it $A \cap B$. If there are no elements common to A and B we say they are *disjoint*. We can express this as $A \cap B = \varnothing$, where $\varnothing$ symbolizes the empty set.

Rule of sum

Let me first state this in terms of sets. The *intersection* of sets A and B is the collection of elements that are in both A and B. We denote it $A \cap B$. If there are no elements common to A and B we say they are *disjoint*. We can express this as $A \cap B = \varnothing$, where $\varnothing$ symbolizes the empty set.

The *union* of sets A and B is the collection of all elements that are in either A or B. We denote it $A \cup B$.

Rule of sum

Let me first state this in terms of sets. The *intersection* of sets A and B is the collection of elements that are in both A and B. We denote it $A \cap B$. If there are no elements common to A and B we say they are *disjoint*. We can express this as $A \cap B = \varnothing$, where $\varnothing$ symbolizes the empty set.

The *union* of sets A and B is the collection of all elements that are in either A or B. We denote it $A \cup B$. To be clear: elements that appear in both A and B are not counted twice in $A \cup B$.

Rule of sum

Let me first state this in terms of sets. The *intersection* of sets A and B is the collection of elements that are in both A and B. We denote it $A \cap B$. If there are no elements common to A and B we say they are *disjoint*. We can express this as $A \cap B = \varnothing$, where $\varnothing$ symbolizes the empty set.

The *union* of sets A and B is the collection of all elements that are in either A or B. We denote it $A \cup B$. To be clear: elements that appear in both A and B are not counted twice in $A \cup B$.

Rule

$|A \cup B| \leq |A| + |B|$. *If A and B are disjoint, then* $|A \cup B| = |A| + |B|$. *In general,* $|A \cup B| = |A| + |B| - |A \cap B|$.

Rule of sum, restated

Our textbook's Rule of Sum is the following.

Rule of sum, restated

Our textbook's Rule of Sum is the following.

Rule

If one task can be performed in m *ways, while another task can be performed in* k *ways, and the two tasks cannot be performed simultaneously, then there are* $m + k$ *possible ways to perform one or the other task.*

Rule of sum, restated

Our textbook's Rule of Sum is the following.

Rule

If one task can be performed in m ways, while another task can be performed in k ways, and the two tasks cannot be performed simultaneously, then there are $m + k$ possible ways to perform one or the other task.

Both the following tasks will produce an element of $A \cup B$: (1) select the element from A or (2) select the element from set B.

Rule of sum, restated

Our textbook's Rule of Sum is the following.

Rule

If one task can be performed in m ways, while another task can be performed in k ways, and the two tasks cannot be performed simultaneously, then there are $m + k$ possible ways to perform one or the other task.

Both the following tasks will produce an element of $A \cup B$: (1) select the element from A or (2) select the element from set B.

The first can be done in $m = |A|$ ways, the second in $k = |B|$ ways.

Rule of sum, restated

Our textbook's Rule of Sum is the following.

Rule

If one task can be performed in m ways, while another task can be performed in k ways, and the two tasks cannot be performed simultaneously, then there are $m + k$ possible ways to perform one or the other task.

Both the following tasks will produce an element of $A \cup B$: (1) select the element from A or (2) select the element from set B.

The first can be done in $m = |A|$ ways, the second in $k = |B|$ ways. If A and B are disjoint, we cannot select the element simultaneously from both A and B.

Rule of sum, restated

Our textbook's Rule of Sum is the following.

Rule

If one task can be performed in m ways, while another task can be performed in k ways, and the two tasks cannot be performed simultaneously, then there are $m + k$ possible ways to perform one or the other task.

Both the following tasks will produce an element of $A \cup B$: (1) select the element from A or (2) select the element from set B.

The first can be done in $m = |A|$ ways, the second in $k = |B|$ ways. If A and B are disjoint, we cannot select the element simultaneously from both A and B. So the number of ways to select from either A or B is $m + k$. This is the same as the number of elements in $A \cup B$.

Rule of sum, restated

Our textbook's Rule of Sum is the following.

Rule

If one task can be performed in m ways, while another task can be performed in k ways, and the two tasks cannot be performed simultaneously, then there are $m + k$ possible ways to perform one or the other task.

Both the following tasks will produce an element of $A \cup B$: (1) select the element from A or (2) select the element from set B.

The first can be done in $m = |A|$ ways, the second in $k = |B|$ ways. If A and B are disjoint, we cannot select the element simultaneously from both A and B. So the number of ways to select from either A or B is $m + k$. This is the same as the number of elements in $A \cup B$.

The rule of sum extends to any number of tasks, as long as no two tasks can be performed simultaneously.

Rule of product

Following the idea of building a set by performing tasks, we have the following rule:

Rule

Suppose one task can be performed in m ways and a second task can be performed in k ways. Then the number of ways to perform the first followed by the second is mk.

Rule of product

Following the idea of building a set by performing tasks, we have the following rule:

Rule

Suppose one task can be performed in m ways and a second task can be performed in k ways. Then the number of ways to perform the first followed by the second is mk.

This rule extends to a sequence of any number of tasks.

Rule of product

Following the idea of building a set by performing tasks, we have the following rule:

Rule

Suppose one task can be performed in m ways and a second task can be performed in k ways. Then the number of ways to perform the first followed by the second is mk.

This rule extends to a sequence of any number of tasks.

It is important, in applications, that the following are satisfied

1. The number of ways to perform a task does not depend on the outcome of the previous tasks.

Rule of product

Following the idea of building a set by performing tasks, we have the following rule:

Rule

Suppose one task can be performed in m ways and a second task can be performed in k ways. Then the number of ways to perform the first followed by the second is mk.

This rule extends to a sequence of any number of tasks.

It is important, in applications, that the following are satisfied

1. The number of ways to perform a task does not depend on the outcome of the previous tasks.
2. If we use this to count final outcomes, we must have
   - If any one task is done differently, the final outcome is different.

Rule of product

Following the idea of building a set by performing tasks, we have the following rule:

Rule

Suppose one task can be performed in m ways and a second task can be performed in k ways. Then the number of ways to perform the first followed by the second is mk.

This rule extends to a sequence of any number of tasks.

It is important, in applications, that the following are satisfied

1. The number of ways to perform a task does not depend on the outcome of the previous tasks.
2. If we use this to count final outcomes, we must have
   - If any one task is done differently, the final outcome is different.
   - All final outcomes are produced by some way of performing the sequence of tasks.

Examples

How many 5-letter strings are possible?

Examples

How many 5-letter strings are possible?

Imagine building a string in 5 steps: (1) select the first letter of the string, (2) select the second letter, etc.

Examples

How many 5-letter strings are possible?

Imagine building a string in 5 steps: (1) select the first letter of the string, (2) select the second letter, etc. There are 26 ways to perform each step.

Examples

How many 5-letter strings are possible?

Imagine building a string in 5 steps: (1) select the first letter of the string, (2) select the second letter, etc. There are 26 ways to perform each step. A different choice at any step produces a different string.

Examples

How many 5-letter strings are possible?

Imagine building a string in 5 steps: (1) select the first letter of the string, (2) select the second letter, etc. There are 26 ways to perform each step. A different choice at any step produces a different string. Every string is the result of some sequence of choices.

Examples

How many 5-letter strings are possible?

Imagine building a string in 5 steps: (1) select the first letter of the string, (2) select the second letter, etc. There are 26 ways to perform each step. A different choice at any step produces a different string. Every string is the result of some sequence of choices. So there are $26 \cdot 26 \cdot 26 \cdot 26 \cdot 26 = 26^5$ possibilities.

Examples

How many 5-letter strings are possible?

Imagine building a string in 5 steps: (1) select the first letter of the string, (2) select the second letter, etc. There are 26 ways to perform each step. A different choice at any step produces a different string. Every string is the result of some sequence of choices. So there are $26 \cdot 26 \cdot 26 \cdot 26 \cdot 26 = 26^5$ possibilities.

How many 5-letter strings have all different letters?

Examples

How many 5-letter strings are possible?

Imagine building a string in 5 steps: (1) select the first letter of the string, (2) select the second letter, etc. There are 26 ways to perform each step. A different choice at any step produces a different string. Every string is the result of some sequence of choices. So there are $26 \cdot 26 \cdot 26 \cdot 26 \cdot 26 = 26^5$ possibilities.

How many 5-letter strings have all different letters?

Use almost the same sequence of tasks: (1) select a first letter, (2) select a different second letter, (3) select a third letter different from the first two, etc.

Examples

How many 5-letter strings are possible?

Imagine building a string in 5 steps: (1) select the first letter of the string, (2) select the second letter, etc. There are 26 ways to perform each step. A different choice at any step produces a different string. Every string is the result of some sequence of choices. So there are $26 \cdot 26 \cdot 26 \cdot 26 \cdot 26 = 26^5$ possibilities.

How many 5-letter strings have all different letters?

Use almost the same sequence of tasks: (1) select a first letter, (2) select a different second letter, (3) select a third letter different from the first two, etc.

There are 26 ways to perform step 1, 25 ways to perform step 2, etc.

Examples

How many 5-letter strings are possible?

Imagine building a string in 5 steps: (1) select the first letter of the string, (2) select the second letter, etc. There are 26 ways to perform each step. A different choice at any step produces a different string. Every string is the result of some sequence of choices. So there are $26 \cdot 26 \cdot 26 \cdot 26 \cdot 26 = 26^5$ possibilities.

How many 5-letter strings have all different letters?

Use almost the same sequence of tasks: (1) select a first letter, (2) select a different second letter, (3) select a third letter different from the first two, etc.

There are 26 ways to perform step 1, 25 ways to perform step 2, etc. As before, a different choice at any step produces a different string.

Examples

How many 5-letter strings are possible?

Imagine building a string in 5 steps: (1) select the first letter of the string, (2) select the second letter, etc. There are 26 ways to perform each step. A different choice at any step produces a different string. Every string is the result of some sequence of choices. So there are $26 \cdot 26 \cdot 26 \cdot 26 \cdot 26 = 26^5$ possibilities.

How many 5-letter strings have all different letters?

Use almost the same sequence of tasks: (1) select a first letter, (2) select a different second letter, (3) select a third letter different from the first two, etc.

There are 26 ways to perform step 1, 25 ways to perform step 2, etc. As before, a different choice at any step produces a different string. Every string is the result of some sequence of choices.

Examples

How many 5-letter strings are possible?

Imagine building a string in 5 steps: (1) select the first letter of the string, (2) select the second letter, etc. There are 26 ways to perform each step. A different choice at any step produces a different string. Every string is the result of some sequence of choices. So there are $26 \cdot 26 \cdot 26 \cdot 26 \cdot 26 = 26^5$ possibilities.

How many 5-letter strings have all different letters?

Use almost the same sequence of tasks: (1) select a first letter, (2) select a different second letter, (3) select a third letter different from the first two, etc.

There are 26 ways to perform step 1, 25 ways to perform step 2, etc. As before, a different choice at any step produces a different string. Every string is the result of some sequence of choices. So there are $26 \cdot 25 \cdot 24 \cdot 23 \cdot 22$ possibilities.

A picture

The following illustrates building strings of length 2 from the letters A, B, and C without repeated letters. This illustrates the rule of product: the first task (select a letter) produces a 3-way branching, and then the second task (select a different letter) produces 2-way branching. There are a total of $3 \cdot 2$ paths to the different final results.