Counting

(a.k.a. Combinatorics)

COUNTING?

BUT I KNOW HOW TO COUNT ALREADY!

There's Counting, and there's *Formal Counting*

- The number of operations performed by nested loops? → *Big-Oh complexity*
- How many different IP addresses are there for a given protocol e.g. IPv4/6?
- How many paths there are between vertices in a graph?
- The number of possible passwords given certain size + possible symbols?
- Number of subsets of a finite set?
- How many unique "addressable" memory locations given address of a certain width?

Ashtung!

You *know* and use the “rules” we are about to study
(specially product and sum rules)
Don’t let the formalism that follows make you think otherwise.

The formalism simply allows us to reason about less intuitive problems, and derive general rules.

PRODUCT RULE

The product rule: Motivating Problem

- Suppose a password on a computer system consists of 4 characters. Let's consider these two *different* scenarios.

1. Each must be a digit
2. Each must be a digit or a letter of the alphabet (small case only)

- There are no other constraints.

- How many such passwords are there for Case #1? Case #2?

Basic Counting Principles: The Product Rule

The Product Rule: A procedure can be broken down into a sequence of two tasks.

IF There are n_1 ways to do the first task and n_2 ways to do the second task

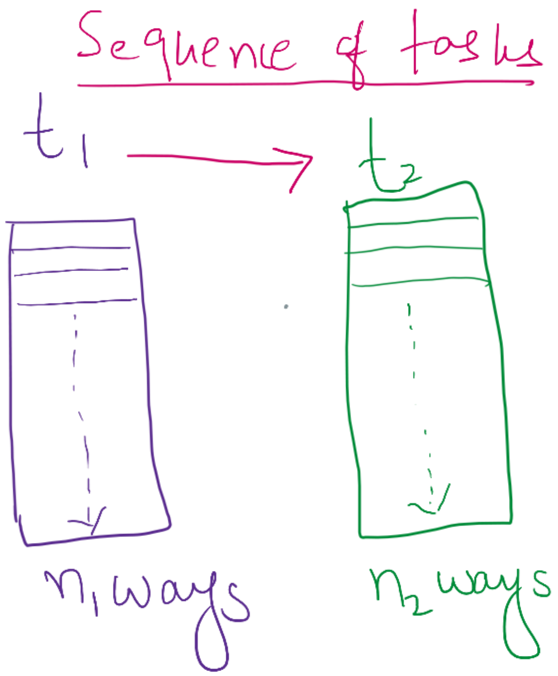
THEN Then there are $n_1 \cdot n_2$ ways to do the procedure.

Basic Counting Principles: The Product Rule

The Product Rule: A procedure can be broken down into a sequence of two tasks.

IF There are n_1 ways to do the first task and n_2 ways to do the second task

THEN Then there are $n_1 \cdot n_2$ ways to do the procedure.

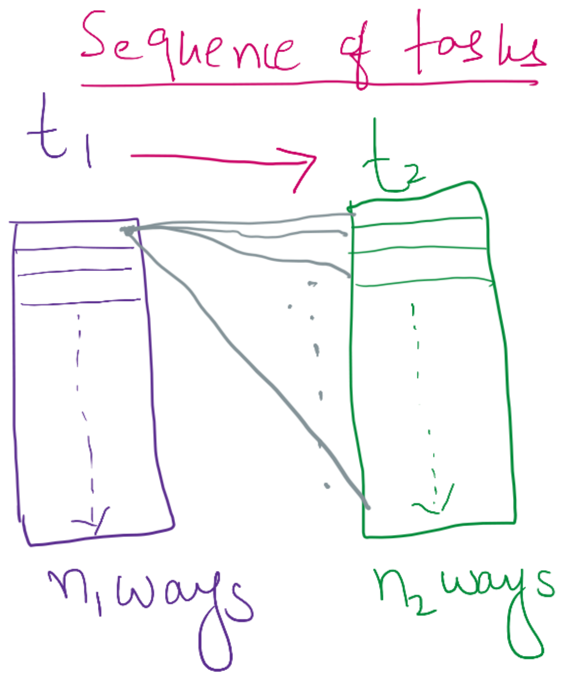


Basic Counting Principles: The Product Rule

The Product Rule: A procedure can be broken down into a sequence of two tasks.

There are n_1 ways to do the first task and n_2 ways to do the second task.

Then there are $n_1 \cdot n_2$ ways to do the procedure.

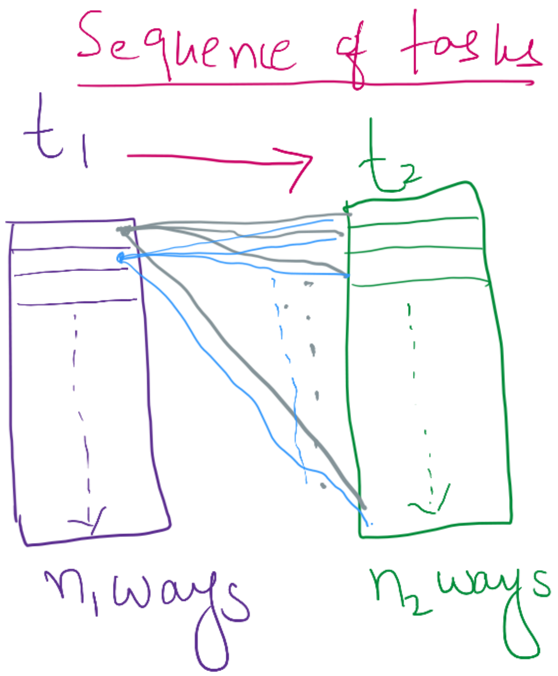


Basic Counting Principles: The Product Rule

The Product Rule: A procedure can be broken down into a **sequence of two tasks**.

There are n_1 ways to do the first task and n_2 ways to do the second task.

Then there are $n_1 \cdot n_2$ ways to do the procedure.



$n_1 \times n_2$ ways to do this sequence of tasks.

Back to the Motivating Problem: The Product Rule

- Suppose a password on a computer system consists of 4 characters.
- Each must be a digit
- There are no other constraints.
- How many such passwords are there?

Back to the Motivating Problem: The Product Rule

- Suppose a password on a computer system consists of 4 characters.
- Each must be a digit
- There are no other constraints.
- How many such passwords are there?

$n_1 \times n_2 \times n_3 \times n_4$

$n_1 = n_2 = n_3 = n_4 = 10$ ways to fill one box.

So $10 \times 10 \times 10 \times 10 = 10^4 = 10{,}000$ such passwords.

0000
0001
⋮
9999

Back to the Motivating Problem: The Product Rule

- Suppose a password on a computer system consists of 4 characters.
- Each must be a digit OR a letter of the alphabet (small case only)
- There are no other constraints.
- How many such passwords are there?

Back to the Motivating Problem: The Product Rule

- Suppose a password on a computer system consists of 4 characters.
- Each must be a digit OR a letter of the alphabet (small case only)
- There are no other constraints.
- How many such passwords are there?

$n_1 \times n_2 \times n_3 \times n_4$

$n_1 = n_2 = n_3 = n_4 = 26+10 = 36$ ways to fill one box.

So $36 \times 36 \times 36 \times 36 = 36^4 = 1{,}679{,}616$ such passwords.

Basic Counting Principles: The Product Rule

The Product Rule: A procedure can be broken down into a sequence of two tasks.

There are n_1 ways to do the first task and n_2 ways to do the second task.

Then there are $n_1 \cdot n_2$ ways to do the procedure.

Example: How many bit strings of length seven are there?

(You can think of placing a bit at one location as a "task")

Basic Counting Principles: The Product Rule

The Product Rule: A procedure can be broken down into a sequence of two tasks.

There are n_1 ways to do the first task and n_2 ways to do the second task.

Then there are $n_1 \cdot n_2$ ways to do the procedure.

Example: How many bit strings of length seven are there?

(You can think of placing a bit at one location as a "task")

0/1	0/1	0/1	0/1	0/1	0/1	0/1

Basic Counting Principles: The Product Rule

The Product Rule: A procedure can be broken down into a sequence of two tasks.

There are n_1 ways to do the first task and n_2 ways to do the second task.

Then there are $n_1 \cdot n_2$ ways to do the procedure.

Example: How many bit strings of length seven are there?

(You can think of placing a bit at one location as a "task")

$2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 = 2^7$

$\boxed{0/1}\boxed{0/1}\boxed{0/1}\boxed{0/1}\boxed{0/1}\boxed{0/1}\boxed{0/1}$

Basic Counting Principles: The Product Rule

The Product Rule: A procedure can be broken down into a sequence of two tasks.

There are n_1 ways to do the first task and n_2 ways to do the second task.

Then there are $n_1 \cdot n_2$ ways to do the procedure.

$2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 = 2^7$

0/1	0/1	0/1	0/1	0/1	0/1	0/1

Example: How many bit strings of length seven are there?

Solution: Since each of the seven bits is either a 0 or a 1, the answer is $2^7 = 128$.

0000000
0000001
0000010
⋮
1111111

Basic Counting Principles: The Product Rule

The Product Rule: A procedure can be broken down into a sequence of two tasks.

There are n_1 ways to do the first task and n_2 ways to do the second task.

Then there are $n_1 \cdot n_2$ ways to do the procedure.

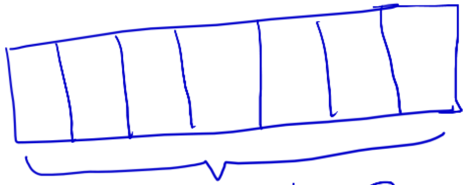
Example: How many *digit* strings (0–9) of length seven are there?

Basic Counting Principles: The Product Rule

The Product Rule: A procedure can be broken down into a sequence of two tasks.

There are n_1 ways to do the first task and n_2 ways to do the second task.

Then there are $n_1 \cdot n_2$ ways to do the procedure.



Each of the 7 "tasks" has 10 possibilities

Example: How many *digit* strings (0–9) of length seven are there?

$10 \times 10 \times 10 \times 10 \times 10 \times 10 \times 10 = 10^7 = 10{,}000{,}000$

(10 million)

i.e 000,0000

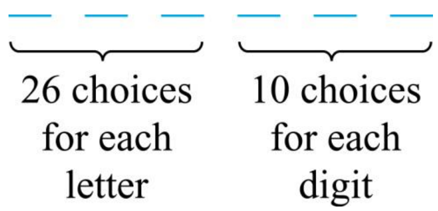
⋮

9,999,999

The Product Rule

Example: How many different license plates can be made if each plate contains a sequence of three uppercase English letters followed by three digits?

Solution: ?



The Product Rule

Example: How many different license plates can be made if each plate contains a sequence of three uppercase English letters followed by three digits?

Solution: By the product rule,

there are $26 \cdot 26 \cdot 26 \cdot 10 \cdot 10 \cdot 10 = 17{,}576{,}000$ different possible license plates.

The Product Rule – We've been using it already!

- How many times is the `<loop-body>` executed?

```
k := 0
for i_1 := 1 to n_1
   for i_2 := 1 to n_2
         .
         .
         .
         for i_m = 1 to n_m
            <loop-body>
```

The Product Rule

- How many times is the `<loop-body>` executed?

```
k := 0
for i₁ := 1 to n₁
   for i₂ := 1 to n₂
         .
         .
         .
         for iₘ = 1 to nₘ
            <loop-body>
```

Each loop can be thought of as a "task", that can be done in n_x ways (ie n_x iterations).

So total possibilities (ie iterations) $= n_1 \times n_2 \times n_3 \times \cdots \times n_m$

The Product Rule

- How

Recall

Rules to compute running times

- Rule 1 – Loops
  - The running time of a loop is at most the running time of the statements inside the loop (including tests) multiplied by the number of iterations

- Rule 2 – Nested loops
  - Should be analysed inside out. Total running time of a statement inside a group of nested loops is running time of statement multiplied by the product of the sizes of all the loops

```
ALG1(n)
  for i = 0 to n-1
   for j = 0 to n-1
     for k = 0 to n-1
       increment x
```

$O(n^3)$

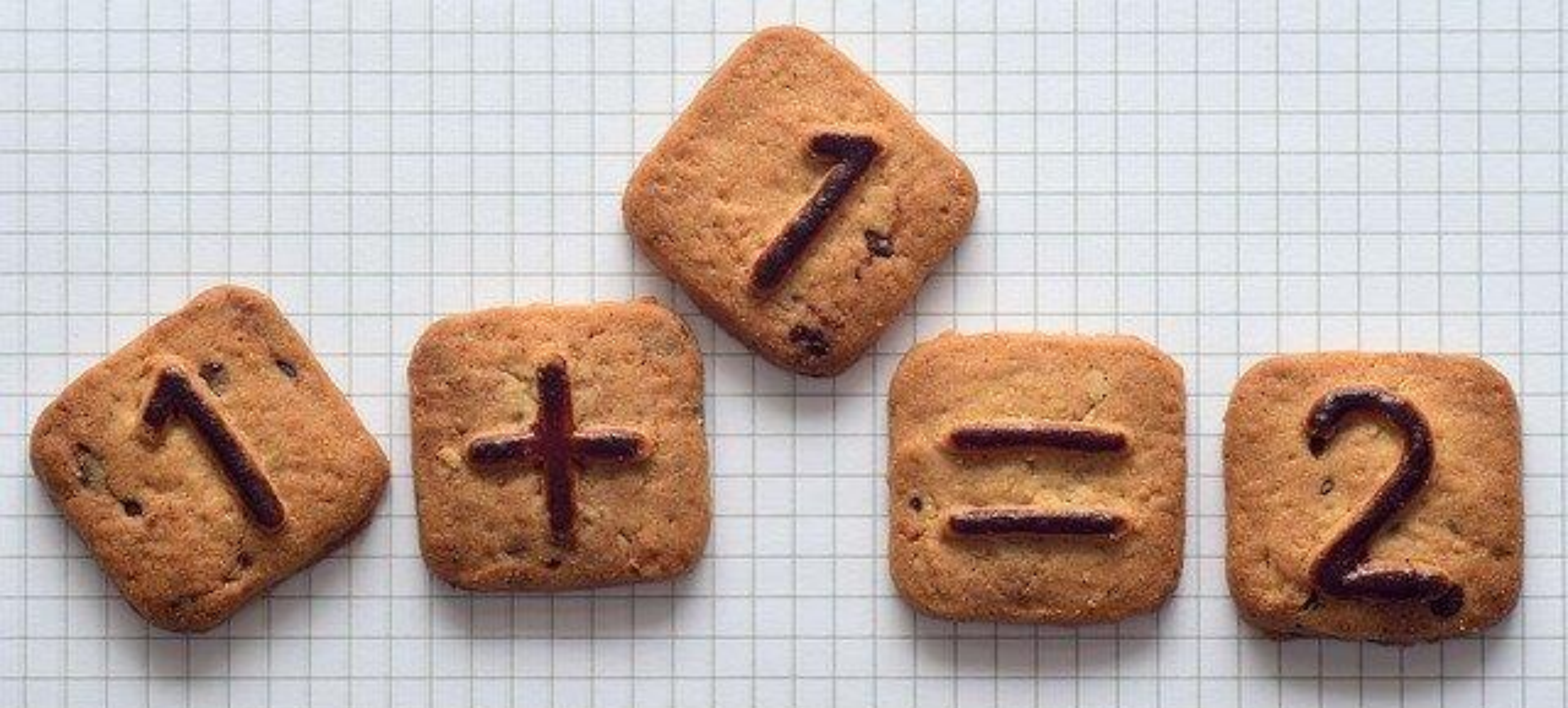
So total possibilities (i.e. iterations)

Product Rule in Cartesian Products

Product Rule in Cartesian Products

- If $A_1, A_2, \ldots, A_m$ are finite sets, then the number of elements in the Cartesian product of these sets is the product of the number of elements of each set.
- The task of choosing an element in the Cartesian product $A_1 \times A_2 \times \cdots \times A_m$ is done by choosing an element in A_1, an element in A_2, …, and an element in A_m.
- By the product rule, it follows that:

$$|A_1 \times A_2 \times \cdots \times A_m| = |A_1| \cdot |A_2| \cdot \cdots \cdot |A_m|.$$



SUM RULE

The sum rule: Motivating Problem

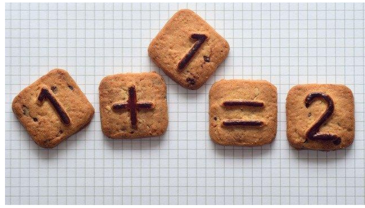
Suppose statement labels in a programming language can be

either a single letter

OR

a single digit.

Find the number of possible labels.



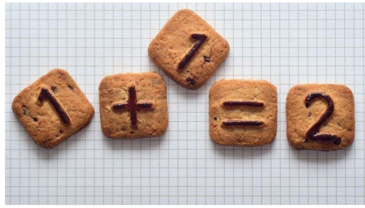
Basic Counting Principles: The Sum Rule

The Sum Rule:

If: a task can be done *either* in one of n_1 ways OR in one of n_2,

where: *none of the elements of set of n_1 ways is the same as any of the n_2 ways*,

then: there are $n_1 + n_2$ ways to do the task.



The sum rule: Motivating Problem

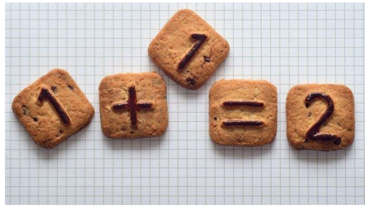
Suppose statement labels in a programming language can be

either a single letter

OR

a single digit.

Find the number of possible labels.



The sum rule: Back to the Motivating Problem

Suppose statement labels in a programming language can be

either a single letter

OR

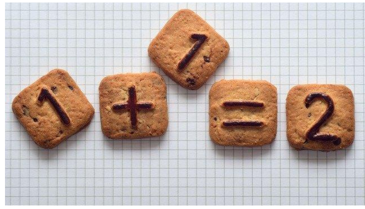
a single digit.

Find the number of possible labels.

26 + 10

(a..z) (0..9)

By sum rule, there are 26 + 10 = 36 possible labels



The Sum Rule: Example

The Sum Rule:

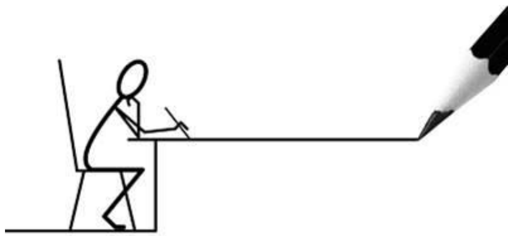
<u>If:</u> a task can be done either in one of n_1 ways OR in one of n_2,

<u>where:</u> *none of the set of n_1 ways is the same as any of the n_2 ways*,

<u>then:</u> there are $n_1 + n_2$ ways to do the task.

The mathematics department must choose either a student OR a faculty member as a representative for a university committee. How many choices are there for this representative if there are 37 members of the mathematics faculty and 83 mathematics majors, <u>*and no one is both a faculty member and a student*</u>.

?



The Sum Rule: Example

The Sum Rule:

> If: a task can be done either in one of n_1 ways OR in one of n_2,
>
> where: *none of the set of n_1 ways is the same as any of the n_2 ways*,
>
> then: there are $n_1 + n_2$ ways to do the task.

The mathematics department must choose either a student OR a faculty member as a representative for a university committee. How many choices are there for this representative if there are 37 members of the mathematics faculty and 83 mathematics majors *and no one is both a faculty member and a student.*

Solution: By the sum rule it follows that there are $37 + 83 = 120$ possible ways to pick a representative.

The Sum Rule in terms of sets

The Sum Rule:

<u>If:</u> a task can be done either in one of n_1 ways OR in one of n_2,

<u>where:</u> *none of the set of n_1 ways is the same as any of the n_2 ways*,

<u>then:</u> there are $n_1 + n_2$ ways to do the task.

Let's say set A consists of the n_1 ways the task can be done in one way

and set B consists of the n_2 ways the task can be done in another way

Then $A \cup B$ contains all possible ways the task can be done

So we can find the total options/ways by finding the *cardinality* of this Union (i.e. the number of elements in $A \cup B$)

The Sum Rule in terms of sets

The Sum Rule:

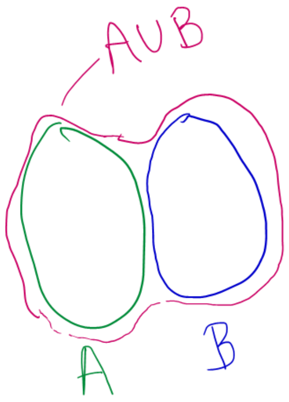
> <u>If:</u> a task can be done either in one of n_1 ways OR in one of n_2,
>
> <u>where:</u> *none of the set of n_1 ways is the same as any of the n_2 ways*,
>
> <u>then:</u> there are $n_1 + n_2$ ways to do the task.

Let's say set A consists of the n_1 ways the task can be done in one way

and set B consists of the n_2 ways the task can be done in another way

Then $A \cup B$ contains all possible ways the task can be done

So we can find the total options/ways by finding the cardinality of this Union (i.e. the number of elements in $A \cup B$)



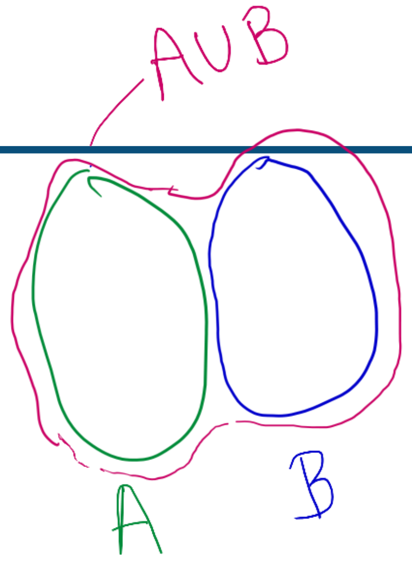
A+B are disjoint; no common elements

The Sum Rule in terms of sets

- Hence, the sum rule can be phrased in terms of sets.

  $|A \cup B| = |A| + |B|$ as long as ***A*** **and** ***B*** **are disjoint sets**.

- Or more generally,



A + B are disjoint; no common elements

The Sum Rule in terms of sets

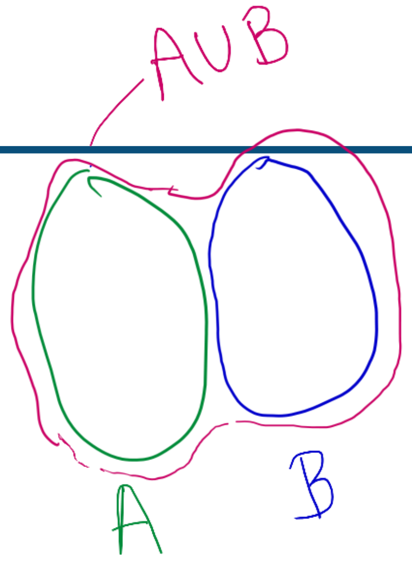
- Hence, the sum rule can be phrased in terms of sets.

  $|A \cup B| = |A| + |B|$ as long as <u>*A* and *B* are disjoint sets</u>.

- Or more generally,

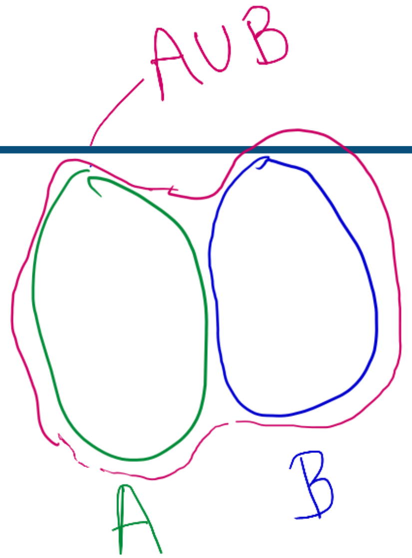
$$|A_1 \cup A_2 \cup \cdots \cup A_m| = |A_1| + |A_2| + \cdots + |A_m|$$

when $A_i \cap A_j = \emptyset$ for all i, j.



A+B are disjoint; no common elements

The Sum Rule in terms of sets



- Hence, the sum rule can be phrased in terms of sets.

  $|A \cup B| = |A| + |B|$ as long as ***A*** **and** ***B*** **are disjoint sets**.

- Or more generally,

$$|A_1 \cup A_2 \cup \cdots \cup A_m| = |A_1| + |A_2| + \cdots + |A_m|$$

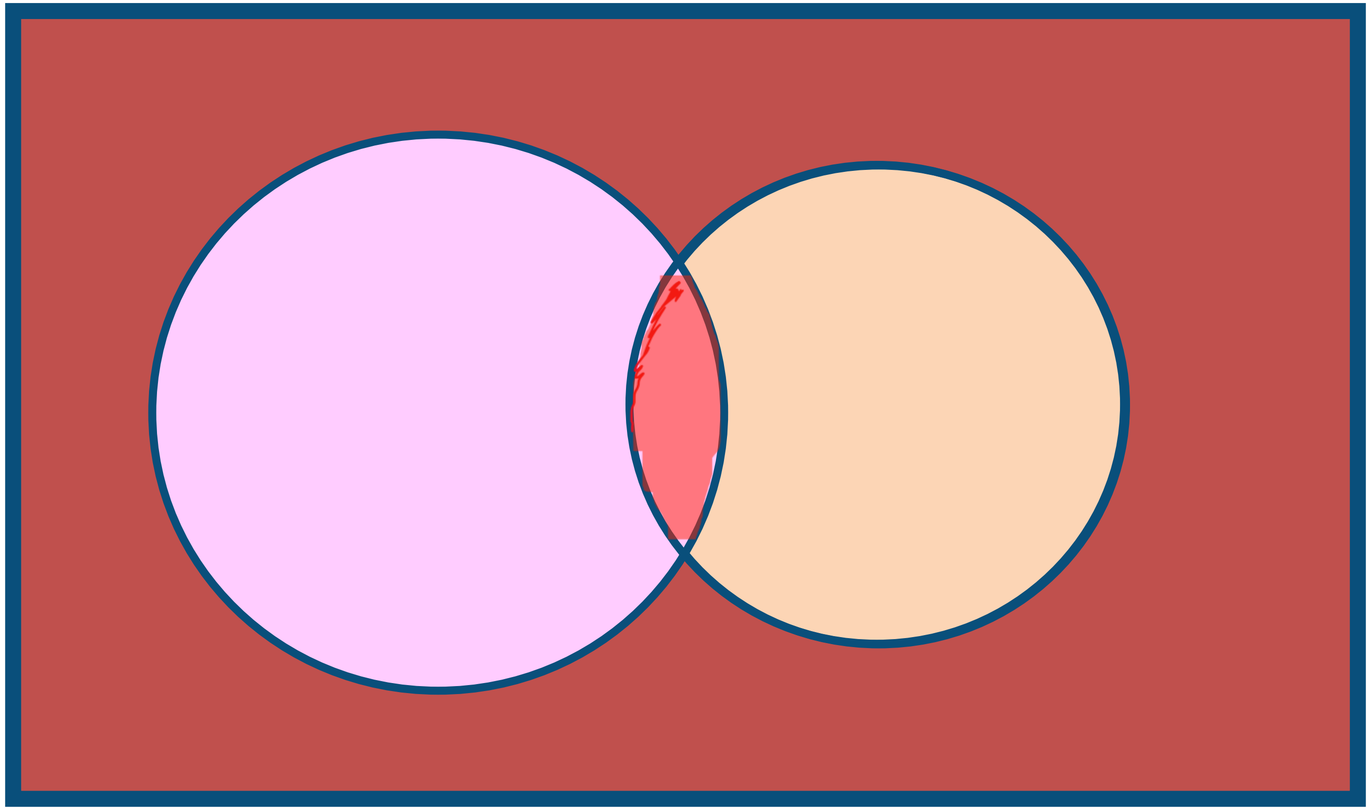
$$\text{when } A_i \cap A_j = \emptyset \text{ for all } i, j.$$

A+B are disjoint;
no common elements

- **What if there are common elements? (sets are not disjoint?)**

THE INCLUSION-EXCLUSION RULE

(ALSO KNOWN AS: SUBTRACTION RULE)



Can you solve this using product and/or sum rule?

- How many bit strings of length 8 either start with a 1 or end with 00 (or both)?

Can you solve this using product and/or sum rule?

- How many bit strings of length 8 either start with a 1 or end with 00 (or both)?

Can you solve this using product and/or sum rule?

- How many bit strings of length 8 either start with a 1 or end with 00 (or both)?

Inclusion–exclusion principle – Introduction

How many bit strings of length 8 either start with a 1 or end with 00?

- there are 2^8 bit strings of length 8

product rule: 8 positions each can be filled in 2 different ways

* * * * * * * *

Inclusion-exclusion principle – Introduction

How many bit strings of length 8 either start with a 1 or end with 00?

- there are 2^8 bit strings of length 8
- there are 2^7 bit strings of length 8 that start with a 1
  - note, this also includes strings that end with 00

product rule: 7 positions each can be filled in 2 different ways

1*******

Inclusion–exclusion principle – Introduction

How many bit strings of length 8 either start with a 1 or end with 00?

- there are 2^8 bit strings of length 8
- there are 2^7 bit strings of length 8 that start with a 1
- there are 2^6 bit strings of length 8 that end with 00
  - note, this also includes strings that start with a 1

product rule: 6 positions each can be filled in 2 different ways

******00

Inclusion-exclusion principle – Introduction

How many bit strings of length 8 either start with a 1 or end with 00?

- there are 2^8 bit strings of length 8
- there are 2^7 bit strings of length 8 that start with a 1
- there are 2^6 bit strings of length 8 that end with 00
- using sum rule
  - there are $2^7 + 2^6$ bit strings that start with 1 or end with 00
    - but we have over-counted
    - adding in twice the strings that start with 1 and end with 00

Inclusion-exclusion principle – Introduction

How many bit strings of length 8 either start with a 1 or end with 00?

- there are 2^8 bit strings of length 8
- there are 2^7 bit strings of length 8 that start with a 1
- there are 2^6 bit strings of length 8 that end with 00
- using sum rule
  - there are $2^7 + 2^6$ bit strings that start with 1 or end with 00
    - but we have over-counted
    - adding in twice the strings that start with 1 and end with 00
- there are 2^5 bit strings of length 8 that start with a 1 and end with 00

product rule: 5 positions each can be filled in 2 different ways

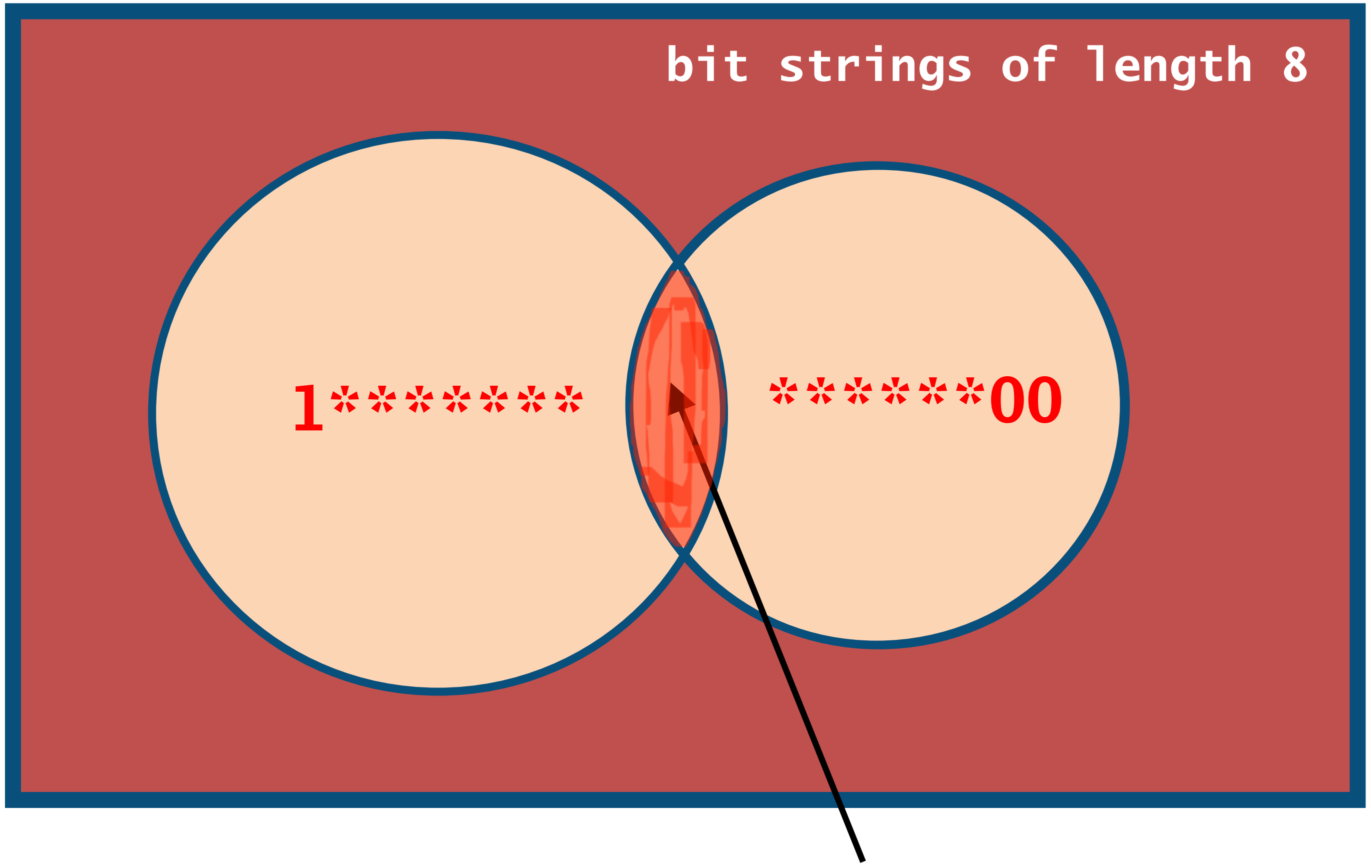
1***00**

Inclusion–exclusion principle – Introduction

How many bit strings of length 8 either start with a 1 or end with 00?

- there are 2^8 bit strings of length 8
- there are 2^7 bit strings of length 8 that start with a 1
- there are 2^6 bit strings of length 8 that end with 00
- using sum rule
  - there are $2^7 + 2^6$ bit strings that start with 1 or end with 00
    - but we have over-counted
    - adding in twice the strings that start with 1 and end with 00
- there are 2^5 bit strings of length 8 that start with a 1 and end with 00
- therefore we actually have $2^7 + 2^6 - 2^5$ bit strings

Inclusion–exclusion principle – Introduction



overcounted strings of the form 1*****00

Basic Counting Principles: Principle of Inclusion-Exclusion

(also known as principle Subtraction Rule)

Principle of Inclusion–Exclusion: If a task can be done either in one of n_1 ways or in one of n_2 ways, then the total number of ways to do the task is $n_1 + n_2$ minus the number of ways to do the task that are common to the two different ways.

- Also known as, the *Subtraction Rule*:

$$|A \cup B| = |A| + |B| - |A \cap B|$$

- The SUM RULE was simply a special case of this principle, where the sets where assumed to be disjoint, so their intersection was empty, and thus $|A \cap B| = 0$

Combining Product and Sum Rules

Combining the Sum and Product Rule

Example: Suppose statement labels in a programming language can be either a single letter OR a letter followed by a digit.

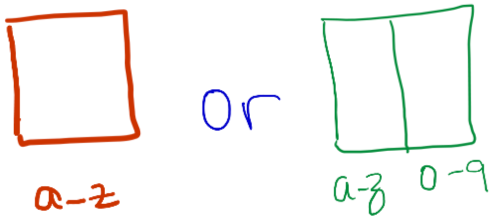
Find the number of possible labels.

Solution: ?

Combining the Sum and Product Rule

Example: Suppose statement labels in a programming language can be either a single letter OR a letter followed by a digit.

Find the number of possible labels.

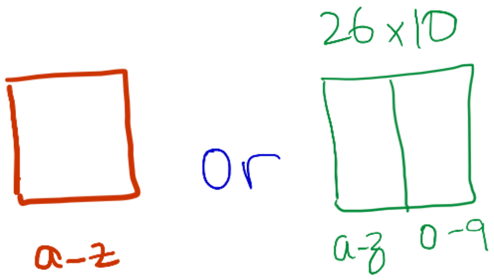


Solution: ?

Combining the Sum and Product Rule

Example: Suppose statement labels in a programming language can be either a single letter OR a letter followed by a digit.

Find the number of possible labels.



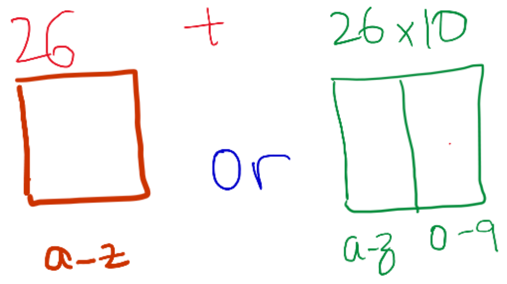
Solution: Use the sum and product rule.

$$\underbrace{26 \cdot 10}_{\text{product rule}} =$$

Combining the Sum and Product Rule

Example: Suppose statement labels in a programming language can be either a single letter OR a letter followed by a digit.

Find the number of possible labels.



Solution: Use the sum and product rule.

$$\underbrace{26 + \underbrace{26 \cdot 10}_{\text{product rule}}}_{\text{sum rule}} = 286$$

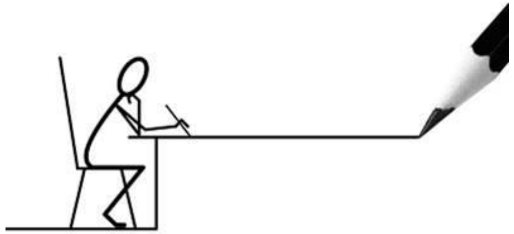
The sum and product rules – Example

How many passwords are there, where the passwords must have

- 6 alpha-numeric characters and first character must be a capital letter

Alpha-numeric: 0 – 9, a – z, A – Z.

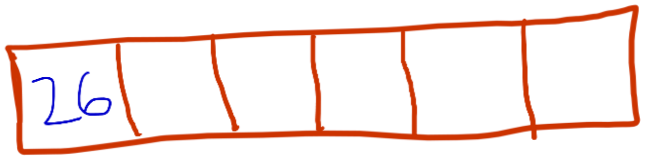
- ?



The sum and product rules – Example

How many passwords are there, where the passwords must have

- 6 alpha-numeric characters and first character must be a capital letter



There are 26 choices for the 1st position

- i.e. A or B or C or … or Z

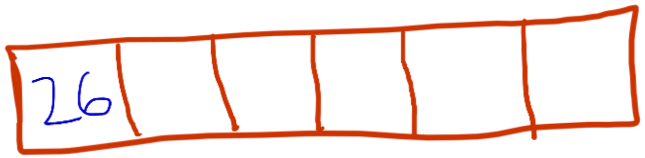
The sum and product rules – Example

How many passwords are there, where the passwords must have

- 6 alpha-numeric characters and first character must be a capital letter

There are 26 choices for the 1^{st} position

- i.e. A or B or C or … or Z



Other positions?

The sum and product rules – Example

How many passwords are there, where the passwords must have

- 6 alpha-numeric characters and first character must be a capital letter

26	62	62	62	62	62

There are 26 choices for the 1st position

- i.e. A or B or C or … or Z

Using the <u>sum rule</u> there are 26+26+10 = 62 choices for each of the 2nd, 3rd, 4th, 5th and 6th positions (a-z, A-Z, or 0-9)

26 + 26 + 10

The sum and product rules – Example

How many passwords are there, where the passwords must have

- 6 alpha-numeric characters and first character must be a capital letter

26	62	62	62	62	62

There are 26 choices for the 1st position

- i.e. A or B or C or … or Z

Using the sum rule there are 26+26+10 = 62 choices for each of the 2nd, 3rd, 4th, 5th and 6th positions (a-z, A-Z, or 0-9)

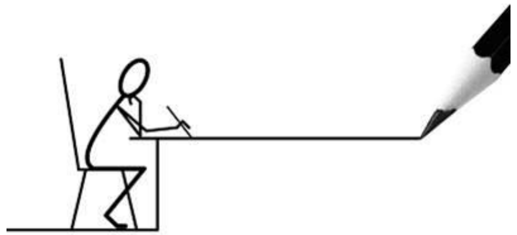
Using the product rule this yields a total of

$26 \cdot 62 \cdot 62 \cdot 62 \cdot 62 \cdot 62 = 23{,}819{,}453{,}632$ choices

The sum and product rules – Example

Suppose that a password is of length between 4 and 6 characters, consists of letters and digits, and is case sensitive

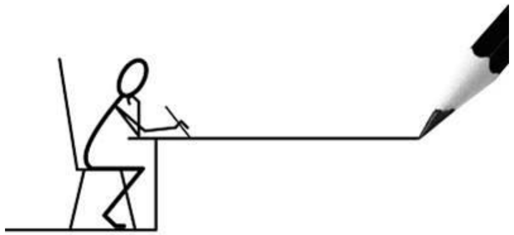
- how many distinct passwords are there?
- how many contain at least one letter and at least one digit?
- how long would it take a hacker to test all passwords at 10^9 tests per second?



The sum and product rules – Example

Suppose that a password is of length between 4 and 6 characters, consists of letters and digits, and is case sensitive

- how many distinct passwords are there?
- how many contain at least one letter and at least one digit?
- how long would it take a hacker to test all passwords at 10^9 tests per second?



The sum and product rules – Example

Suppose that a password is of length between 4 and 6 characters, consists of letters and digits, and is case sensitive

- how many distinct passwords are there?
- how many contain at least one letter and at least one digit?
- how long would it take a hacker to test all passwords at 10^9 tests per second?

There are 26+26+10=62 available characters (sum rule)

62^r passwords of length r (product rule)

The sum and product rules – Example

Suppose that a password is of length between 4 and 6 characters, consists of letters and digits, and is case sensitive

- how many distinct passwords are there?
- how many contain at least one letter and at least one digit?
- how long would it take a hacker to test all passwords at 10^9 tests per second?

There are 26+26+10=62 available characters (sum rule)

62^r passwords of length r (product rule)

So 62^4 passwords of length 4

So 62^5 passwords of length 5

So 62^6 passwords of length 6

The sum and product rules – Example

Suppose that a password is of length between 4 and 6 characters, consists of letters and digits, and is case sensitive

- how many distinct passwords are there?
- how many contain at least one letter and at least one digit?
- how long would it take a hacker to test all passwords at 10^9 tests per second?

There are 26+26+10=62 available characters (sum rule)

62^r passwords of length r (product rule)

So 62^4 passwords of length 4

So 62^5 passwords of length 5

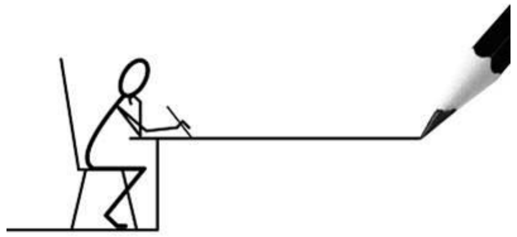
So 62^6 passwords of length 6

$62^4+62^5+62^6$ = 57,731,144,752 passwords in total (sum rule)

The sum and product rules – Example

Suppose that a password is of length between 4 and 6 characters, consists of letters and digits, and is case sensitive

- how many distinct passwords are there?
- **how many contain at least one letter and at least one digit?**
- how long would it take a hacker to test all passwords at 10^9 tests per second?



The sum and product rules – Example

Suppose that a password is of length between 4 and 6 characters, consists of letters and digits, and is case sensitive

- how many distinct passwords are there?
- how many contain at least one letter and at least one digit?
- how long would it take a hacker to test all passwords at 10^9 tests per second?

We have to exclude passwords consisting of

- just letters: $(26+26)^4 + (26+26)^5 + (26+26)^6 = 20{,}158{,}125{,}312$

The sum and product rules – Example

Suppose that a password is of length between 4 and 6 characters, consists of letters and digits, and is case sensitive

- how many distinct passwords are there?
- how many contain at least one letter and at least one digit?
- how long would it take a hacker to test all passwords at 10^9 tests per second?

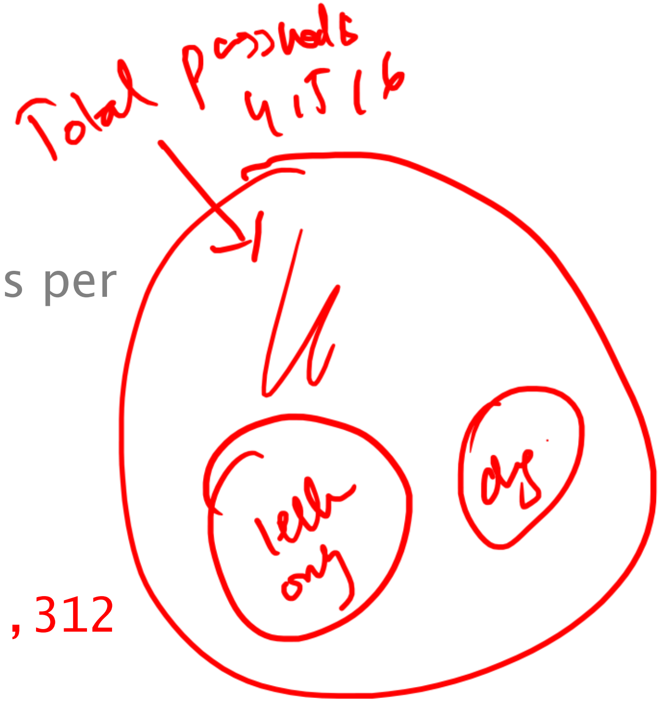
We have to exclude passwords consisting of

- just letters: $(26+26)^4 + (26+26)^5 + (26+26)^6 = 20{,}158{,}125{,}312$
- just digits: $10^4 + 10^5 + 10^6 = 1{,}110{,}000$

The sum and product rules – Example

Suppose that a password is of length between 4 and 6 characters, consists of letters and digits, and is case sensitive

- how many distinct passwords are there?
- how many contain at least one letter and at least one digit?
- how long would it take a hacker to test all passwords at 10^9 tests per second?



We have to exclude passwords consisting of

- just letters: $(26+26)^4 + (26+26)^5 + (26+26)^6 = 20,158,125,312$
- just digits: $10^4 + 10^5 + 10^6 = 1,110,000$

Hence there are

57,731,144,752 – (20,158,125,312 + 1,110,000)

passwords which contain at least one letter and one digit

The sum and product rules – Example

Suppose that a password is of length between **4** and **6** characters, consists of letters and digits, and is case sensitive

- how many distinct passwords are there?
- how many contain at least one letter and at least one digit?
- how long would it take a hacker to test all passwords at 10^9 tests per second?

At 10^9 tests per second, we need $57{,}731{,}144{,}752/10^9=57.7$ seconds so all passwords could be checked in less than 1 minute

Note!

- For this problem…
  - *how many contain at least one letter and at least one digit?*
- …we could have taken another *(maybe more intuitive, but also more difficult to compute)* approach, of adding up the count of all VALID categories of passwords (instead of subtracting all INVALID categories from the grand total).

- So, we could have added all the following counts to arrive at the _same_ answer (check yourself to confirm)
  - 4 character passwords with 1 letter and 3 digits
  - 4 character passwords with 2 letters and 2 digits
  - 4 character passwords with 3 letters and 1 digits
  - 5 character passwords with 1 letter and 4 digits
  - 5 character passwords with 2 letters and 3 digits
  - 5 character passwords with 3 letters and 2 digits
  - 5 character passwords with 4 letters and 1 digit
  - 6 character passwords with 1 letter and 5 digits
  - 6 character passwords with 2 letters and 4 digits
  - 6 character passwords with 3 letters and 3 digits
  - 6 character passwords with 4 letters and 2 digits
  - 6 character passwords with 5 letters and 1 digit

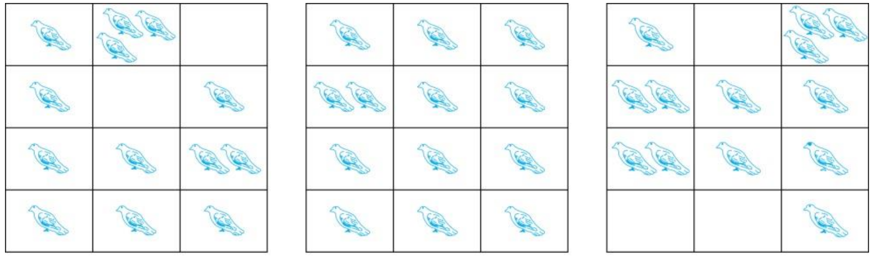
- *Hence: sometimes, when there are less INVALID categoroes and more VALID categories, it is much easier to subtract count of INVALID cagtegories from a total, rather then add all VALID categories.*

THE PIGEONHOLE PRINCIPLE

https://upload.wikimedia.org/wikipedia/commons/thumb/e/e9/Pigeons-in-holes.jpg/592px-Pigeons-in-holes.jpg

The Pigeonhole Principle

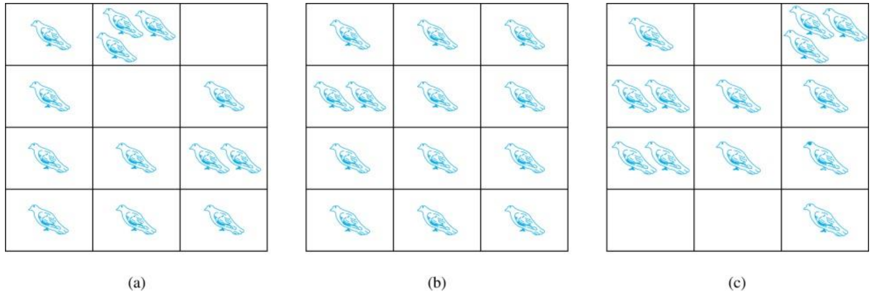
- If a flock of 13 pigeons roosts in a set of 12 pigeonholes, one of the pigeonholes must have more than 1 pigeon.



(a) (b) (c)

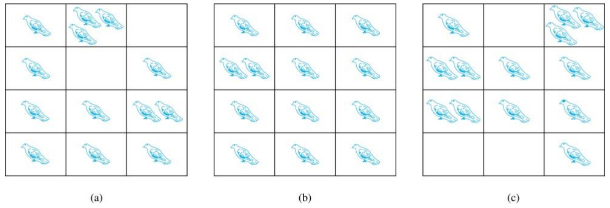
The Pigeonhole Principle

- If a flock of 13 pigeons roosts in a set of 12 pigeonholes, one of the pigeonholes must have more than 1 pigeon.



The Pigeonhole Principle

- If a flock of 13 pigeons roosts in a set of 12 pigeonholes, one of the pigeonholes must have more than 1 pigeon.



Pigeonhole Principle: If k is a positive integer and $k + 1$ objects are placed into k boxes, then at least one box contains two or more objects.

Pigeonhole Principle

Example: Among any group of 367 people, there must be at least two with the same birthday, because there are only 366 possible birthdays.

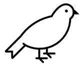
Pigeonhole Principle – How to answer this question?

Example: What if there were 1000 people? there must be at least ___?___ with the same birthday, because there are only 366 possible birthdays.

.

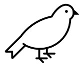
The pigeonhole principle – GENERALIZATION

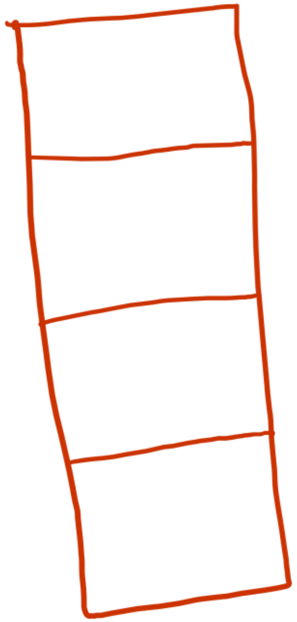
- Let’s say we have 4 pigeonholes.

- Then, if there are 12 pigeons, then at least one box will have ____ (or more) pigeons. 

The pigeonhole principle – GENERALIZATION

- Let's say we have 4 pigeonholes.

- Then, if there are 12 pigeons, then at least one box will have ____ (or more) pigeons. 

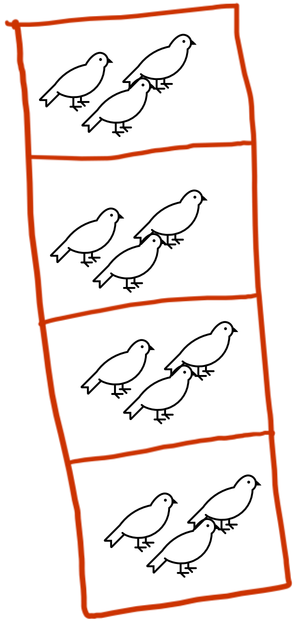


The pigeonhole principle – GENERALIZATION

- Let's say we have 4 pigeonholes.

- Then, if there are 12 pigeons, then *at least* one box will have _3__ (or more) pigeons.

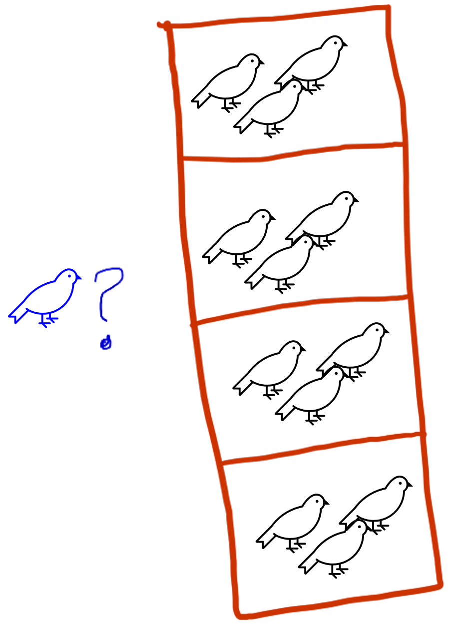
where $3 = 12/4$



The pigeonhole principle – GENERALIZATION

- Let's say we have 4 pigeonholes.

- Then, if there are 13 pigeons, then *at least* one box will have __?_ (or more) pigeons.

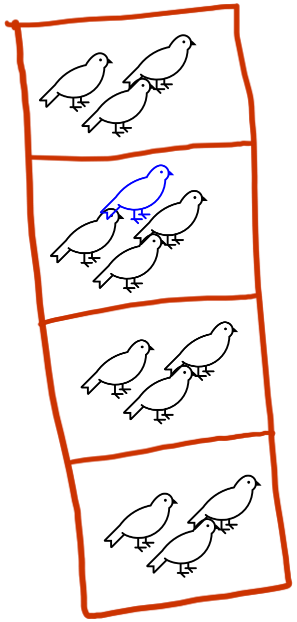


The pigeonhole principle – GENERALIZATION

- Let's say we have 4 pigeonholes.

- Then, if there are 13 pigeons, then at least one box will have _4_ (or more) pigeons.

where 4 = ?

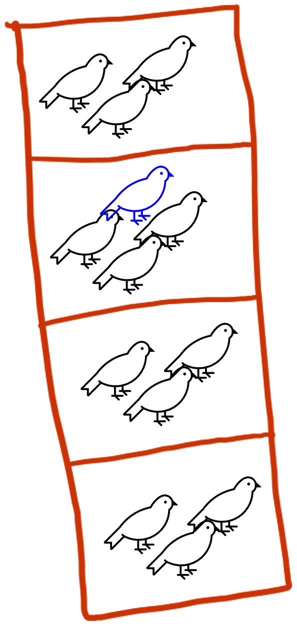


The pigeonhole principle – GENERALIZATION

- Let's say we have 4 pigeonholes.

- Then, if there are 13 pigeons, then at least one box will have _4_ (or more) pigeons.

where $4 = \left(13/4\right) = 3.25 \xrightarrow{\text{round UP}} = 4$



⌈ceiling⌉ OR

ceil operation

The pigeonhole principle – GENERALIZATION

- Let’s say we have 4 pigeonholes.

The pigeonhole principle – GENERALIZATION

- Let’s say we have 4 holes.

- If there are 12 pigeons, then at least one box will have ceil(12/4) = ceil (3) = 3 (or more) pigeons

- If there are 13 pigeons, then at least one box will have ceil(13/4) = ceil (3.25) = 4 (or more) pigeons
- If there are 14 pigeons, then at least one box will have ceil(14/4) = ceil (3.50) = 4 (or more) pigeons
- If there are 15 pigeons, then at least one box will have ceil(15/4) = ceil (3.75) = 4 (or more) pigeons
- If there are 16 pigeons, then at least one box will have ceil(16/4) = ceil (4) = 4 (or more) pigeons

- If there are 17 pigeons, then at least one box will have ceil(17/4) = ceil (4.25) = 5 (or more) pigeons

The pigeonhole principle – GENERALIZATION

If k+1 objects are placed in k containers then one container must contain at least 2 objects

The generalised pigeonhole principle:

The pigeonhole principle – GENERALIZATION

If k+1 objects are placed in k containers then one container must contain at least 2 objects

The generalised pigeonhole principle:

> if **n** objects are placed in **k** containers, then *at least* one container has *at least* **ceil(n/k)** objects
>
> or
>
> $$m = \text{ceil}\,(n/k)$$

– ceil(x) (the ceiling of x) the smallest integer greater than or equal to x
– i.e.round up to the nearest integer (example: 3.001 becomes 4)

Pigeonhole Principle – Coming back to this question

Example: What if there were 1000 people? there must be at least ___?___ with the same birthday, because there are only 366 possible birthdays.

.

The pigeonhole principle – Example 1

if **n** objects are placed in **k** containers, then at least one container has at least **`ceil(n/k)`** objects: $m = ceil\ (n/k)$

Given 100 people in a room, at least how many people are born in the same month?

The pigeonhole principle – Example 1

> if **n** objects are placed in **k** containers, then at least one container has at least **`ceil(n/k)`** objects: $m = ceil\ (n/k)$

Given **100** people in a room, at least how many people are born in the same month?

What are the containers?

months people are born (size **12**)

What are the objects?

people (size **100**)

The pigeonhole principle – Example 1

if **n** objects are placed in **k** containers, then at least one container has at least **ceil(n/k)** objects: $m = ceil\,(n/k)$

Given **100** people in a room, at least how many people are born in the same month?

What are the containers? months people are born (size **12**)

What are the objects? people (size **100**)

Answer: **ceil(100/12)=ceil(8.333)=9**

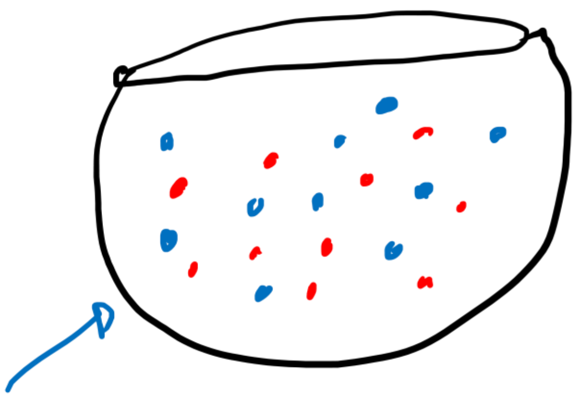
The pigeonhole principle – Example 2

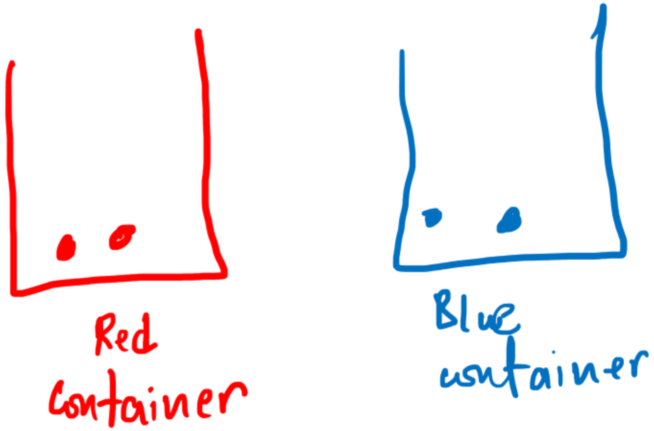
> If **n** objects are placed in **k** containers, then at least one container has at least **`ceil(n/k)`** objects: $m = ceil\,(n/k)$

A bowl contains **10** red and **10** blue balls

A person selects balls at random, *without replacement*

How many balls must be selected to be sure of getting at least three balls of the same colour?





Point to note!

The pigeonhole principle – Example 2

> If **n** objects are placed in **k** containers, then at least one container has at least `ceil(n/k)` objects.

A bowl contains **10** red and **10** blue balls

A person selects balls at random, *without replacement*

How many balls must be selected to be sure of getting at least three balls of the same colour?

$$m = ceil\ (n/k)$$

While this is still a "pigeon-hole" problem, the unknown is *not* what is was in earlier problems.

Before, the question was: given n objects are placed in k containers, then at least one container will have at least how many object? The unknown was: m

Now: We know that, given k containers, at least one container has at least m items. What is the minimum number of items picked? The unknown now is: n

The pigeonhole principle – Example 2

> If **n** objects are placed in **k** containers, then at least one container has at least **`ceil(n/k)`** objects. $m = ceil\ (n/k)$

A bowl contains **10** red and **10** blue balls

A person selects balls at random, *without replacement*

How many balls must be selected to be sure of getting at least three balls of the same colour?

What are the containers?

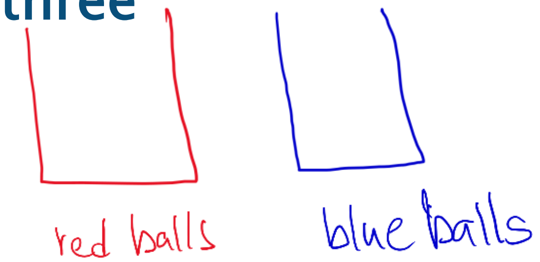
The pigeonhole principle – Example 2

> If **n** objects are placed in **k** containers, then at least one container has at least **`ceil(n/k)`** objects. $m = ceil\,(n/k)$

A bowl contains **10** red and **10** blue balls

A person selects balls at random, *without replacement*

How many balls must be selected to be sure of getting at least three balls of the same colour?



What are the containers?

ball colours (size **2**)

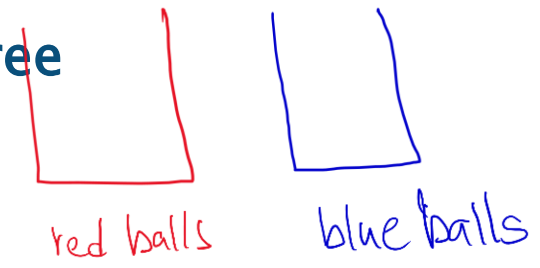
The pigeonhole principle – Example 2

> If **n** objects are placed in **k** containers, then at least one container has at least **`ceil(n/k)`** objects.

A bowl contains **10** red and **10** blue balls

A person selects balls at random, *without replacement*

How many balls must be selected to be sure of getting at least three balls of the same colour?



What are the k containers?

  ball colours (size **2**)

What are the n objects?

  balls <u>selected</u> (what we want to find)

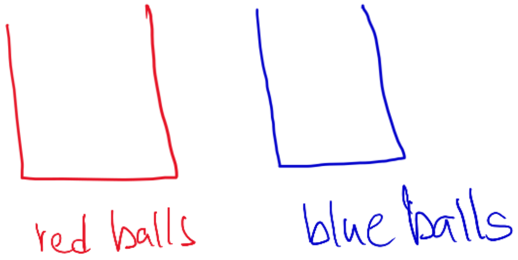
The pigeonhole principle – Example 2

> If **n** objects are placed in **k** containers, then at least one container has at least **`ceil(n/k)`** objects. $m = ceil\ (n/k)$

A bowl contains **10** red and **10** blue balls

A person selects balls at random, *without replacement*

How many balls must be selected to be sure of getting at least three balls of the same colour?



What are the k containers?

ball colours (size **2**)

What are the n objects?

balls selected (size **n** – what we want to find)

Answer: *smallest* **n** such that **`m = ceil(n/2) = 3`**, i.e. **n=5**

– since 4/2=2 and 5/2=2.5

A bit more on the maths

Given `ceil(n/2)=3`

- `how do we find n?`

Or, in general, given: `ceil(n/k) = m`

- how do we find **n** (total number of items placed in buckets)
- given:
  - **k** (number of buckets)
  - **m** (at least one bucket must have these many items)

A bit more on the maths

Given `ceil(n/2)=3`

- `how do we find n?`

Or, in general, given: `ceil(n/k) = m`

- how do we find `n` (total number of items placed in buckets)
- given:
  - `k` (number of buckets)
  - `m` (at least one bucket must have these many items)

First, find n such that `n/k = m-1`

- e.g., `n/2 = 3-1 = 2; then n=4`

A bit more on the maths

Given `ceil(n/2)=3`

- `how do we find n?`

Or, in general, given: `ceil(n/k) = m`

- how do we find **n** (total number of items placed in buckets)
- given:
  - **k** (number of buckets)
  - **m** (at least one bucket must have these many items)

First, find n such that `n/k = m-1`

- e.g., `n/2 = 3-1 then n=4`

Then at this current value for **n**, **n/k** is exactly equal to `m-1`

- e.g., `4/2 = 2`

A bit more on the maths

Given `ceil(n/2)=3`

- `how do we find n?`

Or, in general, given: `ceil(n/k) = m`

- how do we find **n** (total number of items placed in buckets)
- given:
  - **k** (number of buckets)
  - **m** (at least one bucket must have these many items)
- reasoning to get to the answer: find n such that *n/k is exactly equaly to m–1;* ***one less*** *than the number we are looking for.*
- **then, simply add one to n**

First, find n such that `n/k = m-1`

- e.g., `n/2 = 3-1 then n=4`

Then at this current value for `n`, `n/k` is exactly equal to `m-1`

- e.g., `4/2 = 2`

This means if we increase `n` by `1`, then taking `ceil(n/k)` will get us to `m`

- e.g., `ceil((4+1)/2) = ceil(5/2) = 3`
- so the answer is `n = 3`

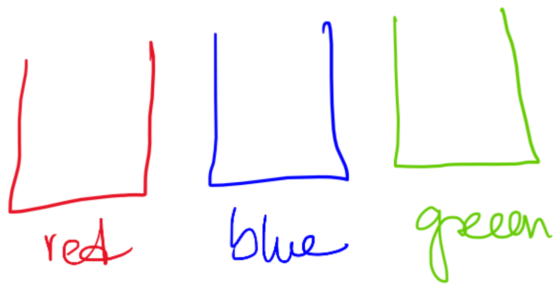
Solve the same example but with different k and m

A bowl contains **10** red, **10** blue balls, and **10** green balls (so $k = 3$)

A person selects balls at random, *without replacement*

How many balls must be selected to be sure of getting at least five (so $m = 5$) balls of the same colour?

- We know that `m = ceil(n/k)`, i.e., `5 = ceil(n/3)`
- How much is `n`?



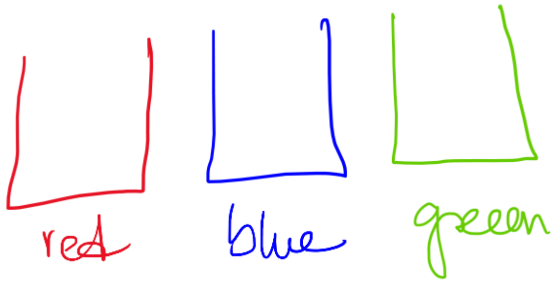
Solve the same example but with different k and m

A bowl contains **10** red, **10** blue balls, and **10** green balls (so **`k = 3`**)

A person selects balls at random, *without replacement*

How many balls must be selected to be sure of getting at least five (so **`m = 5`**) balls of the same colour?

- We know that `m = ceil(n/k)`, i.e., `5 = ceil(n/3)`
- How much is `n`?



- Find `n` such that `n/k=m-1`, i.e., `n/3 = 4`, so `n=12`
- Answer is `12+1 = 13`
- Check: `ceil(13/3) = ceil(4.33) = 5`