



Combinatorics

CS109

Problem Set 1 is out

The screenshot shows a web browser window with the URL `https://cs109psets.netlify.app/sum24/pset1/robot_paths`. The page title is "Pset 1 - Counting for Probability". The main content area is titled "Robot Paths" and contains the following text: "Imagine you have a robot (🤖) that lives on an 7 x 8 grid (it has 7 rows and 8 columns). The robot starts in cell (1, 1) and can take steps either to the right or down (no left or up steps). How many distinct paths can the robot take to the destination (🚩) in cell (7, 8)?" Below the text is a 7x8 grid. A robot icon is in the top-left cell (1,1) and a flag icon is in the bottom-right cell (7,8). The grid is labeled "7 rows" on the left and "8 columns" at the bottom. On the right side of the page, there is an "Answer Editor" section with a "Solution" tab. It contains a "Numeric Answer:" field with the placeholder text "Enter your answer" and a "Check Answer" button. Below this is an "Explanation:" section with a toolbar containing "Block LaTeX", "Inline LaTeX", "Python", and "Image" options. A blue arrow points from the text "Check your answer" to the "Check Answer" button. Another blue arrow points from the text "Insert LaTeX" to the "Block LaTeX" option in the toolbar. At the bottom of the page, there are "Previous Question" and "Next Question" buttons.

Check your answer

Insert LaTeX

Submission
is
automatic!

You can use LaTeX to type fancy math

```
1 \begin{aligned}
2 P(E) |
3 &= \sum_{i=0}^n e^{i} \\
4 &= 0.25
5 \end{aligned}
```

$$P(E) = \sum_{i=0}^n e^i = 0.25$$

Done

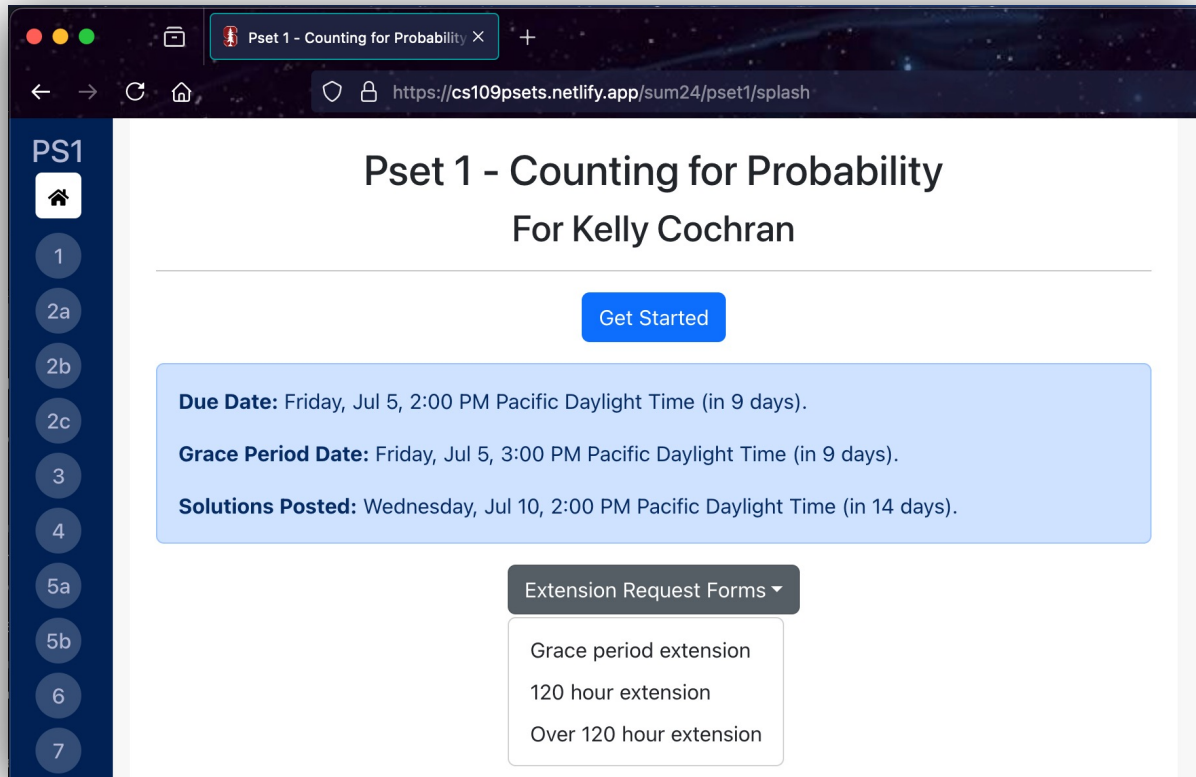


There's a handout on the course website to help you get started!

The screenshot shows a web browser window displaying the CS109 course website. The page title is "CS109 Course Resources" and the URL is "web.stanford.edu/class/cs109/handouts/latex/". A navigation menu on the left includes "Syllabus", "Honor Code", "Office Hours", "Course Reader", "Python Review", "Latex Cheat Sheet", "Fall 2022 Videos", "Midterm", and "Final". The main content area is titled "A Quick Guide to LaTeX" and contains a table of core examples. The table has two columns: "Example" and "Latex".

Example	Latex
4^{20}	<code>4^{20}</code>
x_{12}	<code>x_{12}</code>
$\sqrt{4}$	<code>\sqrt{4}</code>
$\frac{1}{2}$	<code>\frac{1}{2}</code>
code	<code>\texttt{code}</code>
text	<code>\text{text}</code>

Late Policy



Two types of extensions:

1. Grace period (1 hour)
2. Full extension (2 lecture days later)

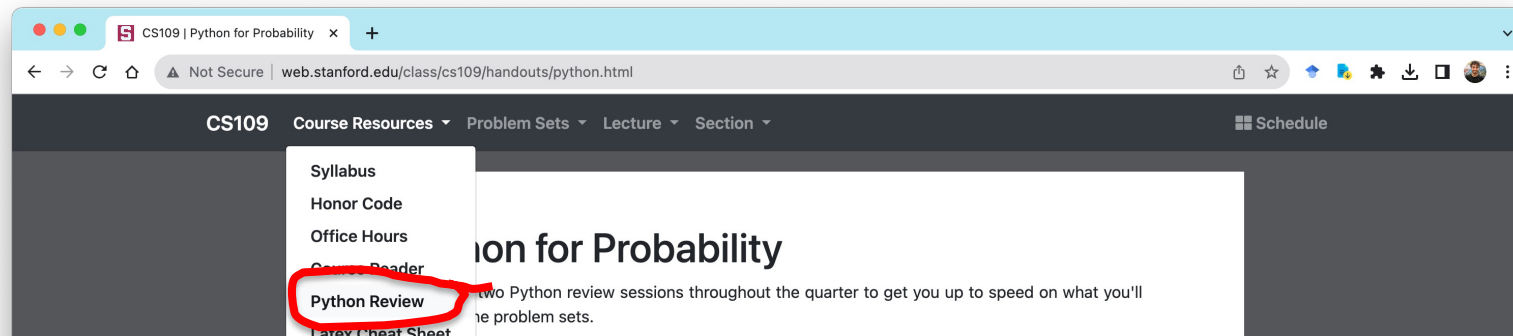
You grant them to yourself in the psetapp.

But CS109 is a fast class, and we don't want you to fall behind – after additional extensions past two, we start to cap the pset grade.

Python Review Session

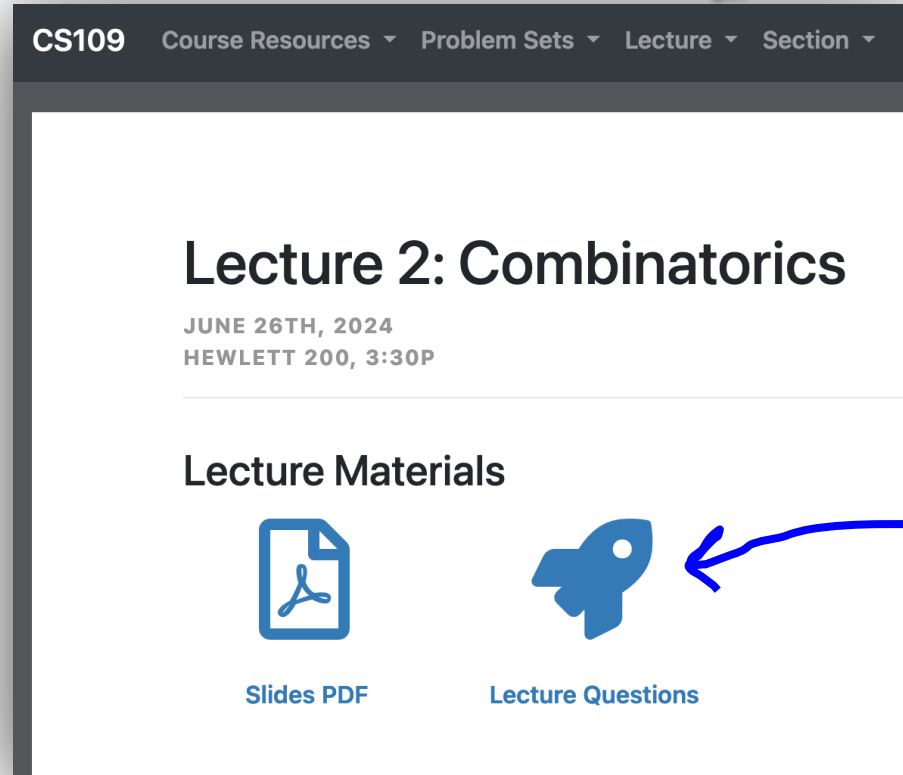
Friday at 5pm PT with Joel
(Zoom)

Find links and recordings on the course website



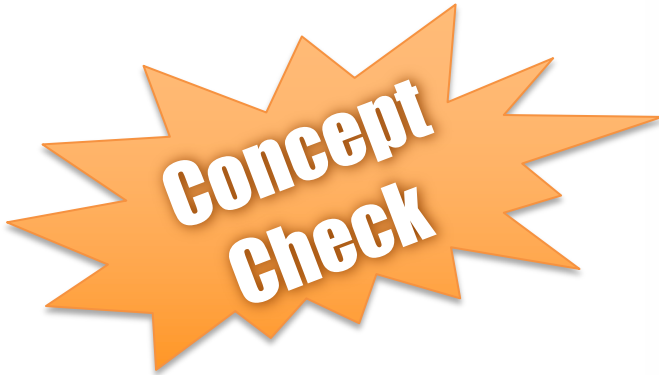
How To Get Attendance Credit

Find today's
lecture here



The screenshot shows the CS109 course page. At the top, there is a navigation bar with 'CS109' and dropdown menus for 'Course Resources', 'Problem Sets', 'Lecture', and 'Section'. The main content area is titled 'Lecture 2: Combinatorics' with the date 'JUNE 26TH, 2024' and location 'HEWLETT 200, 3:30P'. Below this, under the heading 'Lecture Materials', there are two icons: a PDF icon labeled 'Slides PDF' and a rocket icon labeled 'Lecture Questions'. A blue arrow points from the 'Lecture' dropdown menu to the 'Lecture 2: Combinatorics' title. Another blue arrow points from the 'Lecture Questions' icon to the text 'Then click here'.

Then click here



Answer each lecture's concept check question in the psetapp – linked on lecture page

Section Signups Are Due Tonight



Back to Learning!

**Probabilistic
Models**

**Uncertainty
Theory**

**Machine
Learning**

**Random
Variables**

**Core
Probability**

Counting

The Journey of CS109:

CS109: From Counting to Machine Learning



Counting
Theory



Core
Probability



Random
Variables



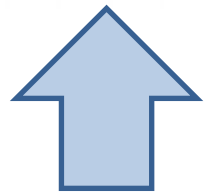
Probabilistic
Models



Uncertainty
Theory



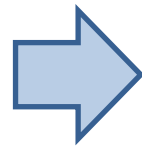
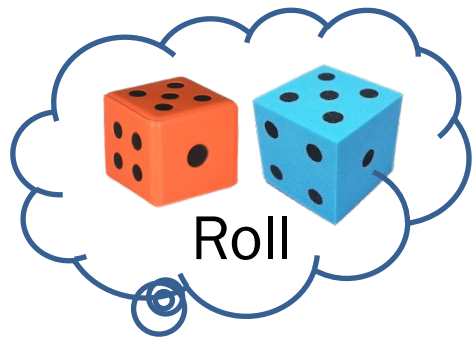
Machine
Learning



Last Lecture: Step Rule of Counting

Definition: Step Rule of Counting (aka Product Rule of Counting)

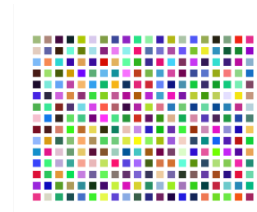
If an experiment has two parts, where the first part can result in one of m outcomes and the second part can result in one of n outcomes regardless of the outcome of the first part, then the total number of outcomes for the experiment is $m \cdot n$.



$\{(1, 1), (1, 2), (1, 3), (1, 4), (1, 5), (1, 6),$
 $(2, 1), (2, 2), (2, 3), (2, 4), (2, 5), (2, 6),$
 $(3, 1), (3, 2), (3, 3), (3, 4), (3, 5), (3, 6),$
 $(4, 1), (4, 2), (4, 3), (4, 4), (4, 5), (4, 6),$
 $(5, 1), (5, 2), (5, 3), (5, 4), (5, 5), (5, 6),$
 $(6, 1), (6, 2), (6, 3), (6, 4), (6, 5), (6, 6)\}$



(a) 12 million pixels



(b) 300 pixels

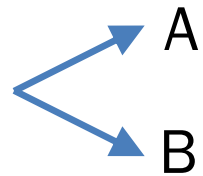


(c) 12 pixels

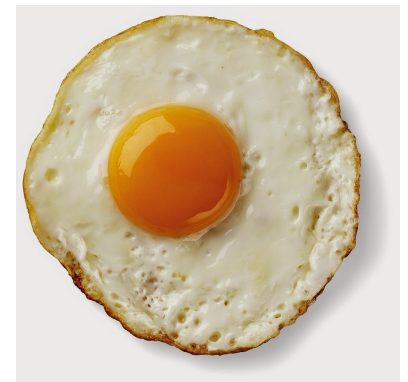
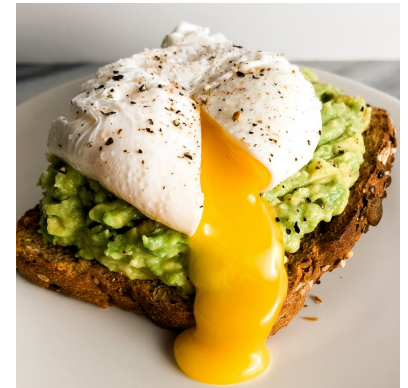
Tip: Associate every rule with example problems that fit the rule

The “Or” Rule, Part 1

One experiment



- If the outcome of an experiment can be either from
 - Set A , where $|A| = m$,
 - or Set B , where $|B| = n$,
 - where $A \cap B = \emptyset$ (no overlap)
- Then the number of outcomes of the experiment is
 - $|A| + |B| = m + n$.



How Many Bit Strings?

Problem: A 6-bit string (made of 1s and 0s) is sent over a network. The valid set of strings recognized by the receiver must either start with "01" or end with "10". How many such strings are there?

How Many Bit Strings?

Problem: A 6-bit string (made of 1s and 0s) is sent over a network. The valid set of strings recognized by the receiver must either start with "01" or end with "10". How many such strings are there?

Tip: Whenever you're stuck starting a counting problem, try writing out example outcomes

How Many Bit Strings?

Problem: A 6-bit string (made of 1s and 0s) is sent over a network. The valid set of strings recognized by the receiver must either start with "01" or end with "10". How many such strings are there?

010000
010001
010010
010011
010100
010101
010110
010111
011000
011001
011010
011011
011100
011101
011110
011111

Set *A*

000010
000110
001010
001110
010010
010110
011010
011110
100010
100110
101010
101110
110010
110110
111010
111110

Set *B*

How Many Bit Strings?

Problem: A 6-bit string (made of 1s and 0s) is sent over a network. The valid set of strings recognized by the receiver must either start with "01" or end with "10". How many such strings are there?

2^4 start with 01

010000
010001
010010
010011
010100
010101
010110
010111
011000
011001
011010
011011
011100
011101
011110
011111

Set *A*

2^4 end with 10

000010
000110
001010
001110
010010
010110
011010
011110
100010
100110
101010
101110
110010
110110
111010
111110

Set *B*

How Many Bit Strings?

Problem: A 6-bit string (made of 1s and 0s) is sent over a network. The valid set of strings recognized by the receiver must either start with "01" or end with "10". How many such strings are there?

2^4 start with 01

010000
010001
010010
010011
010100
010101
010110
010111
011000
011001
011010
011011
011100
011101
011110
011111

Set *A*

2^4 end with 10

000010
000110
001010
001110
010010
010110
011010
011110
100010
100110
101010
101110
110010
110110
111010
111110

Set *B*

How Many Bit Strings?

Problem: A 6-bit string (made of 1s and 0s) is sent over a network. The valid set of strings recognized by the receiver must either start with "01" or end with "10". How many such strings are there?

Answer

$$\begin{aligned} N &= |A| + |B| - |A \text{ and } B| \\ &= 16 + 16 - 4 \\ &= 28 \end{aligned}$$

2^4 start with 01

010000
010001
010010
010011
010100
010101
010110
010111
011000
011001
011010
011011
011100
011101
011110
011111

Set *A*

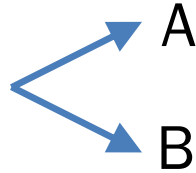
2^4 end with 10

000010
000110
001010
001110
010010
010110
011010
011110
100010
100110
101010
101110
110010
110110
111010
111110

Set *B*

The Real “Or” Rule (aka Inclusion/Exclusion)

One experiment



- If the outcome of an experiment can be either from
 - Set A ,
 - or Set B ,
 - where $A \cap B$ *might not be empty*,
- Then the number of outcomes of the experiment is
 - $N = |A| + |B| - |A \cap B|$.

The Core Counting Rules

Counting with steps

Definition: Step Rule of Counting (aka Product Rule of Counting)

If an experiment has two parts, where the first part can result in one of m outcomes and the second part can result in one of n outcomes regardless of the outcome of the first part, then the total number of outcomes for the experiment is $m \cdot n$.

Counting with “or”

Definition: Inclusion Exclusion Counting

If the outcome of an experiment can either be drawn from set A or set B , and sets A and B may potentially overlap (i.e., it is not the case that A and B are mutually exclusive), then the number of outcomes of the experiment is $|A \text{ or } B| = |A| + |B| - |A \text{ and } B|$.

CS109	CS190	CS019	CS091	CS910	CS901	C1S09	C1S90
C10S9	C109S	C19S0	C190S	C0S19	C0S91	C01S9	C019S
C09S1	C091S	C9S10	C9S01	C91S0	C910S	C90S1	C901S
SC109	SC190	SC019	SC091	SC910	SC901	S1C09	S1C90
S10C9	S109C	S19C0	S190C	S0C19	S0C91	S01C9	S019C
S09C1	S091C	S9C10	S9C01	S91C0	S910C	S90C1	S901C
1CS09	1CS90	1C0S9	1C09S	1C9S0	1C90S	1SC09	1SC90
1S0C9	1S09C	1S9C0	1S90C	10C99	10C9S	10S29	10S9C
109CS	109SC	19CS0	19C0S	19SC0	19S0C	190CS	190SC
0CS19	0CS91	0C1S9	0C19S	0C9S1	0C91S	0SC19	0SC91
0S1C9	0S19C	0S9C1	0S91C	01CS9	01C9S	01SC9	01S9C
019CS	019SC	09CS1	09C1S	09SC1	09S1C	091CS	091SC
9CS10	9CS01	9C1S0	9C10S	9C0S1	9C01S	9SC10	9SC01
9S1C0	9S10C	9S0C1	9S01C	91CS0	91C0S	91SC0	91S0C
910CS	910SC	90CS1	90C1S	90SC1	90S1C	901CS	901SC

Permutations

Counting Possible Orderings

How many letter orderings are possible for the following string?

CS109

All Possible Orderings Of Characters

CS109	CS190	CS019	CS091	CS910	CS901	C1S09	C1S90
C10S9	C109S	C19S0	C190S	C0S19	C0S91	C01S9	C019S
C09S1	C091S	C9S10	C9S01	C91S0	C910S	C90S1	C901S
SC109	SC190	SC019	SC091	SC910	SC901	S1C09	S1C90
S10C9	S109C	S19C0	S190C	S0C19	S0C91	S01C9	S019C
S09C1	S091C	S9C10	S9C01	S91C0	S910C	S90C1	S901C
1CS09	1CS90	1C0S9	1C09S	1C9S0	1C90S	1SC09	1SC90
1S0C9	1S09C	1S9C0	1S90C	10CS9	10C9S	10SC9	10S9C
109CS	109SC	19CS0	19C0S	19SC0	19S0C	190CS	190SC
0CS19	0CS91	0C1S9	0C19S	0C9S1	0C91S	0SC19	0SC91
0S1C9	0S19C	0S9C1	0S91C	01CS9	01C9S	01SC9	01S9C
019CS	019SC	09CS1	09C1S	09SC1	09S1C	091CS	091SC
9CS10	9CS01	9C1S0	9C10S	9C0S1	9C01S	9SC10	9SC01
9S1C0	9S10C	9S0C1	9S01C	91CS0	91C0S	91SC0	91S0C
910CS	910SC	90CS1	90C1S	90SC1	90S1C	901CS	901SC

Tip: Think about a generative story...

Counting Possible Orderings



Counting Possible Orderings



Step 1:
Chose 1st character

Step 2:
Chose 2nd character

Step 3:
Chose 3rd character

Step 4:
Chose 4th character

Step 5:
Chose 5th character

Counting Possible Orderings

(5 options)

9



Step 1:
Chose 1st character

Step 2:
Chose 2nd character

Step 3:
Chose 3rd character

Step 4:
Chose 4th character

Step 5:
Chose 5th character

Counting Possible Orderings

(5 options)

(4 options)



Step 1:

Chose 1st character

Step 2:

Chose 2nd character

Step 3:

Chose 3rd character

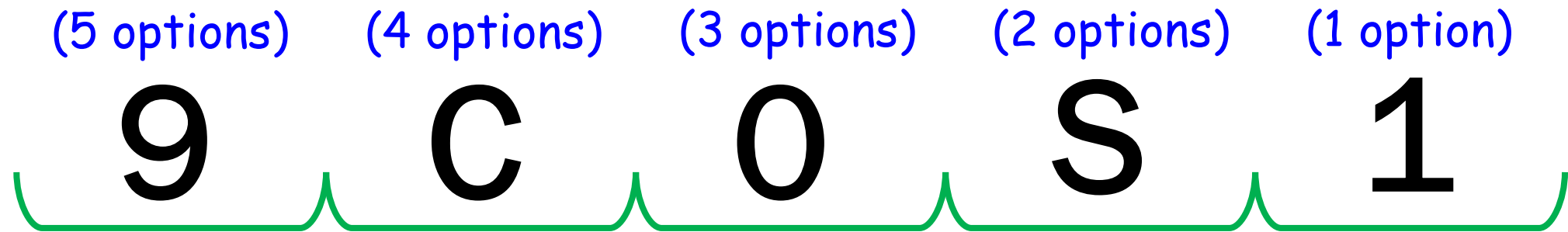
Step 4:

Chose 4th character

Step 5:

Chose 5th character

Counting Possible Orderings



Step 1:
Chose 1st character

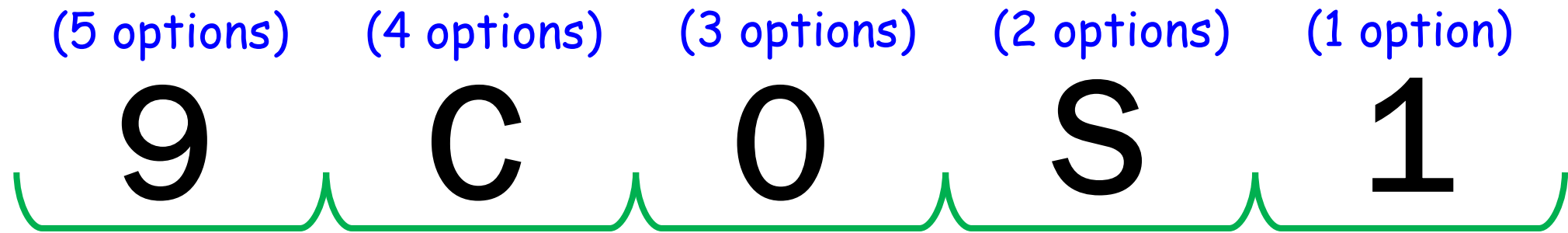
Step 2:
Chose 2nd character

Step 3:
Chose 3rd character

Step 4:
Chose 4th character

Step 5:
Chose 5th character

Counting Possible Orderings



Step 1:
Chose 1st character

Step 2:
Chose 2nd character

Step 3:
Chose 3rd character

Step 4:
Chose 4th character

Step 5:
Chose 5th character

Answer: $5! 5 \times 4 \times 3 \times 2 \times 1$

Possible Orderings Are Called Permutations

Permutation: any *ordered* arrangement of objects.

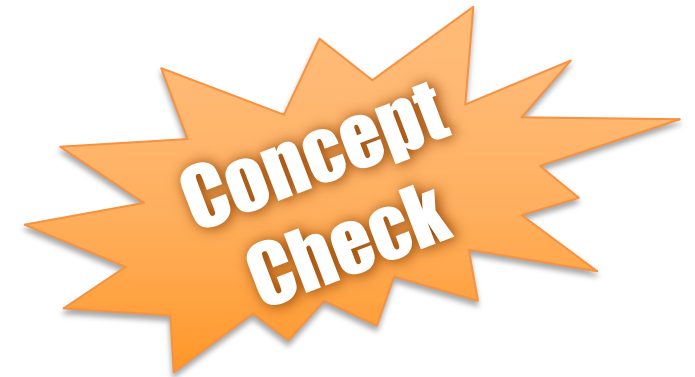
The number of unique orderings (**permutations**) of n distinct objects is:

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

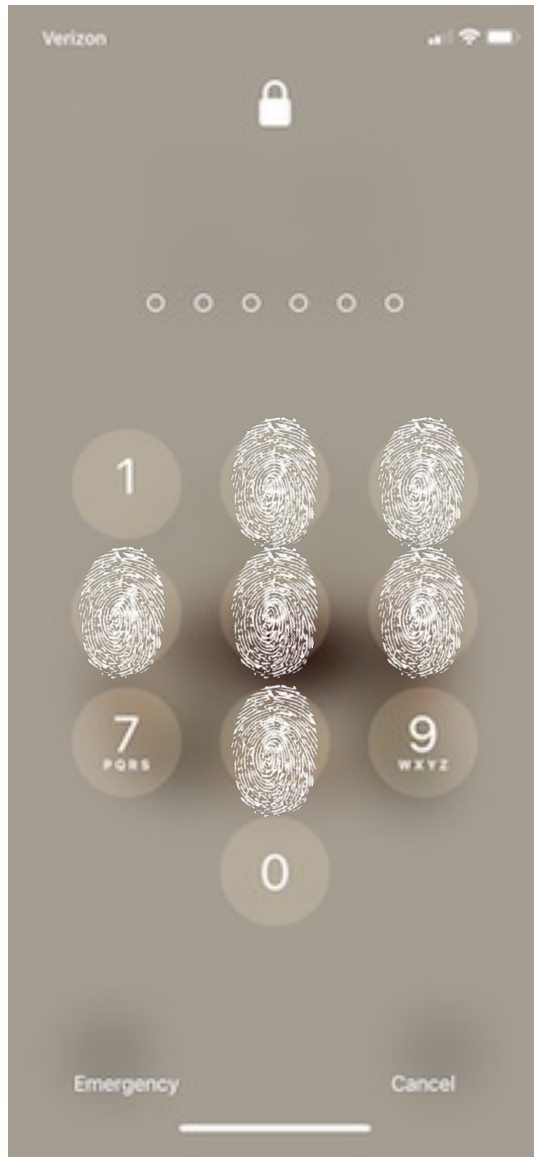
Practice: Unique 6-digit passcodes with **six** smudges



How many unique 6-digit passcodes are possible if each passcode uses **six** distinct numbers?



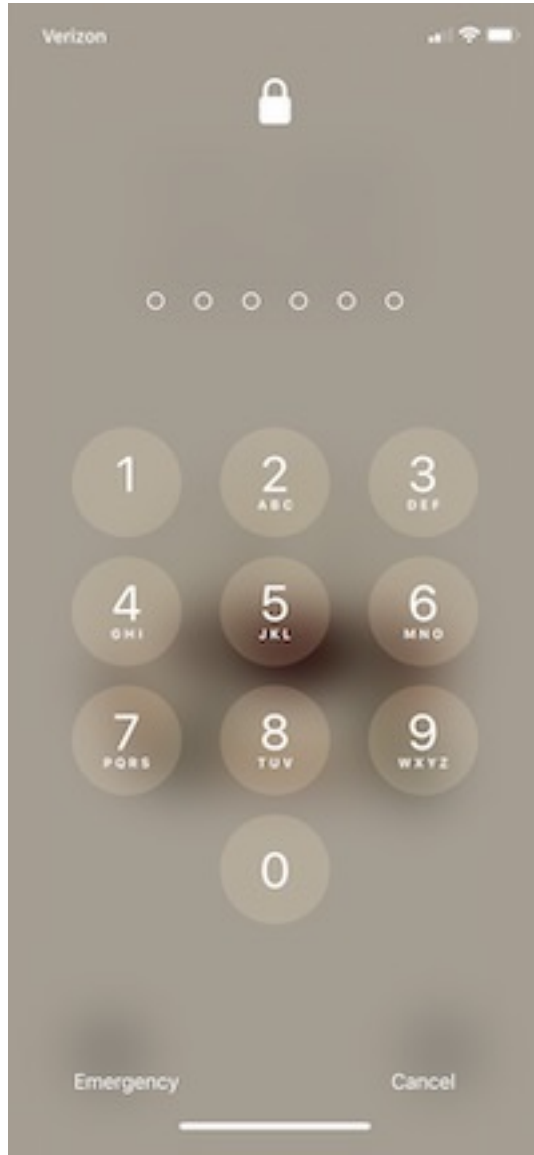
Practice: Unique 6-digit passcodes with **six** smudges



How many unique 6-digit passcodes are possible if each passcode uses **six** distinct numbers?

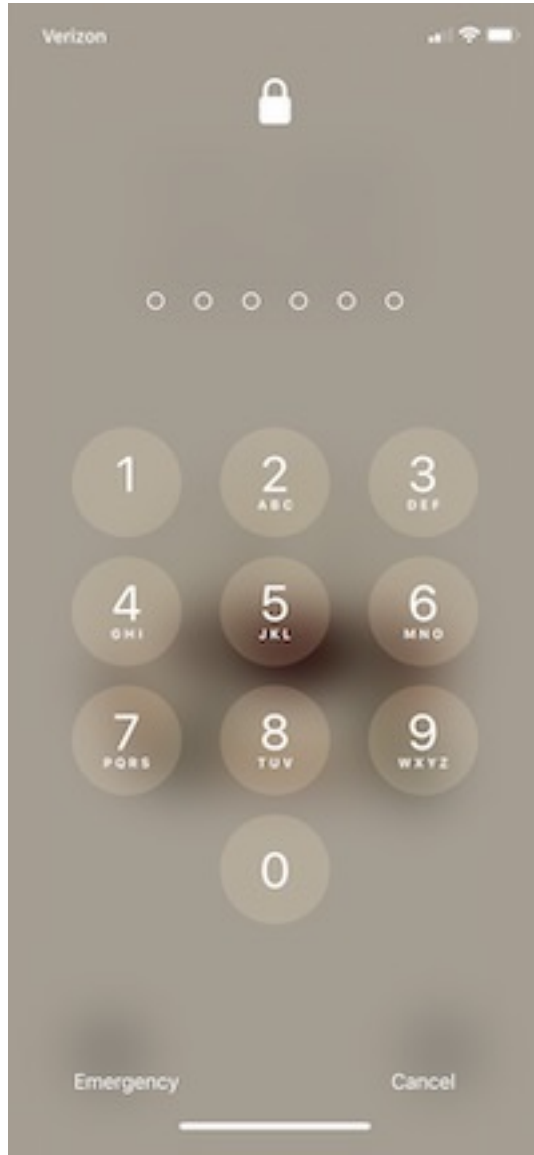
$$6! = 720 \text{ passcodes}$$

Challenge: Unique 6-digit passcodes with *six mystery* smudges



How many unique passcodes are possible if each passcode is *any* 6 unique digits?

Challenge: Unique 6-digit passcodes with **six mystery** smudges



How many unique passcodes are possible if each passcode is **any** 6 unique digits?

$$10 \times 9 \times 8 \times 7 \times 6 \times 5 \\ = \frac{10!}{4!} = 151200 \text{ passcodes}$$

Combinatorics: Formulas For Common Counting Tasks

Counting tasks on n objects

Sort objects
(permutations)

Choose k objects
(combinations)

Put objects in r
buckets

Distinct
(distinguishable)

$n!$

Combinatorics: Formulas For Common Counting Tasks

Counting tasks on n objects

Sort objects
(permutations)

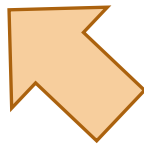
Choose k objects
(combinations)

Put objects in r
buckets

Distinct
(distinguishable)

Some
distinct

$n!$



Permutations With Some Distinct Objects



How many *different* orderings of letters are possible for the string “boba”?

boba, abob, obba...



Permutations With Some Distinct Objects



How many *different* orderings of letters are possible for the string “boba”?

boba	obba	bboa	abob
boab	obab	bbao	abbo
bboa	obba	boba	aobb
bbao	obab	boab	aobb
baob	oabb	babo	abbo
babo	oabb	baob	abob

Are there $4! = 24$ unique permutations?

Permutations With Some Distinct Objects



How many *different* orderings of letters are possible for the string “boba”?

boba	obba	bboa	abob
boab	obab	bbao	abbo
bboa	obba	boba	aobb
bbao	obab	boab	aobb
baob	oabb	babo	abbo
babo	oabb	baob	abob

Are there $4! = 24$ unique permutations?

No!

We're overcounting by a multiple of 2

Permutations With Some Distinct Objects



How many *different* orderings of letters are possible for the string BOBA?

Tip: Starting with a number that you know over- or under-counts, then correcting the difference, is a perfectly valid strategy

$$\frac{4!}{2} = 12$$

Can We Code It? Yes We Can!

baob
bbao
obba
oabb
boab
babo
abbo
aobb
boba
abob
bboa
obab

```
import itertools

def list_outcomes():
    letters = ['b', 'o', 'b', 'a']
    perms = set(itertools.permutations(letters))
    for perm in perms:
        perm_as_string = "".join(perm)
        print(perm_as_string)

import math

def count_outcomes():
    numerator = math.factorial(4)
    denominator = math.factorial(2)
    print(numerator / denominator)
```

Back To Unique Bit Strings: How Many Permutations?

1 0 1 0 0

Back To Unique Bit Strings: How Many Permutations?

1

0

1

0

0



How many ways can we sort coke cans?



Coke



Coke0



Coke



Coke0



Coke0

How many ways can we sort n distinct objects?



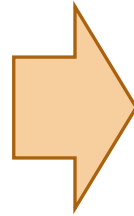
How many ways can we sort n distinct objects?



of permutations = $5!$ $5 \times 4 \times 3 \times 2 \times 1$

Then, how do we sort semi-distinct objects?

All Distinct



Some Indistinct



5! is an overcount... How do we correct it?

General approach to counting permutations

When there are n objects, and

n_1 are the same (indistinguishable or **indistinct**),

n_2 are the same,

...

n_r are the same,

The number of unique permutations is

$$\frac{n!}{n_1! n_2! \cdots n_r!}$$

General approach to counting permutations

When there are n objects, and

n_1 are the same (indistinguishable or **indistinct**),

n_2 are the same,

...

n_r are the same,

The number of unique permutations is

$$\frac{n!}{n_1! n_2! \cdots n_r!}$$

For each group of indistinct objects, divide by the overcounted permutations.

Permutations of Semi-Distinct Objects

Order n semi-
distinct objects $\frac{n!}{n_1! n_2! \cdots n_r!}$



Coke



Coke0



Coke



Coke0



Coke0

Permutations of Semi-Distinct Objects

Order n semi-
distinct objects $\frac{n!}{n_1! n_2! \cdots n_r!}$



Coke



Coke0



Coke



Coke0



Coke0

$$\frac{5!}{2!3!} = 10$$

Boss Battle: String Permutations

$$\text{Order } n \text{ semi-} \quad \frac{n!}{n_1! n_2! \cdots n_r!}$$

distinct objects

How many letter orderings are possible for the following string?

MISSISSIPPI

Boss Battle: String Permutations

$$\text{Order } n \text{ semi-} \quad \frac{n!}{n_1! n_2! \cdots n_r!}$$

distinct objects

How many letter orderings are possible for the following string?

MISSISSIPPI

$$\frac{11!}{1!4!4!2!} = 34,650$$

Combinatorics: Formulas For Common Counting Tasks

Counting tasks on n objects

Sort objects
(permutations)

Choose k objects
(combinations)

Put objects in r
buckets

Distinct
(distinguishable)

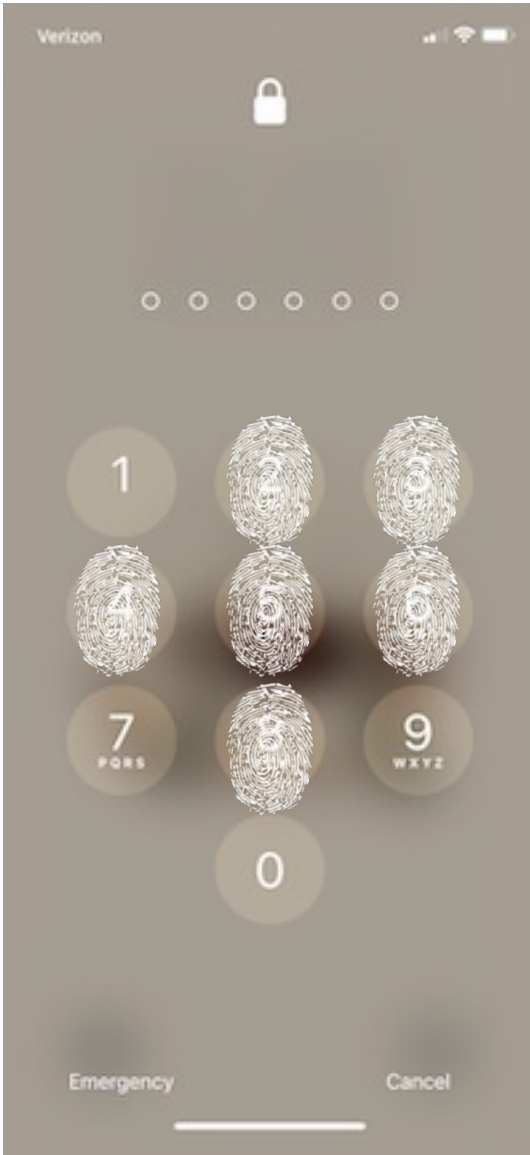
Some
distinct

$$n!$$

$$\frac{n!}{n_1! n_2! \cdots n_r!}$$

Unique 6-digit passcodes with **six** smudges

Order n semi-
distinct objects $\frac{n!}{n_1! n_2! \cdots n_r!}$

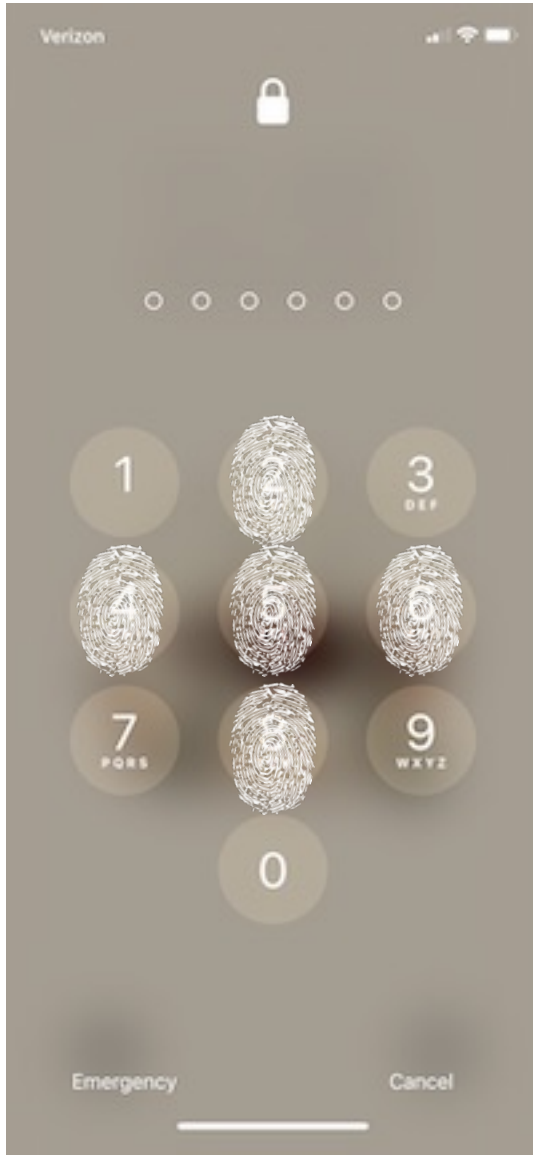


How many unique 6-digit passcodes are possible if each passcode uses **six** distinct numbers?

$$6! = 720 \text{ passcodes}$$

Unique 6-digit passcodes with **five** smudges

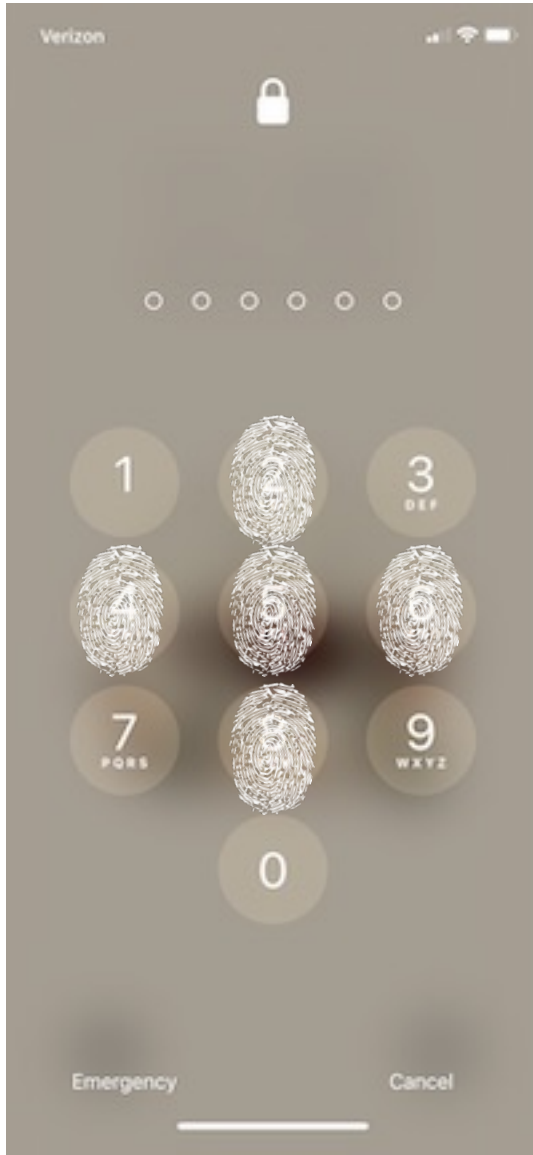
Order n semi-distinct objects $\frac{n!}{n_1! n_2! \cdots n_r!}$



How many unique 6-digit passcodes are possible if each passcode uses exactly **five** distinct numbers?

Unique 6-digit passcodes with **five** smudges

Order n semi-distinct objects $\frac{n!}{n_1! n_2! \cdots n_r!}$



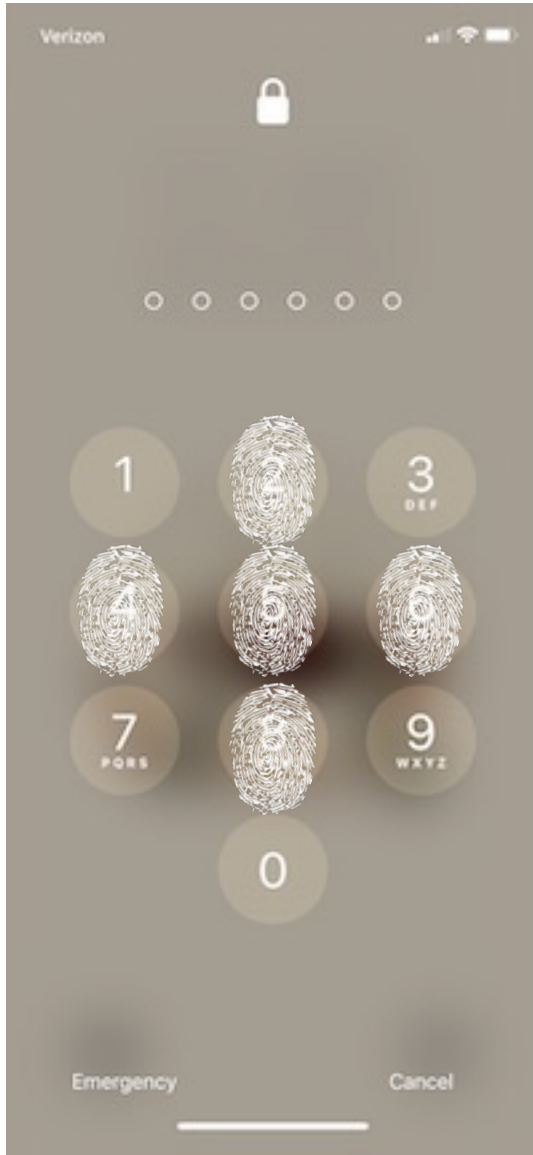
How many unique 6-digit passcodes are possible if each passcode uses exactly **five** distinct numbers?

Steps:

1. Choose digit to repeat 5 outcomes
2. Create passcode (sort 6 digits:
4 distinct, 2 indistinct)

Unique 6-digit passcodes with **five** smudges

Order n semi-distinct objects $\frac{n!}{n_1! n_2! \cdots n_r!}$



How many unique 6-digit passcodes are possible if each passcode uses exactly **five** distinct numbers?

Steps:

1. Choose digit to repeat 5 outcomes
2. Create passcode (sort 6 digits:
4 distinct, 2 indistinct)

$$5 \times \frac{6!}{2!} = 1,800 \text{ passcodes}$$



Combinations

Combinatorics: Formulas For Common Counting Tasks

Counting tasks on n objects

Sort objects
(permutations)

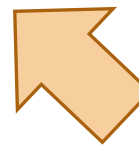
Choose k objects
(combinations)

Put objects in r
buckets

Distinct
(distinguishable)

Some
distinct

Distinct



$$n!$$

$$\frac{n!}{n_1! n_2! \cdots n_r!}$$

Combinations with cake

There are $n = 20$ people.

How many ways can we **choose** $k = 5$ people to get cake?

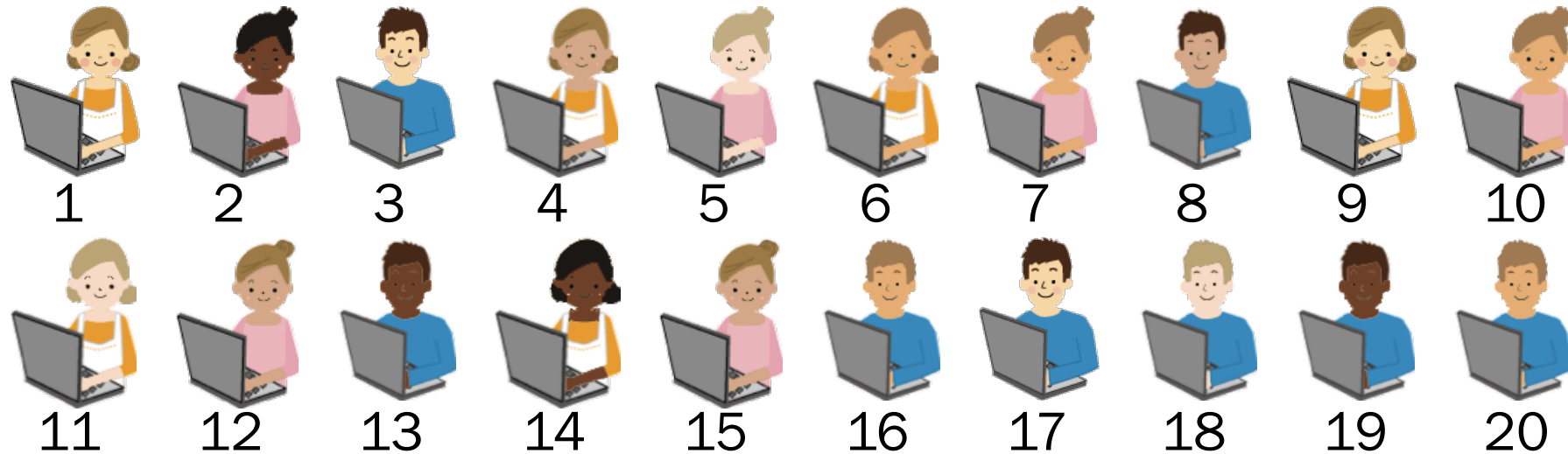


Think about a generative story...

Combinations with cake

There are $n = 20$ people.

How many ways can we **choose** $k = 5$ people to get cake?



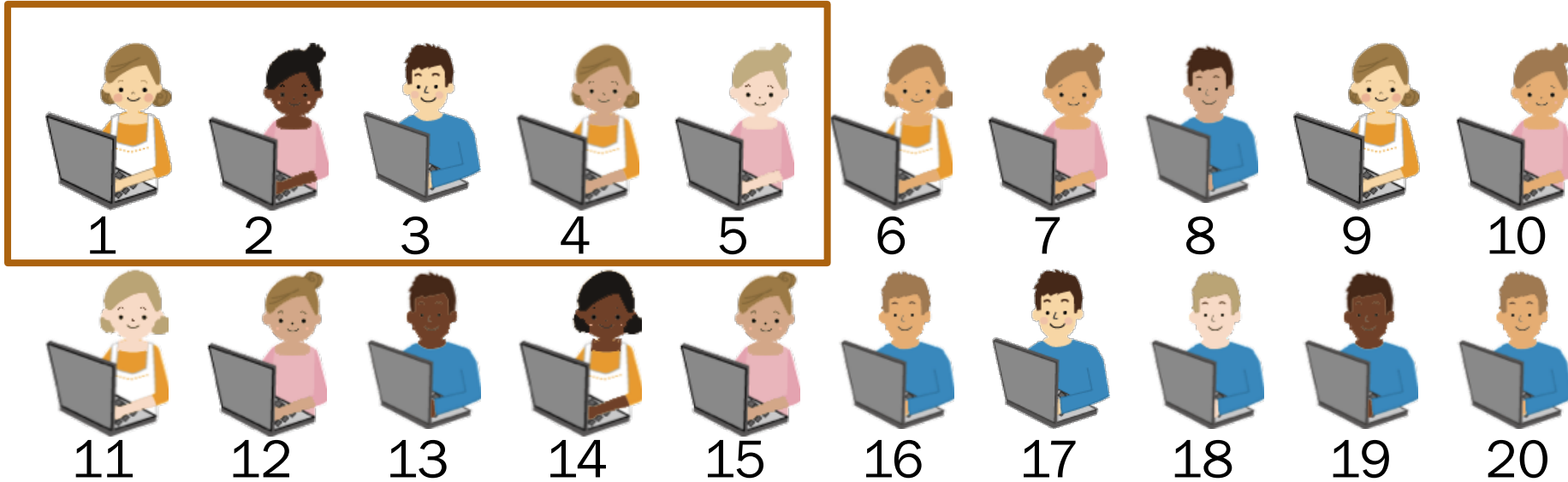
1. n people get in line

$n!$ ways

Combinations with cake

There are $n = 20$ people.

How many ways can we **choose** $k = 5$ people to get cake?



1. n people
get in line

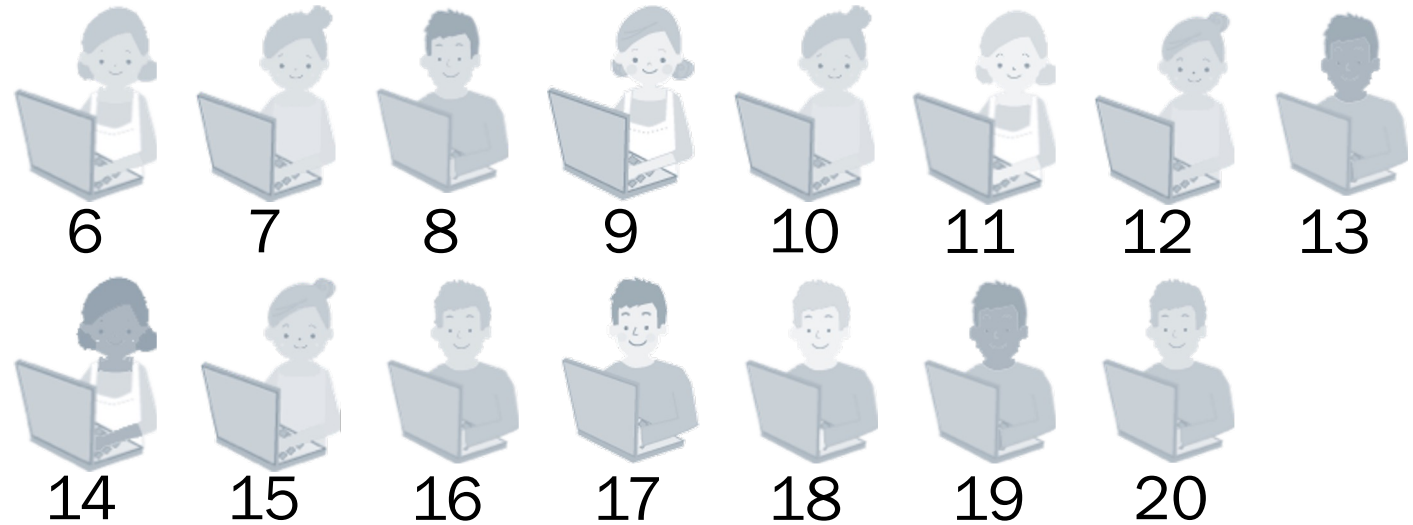
$n!$ ways

2. Put first k
in cake group

Combinations with cake

There are $n = 20$ people.

How many ways can we **choose** $k = 5$ people to get cake?



1. n people
get in line

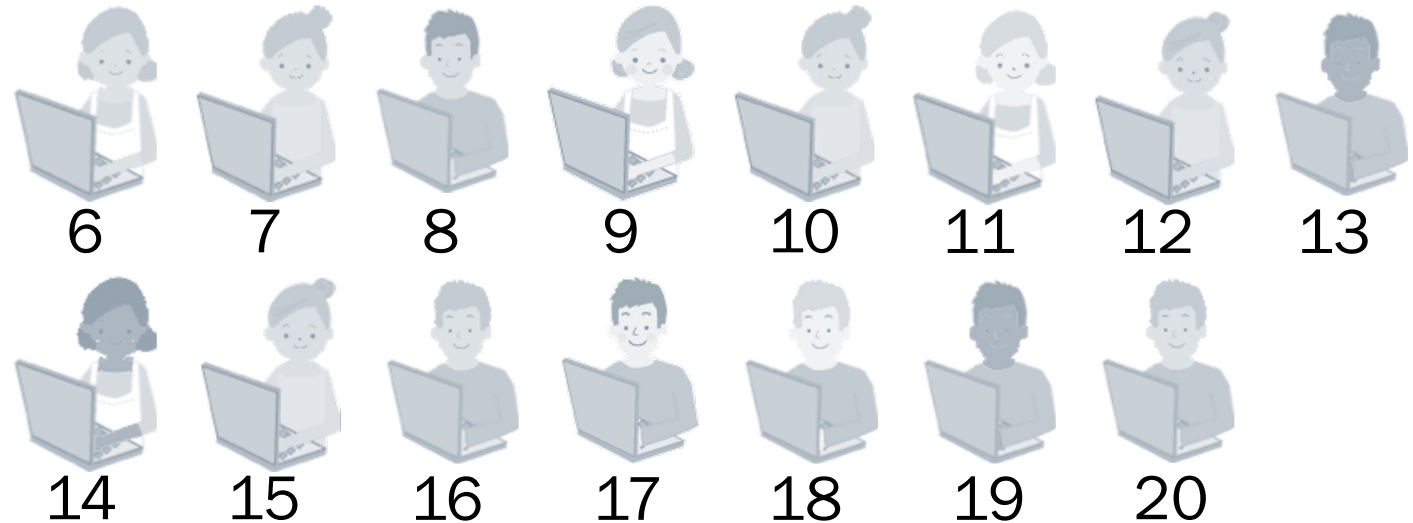
$n!$ ways

2. Put first k
in cake group

Combinations with cake

There are $n = 20$ people.

How many ways can we **choose** $k = 5$ people to get cake?



1. n people
get in line

$n!$ ways

2. Put first k
in cake group

3. **Un-order the
cake group**

$k!$ different
permutations

Combinations with cake

There are $n = 20$ people.

How many ways can we **choose** $k = 5$ people to get cake?



1. n people
get in line

$n!$ ways

2. Put first k
in cake group

3. Un-order the
cake group

$k!$ different
permutations

4. Un-order the
non-cake group

$(n - k)!$ different
permutations

Combinations

A **combination** is an unordered selection of k objects from a set of n **distinct** objects.

The number of ways of making this selection is

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

Combinations

A **combination** is an unordered selection of k objects from a set of n **distinct** objects.

The number of ways of making this selection is

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

1. Order n distinct objects

2. Correct for overcounting: any ordering of the chosen group is the same

3. Correct for overcounting: any ordering of the not-chosen group is the same

Combinations

A **combination** is an **unordered** selection of k objects from a set of n **distinct** objects.

The number of ways of making this selection is

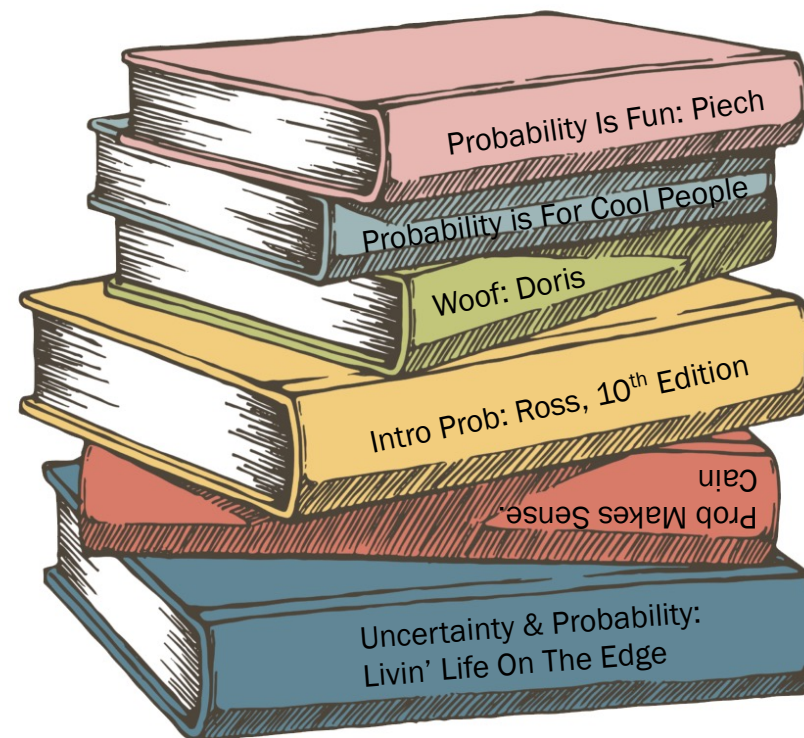
$$\frac{n!}{k!(n-k)!} = n! \times \frac{1}{k!} \times \frac{1}{(n-k)!} = \binom{n}{k}$$



Practice: Combinations of Books

Choose k of
 n distinct objects $\binom{n}{k} = \frac{n!}{k!(n-k)!}$

How many ways are there to choose 3 books from a set of 6 distinct books?

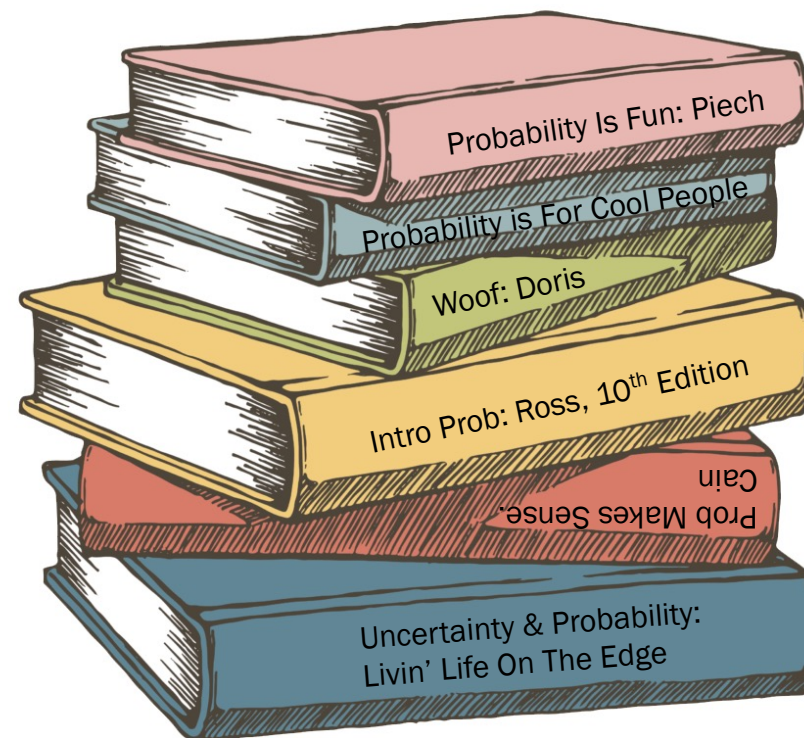


Practice: Combinations of Books

Choose k of
 n distinct objects $\binom{n}{k} = \frac{n!}{k!(n-k)!}$

How many ways are there to choose 3 books from a set of 6 distinct books?

$$\binom{6}{3} = \frac{6!}{3!3!} = 20 \text{ ways}$$



Can We Code It? Yes We Can

How many unique hands of 5 cards are there in a 52 card deck?



Can We Code It? Yes We Can

How many unique hands of 5 cards are there in a 52 card deck?



$$\binom{52}{5}$$

Can We Code It? Yes We Can

How many unique hands of 5 cards are there in a 52 card deck?



```
def main():  
    total = math.comb(52, 5)  
    print(total)
```

```
def main():  
    cards = make_deck()  
    all_hands = itertools.combinations(cards, 5)  
    for hand in all_hands:  
        print(hand)
```

The image depicts a large number of rectangular bins, each filled with a different color of small, spherical balls. The bins are arranged in a grid-like pattern, and the balls are scattered throughout the scene, some floating in the air above the bins. The colors of the balls include red, blue, green, yellow, orange, purple, and black. The background is a light, textured surface. The text "Balls In Bins" is centered in the middle of the image in a large, black, serif font.

Balls In Bins

Combinatorics: Formulas For Common Counting Tasks

Counting tasks on n objects

Sort objects
(permutations)

Choose k objects
(combinations)

Put objects in r
buckets

Distinct
(distinguishable)

Some
distinct

Distinct

Distinct

Indistinct

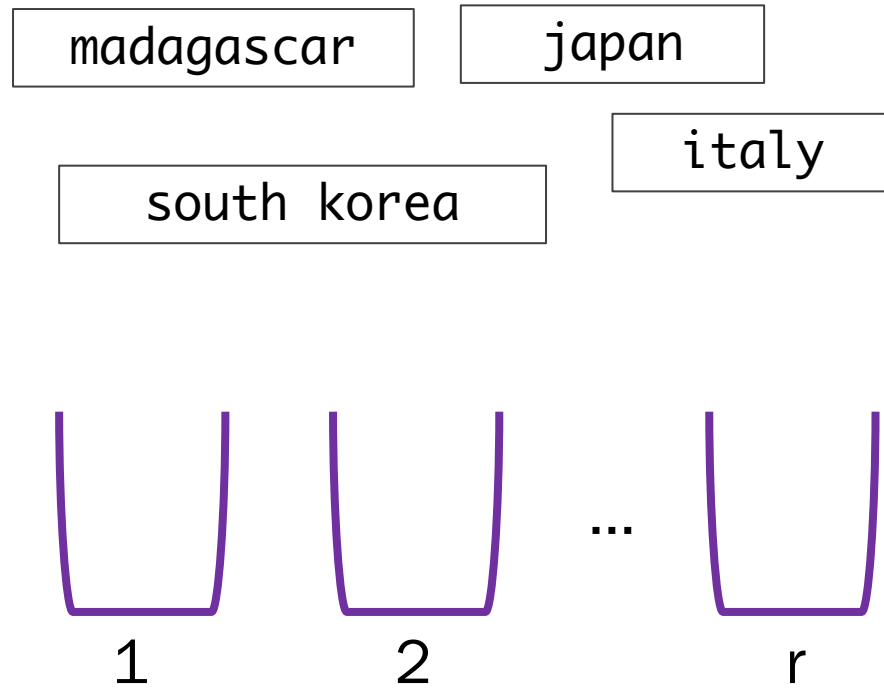
$$n!$$

$$\frac{n!}{n_1! n_2! \cdots n_r!}$$

$$\binom{n}{k}$$

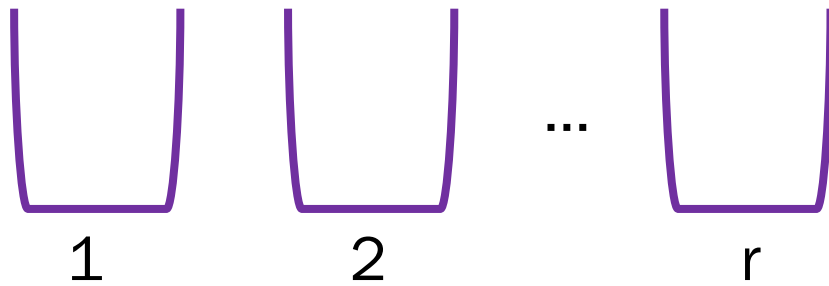
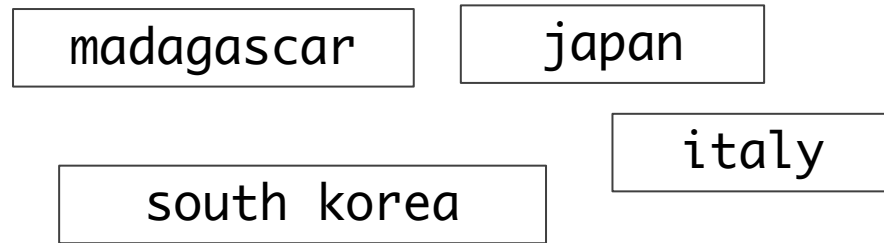
~~Balls in bins~~ Hash tables and **distinct** strings

How many ways are there to hash n **distinct** strings to r buckets?



~~Balls in bins~~ Hash tables and **distinct** strings

How many ways are there to hash n **distinct** strings to r buckets?

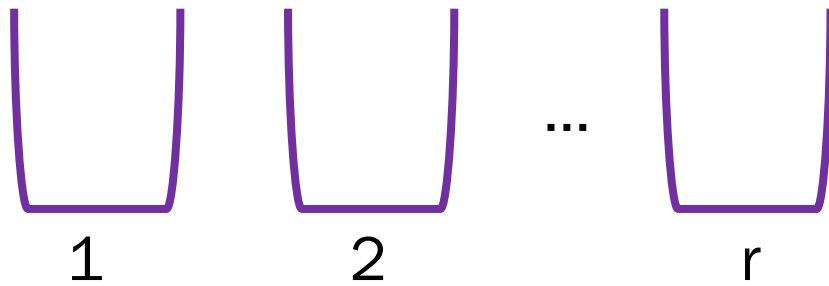
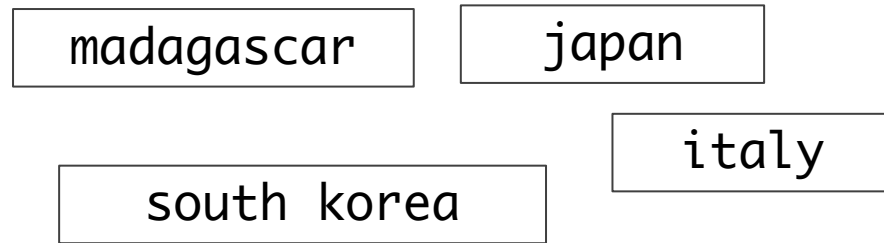


Steps:

1. Bucket 1st string – r choices
2. Bucket 2nd string – r choices
- ...
- n . Bucket n^{th} string – r choices

~~Balls in bins~~ Hash tables and **distinct** strings

How many ways are there to hash n **distinct** strings to r buckets?



Steps:

1. Bucket 1st string – r choices
2. Bucket 2nd string – r choices
- ...
- n . Bucket n^{th} string – r choices

r^n outcomes

Combinatorics: Formulas For Common Counting Tasks

Counting tasks on n objects

Sort objects
(permutations)

Choose k objects
(combinations)

Put objects in r
buckets

Distinct
(distinguishable)

Some
distinct

Distinct

Distinct

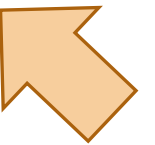
Indistinct

$$n!$$

$$\frac{n!}{n_1! n_2! \cdots n_r!}$$

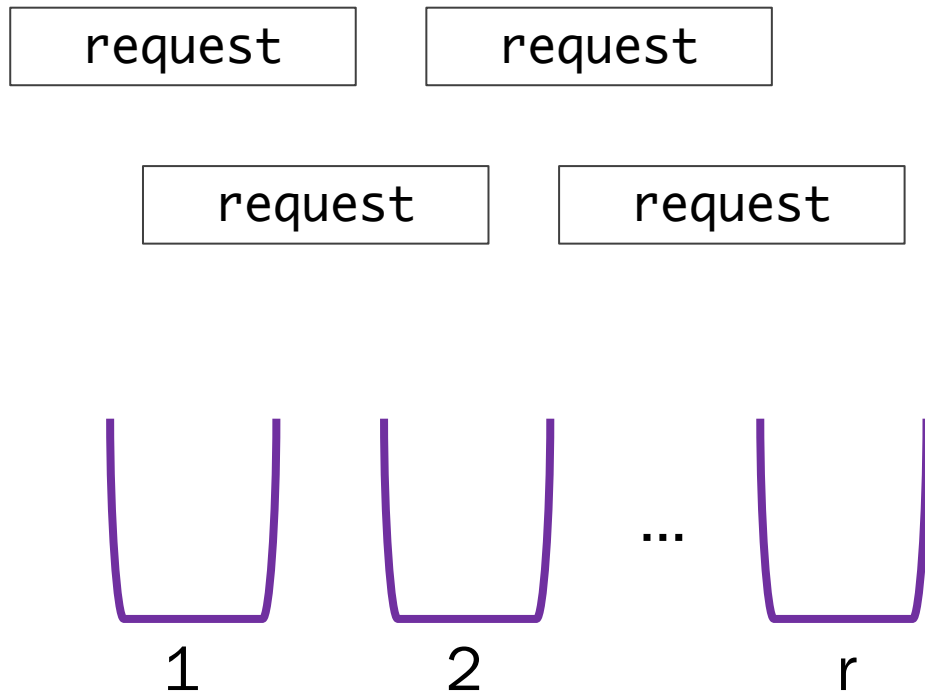
$$\binom{n}{k}$$

$$r^n$$



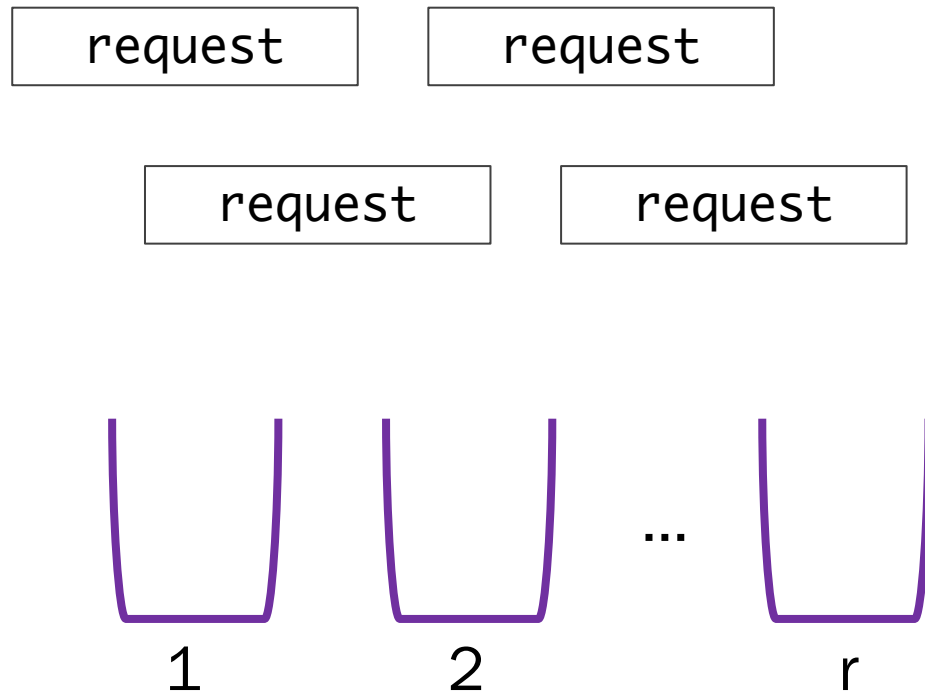
Servers and **indistinct** requests

How many ways are there to distribute n **indistinct** web requests to r servers?



Servers and **indistinct** requests

How many ways are there to distribute n **indistinct** web requests to r servers?



What does one outcome look like?

- Server 1 has x_1 requests,
- Server 2 has x_2 requests,
- ...
- Server r has x_r requests

$$\text{constraint: } \sum_{i=1}^r x_i = n$$

Flowers In Vases

How many ways can we put $n = 5$ indistinct flowers in $r = 3$ distinct vases?

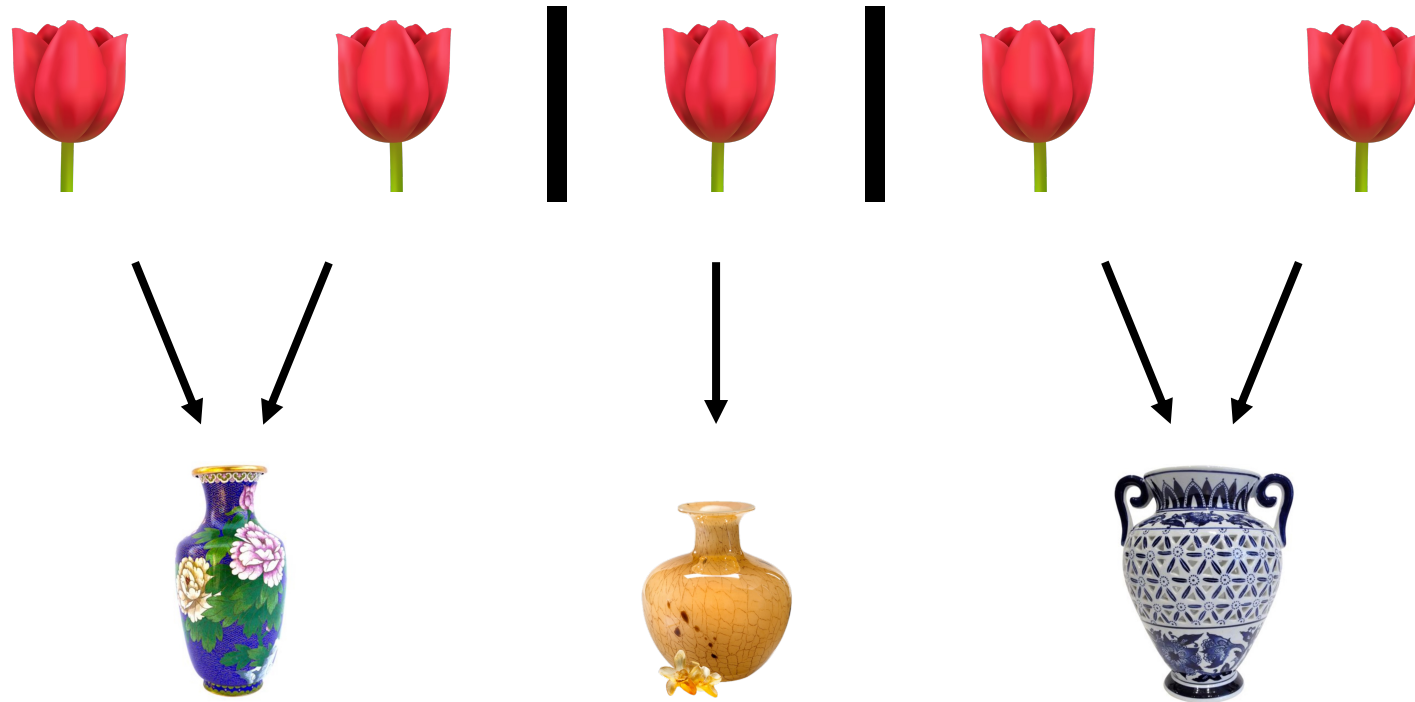


Flowers In Vases

What does one possible outcome look like? Can we build a generative story?

Flowers In Vases

What does one possible outcome look like? Can we build a generative story?



Goal: order n indistinct objects and $r - 1$ indistinct dividers.

Flowers In Vases

What does one possible outcome look like? Can we build a generative story?

Start by treating objects and dividers as distinct.



Goal: order n indistinct objects and $r - 1$ indistinct dividers.

Flowers In Vases

What does one possible outcome look like? Can we build a generative story?

Start by treating objects and dividers as distinct.



1. Order n distinct objects and $r - 1$ distinct dividers

$$(n + r - 1)!$$

Goal: order n **indistinct** objects and $r - 1$ **indistinct** dividers.

Flowers In Vases

What does one possible outcome look like? Can we build a generative story?

Start by treating objects and dividers as distinct.



1. Order n distinct objects and $r - 1$ distinct dividers

$$(n + r - 1)!$$

2. Un-order n objects

$$\frac{1}{n!}$$

Goal: order n indistinct objects and $r - 1$ indistinct dividers.

Flowers In Vases

What does one possible outcome look like? Can we build a generative story?

Start by treating objects and dividers as distinct.



1. Order n distinct objects and $r - 1$ distinct dividers

$$(n + r - 1)!$$

2. Un-order n objects

$$\frac{1}{n!}$$

3. Un-order $r - 1$ dividers

$$\frac{1}{(r - 1)!}$$

Goal: order n **indistinct** objects and $r - 1$ **indistinct** dividers.

The Divider Method: Indistinct Balls In Bins

The number of ways to distribute n indistinct objects into r buckets is

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

This is equivalent to the number of ways to permute $n + r - 1$ objects, such that n are indistinct objects, and $r - 1$ are indistinct dividers.

Integer solutions to equations

How many integer solutions are there to the following equation:

$$x_1 + x_2 + \cdots + x_r = n,$$

where for all i , x_i is an integer such that $0 \leq x_i \leq n$?

Combinatorics: Formulas For Common Counting Tasks

Counting tasks on n objects

Sort objects
(permutations)

Choose k objects
(combinations)

Put objects in r
buckets

Distinct
(distinguishable)

Some
distinct

Distinct

Distinct

Indistinct

$$n!$$

$$\frac{n!}{n_1! n_2! \cdots n_r!}$$

$$\binom{n}{k}$$

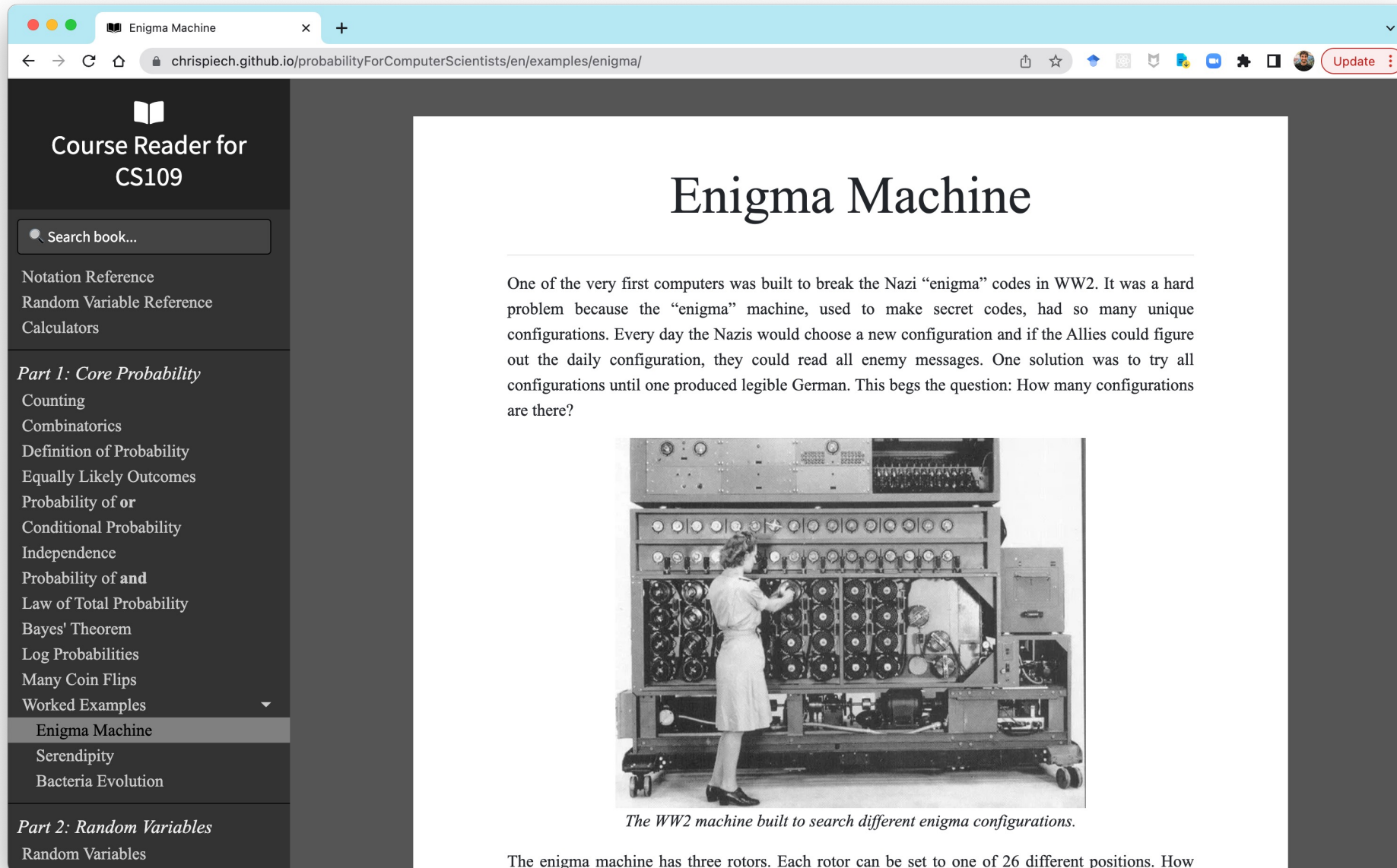
$$r^n$$

$$\binom{n+r-1}{r-1}$$

You're ready for the first 6 problems now!

The screenshot shows a web browser window with two tabs: 'CS109PsetApp - Overview' and 'Pset 1 - Counting for Probability'. The address bar shows the URL 'cs109psets.netlify.app/fall23/pset1/robot_paths'. The page title is 'PS1 Robot Paths'. On the left, a vertical sidebar contains a list of problem numbers from 1a to 14, with '6' highlighted. The main content area contains the following text: 'Imagine you have a robot (🤖) that lives on an 7 x 8 grid (it has 7 rows and 8 columns). The robot starts in cell (1, 1) and can take steps either to the right or down (no left or up steps). How many distinct paths can the robot take to the destination (🚩) in cell (7, 8)?'. Below the text is a 7x8 grid. A robot icon is in the top-left cell (row 1, column 1), and a red flag icon is in the bottom-right cell (row 7, column 8). The text '7 rows' is to the left of the grid, and '8 columns' is below the grid. On the right side of the page, there is an 'Answer Editor' section with a 'Solution' tab. Below it is a 'Numeric Answer' field with the placeholder text 'Enter your answer' and a 'Check Answer' button. Below that is an 'Explanation' section with a toolbar containing 'Block LaTeX', 'Inline LaTeX', 'Python', and 'Image' options. At the bottom of the page, there are 'Previous Question' and 'Next Question' buttons.

Want to go deeper? Cool examples in the course reader



The screenshot shows a web browser window with the address bar displaying `chrispiech.github.io/probabilityForComputerScientists/en/examples/enigma/`. The page title is "Enigma Machine". The left sidebar contains a navigation menu for "Course Reader for CS109" with a search bar and a list of topics. The main content area features the title "Enigma Machine" and a paragraph of text. Below the text is a black and white photograph of a woman in a uniform operating a large, complex mechanical device. The caption below the photo reads: "The WW2 machine built to search different enigma configurations." The bottom of the page shows the start of a paragraph: "The enigma machine has three rotors. Each rotor can be set to one of 26 different positions. How

Course Reader for CS109

Search book...

Notation Reference
Random Variable Reference
Calculators

Part 1: Core Probability

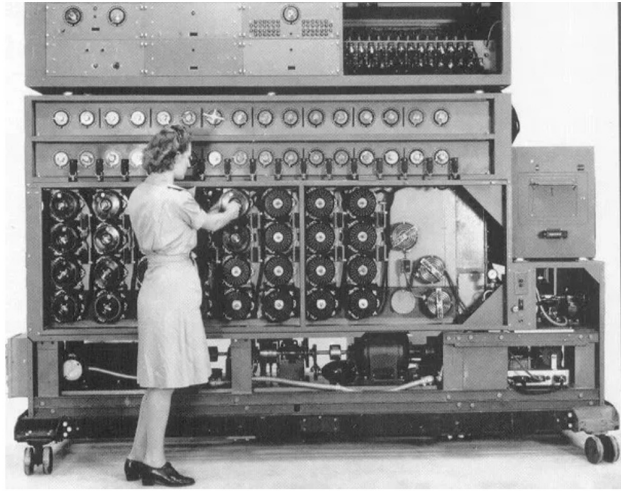
- Counting
- Combinatorics
- Definition of Probability
- Equally Likely Outcomes
- Probability of or
- Conditional Probability
- Independence
- Probability of and
- Law of Total Probability
- Bayes' Theorem
- Log Probabilities
- Many Coin Flips
- Worked Examples
 - Enigma Machine
 - Serendipity
 - Bacteria Evolution

Part 2: Random Variables

Random Variables

Enigma Machine

One of the very first computers was built to break the Nazi “enigma” codes in WW2. It was a hard problem because the “enigma” machine, used to make secret codes, had so many unique configurations. Every day the Nazis would choose a new configuration and if the Allies could figure out the daily configuration, they could read all enemy messages. One solution was to try all configurations until one produced legible German. This begs the question: How many configurations are there?



The WW2 machine built to search different enigma configurations.

The enigma machine has three rotors. Each rotor can be set to one of 26 different positions. How

Up next: going from counts to probabilities...