Before we start

If you or someone you know is suffering from food and/or housing insecurities there are UCSD resources here to help:

Basic Needs Office: https://basicneeds.ucsd.edu/

Triton Food Pantry (in the old Student Center) is free and anonymous, and includes produce:

https://www.facebook.com/tritonfoodpantry/

Mutual Aid UCSD: https://mutualaiducsd.wordpress.com/

If you find yourself in an uncomfortable situation, ask for help. We are committed to upholding University policies regarding nondiscrimination, sexual violence and sexual harassment.

Counseling and Psychological Services (CAPS) at 858 5343755 or http://caps.ucsd.edu

OPHD at (858) 534-8298, ophd@ucsd.edu , http://ophd.ucsd.edu. CARE at Sexual Assault Resource Center at 858 5345793 sarc@ucsd.edu http://care.ucsd.edu

Pandemic resilient instruction

Fall 2021 is a transition quarter so please be patient with us as we do our best to serve the needs of all students while adhering to the university guidelines. First and foremost is the health and safety of everyone. Please do not come to class if you are sick or even think you might be sick. Please reach out (minnes@eng.ucsd.edu) if you need support with extenuating circumstances.

Masks are required in class. All students who attend class must also be fully vaccinated against COVID-19 unless they have a university-approved exemption. Campus policy requires masks and daily "symptom screeners" for everyone and we expect all students to follow these rules.

Welcome to CSE 20: Discrete Math for Computer Science in Fall 2021!

Themes and applications for CSE 20

- **Technical skepticism**: Know, select and apply appropriate computing knowledge and problem-solving techniques. Reason about computation and systems. Use mathematical techniques to solve problems. Determine appropriate conceptual tools to apply to new situations. Know when tools do not apply and try different approaches. Critically analyze and evaluate candidate solutions.
- **Multiple representations**: Understand, guide, shape impact of computing on society/the world. Connect the role of Theory CS classes to other applications (in undergraduate CS curriculum and beyond). Model problems using appropriate mathematical concepts. Clearly and unambiguously communicate computational ideas using appropriate formalism. Translate across levels of abstraction.

Applications: Numbers (how to represent them and use them in Computer Science), Recommendation systems and their roots in machine learning (with applications like Netflix), "Under the hood" of computers (circuits, pixel color representation, data structures), Codes and information (secret message sharing and error correction), Bioinformatics algorithms and genomics (DNA and RNA).

Introductions

Class website: http://cseweb.ucsd.edu/classes/fa21/cse20-a

Pro-tip: the URL structure is your map to finding your course website for other CSE classes.

Pro-tip: you can use MATH109 to replace CSE20 for prerequisites and other requirements.

Instructor: Prof. Mia Minnes "Minnes" rhymes with Guinness, minnes@eng.ucsd.edu, http://cseweb.ucsd.edu/ minnes

Our team: Four TAs and 10 tutors + all of you

Fill in contact info for students around you, if you'd like:

On an average week: **MWF** Lectures + review quizzes, **T** HW due, **W** Discussion, office hours, Piazza. Project parts will be due some weeks.

All dates are on Canvas (click for link) and details are on course calendar (click for link).

Education research: CSE 20 is participating in a project on retention and sense of community in UCSD majors; see research plan. If you consent to participate in this study, no action is needed. If you DO NOT consent to participate in this study, or you choose to opt-out at any time during the a cademic year, sign and submit this form to the research contact at retentionstudy@cs.ucsd.edu.

Friday September 24

What data should we encode about each Netflix account holder to help us make effective recommendations?

In machine learning, clustering can be used to group similar data for prediction and recommendation. For example, each Netflix user's viewing history can be represented as a n-tuple indicating their preferences about movies in the database, where n is the number of movies in the database. People with similar tastes in movies can then be clustered to provide recommendations of movies for one another. Mathematically, clustering is based on a notion of distance between pairs of n-tuples.

In the table below, each row represents a user's ratings of movies: ✓ (check) indicates the person liked the movie, ✗ (x) that they didn't, and • (dot) that they didn't rate it one way or another (neutral rating or didn't watch).

Person	Fyre	Frozen II	Picard	Ratings written as a 3-tuple
P_1	✗	•	✓	
P_2	✓	✓	✗	
P_3	✓	✓	✓	
P_4	•	✗	✓	

Conclusion: Modeling involves choosing data types to represent and organize data

Review: Week 0 Friday

1. Please complete the beginning of the quarter survey https://forms.gle/gvibFnNixxqcWbaU8
2. We want you to be familiar with class policies and procedures so you are ready to have a successful quarter. Please take a look at the class website http://cseweb.ucsd.edu/classes/fa21/cse20-a and answer the questions about it on Gradescope.
3. Modeling:
   (a) Using the example movie database from class with the 3 movies Fyre, Frozen II, Picard, which of the following is a 3-tuple that represents the ratings of a user who liked Frozen II? (Select all and only that apply.)
      i. 1
      ii. $(0, 0, 0)$
      iii. $[1, 1, 1]$
      iv. $\{-1, 0, 1\}$
      v. $(1, -1, 0)$
      vi. $(0, 1, 1)$
      vii. $(1, 1, 1, 1)$

   (b) Using the example movie database from class with the 3 movies Fyre, Frozen II, Picard, how many distinct (different) 3-tuples of ratings are there?

Monday September 27

Notation and prerequisites

Term	Notation Example(s)	We say in English ...
sequence	$x_1, \ldots, x_n$	A sequence x_1 to x_n
	$x_1, \ldots, x_n$ where $n = 0$	An empty sequence
	$x_1, \ldots, x_n$ where $n = 1$	A sequence containing just x_1
	$x_1, \ldots, x_n$ where $n = 2$	A sequence containing just x_1 and x_2 in order
	x_1, x_2	A sequence containing just x_1 and x_2 in order
all integers	$\mathbb{Z}$	The (set of all) integers (whole numbers including negatives, zero, and positives)
all positive integers	$\mathbb{Z}^+$	The (set of all) strictly positive integers
all natural numbers	$\mathbb{N}$	The (set of all) natural numbers. **Note**: we use the convention that 0 is a natural number.
function rule definition	$f(x) = x + 4$	Define f of x to be $x + 4$
piecewise rule definition	$f(x) = \begin{cases} x & \text{if } x \geq 0 \\ -x & \text{if } x < 0 \end{cases}$	Define f of x to be x when x is nonnegative and to be $-x$ when x is negative
function application	$f(7)$	f of 7 **or** f applied to 7 **or** the image of 7 under f
	$f(z)$	f of z **or** f applied to z **or** the image of z under f
	$f(g(z))$	f of g of z **or** f applied to the result of g applied to z
absolute value	$\lvert -3 \rvert$	The absolute value of -3
square root	$\sqrt{9}$	The non-negative square root of 9

Data Types: sets, n-tuples, and strings

Term	Examples: (add additional examples from class)
set unordered collection of elements *repetition doesn't matter* *Equal sets agree on membership of all elements*	$7 \in \{43, 7, 9\}$ $\quad$ $2 \notin \{43, 7, 9\}$
n-tuple ordered sequence of elements with n "slots" $(n > 0)$ *repetition matters, fixed length* *Equal n-tuples have corresponding components equal*	
string ordered finite sequence of elements each from specified set *repetition matters, arbitrary finite length* *Equal strings have same length and corresponding characters equal*	

Special cases:

When $n = 2$, the 2-tuple is called an **ordered pair**.

A string of length 0 is called the **empty string** and is denoted λ.

A set with no elements is called the **empty set** and is denoted $\{\}$ or $\emptyset$.

To define a set using **roster method**, explicitly list its elements. That is, start with $\{$ then list elements of the set separated by commas and close with $\}$.

To define a set using **set builder definition**, either form "The set of all x from the universe U such that x is ..." by writing

$$\{x \in U \mid ...x...\}$$

or form "the collection of all outputs of some operation when the input ranges over the universe U" by writing

$$\{...x... \mid x \in U\}$$

We use the symbol $\in$ as "is an element of" to indicate membership in a set.

Example sets: For each of the following, identify whether it's defined using the roster method or set builder notation and give an example element.

$\{-1, 1\}$

$\{0, 0\}$

$\{-1, 0, 1\}$

$\{(x, x, x) \mid x \in \{-1, 0, 1\}\}$

$\{\}$

$\{x \in \mathbb{Z} \mid x \geq 0\}$

$\{x \in \mathbb{Z} \mid x > 0\}$

$\{\mathtt{A}, \mathtt{C}, \mathtt{U}, \mathtt{G}\}$

$\{\mathtt{AUG}, \mathtt{UAG}, \mathtt{UGA}, \mathtt{UAA}\}$

RNA is made up of strands of four different bases that match up in specific ways.
The bases are elements of the set $B = \{\texttt{A}, \texttt{C}, \texttt{U}, \texttt{G}\}$.

Formally, to define the set of all RNA strands, we need more than roster method or set builder descriptions.

New! Recursive Definitions of Sets: The set S (pick a name) is defined by:

Basis Step:	Specify finitely many elements of S
Recursive Step:	Give rule(s) for creating a new element of S from known values existing in S, and potentially other values.

The set S then consists of all and only elements that are put in S by finitely many (a nonnegative integer number) of applications of the recursive step after the basis step.

Definition The set of nonnegative integers $\mathbb{N}$ is defined (recursively) by:

Basis Step:
Recursive Step:

Examples:

Definition The set of all integers $\mathbb{Z}$ is defined (recursively) by:

Basis Step:
Recursive Step:

Examples:

Definition The set of RNA strands S is defined (recursively) by:

Basis Step:	$\texttt{A} \in S, \texttt{C} \in S, \texttt{U} \in S, \texttt{G} \in S$
Recursive Step:	If $s \in S$ and $b \in B$, then $sb \in S$

where sb is string concatenation.

Examples:

Definition The set of bitstrings (strings of 0s and 1s) is defined (recursively) by:

Basis Step:
Recursive Step:

Notation: We call the set of bitstrings $\{0,1\}^*$.

Examples:

Review: Week 1 Monday

1. Colors can be described as amounts of red, green, and blue mixed together[1] Mathematically, a color can be represented as a 3-tuple (r, g, b) where r represents the red component, g the green component, b the blue component and where each of r, g, b must be a value from this collection of numbers:

   {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234, 235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245, 246, 247, 248, 249, 250, 251, 252, 253, 254, 255}

   (a) **True** or **False**: $(1, 3, 4)$ fits the definition of a color above.

   (b) **True** or **False**: $(1, 100, 200, 0)$ fits the definition of a color above.

   (c) **True** or **False**: $(510, 255)$ fits the definition of a color above.

   (d) **True** or **False**: There is a color (r_1, g_1, b_1) where $r_1 + g_1 + b_1$ is greater than 765.

   (e) **True** or **False**: There is a color (r_2, g_2, b_2) where $r_2 + g_2 + b_2$ is equal to 1.

   (f) **True** or **False**: Another way to write the collection of allowed values for red, green, and blue components is
   $$\{x \in \mathbb{N} \mid 0 \leq x \leq 255\}$$
   .

   (g) **True** or **False**: Another way to write the collection of allowed values for red, green, and blue components is
   $$\{n \in \mathbb{Z} \mid 0 \leq n \leq 255\}$$
   .

   (h) **True** or **False**: Another way to write the collection of allowed values for red, green, and blue components is
   $$\{y \in \mathbb{Z} \mid -1 < y \leq 255\}$$
   .

2. Sets are unordered collections. In class, we saw some examples of sets and also how to define sets using roster method and set builder notation.

   (a) Select all and only the sets below that have 0 as an element.

      i. $\{-1, 1\}$

[1] This RGB representation is common in web applications. Many online tools are available to play around with mixing these colors, e.g. `https://www.w3schools.com/colors/colors_rgb.asp`.

ii. $\{0, 0\}$
iii. $\{-1, 0, 1\}$
iv. $\mathbb{Z}$
v. $\mathbb{Z}^+$
vi. $\mathbb{N}$

(b) Select all and only the sets below that have the ordered pair $(2, 0)$ as an element.

i. $\{x \mid x \in \mathbb{N}\}$
ii. $\{(x, x) \mid x \in \mathbb{N}\}$
iii. $\{(x, x-2) \mid x \in \mathbb{N}\}$
iv. $\{(x, y) \mid x \in \mathbb{Z}^+, y \in \mathbb{Z}\}$

3. Which of the following are (recursive) definitions of the set of integers $\mathbb{Z}$? (Select True/False for each one.)

(a)

Basis Step:	$5 \in \mathbb{Z}$
Recursive Step:	If $x \in \mathbb{Z}$, then $x + 1 \in \mathbb{Z}$ and $x - 1 \in \mathbb{Z}$

(b)

Basis Step:	$0 \in \mathbb{Z}$
Recursive Step:	If $x \in \mathbb{Z}$, then $x + 1 \in \mathbb{Z}$ and $x - 1 \in \mathbb{Z}$ and $x + 2 \in \mathbb{Z}$ and $x - 2 \in \mathbb{Z}$

(c)

Basis Step:	$0 \in \mathbb{Z}$
Recursive Step:	If $x \in \mathbb{Z}$, then $x + 2 \in \mathbb{Z}$ and $x - 1 \in \mathbb{Z}$

(d)

Basis Step:	$0 \in \mathbb{Z}$
Recursive Step:	If $x \in \mathbb{Z}$, then $x + 1 \in \mathbb{Z}$ and $x + 2 \in \mathbb{Z}$

Wednesday September 29

To define a set we can use the roster method, set builder notation, a recursive definition, and also we can apply a set operation to other sets.

New! Cartesian product of sets and **set-wise concatenation of sets of strings**

Definition: Let A and B be sets. The **Cartesian product** of A and B, denoted $A \times B$, is the set of all ordered pairs (a, b) where $a \in A$ and $b \in B$

$$A \times B = \{(a, b) \mid a \in A \text{ and } b \in B\}$$

Definition: Let A and B be sets of strings over the same alphabet. The **set-wise concatenation** of A and B, denoted $A \circ B$, is the set of all results of string concatenation ab where $a \in A$ and $b \in B$

$$A \circ B = \{ab \mid a \in A \text{ and } b \in B\}$$

Fill in the missing entries in the table:

Set	**Example elements in this set**:
B	`A` `C` `G` `U`
	$(\texttt{A}, \texttt{C})$ $(\texttt{U}, \texttt{U})$
$B \times \{-1, 0, 1\}$	
$\{-1, 0, 1\} \times B$	
	$(0, 0, 0)$
$\{\texttt{A}, \texttt{C}, \texttt{G}, \texttt{U}\} \circ \{\texttt{A}, \texttt{C}, \texttt{G}, \texttt{U}\}$	
	`GGGG`

New! Defining functions A function is defined by its (1) domain, (2) codomain, and (3) rule assigning each element in the domain exactly one element in the codomain.

The domain and codomain are nonempty sets.
The rule can be depicted as a table, formula, or English description.

Example: The absolute value function

Domain

Codomain

Rule

In the table below, each row represents a user's ratings of movies: ✓ (check) indicates the person liked the movie, ✗ (x) that they didn't, and • (dot) that they didn't rate it one way or another (neutral rating or didn't watch).

Person	Fyre	Frozen II	Picard	Ratings written as a 3-tuple
P_1	✗	•	✓	$(-1, 0, 1)$
P_2	✓	✓	✗	$(1, 1, -1)$
P_3	✓	✓	✓	$(1, 1, 1)$
P_4	•	✗	✓	

Which of P_1, P_2, P_3 has movie preferences most similar to P_4?

One approach to answer this question: use **functions** to define distance between user preferences.

<table>
<tr><td colspan="2">Define the following functions whose inputs are ordered pairs of 3-tuples each of whose components comes from the set $\{-1, 0, 1\}$</td></tr>
<tr><td>$d_1(\ (x_1, x_2, x_3), (y_1, y_2, y_3)\ ) = \sum_{i=1}^{3} ((\mid x_i - y_i \mid + 1) \textbf{ div } 2)$</td><td>$d_2(\ (x_1, x_2, x_3), (y_1, y_2, y_3)\ ) = \sqrt{\sum_{i=1}^{3} (x_i - y_i)^2}$</td></tr>
</table>

$d_1(P_4, P_1)$	$d_1(P_4, P_2)$	$d_1(P_4, P_3)$
$d_2(P_4, P_1)$	$d_2(P_4, P_2)$	$d_2(P_4, P_3)$

Extra example: A new movie is released, and P_1 and P_2 watch it before P_3, and give it ratings; P_1 gives ✓ and P_2 gives ✗. Should this movie be recommended to P_3? Why or why not?

Extra example: Define the new functions that would be used to compare the 4-tuples of ratings encoding movie preferences now that there are four movies in the database.

Definition (Of a function, recursively) A function *rnalen* that computes the length of RNA strands in S is defined by:

		$rnalen : S$	$\rightarrow \mathbb{Z}^+$
Basis Step:	If $b \in B$ then	$rnalen(b)$	$= 1$
Recursive Step:	If $s \in S$ and $b \in B$, then	$rnalen(sb)$	$= 1 + rnalen(s)$

The domain of *rnalen* is ____________________. The codomain of *rnalen* is ____________________.

$rnalen(\texttt{ACU}) =$ ____

Extra example: A function *basecount* that computes the number of a given base b appearing in a RNA strand s is defined recursively: *fill in codomain and sample function applications*

$$basecount : S \times B \rightarrow$$

Basis Step: If $b_1 \in B, b_2 \in B$

$$basecount(b_1, b_2) = \begin{cases} 1 & \text{when } b_1 = b_2 \\ 0 & \text{when } b_1 \neq b_2 \end{cases}$$

Recursive Step: If $s \in S, b_1 \in B, b_2 \in B$

$$basecount(sb_1, b_2) = \begin{cases} 1 + basecount(s, b_2) & \text{when } b_1 = b_2 \\ basecount(s, b_2) & \text{when } b_1 \neq b_2 \end{cases}$$

$basecount(\texttt{ACU}, \texttt{A}) =$ ____

$basecount(\texttt{ACU}, \texttt{G}) =$ ____

Review: Week 1 Wednesday

Friday October 1

Number representations

Review: Week 1 Friday