Defining functions more examples

Let's practice with functions related to some of our applications so far.

Recall: We model the collection of user ratings of the four movies Dune, Oppenheimer, Barbie, Nimona as the set $\{-1, 0, 1\}^4$. One function that compares pairs of ratings is

$$d_0 : \{-1, 0, 1\}^4 \times \{-1, 0, 1\}^4 \to \mathbb{R}$$

given by

$$d_0(\ (\ (x_1, x_2, x_3, x_4), (y_1, y_2, y_3, y_4)\ )\ ) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + (x_3 - y_3)^2 + (x_4 - y_4)^2}$$

Notice: any ordered pair of ratings is an okay input to d_0.

Notice: there are (at most)

$$(3 \cdot 3 \cdot 3 \cdot 3) \cdot (3 \cdot 3 \cdot 3 \cdot 3) = 3^8 = 6561$$

many pairs of ratings. There are therefore lots and lots of real numbers that are not the output of d_0.

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

Pro-tip: informal definitions sometime use $\cdots$ to indicate "continue the pattern". Often, to make this pattern precise we use recursive definitions.

Name	Domain	Codomain	Rule	Example
rnalen	S	$\mathbb{Z}^+$	Basis Step: If $b \in B$ then $rnalen(b) = 1$ Recursive Step: If $s \in S$ and $b \in B$, then $rnalen(sb) = 1 + rnalen(s)$	$rnalen(\texttt{AC}) \overset{\text{rec step}}{=} 1 + rnalen(\texttt{A})$ $\overset{\text{basis step}}{=} 1 + 1 = 2$
basecount	$S \times B$	$\mathbb{N}$	Basis Step: If $b_1 \in B, b_2 \in B$ then $basecount(\ (b_1, b_2)\ ) =$ $\begin{cases} 1 & \text{when } b_1 = b_2 \\ 0 & \text{when } b_1 \neq b_2 \end{cases}$ Recursive Step: If $s \in S, b_1 \in B, b_2 \in B$ $basecount(\ (sb_1, b_2)\ ) =$ $\begin{cases} 1 + basecount(\ (s, b_2)\ ) & \text{when } b_1 = b_2 \\ basecount(\ (s, b_2)\ ) & \text{when } b_1 \neq b_2 \end{cases}$	$basecount(\ (\texttt{ACU}, \texttt{C})\ ) =$
"2 to the power of"	$\mathbb{N}$	$\mathbb{N}$	Basis Step: $2^0 = 1$ Recursive Step: If $n \in \mathbb{N}, 2^{n+1} =$	
"b to the power of i"	$\mathbb{Z}^+ \times \mathbb{N}$	$\mathbb{N}$	Basis Step: $b^0 = 1$ Recursive Step: If $i \in \mathbb{N}, b^{i+1} = b \cdot b^i$	

$2^0 = 1$	$2^1 = 2$	$2^2 = 4$	$2^3 = 8$	$2^4 = 16$	$2^5 = 32$	$2^6 = 64$	$2^7 = 128$	$2^8 = 256$	$2^9 = 512$	$2^{10} = 1024$

Division algorithm

Integer division and remainders (aka The Division Algorithm) Let n be an integer and d a positive integer. There are unique integers q and r, with $0 \leq r < d$, such that $n = dq + r$. In this case, d is called the divisor, n is called the dividend, q is called the quotient, and r is called the remainder.

Because these numbers are guaranteed to exist, the following functions are well-defined:

- **div** $: \mathbb{Z} \times \mathbb{Z}^+ \to \mathbb{Z}$ given by **div** $(\ (n, d)\ )$ is the quotient when n is the dividend and d is the divisor.
- **mod** $: \mathbb{Z} \times \mathbb{Z}^+ \to \mathbb{Z}$ given by **mod** $(\ (n, d)\ )$ is the remainder when n is the dividend and d is the divisor.

Because these functions are so important, we sometimes use the notation n **div** $d =$ **div** $(\ (n, d)\ )$ and n **mod** $d =$ **mod** $(\ (n, d)\ )$.

Pro-tip: The functions **div** and **mod** are similar to (but not exactly the same as) the operators / and % in Java and python.

Example calculations:

20 **div** 4

20 **mod** 4

20 **div** 3

20 **mod** 3

-20 **div** 3

-20 **mod** 3

Netflix intro

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.

Data types

Term	**Examples**: (add additional examples from class)
set unordered collection of elements *repetition doesn't matter* *Equal sets agree on membership of all elements*	$7 \in \{43, 7, 9\}$ $2 \notin \{43, 7, 9\}$
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 (called the alphabet over which the string is defined) *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 $\emptyset$.

Ratings encoding

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). Can encode these ratings numerically with 1 for ✓ (check), -1 for ✗ (x), and 0 for • (dot).

Person	Dune	Oppenheimer	Barbie	Nimona	Ratings written as a 4-tuple
P_1	✗	•	✓		
P_2	✓	✓	✗		
P_3	✓	✓	✓		
P_4	•	✗	✓		
You					

Definitions set prereqs

Term	**Notation Example(s)**	**We say in English . . .**
all reals	$\mathbb{R}$	The (set of all) real numbers (numbers on the number line)
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.

Defining sets

To define sets:

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.

Can we infer the data type of the example element from the notation?

$\{-1, 1\}$

$\{0, 0\}$

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

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

$\{\}$

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

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

$\{\smile, \sun\}$

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

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

Definitions functions prereqs

Term	Notation Example(s)	We say in English ...
sequence	$x_1, \ldots, x_n$	A sequence x_1 to x_n
summation	$\sum_{i=1}^{n} x_i$ or $\sum_{i=1}^{n} x_i$	The sum of the terms of the sequence x_1 to x_n
piecewise rule definition	$f(x) = \begin{cases} \text{rule 1 for } x & \text{when COND 1} \\ \text{rule 2 for } x & \text{when COND 2} \end{cases}$	Define f of x to be the result of applying rule 1 to x when condition COND 1 is true and the result of applying rule 2 to x when condition COND 2 is true. This can be generalized to having more than two conditions (or cases).
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

Pro-tip: the meaning of two vertical lines | | depends on the data-types of what's between the lines. For example, when placed around a number, the two vertical lines represent absolute value. We've seen a single vertial line | used as part of set builder definitions to represent "such that". Again, this is (one of the many reasons) why is it very important to declare the data-type of variables before we use them.

Defining functions

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, piecewise definition, or English description.
The notation is

"Let the function FUNCTION-NAME: DOMAIN $\to$ CODOMAIN be given by
FUNCTION-NAME(x) = ... for every $x \in DOMAIN$".

or

"Consider the function FUNCTION-NAME: DOMAIN $\to$ CODOMAIN defined as
FUNCTION-NAME(x) = ... for every $x \in DOMAIN$".

Example: The absolute value function

Domain

Codomain

Rule

Defining functions ratings

Recall our representation of Netflix users' ratings of movies as n-tuples, where n is the number of movies in the database. Each component of the n-tuple is -1 (didn't like the movie), 0 (neutral rating or didn't watch the movie), or 1 (liked the movie).

Consider the ratings $P_1 = (-1, 0, 1, 0)$, $P_2 = (1, 1, -1, 0)$, $P_3 = (1, 1, 1, 0)$, $P_4 = (0, -1, 1, 0)$

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 quantify difference among user preferences.

For example, consider the function $d_0 : \{-1, 0, 1\}^4 \times \{-1, 0, 1\}^4 \to \mathbb{R}$ given by

$$d_0(\ (\ (x_1, x_2, x_3, x_4), (y_1, y_2, y_3, y_4)\ )\ ) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + (x_3 - y_3)^2 + (x_4 - y_4)^2}$$

Defining functions recursively

When the domain of a function is a *recursively defined set*, the rule assigning images to domain elements (outputs) can also be defined recursively.

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

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

where sb is string concatenation.

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

		$rnalen : S$	$\to \mathbb{Z}^+$
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

Example function application:

$$rnalen(\mathtt{ACU}) =$$

Example: A function *basecount* that computes the number of a given base b appearing in a RNA strand s is defined recursively:

$$basecount : S \times B \to \mathbb{N}$$

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 $s \in S, b_1 \in B, b_2 \in B$

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