Domain and Range

(Also available for Pyret)

Students encounter String and Image datatypes and use "contracts" to make sense of the domain and range of functions.

Prerequisites	Order of Operations
Relevant Standards <select multiple><option>OK</option><option>CC-Math</option></select>	*Select one or more standards from the menu on the left (⌘-click on Mac, Ctrl-click elsewhere).*
Lesson Goals	Students will be able to: • Demonstrate understanding of ***Domain*** and ***Range*** and how they relate to ***functions***
Student-facing Goals	• I can identify the domain and range of a function. • I can identify the data types ***Number***, ***String***, and ***Image***
Materials	• Lesson slides template (Google Slides) • Exploring Image Functions (original (Page 17), solutions) • Reading for Domain and Range (original (Page 18), solution)
Preparation	• Make sure all materials have been gathered • Decide how students will be grouped in pairs
Supplemental Resources	• Functions Review (Quizizz) • Domain and Range Review (Desmos Activity)
Key Points For The Facilitator	• Check frequently for understanding of ***data types*** and ***contract***s during this lesson and throughout subsequent lessons. • Students will use their Contracts page frequently, so it should be kept in an accessible, convenient location.

For a textbook-like version of materials similar to these, you may wish to see the prior unit-based version.

Glossary

contract :: a statement of the name, domain, and range of a function

datatypes :: a way of classifying values, such as: Number, String, Image, Boolean, or any user-defined data structure

domain :: the type or set of inputs that a function expects

error message :: information from the computer about errors in code

function :: a mathematical object that consumes inputs and produces an output

Image :: a type of data for pictures

Number :: a data type representing a real number

range :: the type or set of outputs that a function produces

String :: a data type for any sequence of characters between quotation marks (examples: "hello", "42", "this is a string!")

Warmup

Students should open WeScheme in their browser, and click "Log In". This will ask them to log in with a valid Google account (Gmail, Google Classroom, YouTube, etc.), and then show them the "My Programs" page. This page is empty - they don't have any programs yet! Have them click "Start a New Program".
They will also want to have their Contracts page (back of workbook) ready, preferably in paper form.

Contracts

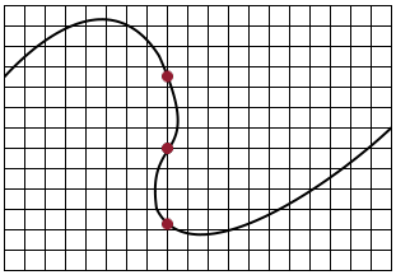
15 minutes

Overview

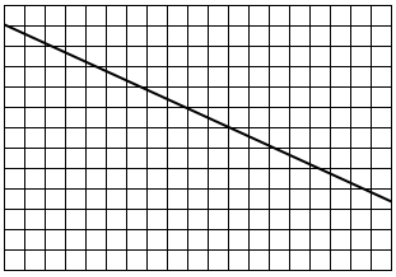
This activity introduces the notion of ***Contracts***, which are a simple notation for keeping track of the set all of possible inputs and outputs for a function. They are also closely related to the concept of a *function machine* , which is introduced as well. *Note: Contracts are based on the same notation found in Algebra!*

Launch

For each input to a function, there is exactly one output



Not a function



Function

Functions are a lot like machines: values go in, something happens, and new values come out. Let's start with an example of a function we all know: adding two numbers! Addition is like a machine that takes in pairs of numbers and produces a sum.
Consider the graphs on the right: for every input on the x-axis, a function will produce a *single* output. If we draw a vertical line and it hits the graph more than once, it means there is *more than one output* for the same input. Like any good machine, function machines must be **reliable**.
Whenever we use any machine, we always think about what goes in and what comes out. A coffee maker takes in coffee beans and water, and produces coffee. A toaster takes in bread and produces toast. We don't have to know exactly how coffee makers or toasters work in order to *use* them. All we need to know is what type of thing goes in and what type of thing should come out!
In our coffee-maker example, we expect to get the exact same coffee out if we use the exact same beans and water each time. If you put bread in a toaster and got a bagel out, you'd be pretty surprised! *Functions work the same way:* no matter how many times you plug in the same number, you will *always* get the same result. And if you don't? **It's not a function!**

Investigate

We use something called a ***Contract*** to keep track of what goes in and out of these machines called functions. Contracts are like a "cheat sheet" for using functions. Once you know how to read one, you can quickly figure out how to use a function just by looking at its contract!

> The Contract for a function has three parts: the Name of the function, the ***Domain***, and the ***Range***

- The Name is simply how we refer to the function: `*`, `+`, `sqrt`, etc.
- The ***Domain*** tells us what the function "takes in", or *consumes*. These are also known as the *arguments* to the function.
- The ***Range*** tells us what the function "gives back", *produces*.

Memorizing contracts is hard, and why memorize when we can just keep a log of them! Let's write them down so we can use them later! At the back of your workbook, you'll find pages with space to write down every contract you see in the course.

- What does Multiplication need as an input? What does it produce?
- What inputs does the Square Root function consume? What does it produce?
- When we Square something, what does the Square function consume and produce?
- Write the contracts for `+`, `-`, `*`, `/`, `sqr`, and `sqrt` into the Contracts page.

A Sample Contracts Table

Name		Domain		Range
`; +`	`:`	`Number Number`	`->`	`Number`
`; -`	`:`	`Number Number`	`->`	`Number`
`; sqr`	`:`	`Number`	`->`	`Number`
`; sqrt`	`:`	`Number`	`->`	`Number`

It would be silly to buy a coffee-maker that only works with one specific coffee! Similarly, Contracts don't tell us *specific* inputs. They tell us the ***Datatype*** of input a function needs. For example, a Contract wouldn't say that addition requires "3 and 4". Addition works on more than just those two inputs! Instead, it would tells us that addition requires "two Numbers". When we *use* a Contract, we plug specific numbers into a mathematical expression.

> Contracts are general. Expressions are specific.

Optional: Have students make a Domain and Range Frayer model and use the visual organizer to explain the concepts of Domain and Range in their own words.

Synthesize

- What is wrong with the contract `; + : 3 4 -> 7`?
- What is the difference between a value like `17` and a type like `Number`?

Exploring Image Functions

25 minutes

Overview

Students explore functions that go beyond numbers, producing all sorts of simple geometric shapes and images in the process. Making images is highly motivating, and encourages students to get better at both reading error messages and persisting in catching bugs.

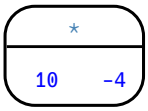
Launch

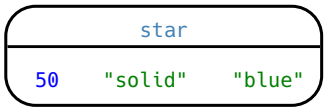
Students have already seen `Number` values like `42`, `-91`, `1/4` or `0.25`, but computer programs can work with a much larger set of ***datatypes***. Show students examples of the `String` datatype, by having them type various things in quotation marks:

- `"hello"`
- `"many words, one string"`
- `"42"`
- `"1/3"`
- Something students come up with on their own…

A String is *anything* in quotation marks. Like Number values, String values evaluate to themselves.

Here are two Circles of Evaluation. One of them is familiar, but the other very different from what you've seen before. What's different about the Circle on the right?





Possible responses:

- We've never seen the function `star` before
- We've never seen Strings used in a Circle of Evaluation before
- We've never seen a function take in three inputs
- We've never seen a function take in a mix of Numbers and Strings

Can you figure out the Name and ***Domain*** for the function in the second Circle? This is a chance to look for and make use of structure in deciphering a novel expression!

Possible responses:

- We know the name of the function is `star`, because that's what is at the top of the circle
- We know it has three things in its Domain
- We know the Domain consists of a Number and two Strings

- But what about the ***Range***? What do you think this expression will evaluate to?
- Convert this Circle to code and try out!
- What does the `50` mean to the computer? Try replacing it with different values, and see what you get.
- What does the `"blue"` mean to the computer? Try replacing it with different values, and see what you get.
- What does the `"solid"` mean to the computer? Try replacing it with different values, and see what you get. **If you get an error, read it!** It just might give you a hint about what to do…

You've seen two ***datatypes*** already: Numbers and Strings. Did we get back either on of those? The ***Range*** of `star` is a datatype we haven't seen before: an `Image`!

Error Messages

The error messages in this environment are *designed* to be as student-friendly as possible. Encourage students to read these messages aloud to one another, and ask them what they think the error message *means*. By explicitly drawing their attention to errors, you will be setting them up to be more independent in the next activity!

Suppose we had never seen `star` before. How could we figure out how to use it, using the helpful error messages?

- Type `star` into the Interactions Area and hit "Enter". What did you get back? What does that mean? *There is*

something called "star", and the computer knows it's a function!

- If it's a function, we know that it will need an open parentheses and at least one input. Have students try `(star 50)`
- What error did we get? What *hint* does it give us about how to use this function?

Investigate

- Have students turn to Exploring Image Functions (Page 17) in the workbook.
- Have students open a new program file and name it "Exploring Images".

Give students time to investigate image functions and see how many they can discover, using the Contracts page to organize their findings.

Strategies for English Language Learners

MLR 2 - Collect and Display: As students explore, walk the room and record student language relating to functions, domain, range, contracts, or what they perceive from ***error messages***. This output can be used for a concept map, which can be updated and built upon, bridging student language with disciplinary language while increasing sense-making.

Synthesize

- **What image functions did you and your partner discover?** `rectangle`, `triangle`, `ellipse`, `circle`, etc.
- **How did you decide what to try?**
- **What error messages did you see?** *Input mismatches, missing parentheses, etc.*
- **How did you figure out what to do after seeing an error message?** *Read the error message, think about what the computer is trying to tell us, etc.*

Making Sense of Contracts

10 minutes

Overview

This activity digs deeper into Contracts, and has students create their own Contracts trackers to take ownership of the concept and create an artifact they can refer back to.

Launch

`star` has three elements in its Domain: A Number, a String, and another String.

- **What do these elements represent?** *The Number is the radius, the first String is the style (either* `outline` *or* `solid` *), the second String is the color.*
- **What happens if I don't give it those things?** *We won't get the star we want, we'll probably get an error!*
- **If I give** `star` **what it needs, what do I get in return?** *An Image of the star that matches the arguments*
- `square` **has the same Domain as** `star` **. What do the arguments in** `square` **represent?** *length, style, color*
- **Can different functions have the same Domain? The same Range? Are they still different functions?** *Yes, yes, and yes!*
- **Can we come up with an example of two math functions that have the same Domain and Range?**

When the input matches what the function consumes, the function produces the output we expect.

Where else have you heard the word "contract"? How can you connect that meaning to contracts in programming?
An actor signs a contract agreeing to perform in a film in exchange for compensation, a contractor makes an agreement with a homeowner to build or repair something in a set amount of time for compensation, or a parent agrees to pizza for dinner in exchange for the child completing their chores. Similarly, a contract in programming is an ***agreement*** *between what the function is given and what it produces.*

Investigate

- Students complete Reading for Domain and Range (Page 18) with their partner.

Students create a visual "Contracts page" either digitally or physically. Ask students to think about how they visualize contracts in their own minds and how they could use that imagery to explain functions and their contracts to others.

Additional Exercises:

- Bootstrap:Algebra - Contracts (Quizizz)
- Bootstrap:Algebra - Data Types & Circles of Evaluation (Desmos Activity)
- Bootstrap:Algebra - Data Types (Desmos Activity)
- Converting Circles of Evaluation to Code (1) (original , answers)
- Converting Circles of Evaluation to Code (2) (original , answers)
- Identifying Parts of Expressions (1) (original , answers)
- Identifying Parts of Expressions (2) (original , answers)
- Matching Expressions & Contracts (original , answers)