# Function Composition


@BootstrapWorld

# Composing Functions

For this activity we will be working in groups of 3-4 students.

Each group has a set of "Function Cards", each of which takes an input and produces an output.

# Composing Functions

Each **function card** has a Contract (Name, Domain, Range), and a description of what it does.

Starting with 4,

- you could play the `add 1` card to turn it into 5.
- you could play `add 6` and turn it into 10.

What other cards could you play to get from 4 to 10?

# Composing Functions

Select the cards you would play with to get from the starting number to the ending number.

- Start at 4, end at 26
- Start at -22, end at 13
- Start at 64, end at 0

Let's try some more numbers from this random integer generator.

You will need to use some functions more than once, and that's ok! If you're ready for a challenge, try to find the *shortest path* from start to end, using the smallest number of compositions.

# Diagramming Function Composition

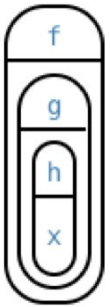
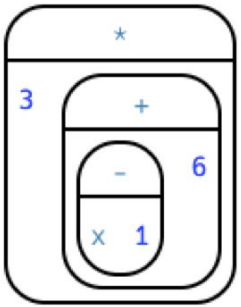
Three of the function cards we just used were for the functions `f`, `g` and `h`:

- `f` multiplies its input by 3
- `g` adds six to its input
- `h` subtracts one from its input

We can compose those functions in any order. If we composed them as `f(g(h(x)))` and evaluated them for `x = 4` what would happen?

# Diagramming Function Composition

The circles of evaluation for `f(g(h(4)))` would look like this:

| Function Composition | Order of Operations |
| --- | --- |
|  |  |

# Diagramming Function Composition

Complete Diagramming Function Composition with your partner.

# Diagramming Function Composition

Do `f(g(h(x)))` and `g(h(f(x)))` evaluate to the same thing?

# Composing Functions in Code

- Log into WeScheme, and open a new program.
- Save the file as "Function Composition", and click "Run".
- Complete the following Function Composition -- Green Star. When you're done, change the color of all the stars in the file to gold.
- Then try Function Composition -- Your Name.
- If you have time, work with Function Composition -- scale-xy and/or Function Composition Matching Activity

# Composing Functions in Code

- What do all of these functions have in common?
- Does using one of these functions change the original image?
- What does the number in `scale` represent?
- What does the number in `rotate` represent?
- The Domain and Range for `flip-horizontal` is `Image -> Image`. Why can we use the output of the `text` function as an *input* for `flip-horizontal`?

# Composing Multiple Ways

As is often true with solving math problems, there is more than one way to get the same composed image.

Suppose I wrote the code:

```
(scale 3 (circle 50 "solid" "red"))
```

What's another line of code I could write that would produce the exact same image?

# Composing Multiple Ways

Complete More than one way to Compose an Image!.

There is a special function in WeScheme that lets us test whether or not two images are equal.

```
image=? :: Image, Image -> Boolean
```

Use it to test whether all of the expressions you wrote successfully build the same images.

# Composing Multiple Ways

- Could we have written more expressions to create the same images?
- Are all of the ways to write the code equally efficient?