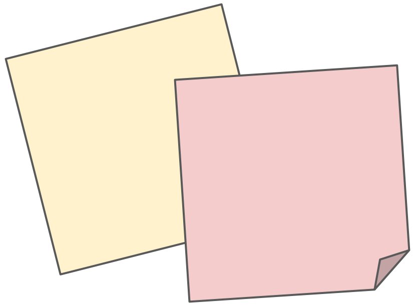
Unit 4 - Lesson 1
Variables Explore

Variables Explore

You and your partner should have:
Small stacks of red and yellow stickies
3 plastic baggies
Pen/Pencil
Dry erase marker



Unit 4 Lesson 1 - Activity

Value

One piece of information
Goes on a sticky

Numbers

Made of the digits 0...9
No quotes
Yellow sticky

9

22

548

123

Strings

Made of any characters
Inside double quotes
Red sticky

"hi"

"hi there"

"c u l8r"

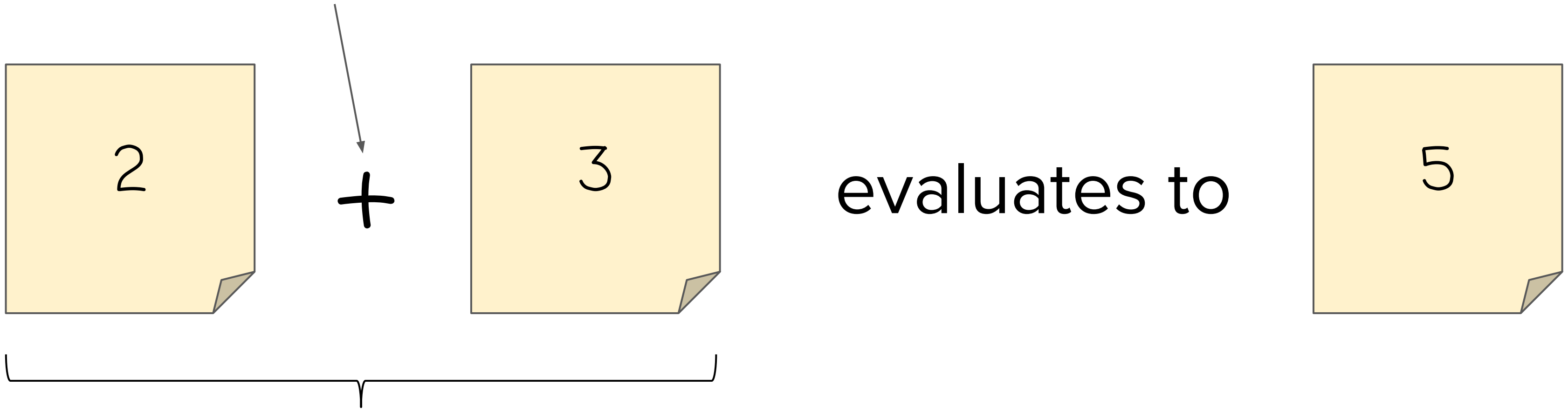
"123"

Do This:
Make one number and one string.
Share it at your table.

Unit 4 Lesson 1 - Activity

Operators

Fancy name for + - * /

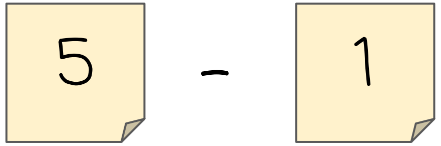


Expression

Combination of operators and values

Evaluates to single value

Do This: Evaluate this expression



Unit 4 Lesson 1 - Activity

3	+	4	evaluates to	7
5	-	2	evaluates to	3
11	*	2	evaluates to	22
10	/	2	evaluates to	5
"for"	+	"ever"	evaluates to	"forever"
"gr"	+	8	evaluates to	"gr8"
2	+	"day"	evaluates to	"2day"

Do This: Evaluate these expressions. Pay attention to what color stickies you create and if you use quotes.

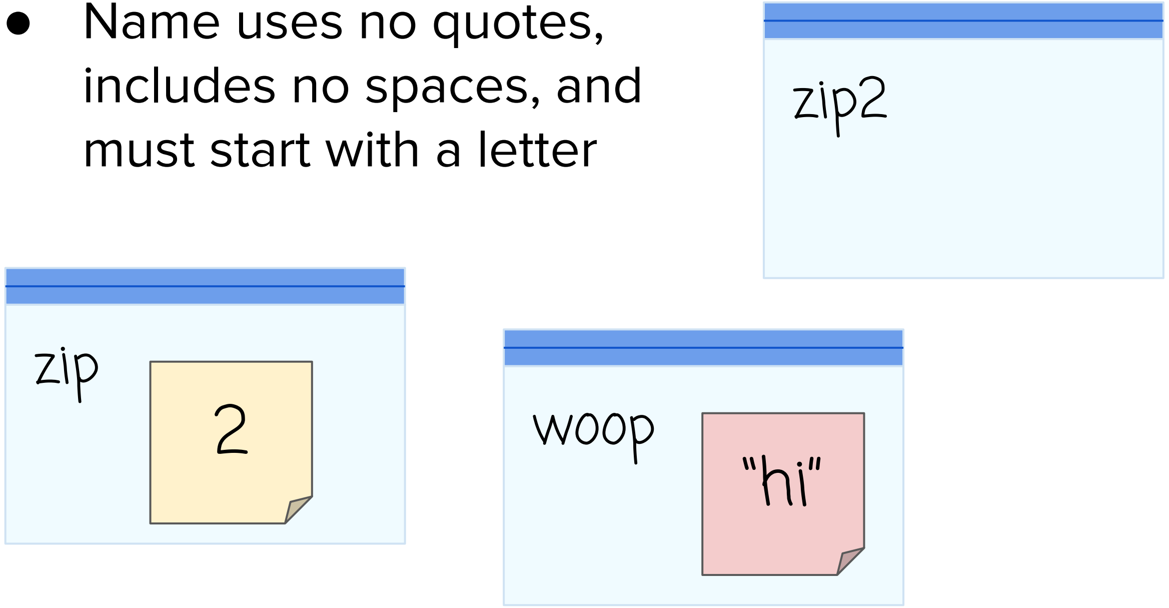
4	+	5	9
10	-	9	1
"tree"	+	"house"	"treehouse"
"you"	+	"r"	"your"
3	+	"D"	"3D"

If you're using one or two strings, you can only use the + operator. The others don't make sense!

Variables

- Plastic baggies
- Can hold at most one value
- Name uses no quotes, includes no spaces, and must start with a letter

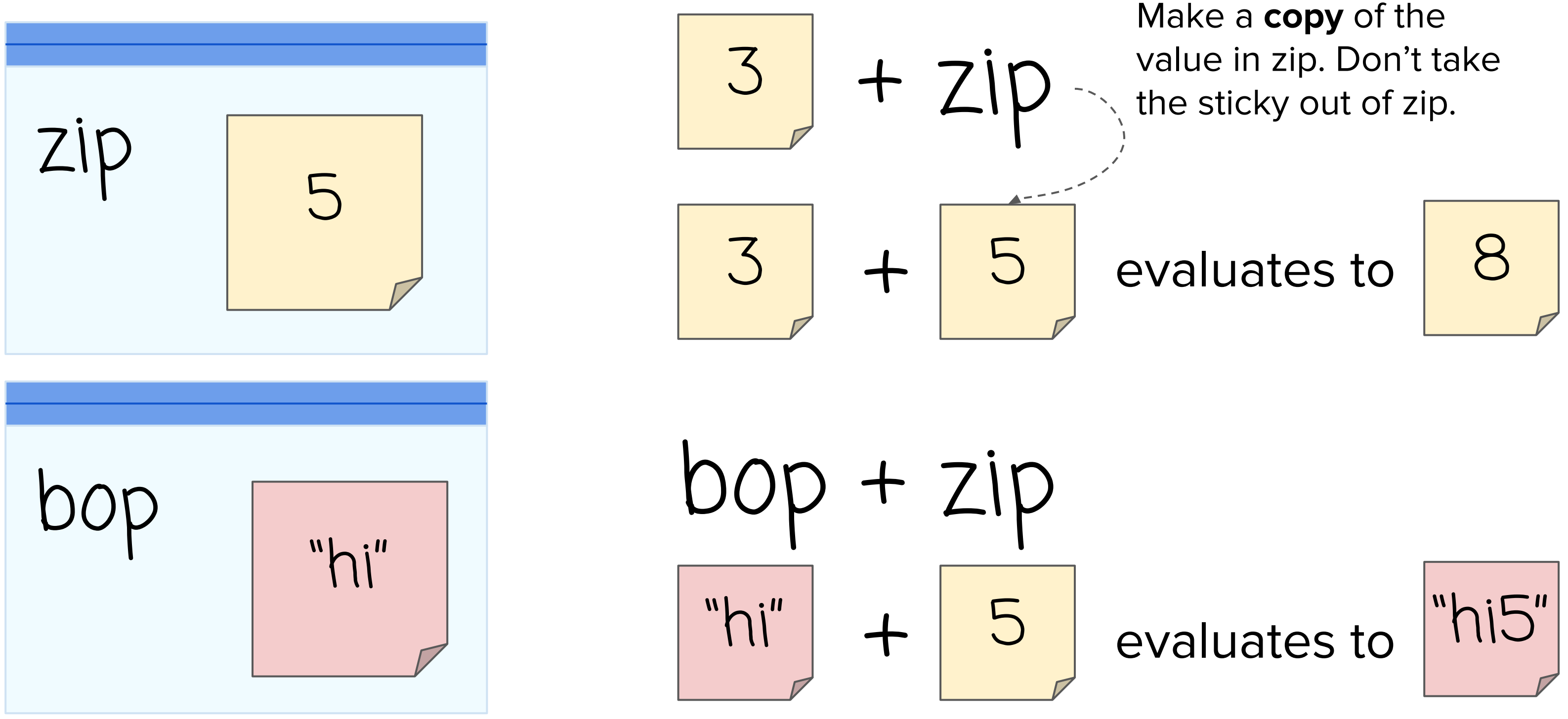
Do This:
Make one variable with any name you like. Share it with another group.



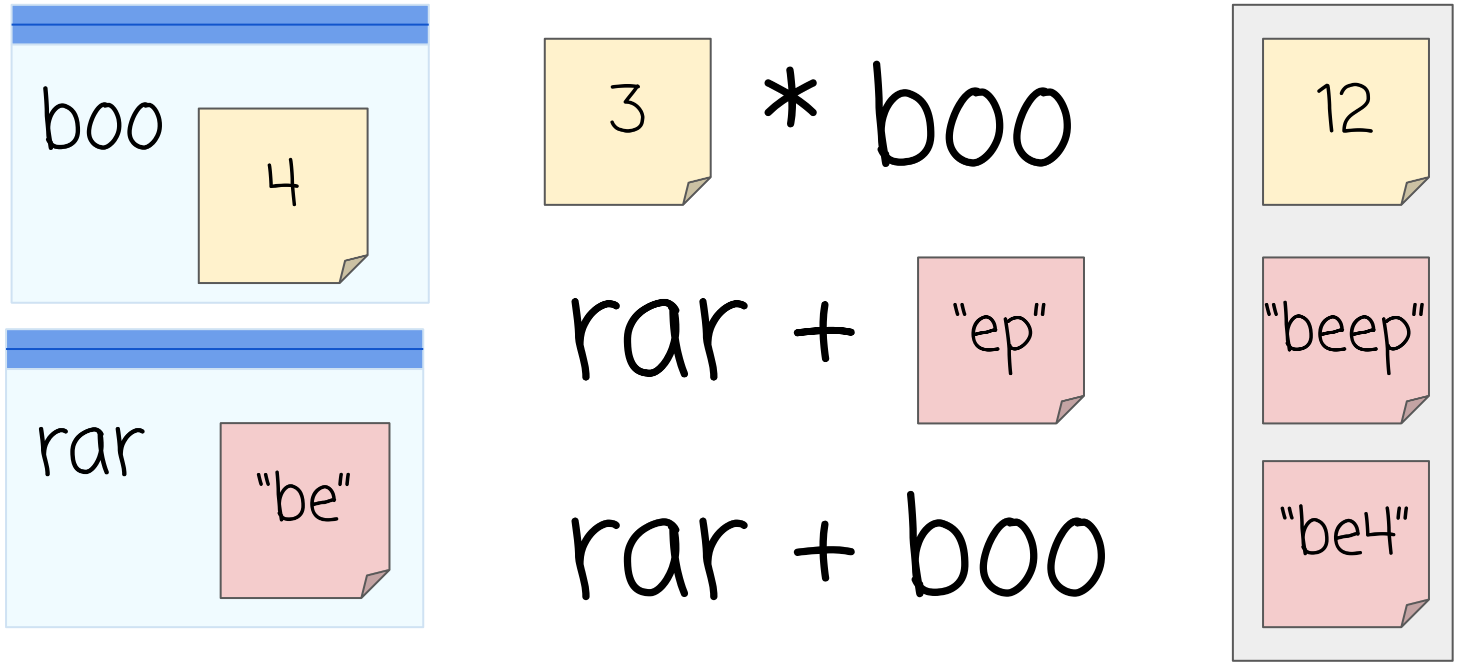
Variables and Expressions

Replace variable name with a copy of the value it holds

Evaluate the expression as normal


zip
5
bop
"hi"
3 + zip
Make a **copy** of the value in zip. Don't take the sticky out of zip.
3 + 5 evaluates to 8
bop + zip
"hi" + 5 evaluates to "hi5"

Do This: Evaluate these expressions. Make sure you pay attention to whether it evaluates to a string or a number.



Let’s start writing programs that control our variables.

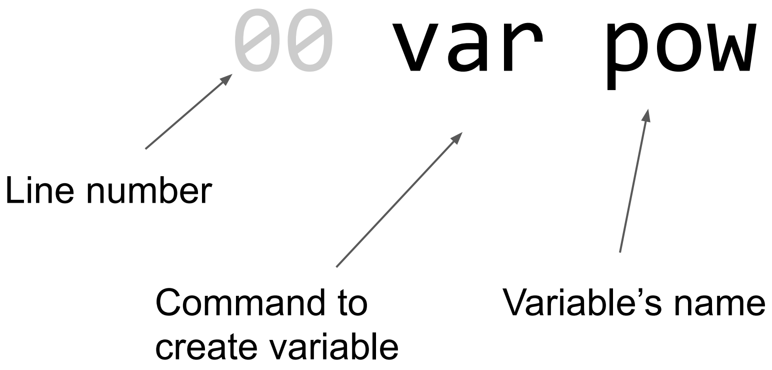
We’re going to stop using stickies but will highlight strings and numbers to help you remember the difference.

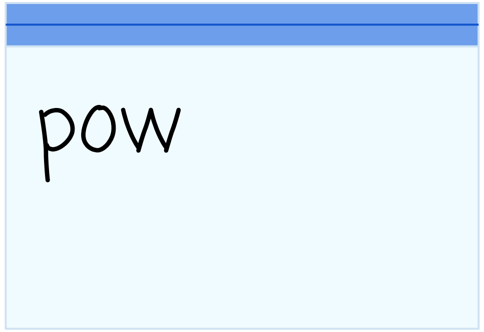
`var`

Creates a new variable

Grab a new baggie

Write the variable's name on the baggie


```
00 var pow
```



←

Do This: Run this program

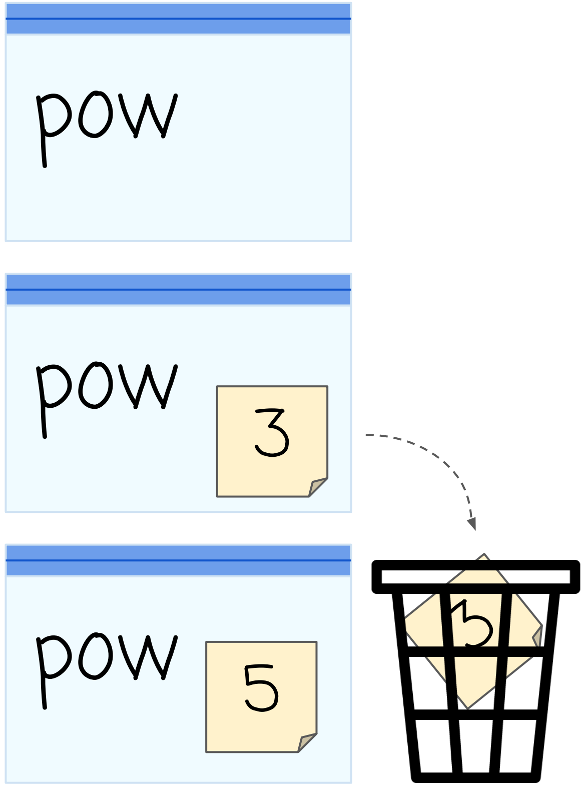
"Assignment operator"

"Assign": a fancy name for putting a value inside the baggie.

Variables can only hold one stickie. If there's already a sticky note in there, throw it away.

"pow gets 3" and "pow gets 5"

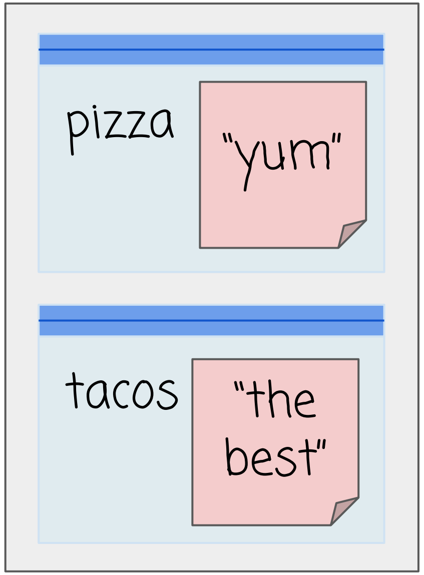
```
00 var pow
01 pow ← 3
02 pow ← 5
```



Do This:

Run this program. Compare your result with another group.

```
var pizza
pizza ← 3
var tacos
pizza ← "yum"
tacos ← "the
best"
```

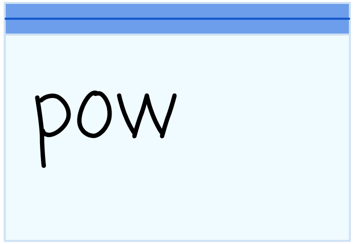


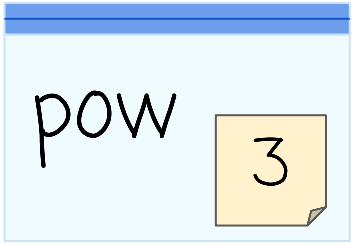
Assign a Variable with Expression

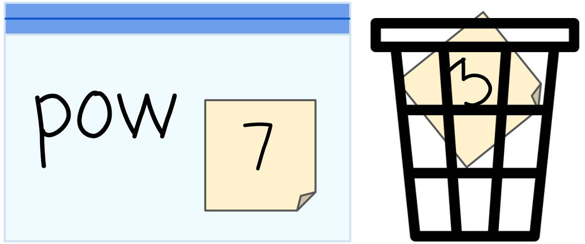
Evaluate the expression first to get one value.
Assign the value as normal

```
00 var pow

01 pow ← 1 + 2

02 pow ← 3 + 4
```

Evaluate expression first



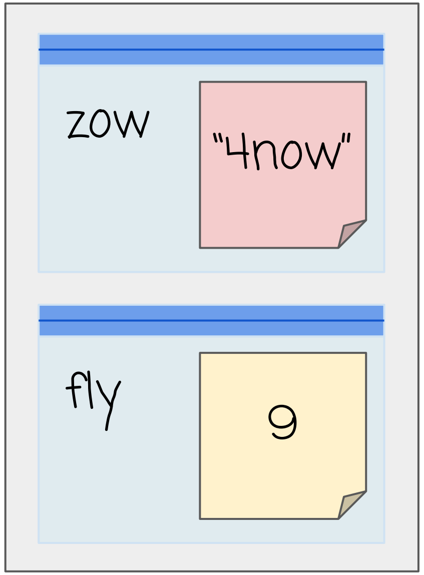




Do This:

Run this program. Compare your result with another group.

```
00 var zow
01 var fly
02 fly ← "to" + "day"
03 zow ← 4 - 1
04 fly ← 3 * 3
05 zow ← 4 + "now"
```

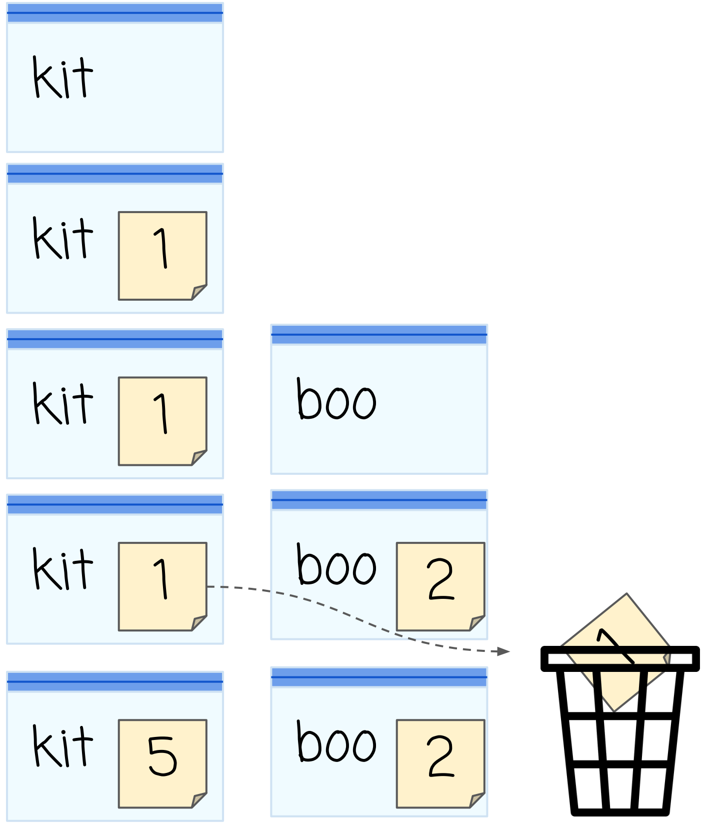


We’re not going to highlight our strings and numbers anymore. We can just use double quotes around the strings to tell the difference.

Assign a Variable: Expressions with Variables

Evaluate the expression on the right first to get one value.
Assign the value as normal

```
var kit
kit ← 1
var boo
boo ← kit + 1
kit ← 5
```

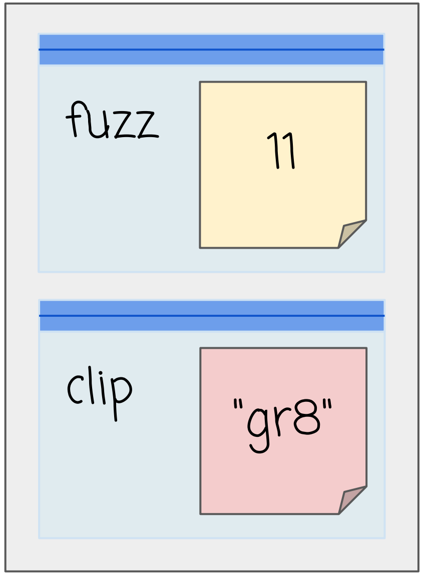


Note: Variables aren't "connected". Changing kit doesn't change boo.

Do This:

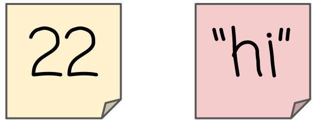
Run this program. Compare your result with another group.

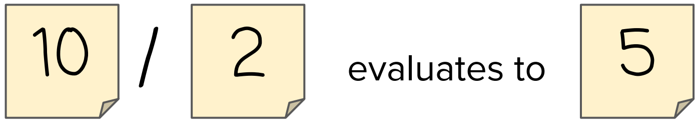
```
var fuzz
var clip
fuzz ← 5
clip ← fuzz + 2
fuzz ← clip + 1
clip ← “gr” + fuzz
fuzz ← fuzz + 1
fuzz ← fuzz + 1
fuzz ← fuzz + 1
```

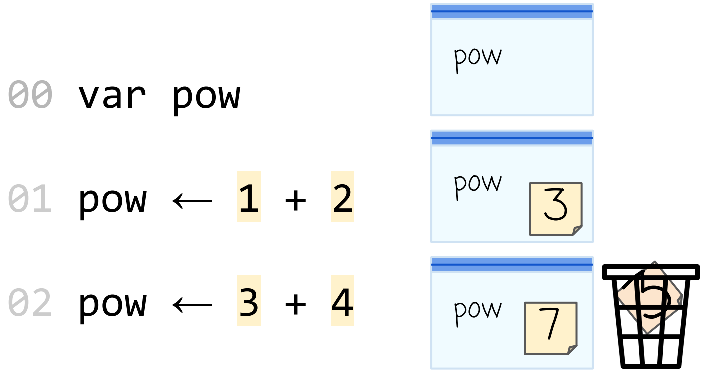


Key Takeaways

- Numbers and strings are two different types of values
- Expressions evaluate to a single new value
- When variables are in the expression just make a copy, don't change the actual variable.
- Variables are "assigned" a new value
- Evaluate first, then assign
- Old values are deleted forever.
- Assignment just moves information around. It does not "connect" variables.






```
00 var pow
01 pow ← 1 + 2
02 pow ← 3 + 4
```

In some languages (including Javascript) the assignment operator is not written

$\leftarrow$

it is written as

$=$

So the command

```
fuzz ← fuzz + 1
```

it is written as

```
fuzz = fuzz + 1
```

In math = means “are equal forever”
In programming = means “put this value in this variable”

We’ll see this more next time.

Wrap Up