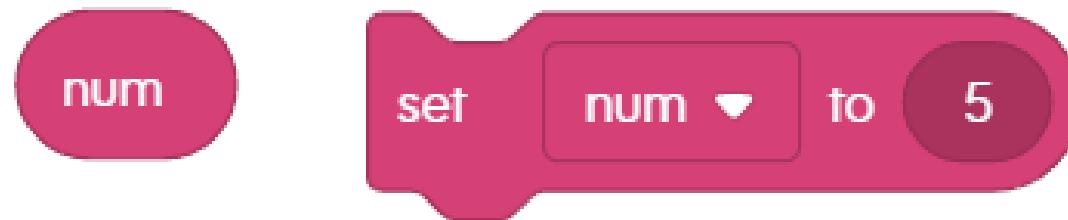


Spheros & Variables

Variables to Control Loops

MTH1W – Day 1 (and 2), Part C



MTH1W Curriculum Covered in This Lesson:

- C2.1a [Coding] Use coding to demonstrate understanding of **variables**
- C2.1b [Coding] Use coding to demonstrate understanding of **parameters**
- C2.1c [Coding] Use coding to demonstrate understanding of **equations**
- C2.1d [Coding] Use coding to demonstrate understanding of **inequalities**

- C2.2a [Coding] Create code by decomposing situations into computational steps in order to **represent mathematical concepts and relationships**
- C2.2b [Coding] Create code by decomposing situations into computational steps in order to **solve problems**

- C2.3a [Coding] **Read code to predict** its outcome
- C2.3b [Coding] **Alter code** to adjust **constraints, parameters** to represent a similar or new mathematical situation
- C2.3c [Coding] **Alter code** to adjust **outcomes** to represent a similar or new mathematical situation

Sequencing and Timing:

- On Day 1, students will cover part A, Part B and some of Part C (this lesson).
- On Day 2, students finish Part C (this lesson) and cover Part D.

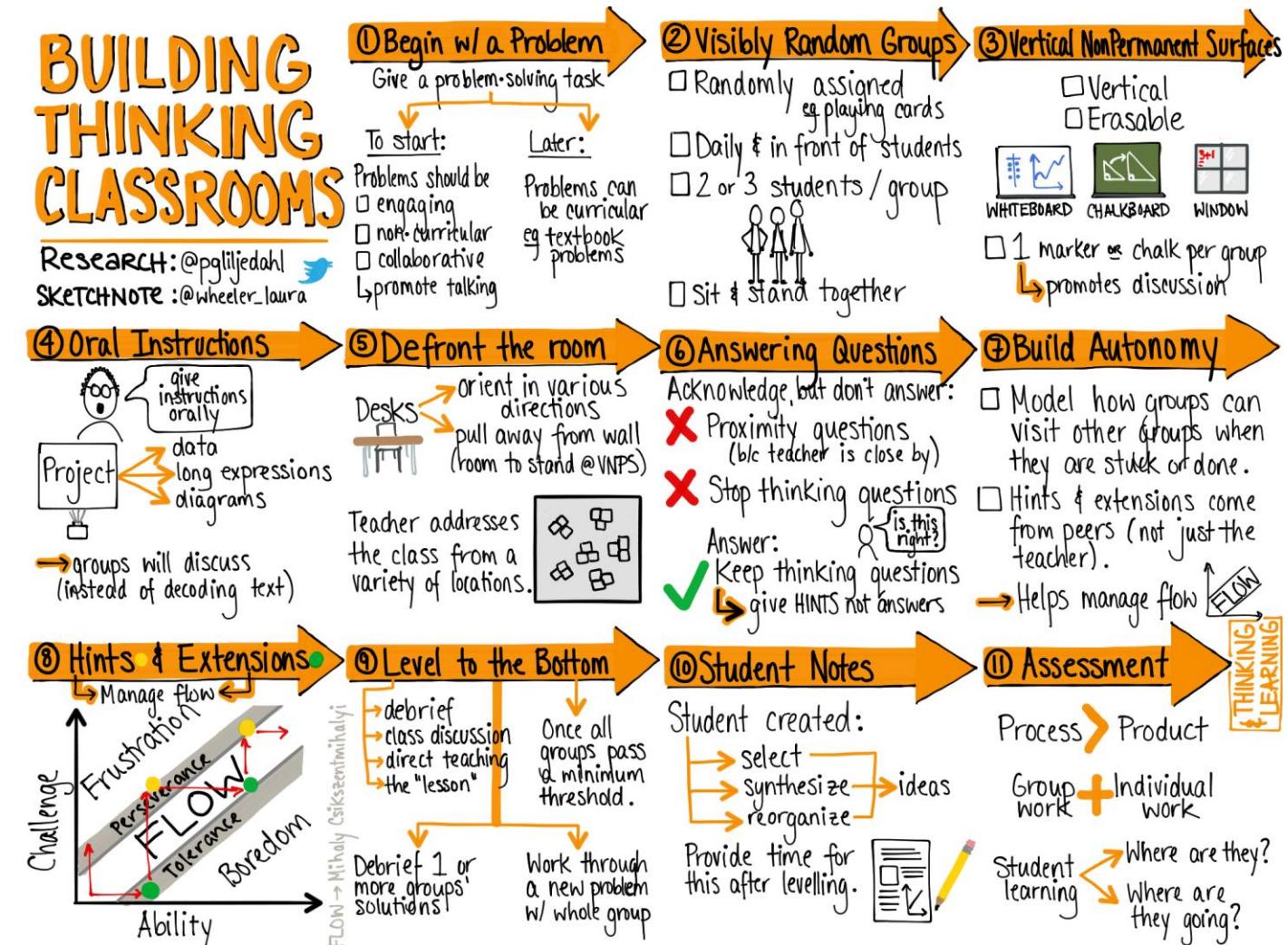
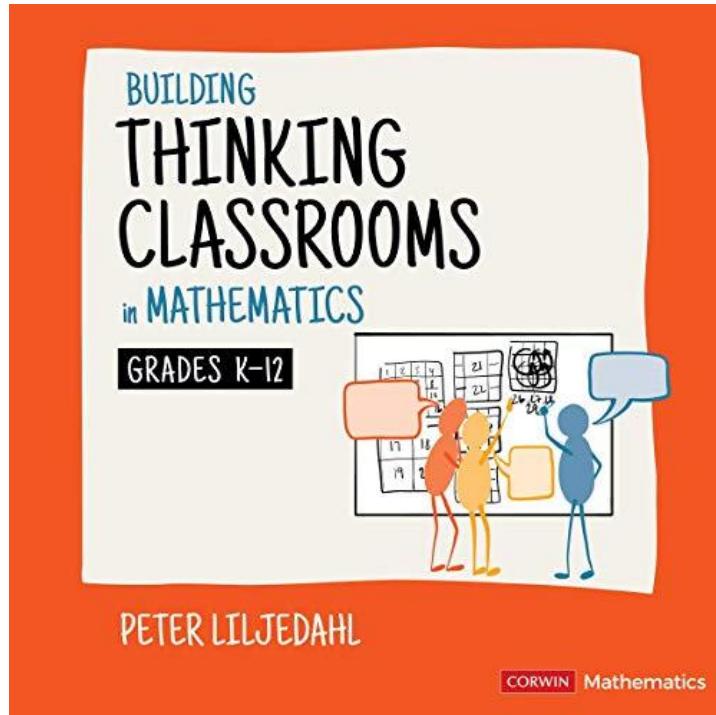
Original Curriculum Language:

C2.1 [Coding] use coding to demonstrate an understanding of algebraic concepts including variables, parameters, equations, and inequalities

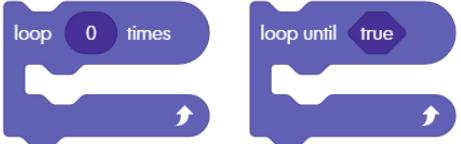
C2.2 [Coding] create code by decomposing situations into computational steps in order to represent mathematical concepts and relationships, and to solve problems

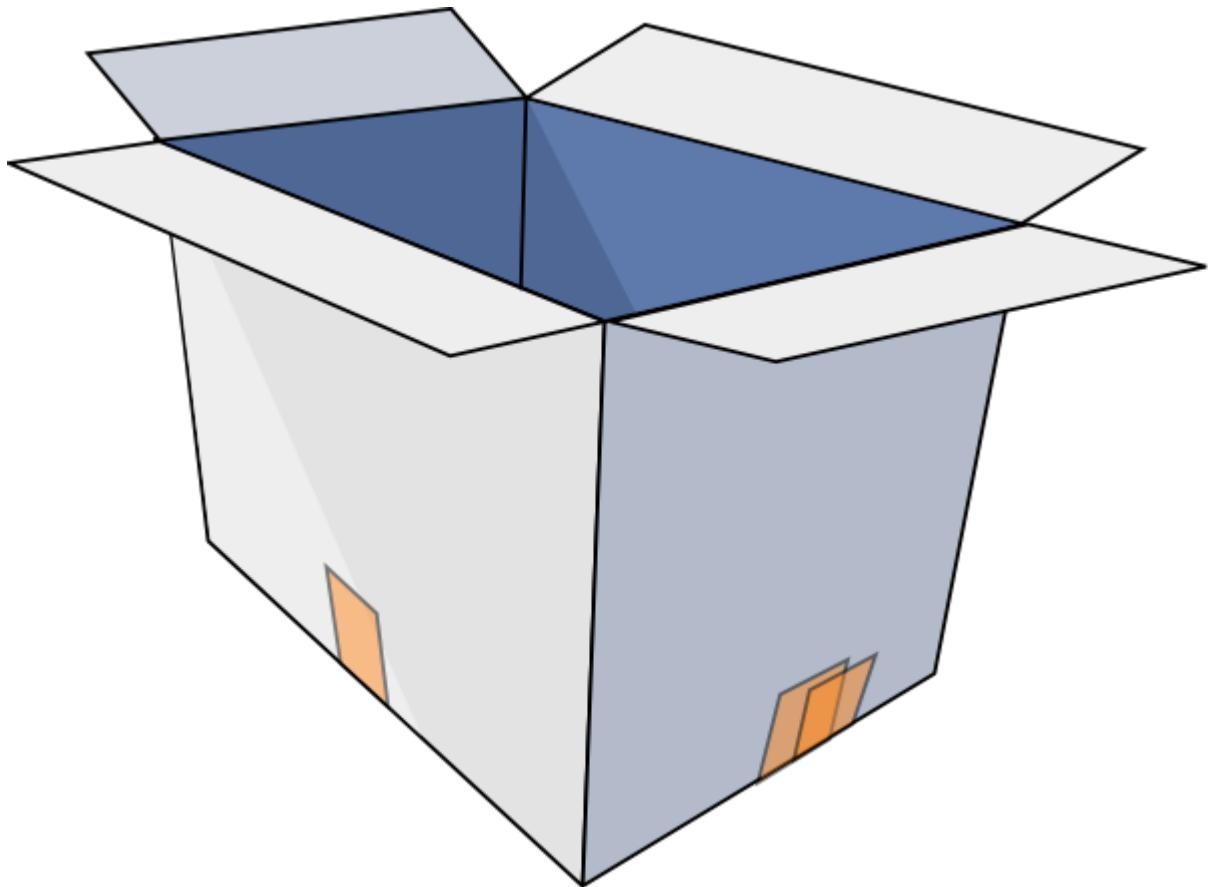
C2.3 [Coding] read code to predict its outcome, and alter code to adjust constraints, parameters, and outcomes to represent a similar or new mathematical situation

This lesson is set up to follow the format of Building Thinking Classrooms by Peter Liljedahl.



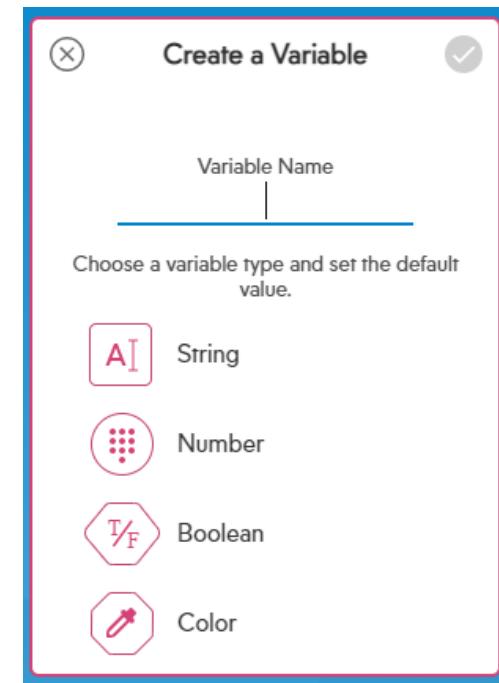
Basic Coding Components

Output	<p><i>Changes the robot:</i> Moves. Spins. Re-directs to a new angle. Makes sound. Speeds up. Lights up.</p>	
Math	<p><i>Calculation that results in a number:</i> +, -, *, /, square root</p>	
Boolean Expressions	<p><i>Calculation that results in true or false:</i> >, <, =, >=, <=, and, or, not</p>	
Control	<p>Ifs <i>Decides which piece of code to run:</i> Uses a Boolean expression and output or math.</p>	
	<p>Loops <i>Repeats code:</i> Uses a Boolean expression and output or math.</p>	
Variables	<p>Named pieces of memory where you can store things to use later OR to store calculation results.</p>	 Let's look at variables!
Functions	<p>Named pieces of code where you can group code together to use it later.</p>	

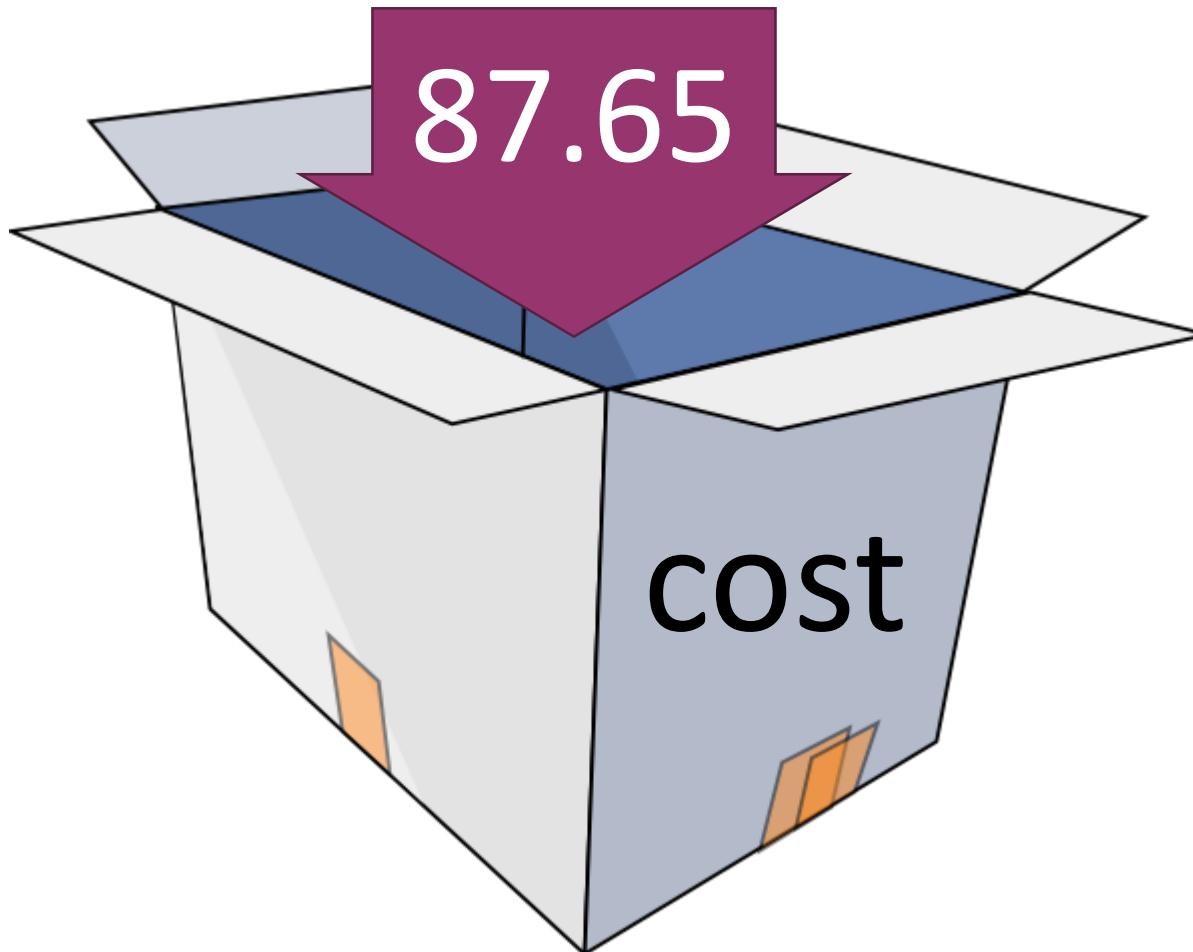


A **variable** is used in programming to store a value until we need it again.

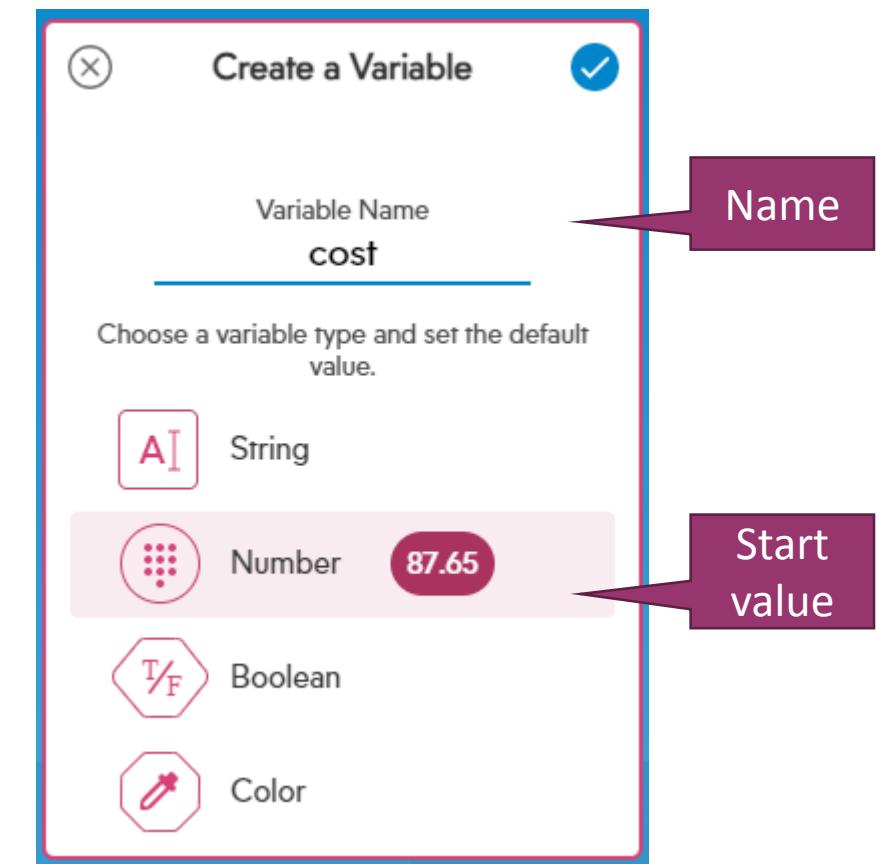
It is sort of like a box.



How sphero creates variables



Variables have **names** so that you can call them again in your code.

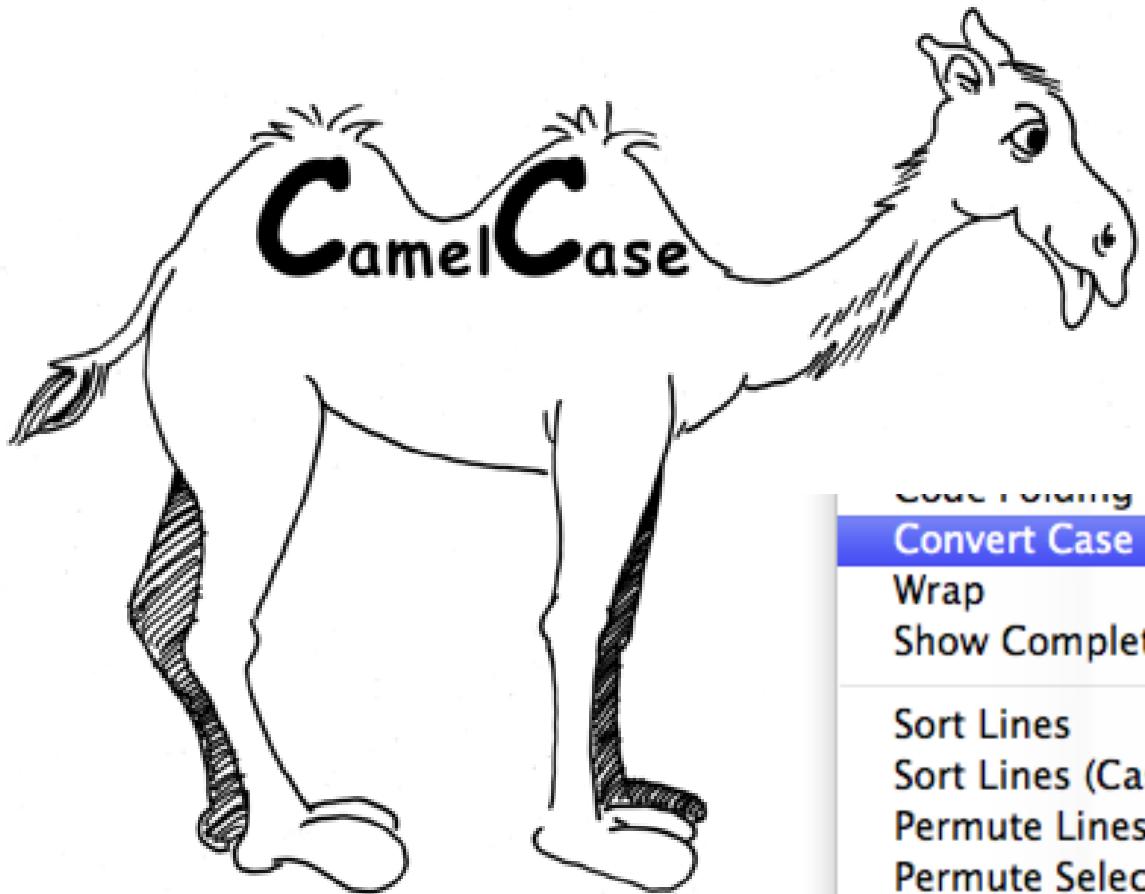


Variable Naming Rules:

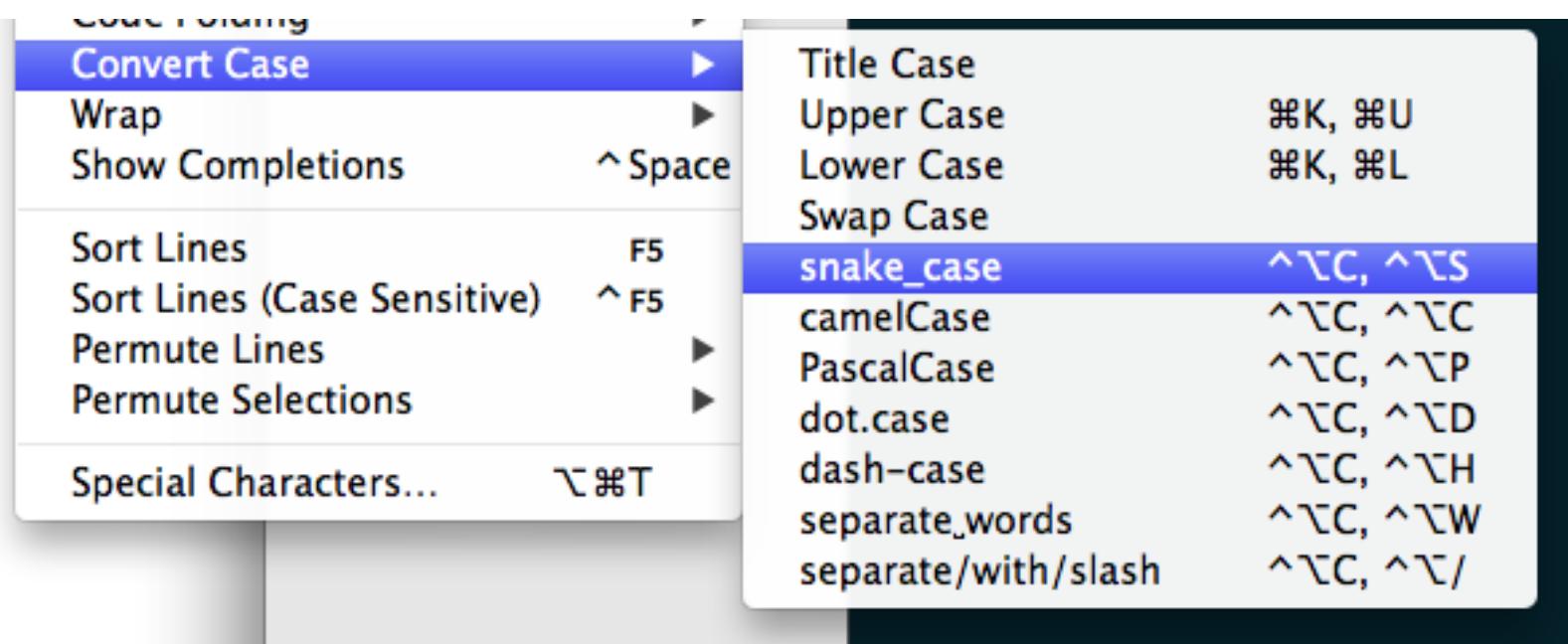
Because variables are used in the code, they have to be named carefully so the computer can understand.



1. No spaces. Camel case or underscores are fine.
2. Meaningful is easier in the long run.
3. Cannot start with a number. Otherwise, numbers are fine.
4. Cannot contain odd characters.
5. Cannot contain reserved words. (eg. setText or onEvent).
These already have a purpose in javascript.



What is
camelCase?



Which are valid variable names?

2many

impt*

T or F

tallTree

yOrN?

chicken

Naming Rules

1. No spaces.
2. Meaningful.
3. Cannot start with a number.
4. Cannot contain odd characters.
5. Cannot contain reserved words.

Sphero's variables also types: **String** and **Number**.

- A type is a kind of data – for example, words are text.
- A type is also an amount of memory allocated to a variable.
- A type is also the kinds of operations that can be done with a variable.

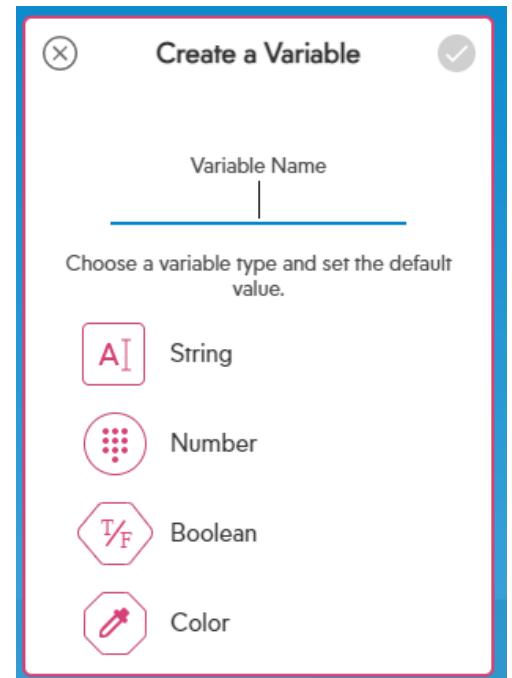
String Data Examples

- Gorski
- cat
- Frogs are green
- (905) 567-9345

Number Data Examples

- 78.545
- -4
- 98
- 0

There are two other types, what are they?



Variables

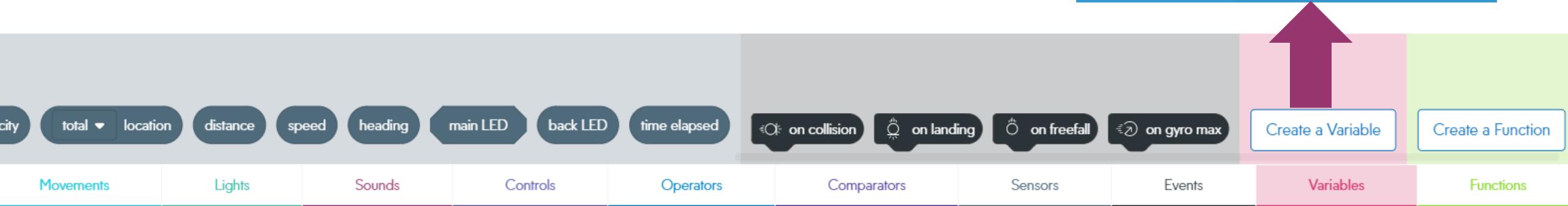
Create a Variable

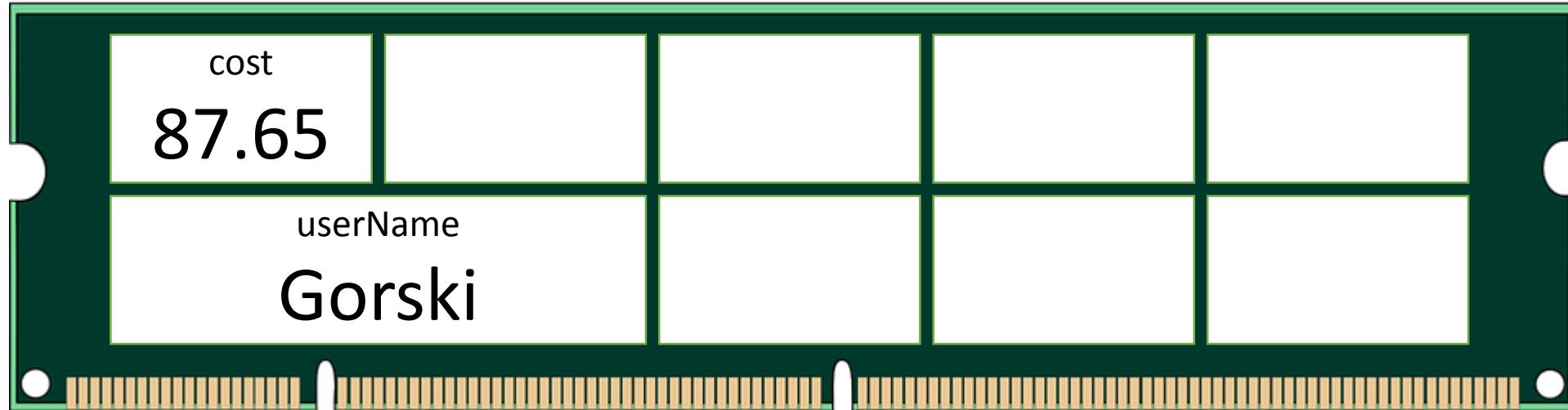
Variable Name
numSides

Choose a variable type and set the default value.

- String**
- Number** **0**
- Boolean**
- Color**

Steps to make a new variable in sphero.edu





Create a Variable

Variable Name cost

Choose a variable type and set the default value.

String Gorski

Number 87.65

Boolean

Color

Create a Variable

Variable Name user|Name

Choose a variable type and set the default value.

String Gorski

Number

Boolean

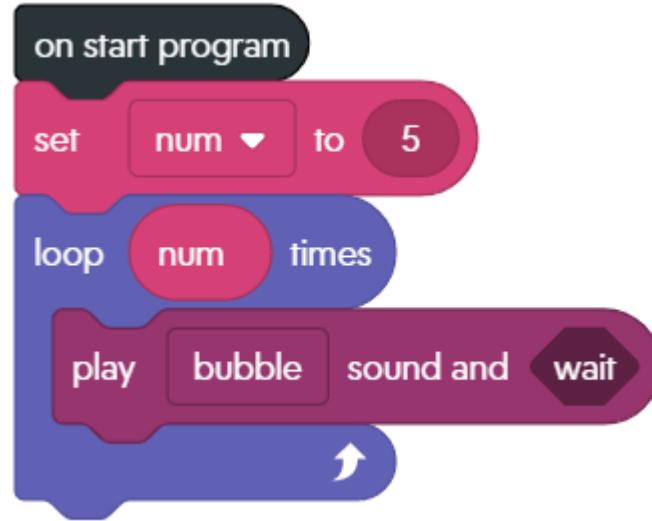
Color

A **variable** is a space in memory.
It has a name, a value, and a type.

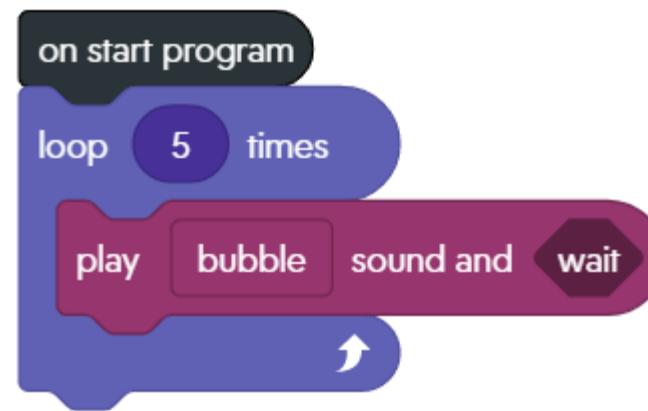
Its value can change whenever you want. Its name and type cannot.

Which type is smaller: String or Number?

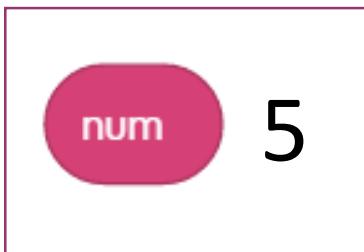
1st



2nd



variables

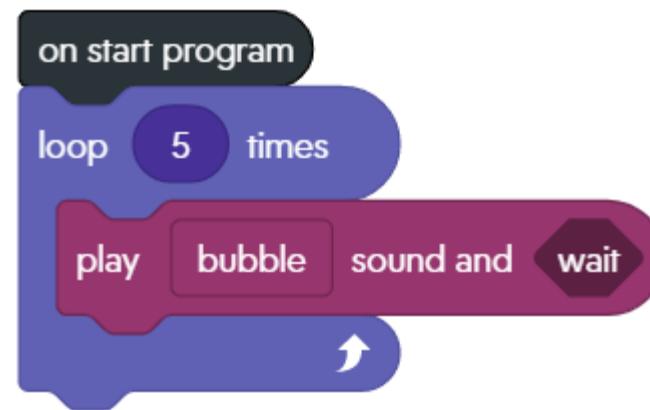


These programs
would produce
the same result.

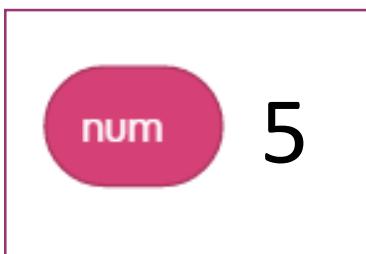
1st



2nd



variables



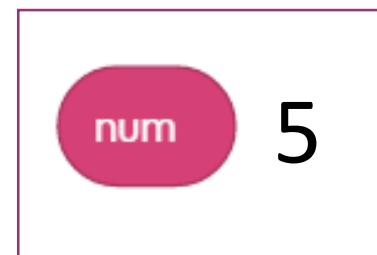
These programs would produce the same result.

What are some reasons that the first is better?

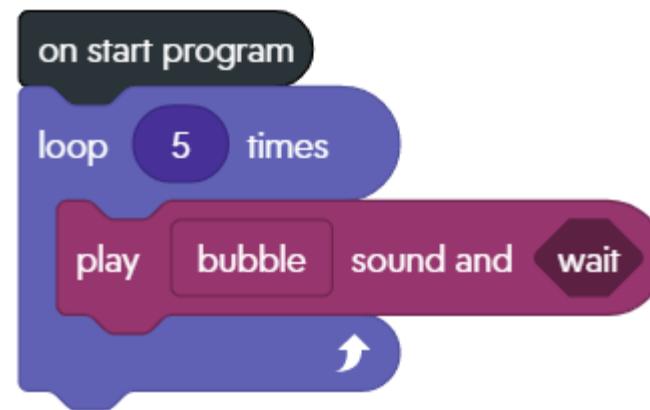
1st



variables



2nd



These programs would produce the same result.

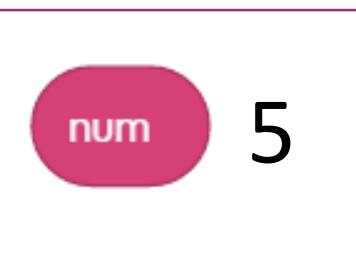
What are some reasons that the first is better?

Everything that needs editing is in a variable at the top.

1st

```
on start [ ]  
  set [num v] to [5]  
  [loop (5) [play [bubble v] sound and wait]]
```

variables



2nd

```
on start [ ]  
  [loop (5) [play [bubble v] sound and wait]]
```

These programs would produce the same result.

What are some reasons that the first is better?

Everything that needs editing is in a variable at the top.

Others using your code don't need to root through it to figure it out.

1st

```
on start [ ]  
  set [num v] to [5]  
  [loop v [5 times]  
    [play sound [bubble v] and wait v]  
  ] v]
```

variables

```
num [5 v]
```

2nd

```
on start [ ]  
  [loop v [5 times]  
    [play sound [bubble v] and wait v]  
  ] v]
```

These programs would produce the same result.

What are some reasons that the first is better?

Everything that needs editing is in a variable at the top.

Others using your code don't need to root through it to figure it out.

They edit the variable and they are done!

1st

```
on start [ ]  
  set [num v] to [5]  
  [loop (5) [play sound [bubble v] and wait]]
```

variables

```
num  
5
```

2nd

```
on start [ ]  
  [loop (5) [play sound [bubble v] and wait]]
```

These programs would produce the same result.

Depending on what I wanted to do with the code, I might use the first OR the second.

What are some reasons that the first is better?

Everything that needs editing is in a variable at the top.

Others using your code don't need to root through it to figure it out.

They edit the variable and they are done!

Basic Coding Components

Output

Changes the robot: Moves. Spins. Re-directs to a new angle. Makes sound. Speeds up. Lights up.



Math

Calculation that results in a number: +, -, *, /, square root

Let's look at math!

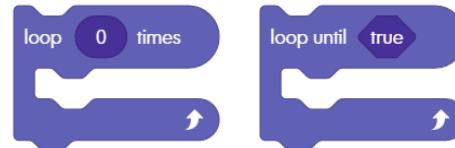
Boolean Expressions

Calculation that results in true or false: >, <, =, >=, <=, and, or, not

Control

Ifs *Decides which piece of code to run:* Uses a Boolean expression and output or math.

Loops *Repeats code:* Uses a Boolean expression and output or math.



Variables

Named pieces of memory where you can store things to use later OR to store calculation results.



Functions

Named pieces of code where you can group code together to use it later.

To build up math statements, you set with different pieces....



That you combine to create a mathematical expression:



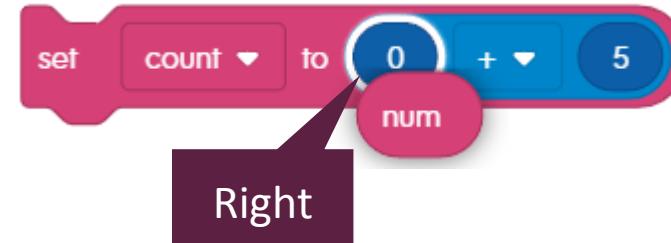
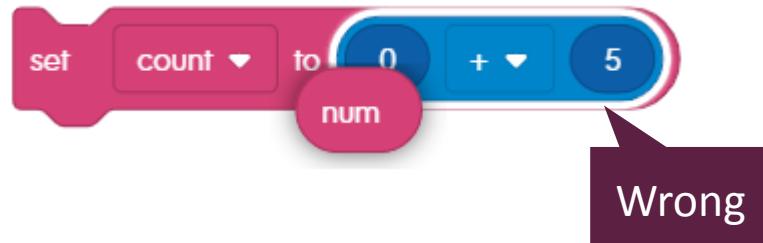
To build up math statements, you set with different pieces....



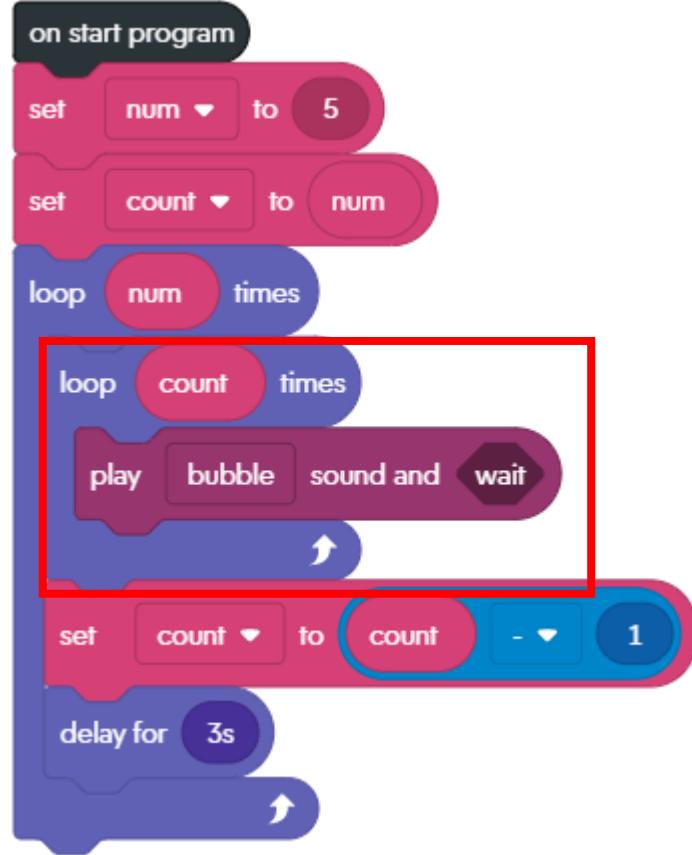
That you combine to create a mathematical expression:



Be careful that the right thing is selected before you place the block:



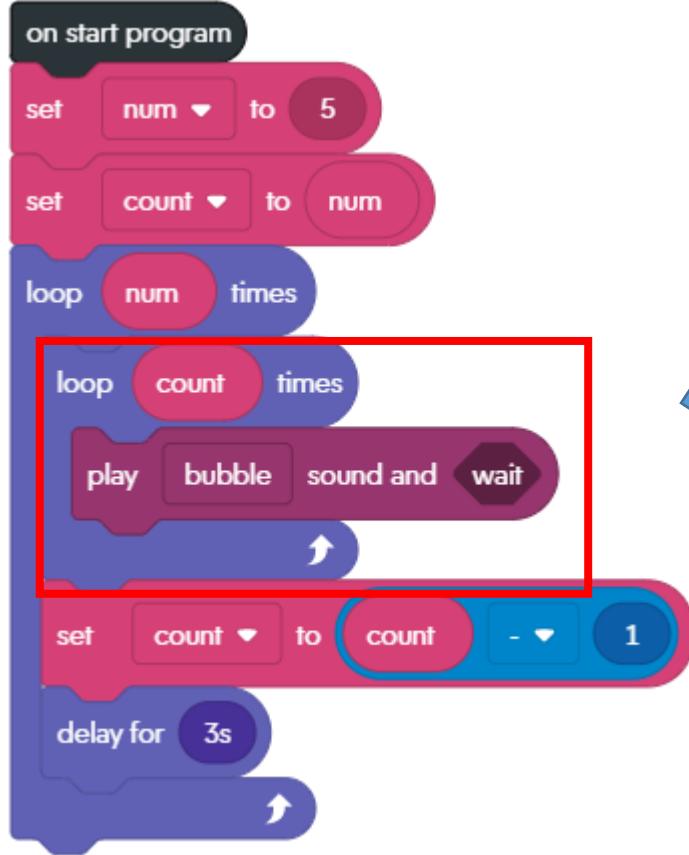
The Bubble Count Down



variable

The code on the right
puts the loop that does
the counting INSIDE
another loop.

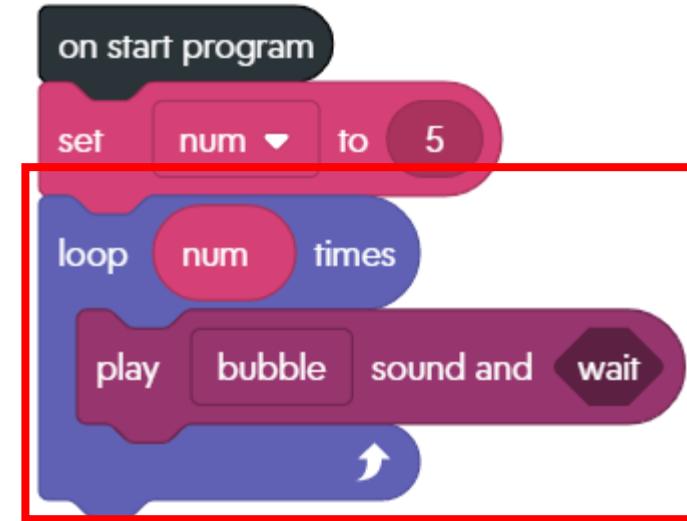
The Bubble Count Down



on start program

- set num to 5
- set count to num
- loop (num times)
 - loop (count times)
 - play bubble sound and wait
- set count to (count - 1)
- delay for 3s

variable



on start program

- set num to 5
- loop (num times)
 - play bubble sound and wait

Previous
loop inside
other loop.

The code on the right
puts the loop that does
the counting INSIDE
another loop.

The Bubble Count Down

on start program

- set num to 5
- set count to num
- loop (num times)
 - loop (count times)
 - play bubble sound and wait
 - set count to (count - 1)
- delay for 3s

variable

Outer loop

Inner loop

More Outer loop

This Scratch script starts with an "on start program" hat block. It initializes the variable "num" to 5 and the variable "count" to the value of "num". It then enters an "outer loop" that runs 5 times. Inside this loop, it enters an "inner loop" that runs the current value of "count" times. Each iteration of the inner loop plays a "bubble" sound and waits. After the inner loop completes, the script sets "count" to one less than its current value. Finally, the script waits for 3 seconds before starting the outer loop again.

on start program

- set num to 5
- loop (num times)
 - play bubble sound and wait

Previous loop inside other loop.

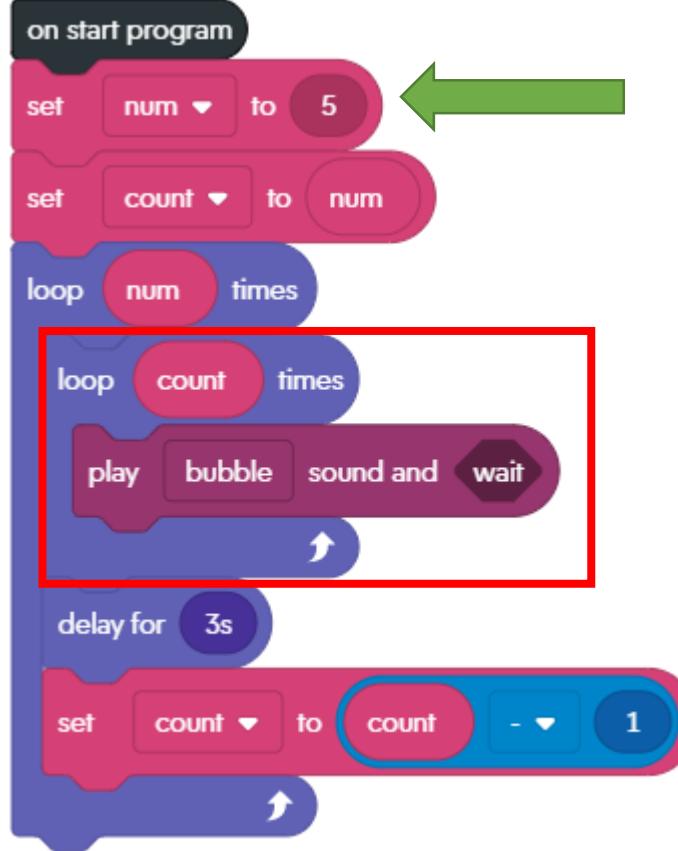
This Scratch script starts with an "on start program" hat block. It initializes the variable "num" to 5 and enters a loop that runs 5 times. Inside this loop, it plays a "bubble" sound and waits for each iteration.

The code on the right puts the loop that does the counting INSIDE another loop.

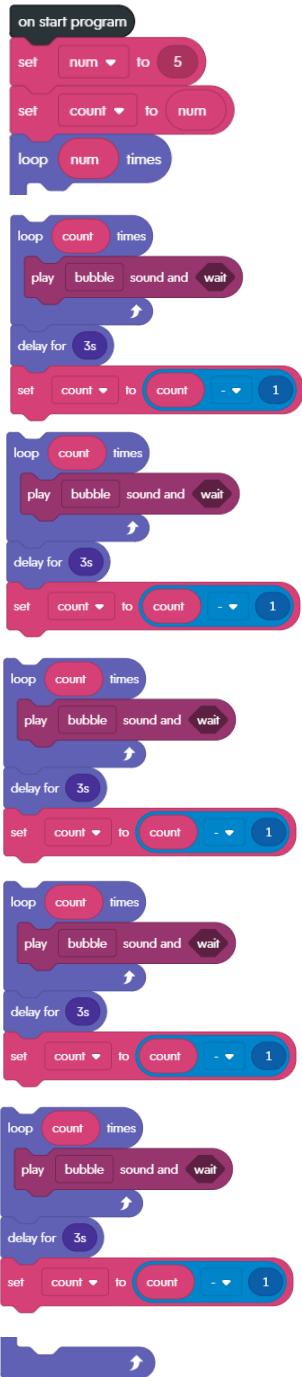
The outer loop counts down from 5 to 1.

The inner loop makes the “current count” of bubble noises

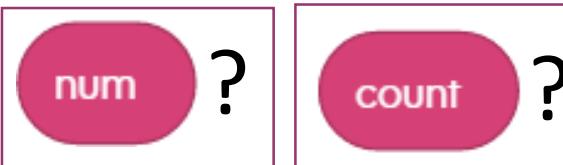
The Bubble Count Down



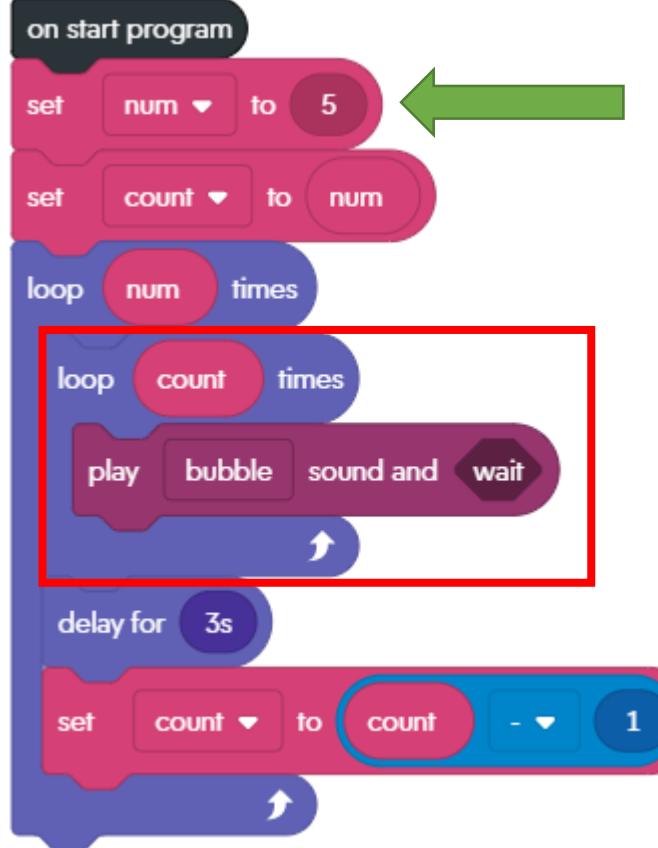
What does this line of code change?



variables

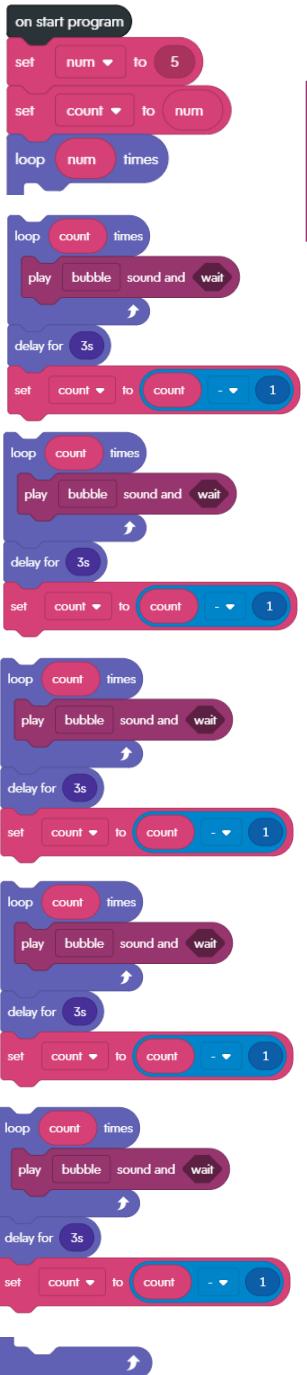


The Bubble Count Down



on start program

- set num to 5
- set count to num
- loop [play bubble sound and wait, delay for 3s, set count to count - 1] times num



on start program

- set num to 5
- set count to num
- loop [play bubble sound and wait, delay for 3s, set count to count - 1] times num
- set count to 1

variables

num 5

count ?

The num variable gets the value 5.

The Bubble Count Down

```
on start program
  set num ▾ to 5
  set count ▾ to num
  loop num times
    loop count times
      play bubble sound and wait
    delay for 3s
    set count ▾ to count - 1
```

What does this line of code change?

```
on start program
  set num ▾ to 5
  set count ▾ to num
  loop num times
    loop count times
      play bubble sound and wait
    delay for 3s
    set count ▾ to count - 1
```

variables

num 5

count ?

The Bubble Count Down

```
on start program
  set num ▾ to 5
  set count ▾ to num
  loop num times
    loop count times
      play bubble sound and wait
      delay for 3s
    set count ▾ to count - ▾ 1
  end
end
```

What do these lines of code change?

```
on start program
  set num ▾ to 5
  set count ▾ to num
  loop num times
    loop count times
      play bubble sound and wait
      delay for 3s
    set count ▾ to count - ▾ 1
  end
end
```

variables

num 5

count 5

The Bubble Count Down

```
on start program
  set num ▾ to 5
  set count ▾ to num
  loop num times
    loop count times
      play bubble sound and wait
      delay for 3s
    set count ▾ to count - 1
  delay for 3s
  set count ▾ to count - 1
```

The loop makes
5 bubble
sounds.

```
on start program
  set num ▾ to 5
  set count ▾ to num
  loop num times
    loop count times
      play bubble sound and wait
      delay for 3s
    set count ▾ to count - 1
  delay for 3s
  set count ▾ to count - 1
```

variables

num 5

count 5



The Bubble Count Down

```
on start program
  set [num v] to [5]
  set [count v] to [num]
  loop (num) [times]
    play [bubble v] sound and wait
    loop (count) [times]
      play [bubble v] sound and wait
      set [count v] to [count - 1]
    end
    delay (3s)
    set [count v] to [count - 1]
  end
end
```

What does this line of code change?

```
on start program
  set [num v] to [5]
  set [count v] to [num]
  loop (num) [times]
    play [bubble v] sound and wait
    loop (count) [times]
      play [bubble v] sound and wait
      set [count v] to [count - 1]
    end
    delay (3s)
    set [count v] to [count - 1]
  end
end
```

variables

num 5

count 5



The Bubble Count Down

```
on start program
  set [num v] to [5]
  set [count v] to [num]
  loop (num) [times]
    play [bubble v] sound and wait
    loop (count) [times]
      play [bubble v] sound and wait
      set [count v] to [count - 1]
    end
    delay for (3s)
    set [count v] to [count - 1]
  end
end
```

It causes a 3s delay to separate the bubble sounds.

variables

num 5

count 5



3s delay

```
on start program
  set [num v] to [5]
  set [count v] to [num]
  loop (num) [times]
    play [bubble v] sound and wait
    loop (count) [times]
      play [bubble v] sound and wait
      set [count v] to [count - 1]
    end
    delay for (3s)
    set [count v] to [count - 1]
  end
end
```

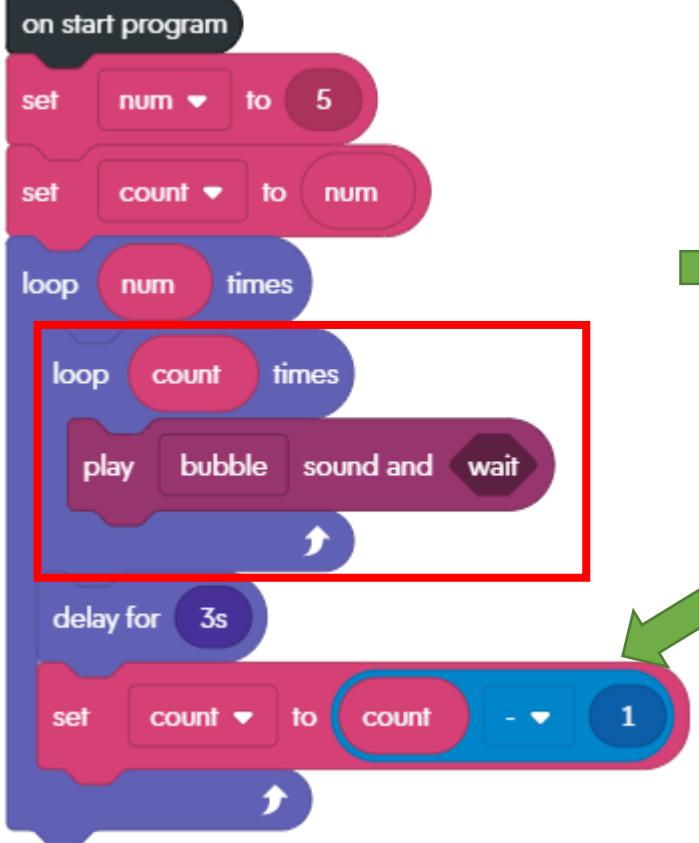
```
on start program
  set [num v] to [5]
  set [count v] to [num]
  loop (num) [times]
    play [bubble v] sound and wait
    loop (count) [times]
      play [bubble v] sound and wait
      set [count v] to [count - 1]
    end
    delay for (3s)
    set [count v] to [count - 1]
  end
end
```

```
on start program
  set [num v] to [5]
  set [count v] to [num]
  loop (num) [times]
    play [bubble v] sound and wait
    loop (count) [times]
      play [bubble v] sound and wait
      set [count v] to [count - 1]
    end
    delay for (3s)
    set [count v] to [count - 1]
  end
end
```

```
on start program
  set [num v] to [5]
  set [count v] to [num]
  loop (num) [times]
    play [bubble v] sound and wait
    loop (count) [times]
      play [bubble v] sound and wait
      set [count v] to [count - 1]
    end
    delay for (3s)
    set [count v] to [count - 1]
  end
end
```

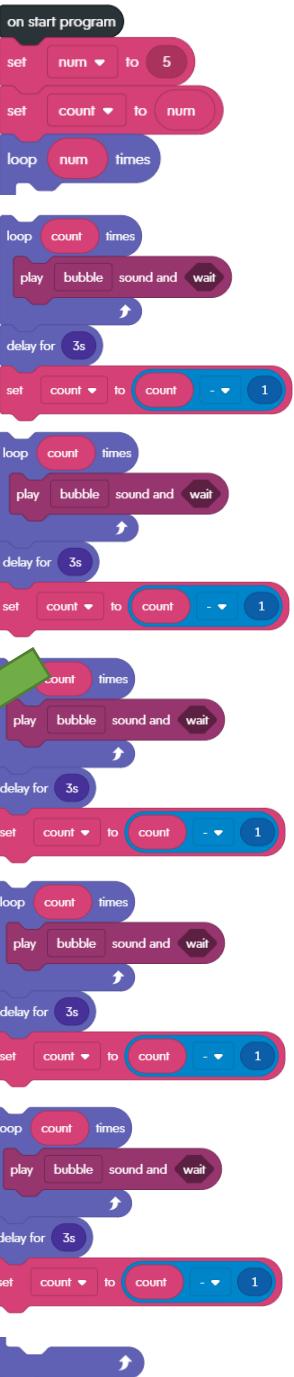
```
on start program
  set [num v] to [5]
  set [count v] to [num]
  loop (num) [times]
    play [bubble v] sound and wait
    loop (count) [times]
      play [bubble v] sound and wait
      set [count v] to [count - 1]
    end
    delay for (3s)
    set [count v] to [count - 1]
  end
end
```

The Bubble Count Down



```
on start program
  set num to 5
  set count to num
  loop (num times)
    play bubble sound and wait
    delay (3 seconds)
    set count to (count - 1)
  end
end
```

What does this line of code change?

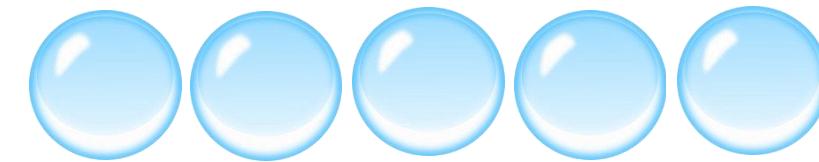


```
on start program
  set num to 5
  set count to num
  loop (num times)
    play bubble sound and wait
    delay (3 seconds)
    set count to (count - 1)
  end
end
```

variables

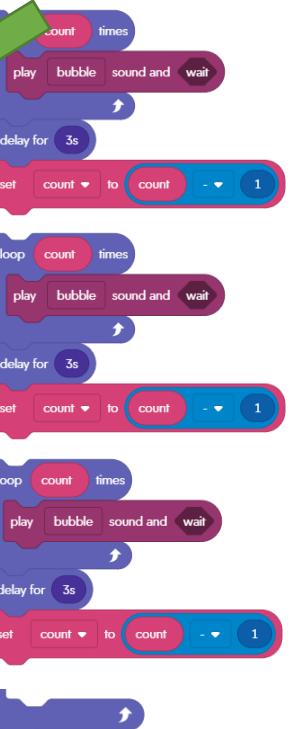
num 5

count 5

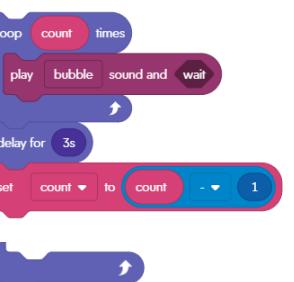


3s delay

count ?



```
on start program
  set num to 5
  set count to num
  loop (num times)
    play bubble sound and wait
    delay (3 seconds)
    set count to (count - 1)
  end
end
```



```
on start program
  set num to 5
  set count to num
  loop (num times)
    play bubble sound and wait
    delay (3 seconds)
    set count to (count - 1)
  end
end
```



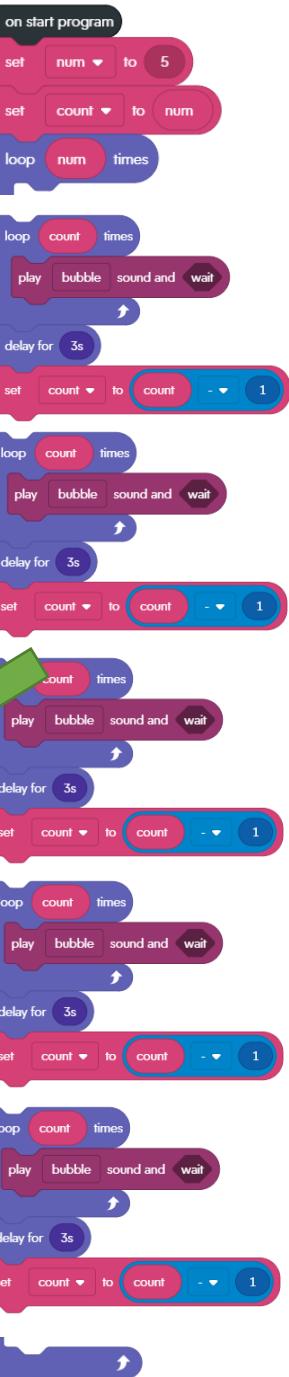
The Bubble Count Down



```
on start program
  set num to 5
  set count to num
  loop (num) [play bubble sound and wait
    loop (count) [play bubble sound and wait
    ]
    delay (3s)
    set count to (count - 1)
  ]

```

Count becomes 4.



```
on start program
  set num to 5
  set count to 5
  loop (num) [play bubble sound and wait
    loop (count) [play bubble sound and wait
    ]
    delay (3s)
    set count to (count - 1)
  ]

```

variables

num 5

count 5



3s delay

count 4



The Bubble Count Down

```
on start program
  set num ▾ to 5
  set count ▾ to num
  loop num times
    loop count times
      play bubble sound and wait
    delay for 3s
    set count ▾ to count - ▾ 1
  end
end
```

What does this line of code change?

```
on start program
  set num ▾ to 5
  set count ▾ to num
  loop num times
    loop count times
      play bubble sound and wait
    delay for 3s
    set count ▾ to count - ▾ 1
  end
end
```

variables

num 5

count 5



3s delay

count 4



The Bubble Count Down

```
on start program
  set num ▾ to 5
  set count ▾ to num
  loop num times
    loop count times
      play bubble sound and wait
    end
    delay for 3s
    set count ▾ to count - 1
  end
end
```

Now, only 4 bubbles are displayed.

```
on start program
  set num ▾ to 5
  set count ▾ to num
  loop num times
    loop count times
      play bubble sound and wait
    end
    delay for 3s
    set count ▾ to count - 1
  end
end
```

variables

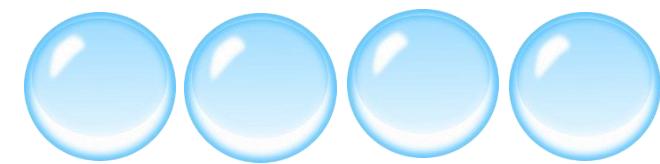
num 5

count 5

count 4



3s delay



The Bubble Count Down

```
on start program
  set num to 5
  set count to num
  loop (num)
    play bubble sound and wait
    delay (3 seconds)
  end
  set count to (count) - 1
  next
```

What does this line of code change?

```
on start program
  set num to 5
  set count to num
  loop (count)
    play bubble sound and wait
    delay (3 seconds)
  end
  set count to (count) - 1
  next
```

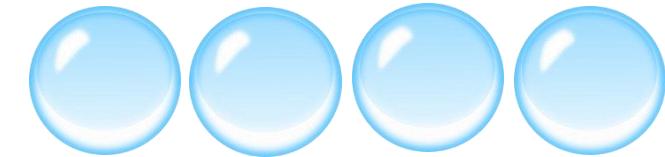
variables

num 5

count 5

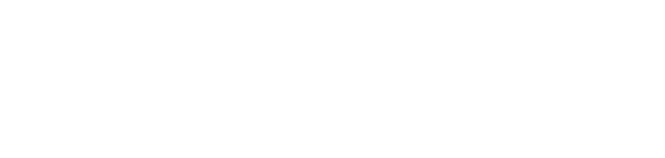
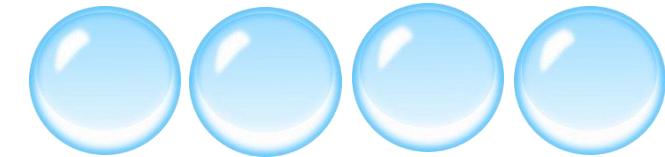


3s delay

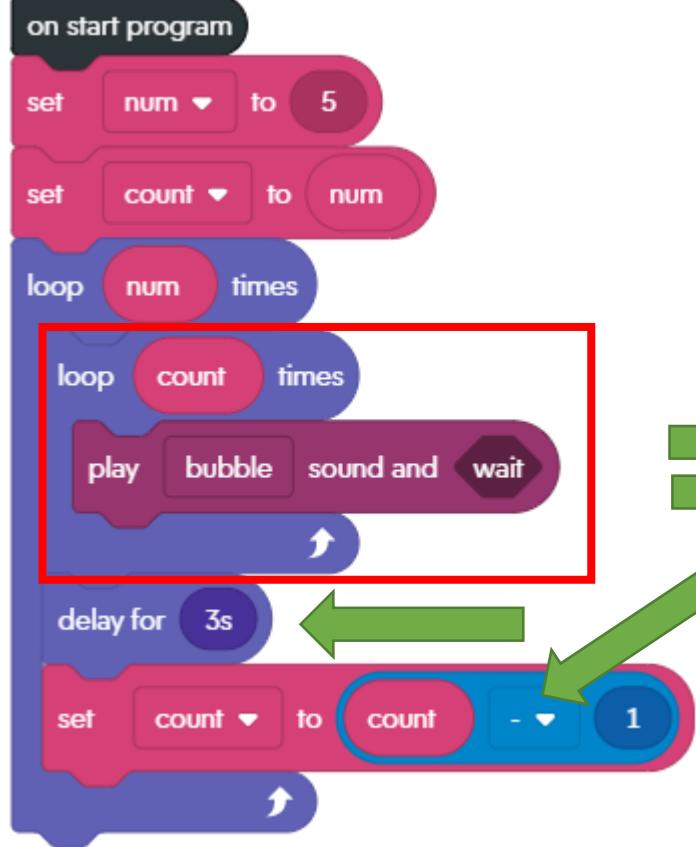


count 4

count ?



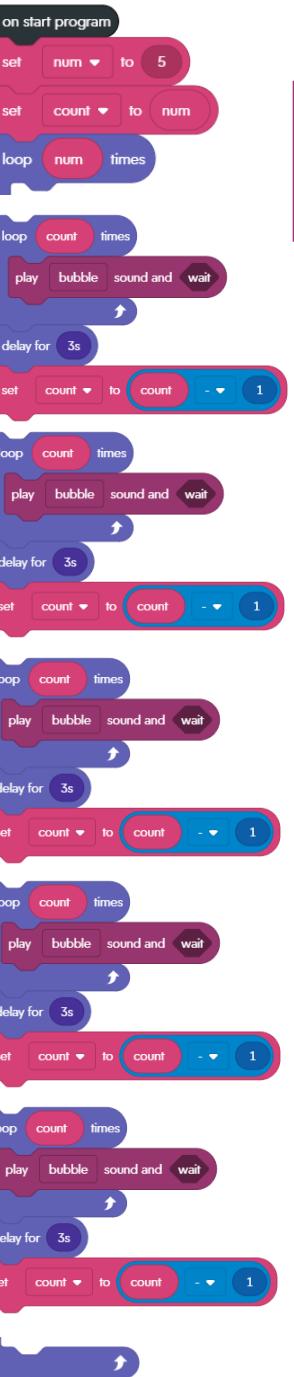
The Bubble Count Down



on start program

- set num to 5
- set count to num
- loop num times
 - loop count times
 - play bubble sound and wait
 - delay for 3s
 - set count to count - 1

3s delay and the count decreases.



on start program

- set num to 5
- set count to num
- loop num times
 - loop count times
 - play bubble sound and wait
 - delay for 3s
 - set count to count - 1

variables

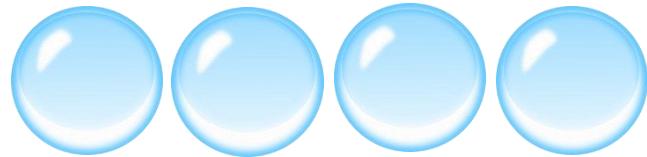
num 5

count 5



3s delay

count 4

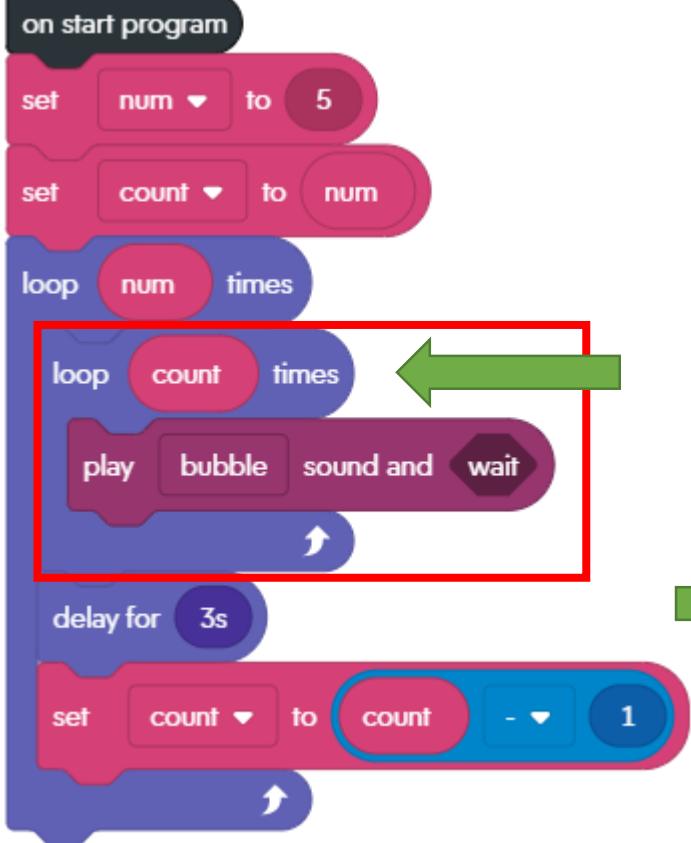


3s delay

count 3



The Bubble Count Down



on start program

- set num to 5
- set count to num
- loop num times
 - loop count times
 - play bubble sound and wait
 - delay for 3s
 - set count to count - 1

What does this line of code change?



on start program

- set num to 5
- set count to num
- loop num times
 - loop count times
 - play bubble sound and wait
 - delay for 3s
 - set count to count - 1

variables

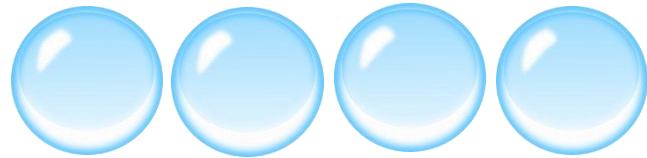
num 5

count 5



3s delay

count 4



3s delay

count 3



The Bubble Count Down

on start program

- set num to 5
- set count to num
- loop num times
 - loop count times
 - play bubble sound and wait
 - delay for 3s
 - set count to count - 1

3 bubble sounds this time in the loop.

on start program

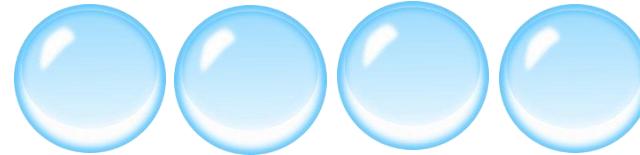
- set num to 5
- set count to num
- loop num times
 - loop count times
 - play bubble sound and wait
 - delay for 3s
 - set count to count - 1

variables

num 5

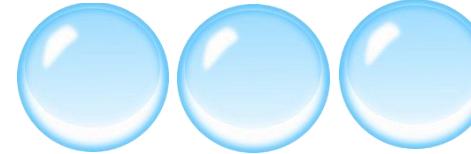
count 5

3s delay



count 4

3s delay



count 3

The Bubble Count Down

```
on start program
  set [num v] to [5]
  set [count v] to [num]
  loop (count) [times]
    play [bubble v] sound and wait
    delay (3s)
    loop (count) [times]
      play [bubble v] sound and wait
      delay (3s)
      set [count v] to [count - 1]
    end
  end
end
```

What do these lines of code change?

variables

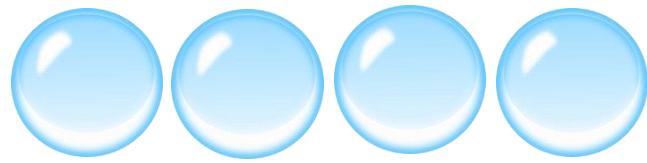
num 5

count 5



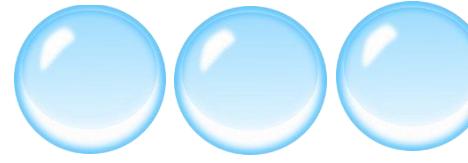
3s delay

count 4



3s delay

count 3



count ?



The Bubble Count Down

```
on start program
  set num to 5
  set count to num
  loop (num) [play bubble sound and wait
    delay (3s)
    loop (count) [play bubble sound and wait
      delay (3s)
      set count to (count) - 1
    ]
  ]

```

3s delay and
count
decreases.

variables

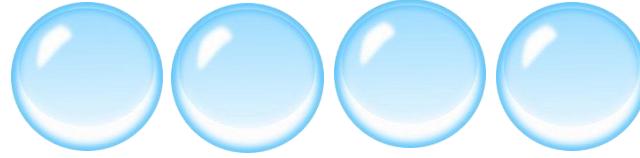
num 5

count 5



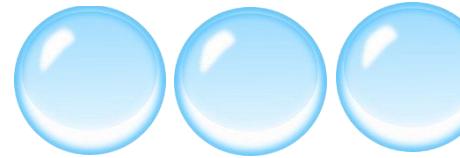
3s delay

count 4



3s delay

count 3



3s delay

count 2

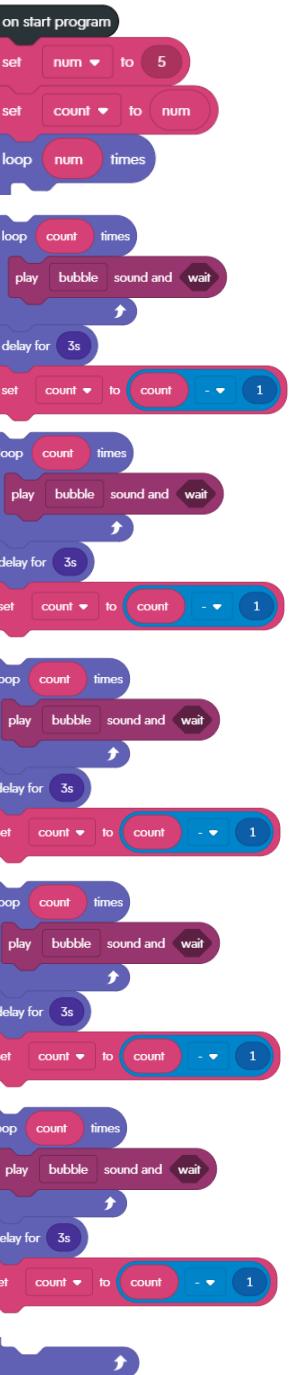


The Bubble Count Down



```
on start program
  set num to 5
  set count to num
  loop (count) [play bubble sound and wait
    delay (3 seconds)
    set count to (count) - 1]
  end
```

The 2 Bubbles
are displayed.



```
on start program
  set num to 5
  set count to num
  loop (count) [play bubble sound and wait
    delay (3 seconds)
    set count to (count) - 1]
  end
```

variables

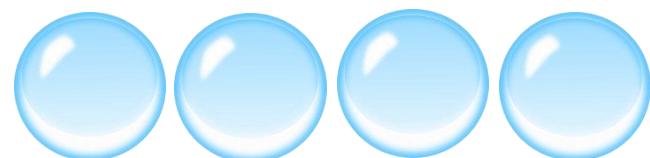
num 5

count 5



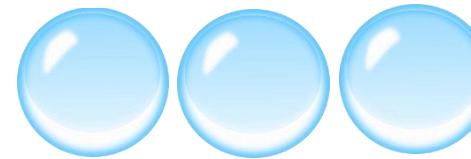
3s delay

count 4



3s delay

count 3



3s delay

count 2



The Bubble Count Down

```
on start program
  set num ▾ to 5
  set count ▾ to num
  loop num times
    loop count times
      play bubble sound and wait
      delay for 3s
    set count ▾ to count - 1
  end
```

We delay and decrease count.

```
on start program
  set num ▾ to 5
  set count ▾ to num
  loop num times
    loop count times
      play bubble sound and wait
      delay for 3s
    set count ▾ to count - 1
  end
```

variables

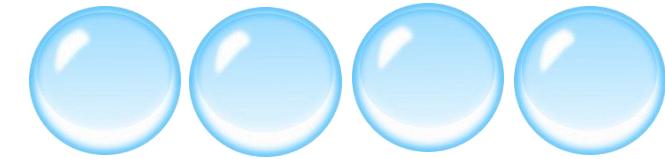
num 5

count 5



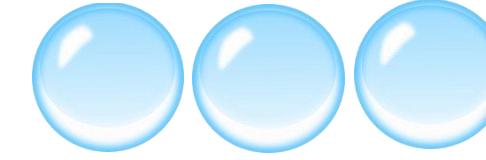
3s delay

count 4



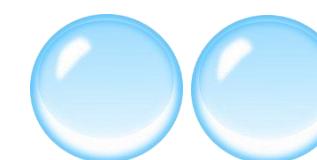
3s delay

count 3



3s delay

count 2



3s delay

count 1



The Bubble Count Down

on start program

- set num to 5
- set count to num
- loop num times
 - loop count times
 - play bubble sound and wait
 - delay for 3s
 - set count to count - 1

The one bubble is displayed.

on start program

- set num to 5
- set count to num
- loop num times
 - loop count times
 - play bubble sound and wait
 - delay for 3s
 - set count to count - 1

variables

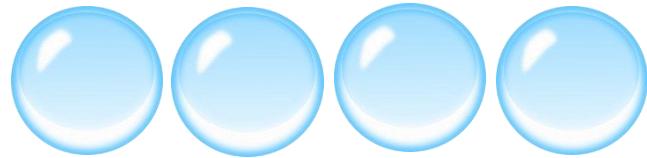
num 5

count 5



3s delay

count 4



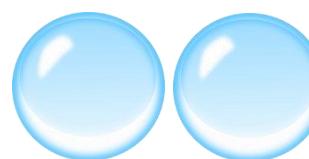
3s delay

count 3



3s delay

count 2



3s delay

count 1



The Bubble Count Down

```
on start program
  set num ▾ to 5
  set count ▾ to num
  loop num times
    loop count times
      play bubble sound and wait
      delay for 3s
    set count ▾ to count - 1
  end
```

We delay 3s and decrease count.

```
on start program
  set num ▾ to 5
  set count ▾ to num
  loop num times
    loop count times
      play bubble sound and wait
      delay for 3s
    set count ▾ to count - 1
  end
```

variables

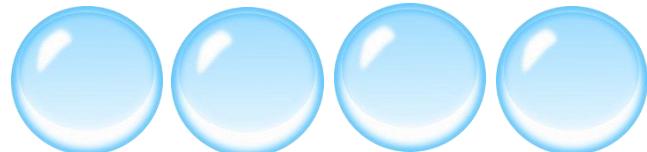
num 5

count 5



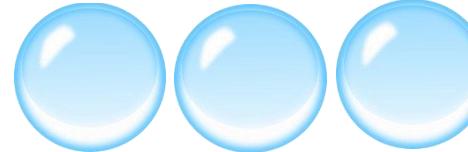
3s delay

count 4



3s delay

count 3



3s delay

count 2



3s delay

count 1



3s delay

count 0

The Bubble Count Down



```
on start program
  set num ▾ to 5
  set count ▾ to num
  loop num times
    loop count times
      play bubble sound and wait
    delay for 3s
    set count ▾ to count - 1
  end
  set count ▾ to 1
```

Both loops are now finished

```
on start program
  set num ▾ to 5
  set count ▾ to num
  loop num times
    loop count times
      play bubble sound and wait
    delay for 3s
    set count ▾ to count - 1
  end
  set count ▾ to 1
```

variables

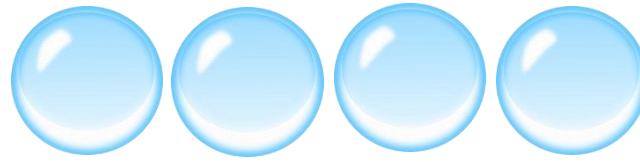
num 5

count 5



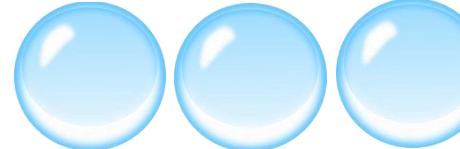
3s delay

count 4



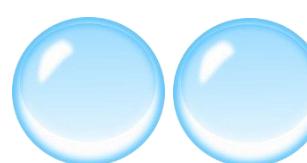
3s delay

count 3



3s delay

count 2



3s delay

count 1



3s delay

count 0



Basic Coding Components

Output

Changes the robot: Moves. Spins. Re-directs to a new angle. Makes sound. Speeds up. Lights up.



Math

Calculation that results in a number: +, -, *, /, square root



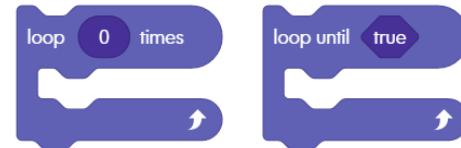
Boolean Expressions

Calculation that results in true or false: >, <, =, >=, <=, and, or, not

Control

Ifs *Decides which piece of code to run:* Uses a Boolean expression and output or math.

Loops *Repeats code:* Uses a Boolean expression and output or math.



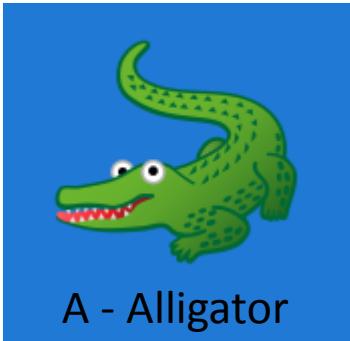
Variables

Named pieces of memory where you can store things to use later OR to store calculation results.

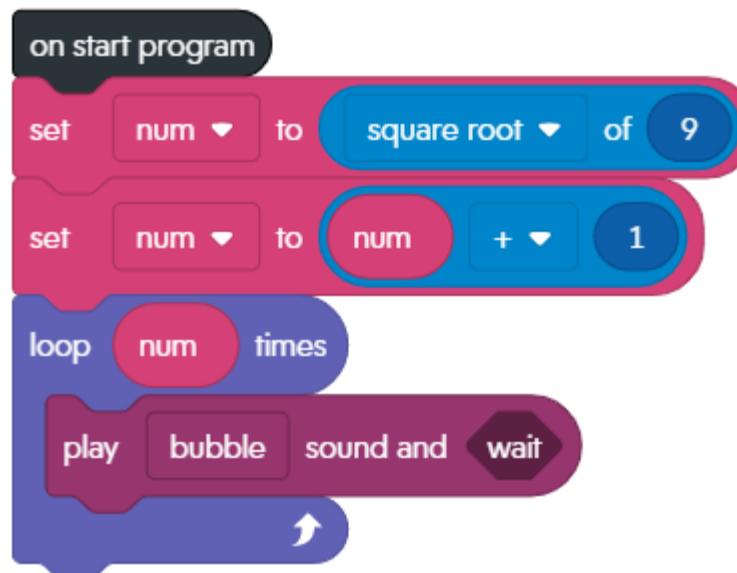


Functions

Named pieces of code where you can group code together to use it later.



In this code, how many bubble sounds are played? Make 2 new math equations using other functions to result in 8 bubble sounds being played.





B - Bat



Run this code and figure out what it does.

Make two new math equations that make it run 7 times. Make the colour sequence inside the loop red, green, blue.

Test it on your Sphero.

```
on start [ ]\n  set [num v] to [3]\n  [set [num v] to [num v] + [1] v\n    [loop [6] [\n      [main LED v]\n      [delay for [2s] v]\n      [main LED v]\n      [delay for [2s] v]\n      [main LED v]\n    ]]]\n  [main LED v]
```



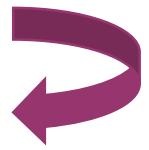
Run this code and figure out what it does. (Put the Sphero on the ground first)

Make two new math equations that make it run 5 times. Make the Sphero roll in a zig-zag pattern.

```
on start [ ] 
  set [num v] to [square root of 9]
  [set [num v] to [num v] + [1] v
  repeat (5) [
    [roll [0°] at [15] speed for [2s]
    [delay for [2s]
    [main LED [white v]
    [roll [0°] at [15] speed for [2s]
    [delay for [2s]
    [main LED [blue v]
    [roll [0°] at [15] speed for [2s]
    [delay for [2s]
    [main LED [orange v]
  end
end
```

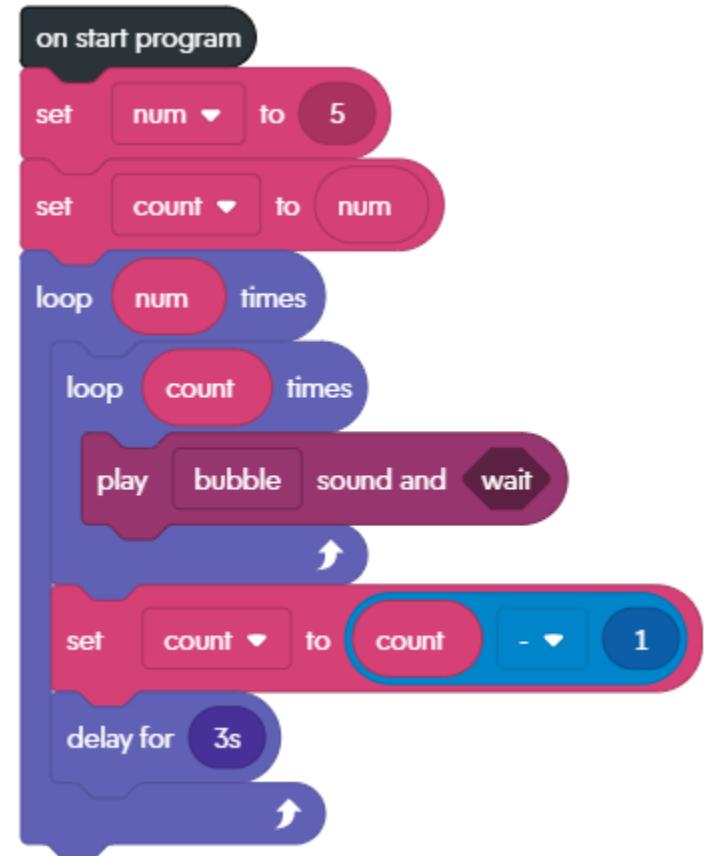


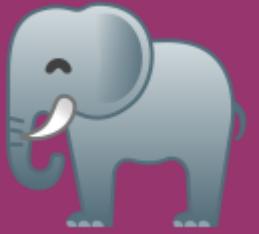
D - Dog



Make the Bubble sounds (or whatever you choose) count up instead of counting down.

Test it on your Sphero.



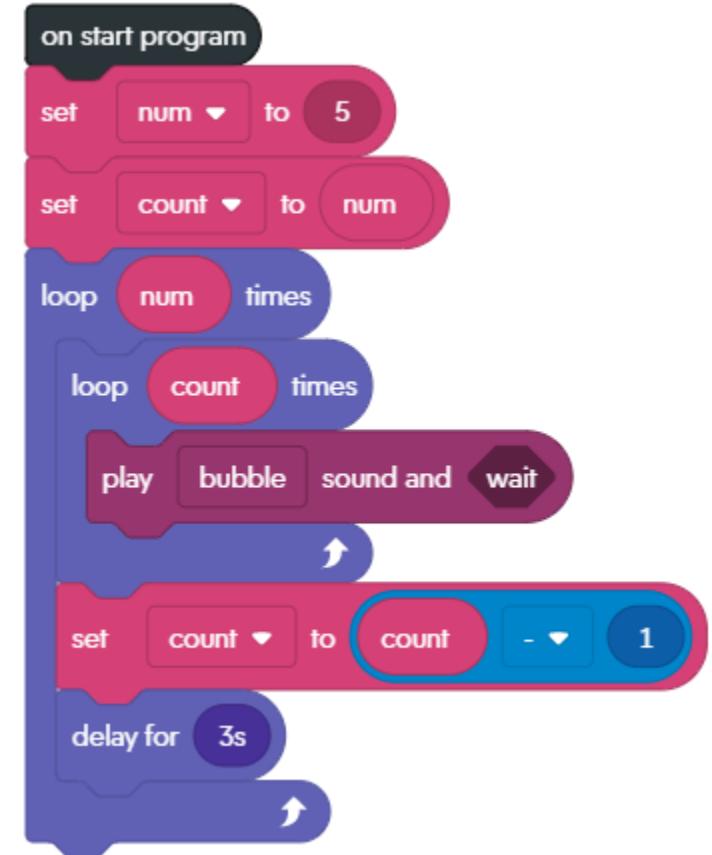


E - Elephant



Make the Bubble sounds (or whatever you choose) start at 12 and count down by 2s.

Test it on your Sphero.



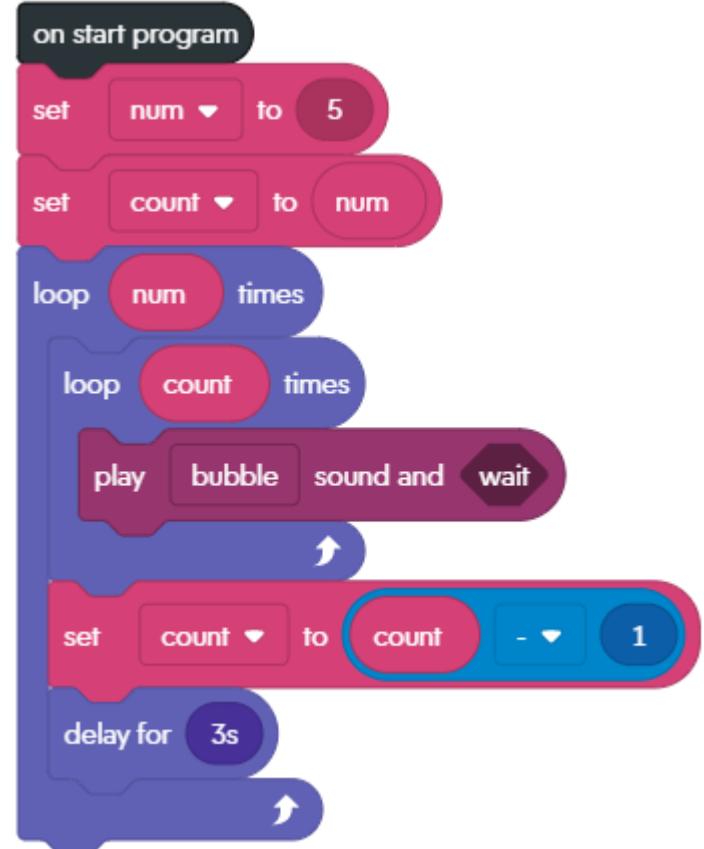


F - Fish



Make the Bubble sounds (or whatever you choose) start at 3 and count up to 15 by 3s.

Test it on your Sphero.

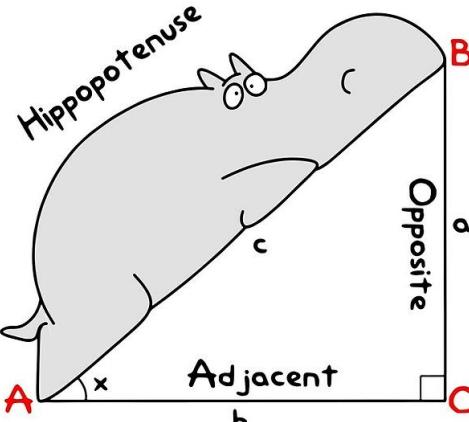




G - Giraffe



Write a program that calculates the size of the hypotenuse of a triangle.



A diagram of a right-angled triangle with vertices labeled A, B, and C. Vertex C is at the bottom right and is marked with a right-angle symbol. The horizontal side is labeled 'Adjacent' and the vertical side is labeled 'Opposite'. The hypotenuse, labeled 'Hippopoteneuse', is labeled 'c' and has a small drawing of a hippopotamus on it. The adjacent side is labeled 'b' and the opposite side is labeled 'a'.

$c = \sqrt{a^2 + b^2}$

Scratch script:

- on start program
 - set **a** to 3
 - set **b** to 4
 - set **c** to [square root of ? * ? + ? * ?]
- loop [? times]
 - play bubble sound and wait



H - Horse



Write a program that counts down from ten and then “blasts off”. You can do this with a loop. You can also start the Sphero moving after the “launch sequence”.

Test it on your Sphero.

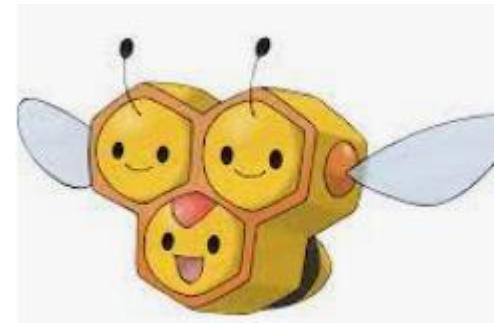
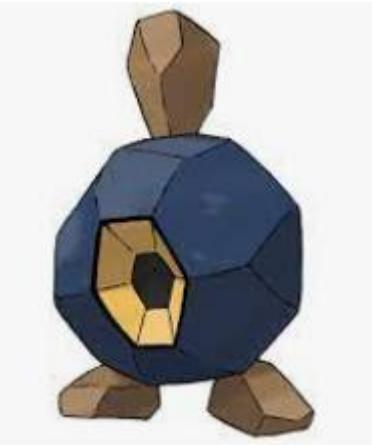




I – Iguana



Write a Sphero program that creates an hexagon. This can be done using a variable for the angle.



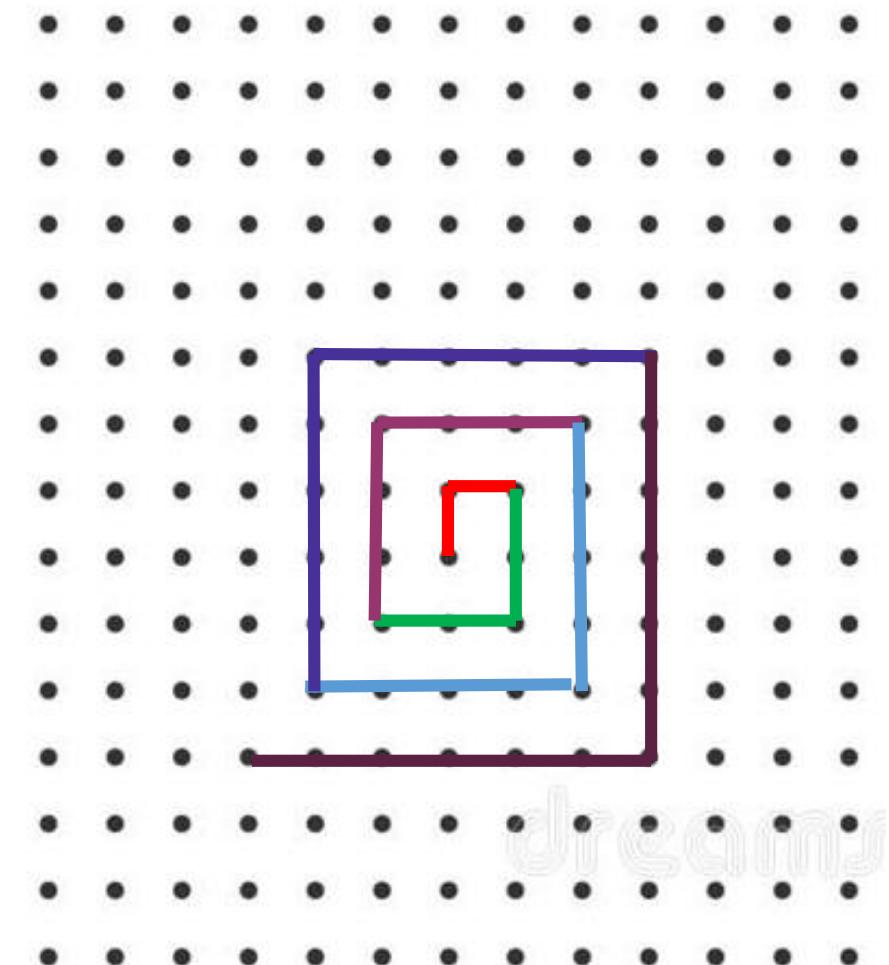


TIPS

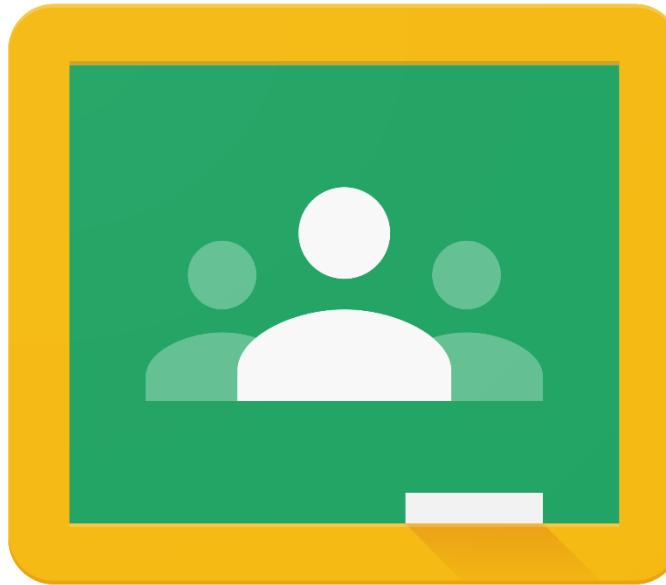
J - Jaguar



Write a program that allows the Sphero to travel in a spiral pattern like this Anishinaabe basket.



When you are done,
there are check-your-
understanding
questions on Google
Classroom.



Google Classroom