Announcements

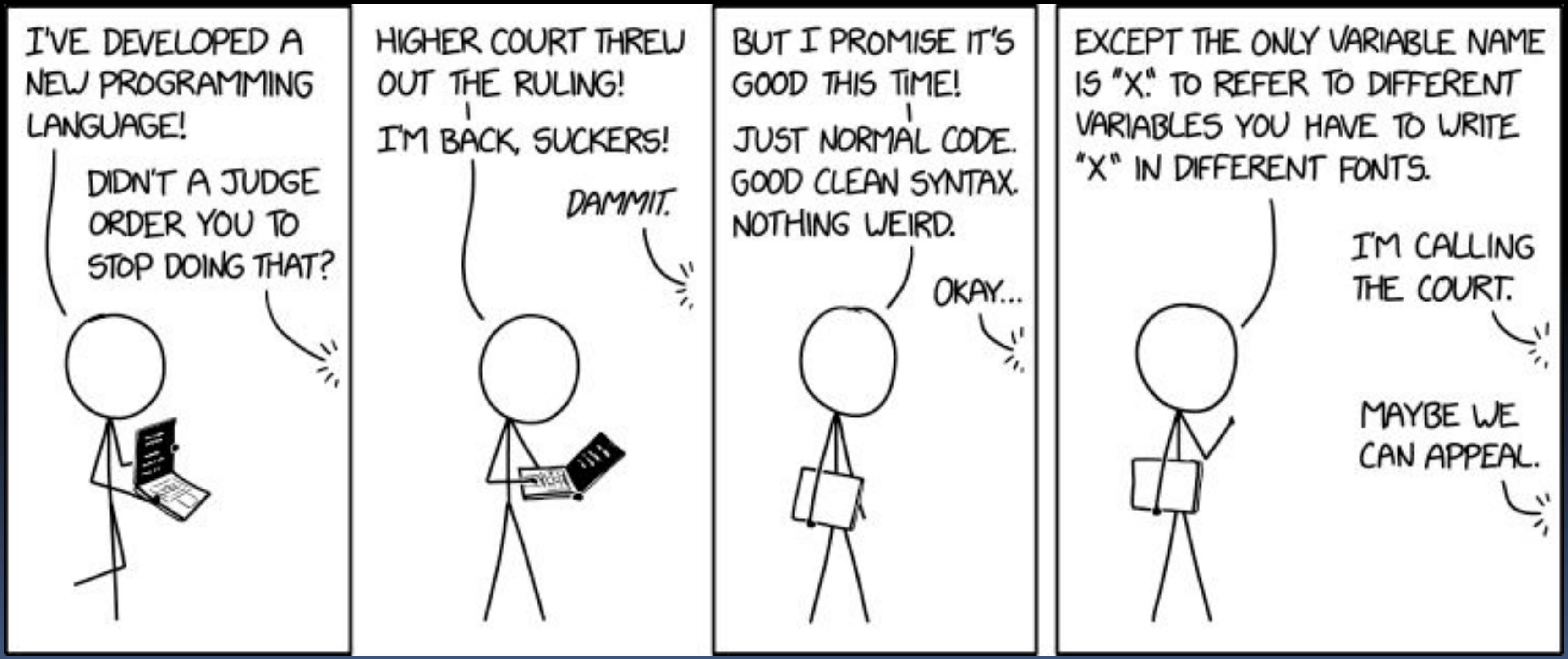
- Log into iClicker. To enroll, go to EdStem post #4
- Added CS10 recently? Read this EdStem post
- Project 1 due tonight
- Come to the front after class to meet each other to partner up

UC Berkeley
Teaching Professor
Dan Garcia

The Beauty and Joy of Computing

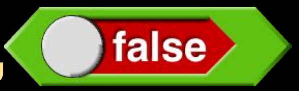
Variables, Scope, Lists, and HOFs

xkcd.com/2309

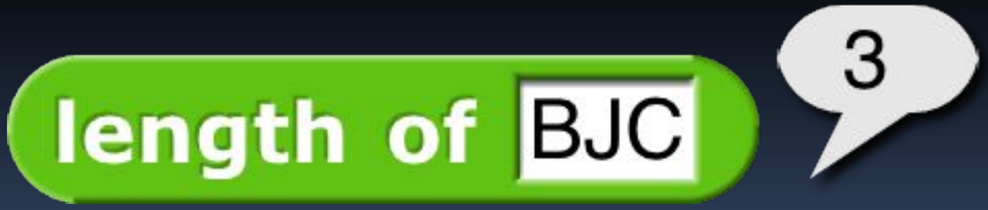

I'VE DEVELOPED A NEW PROGRAMMING LANGUAGE!
DIDN'T A JUDGE ORDER YOU TO STOP DOING THAT?
HIGHER COURT THREW OUT THE RULING!
I'M BACK, SUCKERS!
DAMMIT.
BUT I PROMISE IT'S GOOD THIS TIME!
JUST NORMAL CODE. GOOD CLEAN SYNTAX. NOTHING WEIRD.
OKAY...
EXCEPT THE ONLY VARIABLE NAME IS "X." TO REFER TO DIFFERENT VARIABLES YOU HAVE TO WRITE "X" IN DIFFERENT FONTS.
I'M CALLING THE COURT.
MAYBE WE CAN APPEAL.

Variables

Review: Types & Domain/Range

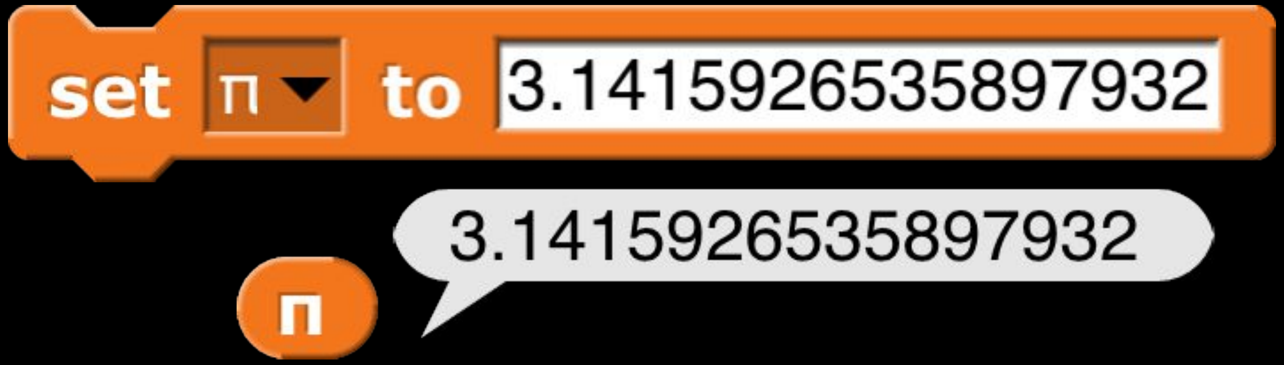
- Type: The classification of our data
  - E.g., 42 is a *number*
  - E.g., Hello World! is a sentence
  - E.g.,  is a Boolean

- Domain: The type(s) of <u>input</u> of a procedure
- Range: The type of <u>output</u> a function reports



What is a variable?

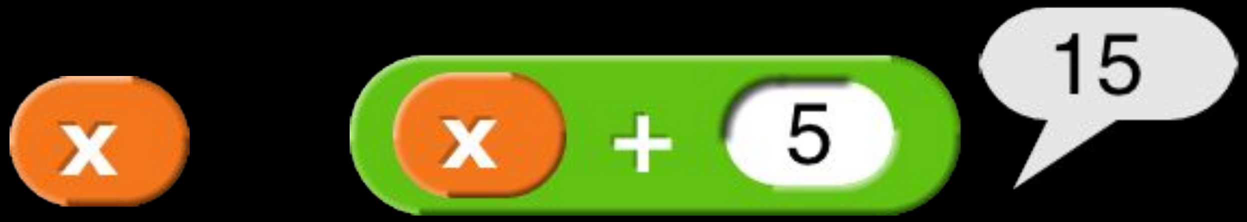
- A variable is a storage location for data values. The set block binds names to values.
  - Names (aka Identifiers)
  - Values
  - E.g.,



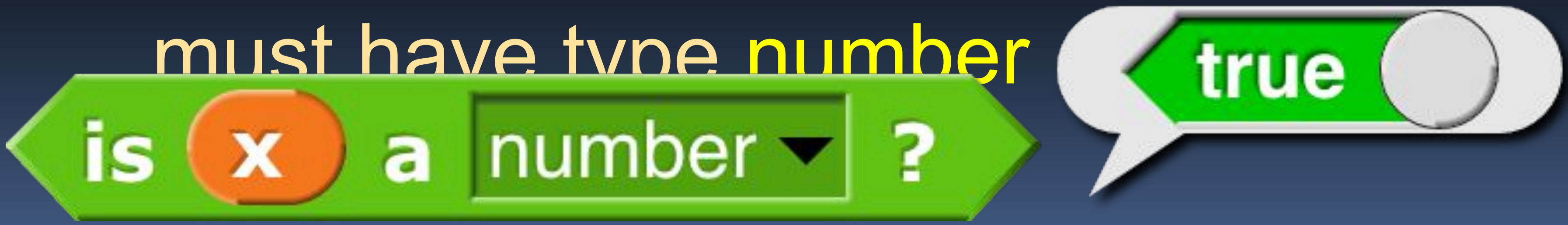
- The variable name is what we use to reference the stored value.
- This allows us to use the name independently of what it actually represents.
- This is one form of abstraction!

Variables and Types (1/2)

- The types of the variables used as input must be within the domain of the procedures.
  - E.g., What's the domain of the procedure +?

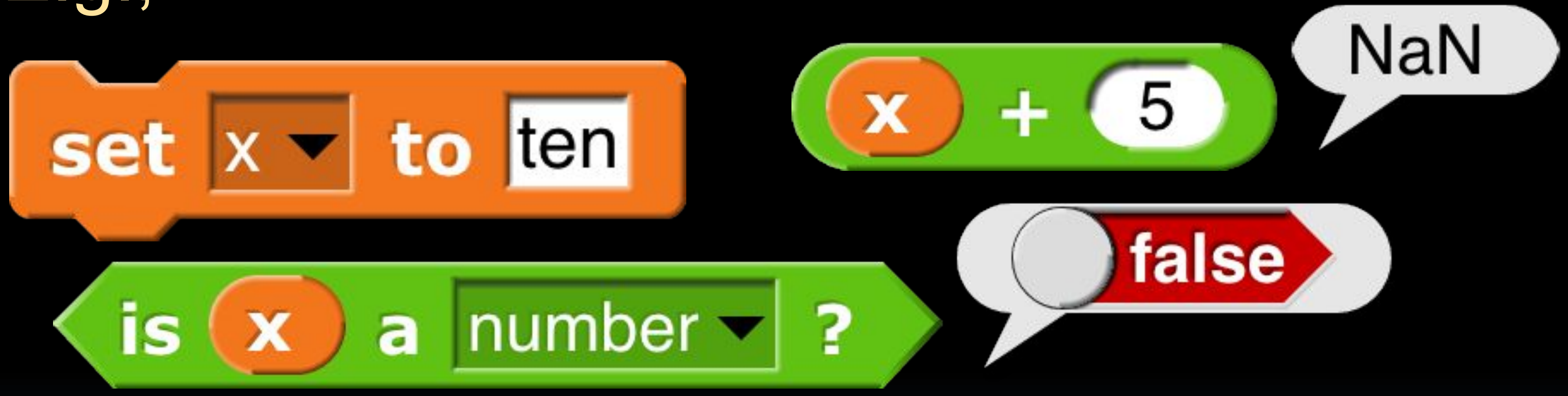


must have type number



Variables and Types (2/2)

- What happens if the domain is not satisfied? I.e., what if the variable's type isn't in the function's domain?
  - E.g.,

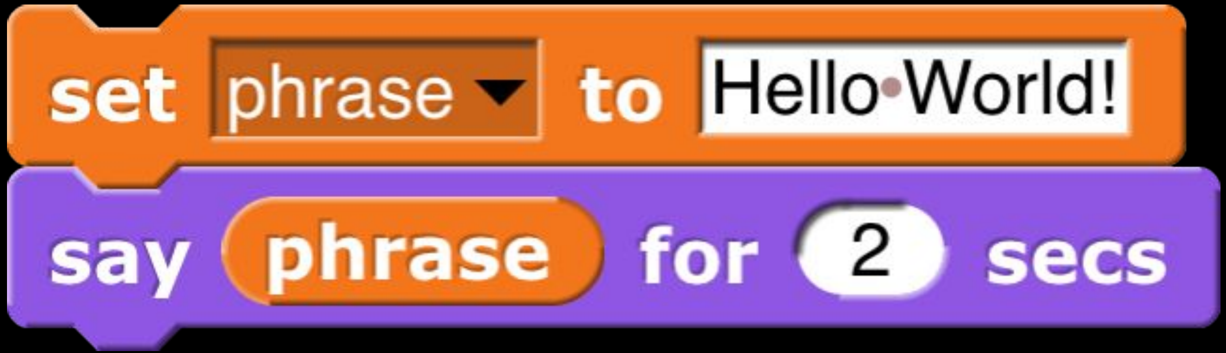


- The functions will yield undefined behavior.
  - But it's not the function's fault!
  - The call violated the spec!

Scope

Variable Scope

- The scope of a variable is the part(s) of code where that variable name binding is valid.
  - I.e. where it exists…

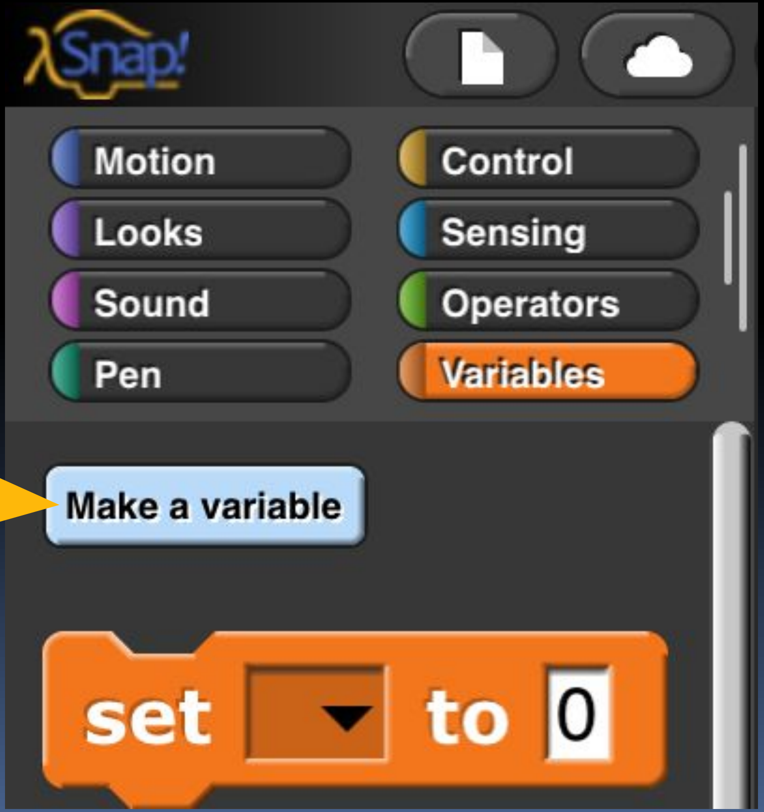


```
set phrase to Hello World!
say phrase for 2 secs
```

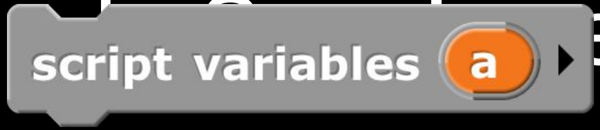
- We can run the above code successfully phrase e is in scope when the **say** block asks for its value.

Global Variables

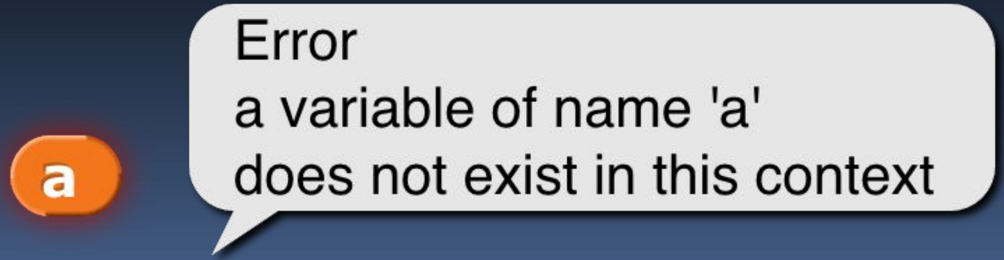
- A variable that is declared globally is accessible anywhere in the code
- Global variables have global scope
- In Snap*!*, we create global variables by clicking the "Make a variable" button in the "Variables" tab



Local (Script) Variables (1/2)

- A local variable is only accessible in the program's current scope, which in Snap*!* is the current script
  - I.e., the current set of connected commands
- Local variables have local scope
- 

e create local variables with the block in the "Variables" tab
- When variables are out of scope, we get this:

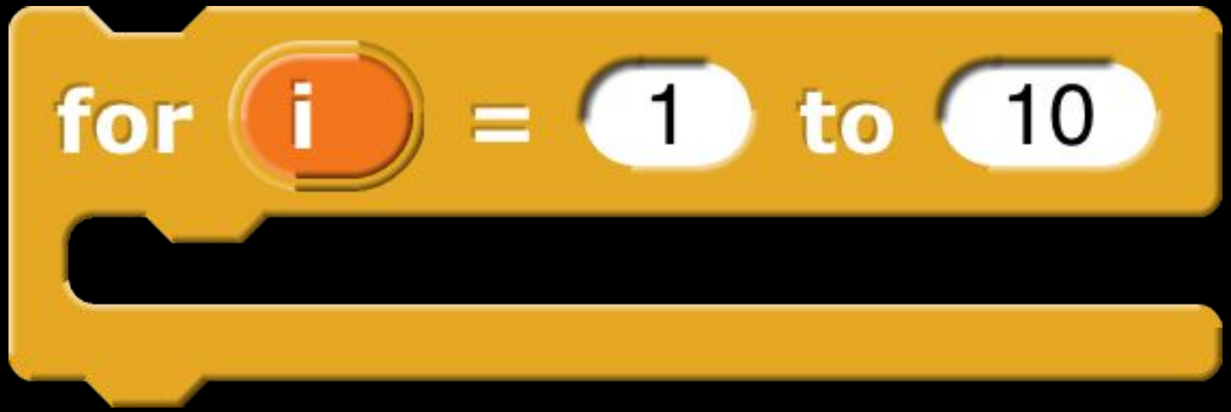


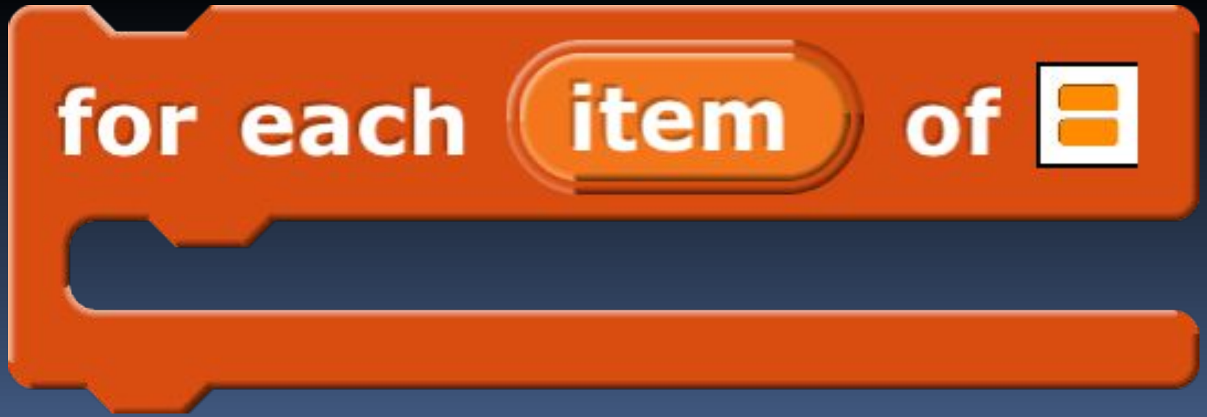
Error
a variable of name 'a'
does not exist in this context

Garcia

Local (Script) Variables (2/2)

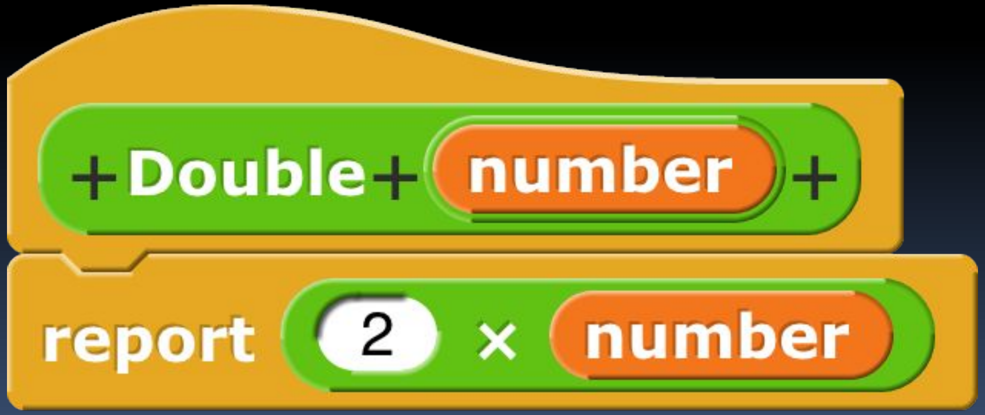
- Other ways to create local variables
  - The scope is inside the slot AND the rest of the script



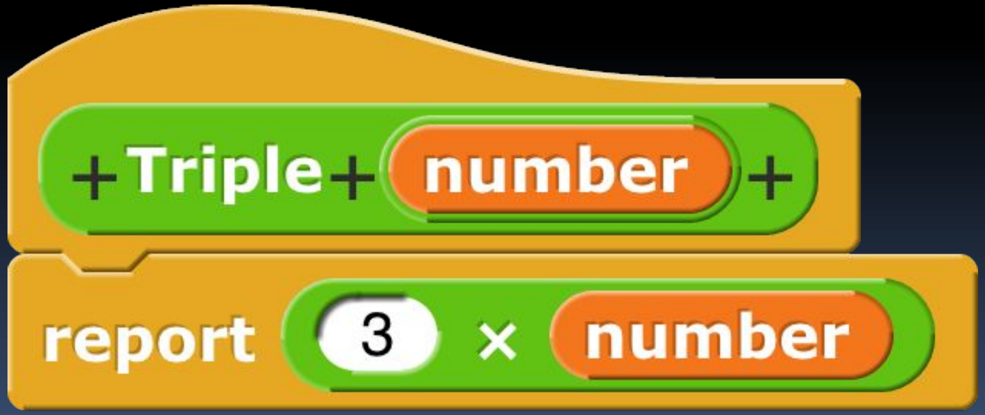


Variable Scope in Functions

- Procedures definitions (i.e., commands and reporters) create a new scope
- Inputs are variables that only exist within the procedure in which they are defined
  - This allows procedures to have the same input names as others with no conflicts, yay!



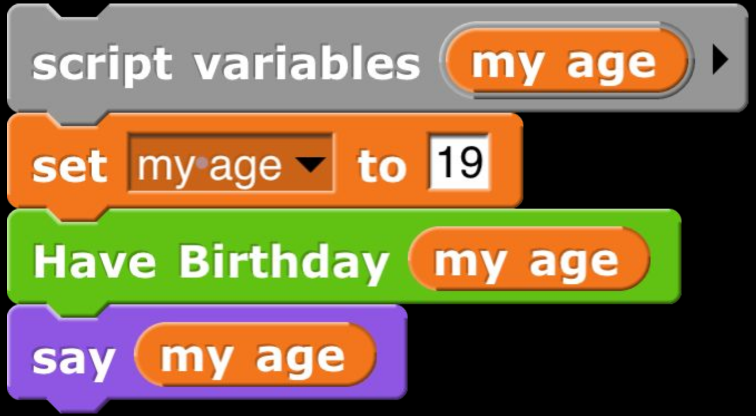
```
+Double+ (number) +
report (2 × number)
```



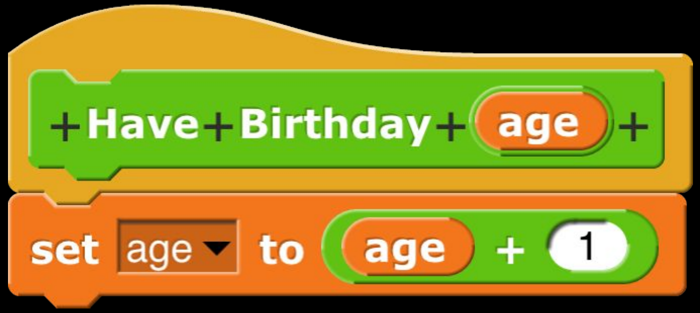
```
+Triple+ (number) +
report (3 × number)
```

What happens on a procedure call?

- The value of the variable is sent in, and the formal parameter is bound to that value



```
script variables (my age)
set [my age] to 19
Have Birthday (my age)
say (my age)
```



```
+Have+Birthday+(age)+
set [age] to ((age) + 1)
```

What is said by the purple **say** block?

A) 18

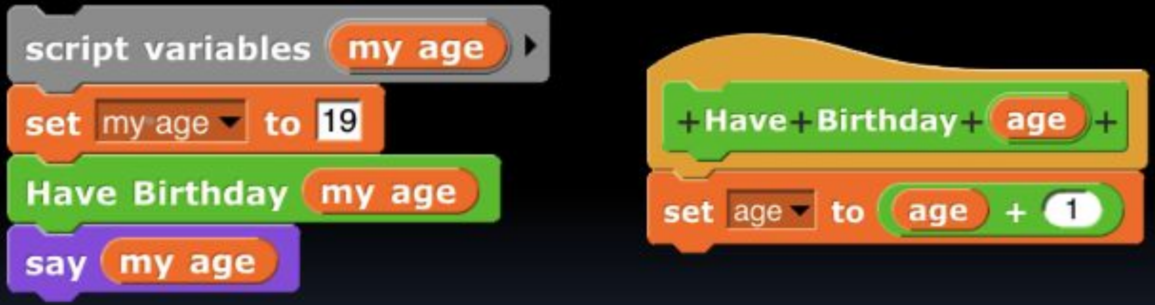
B) 19

C) 20

D) Error, age not bound in this scope/context

E) Error, my age not bound in this scope/context

L04 What is said by the purple "say" block?



```
script variables (my age)
set [my age] to 19
Have Birthday (my age)
say (my age)
```

```
+Have+Birthday+(age)+
set [age] to ((age) + 1)
```

18

19

20

Error, age not bound in this scope/context

Error, my age is not bound in this scope/context

Variables, Scope, Lists, HOFs Demos