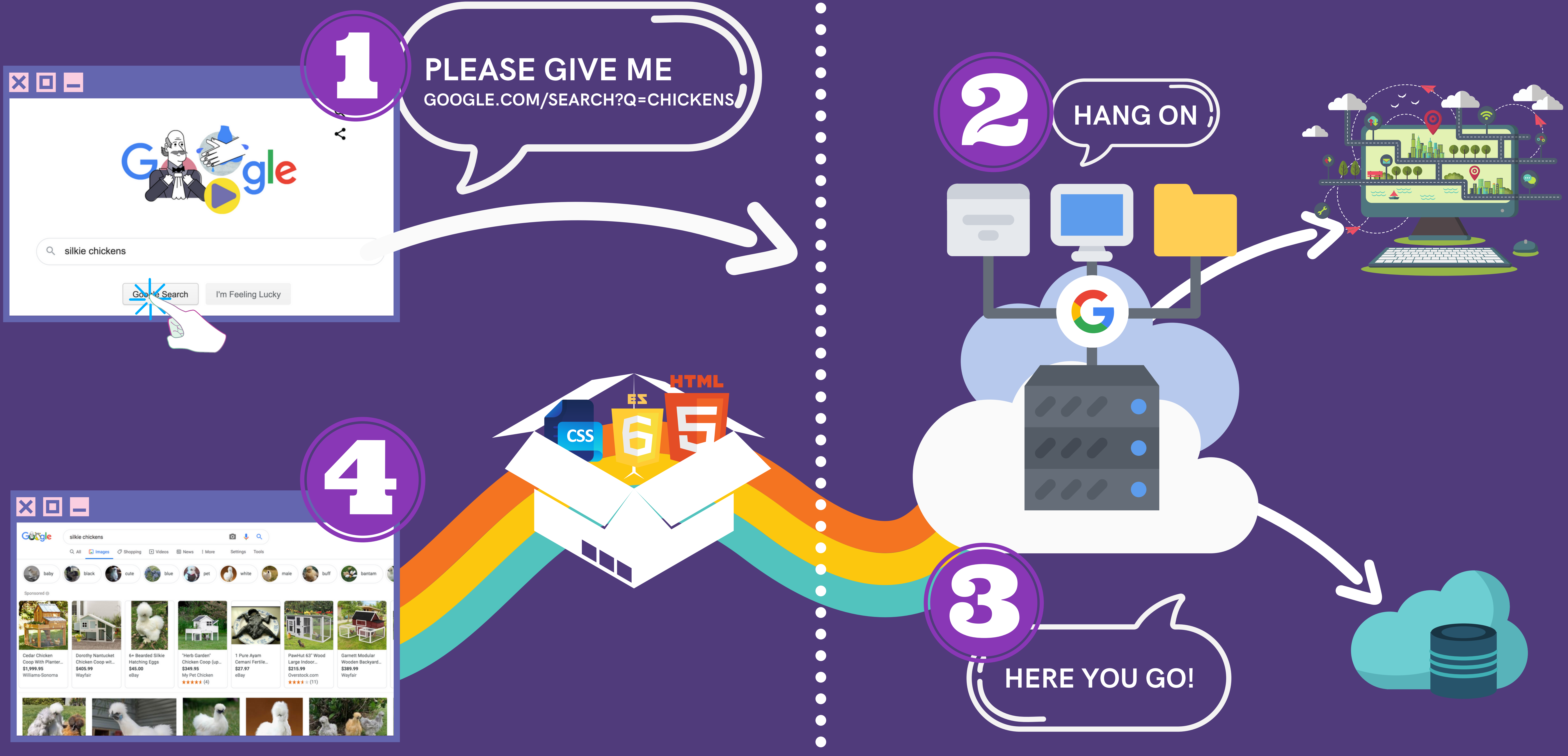
Web Developer Bootcamp

JavaScript Basics

VALUES & VARIABLES

JS-LAND



FRONT END

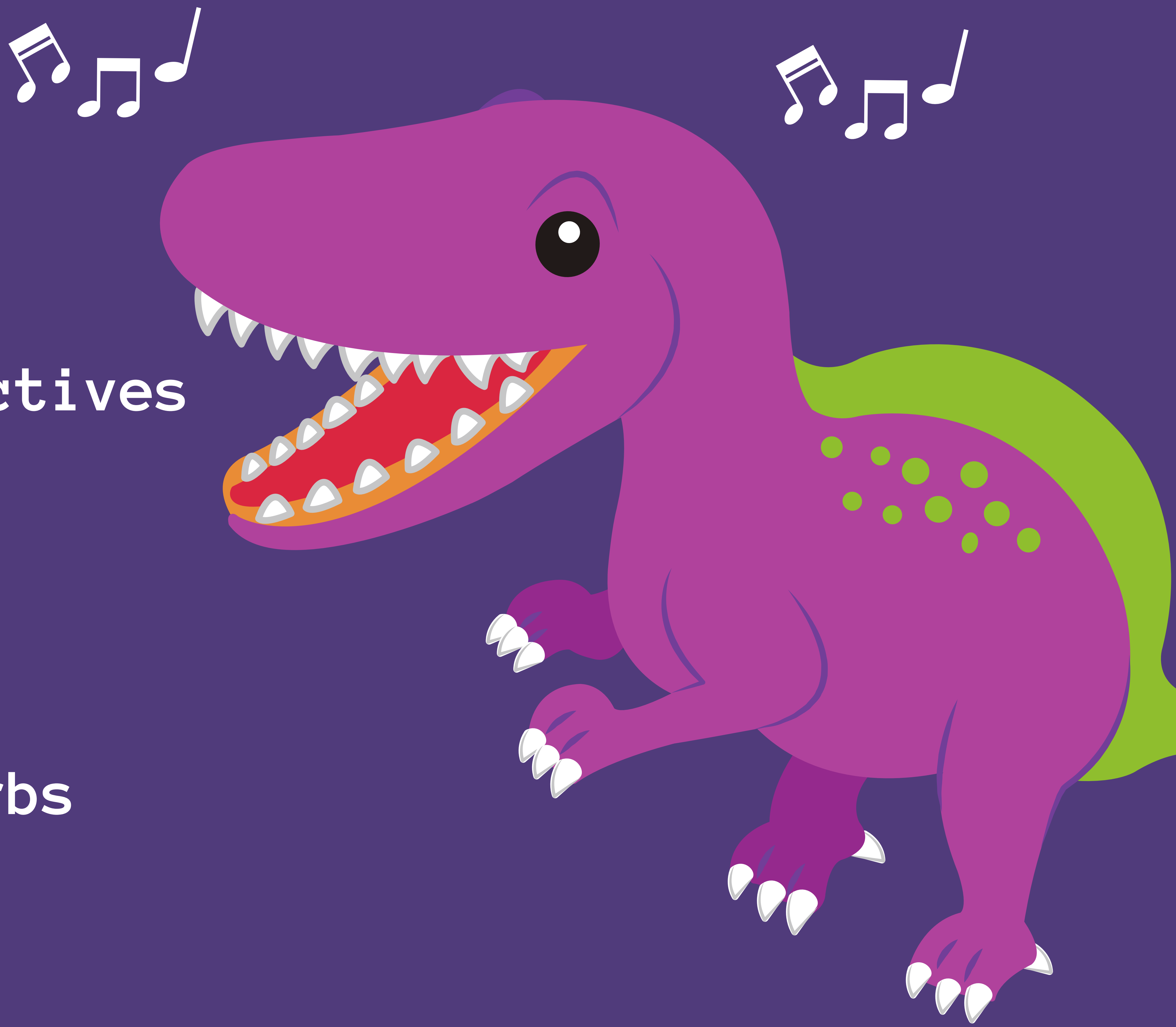
BACK END

THE PURPLE DINO DANCED

CSS – adjectives

HTML – nouns

JS – verbs



1 LEARN JS ON ITS OWN. NO HTML/CSS.

2 USE JS TO MANIPULATE HTML/CSS

Primitive Types

THE BASIC BUILDING BLOCKS

Number

String

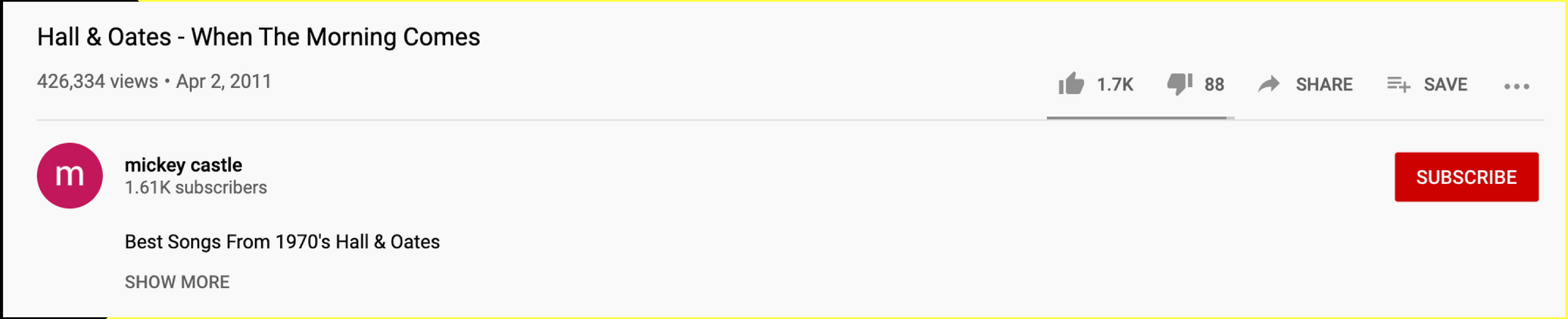
Boolean

Null

Undefined

* Technically there are two others: Symbol and BigInt

DIFFERENT DATA TYPES



Hall & Oates - When The Morning Comes

426,334 views • Apr 2, 2011

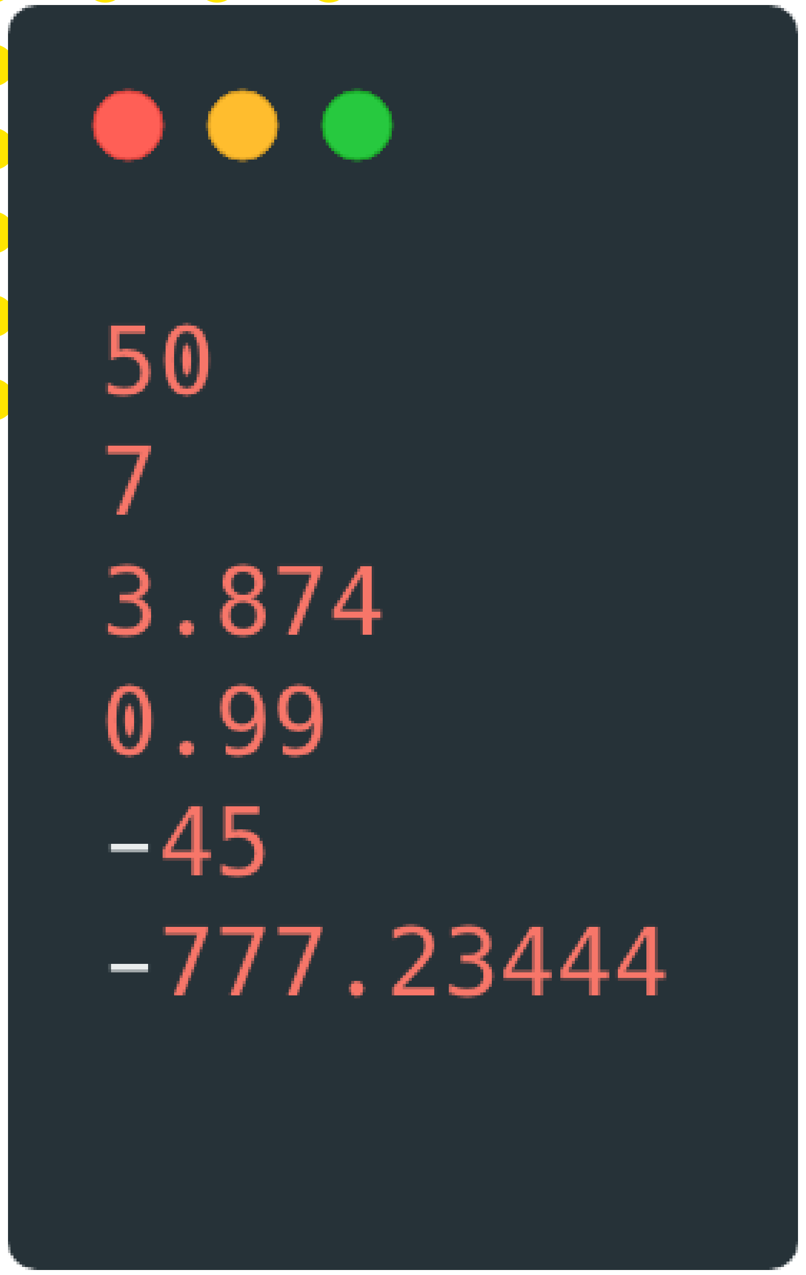
mickey castle
1.61K subscribers

Best Songs From 1970's Hall & Oates

Running Code in The Console

THE EASIEST PLACE TO START

Early on, we'll run our code using the Chrome developer tools console. Then, we'll learn how to write external scripts.



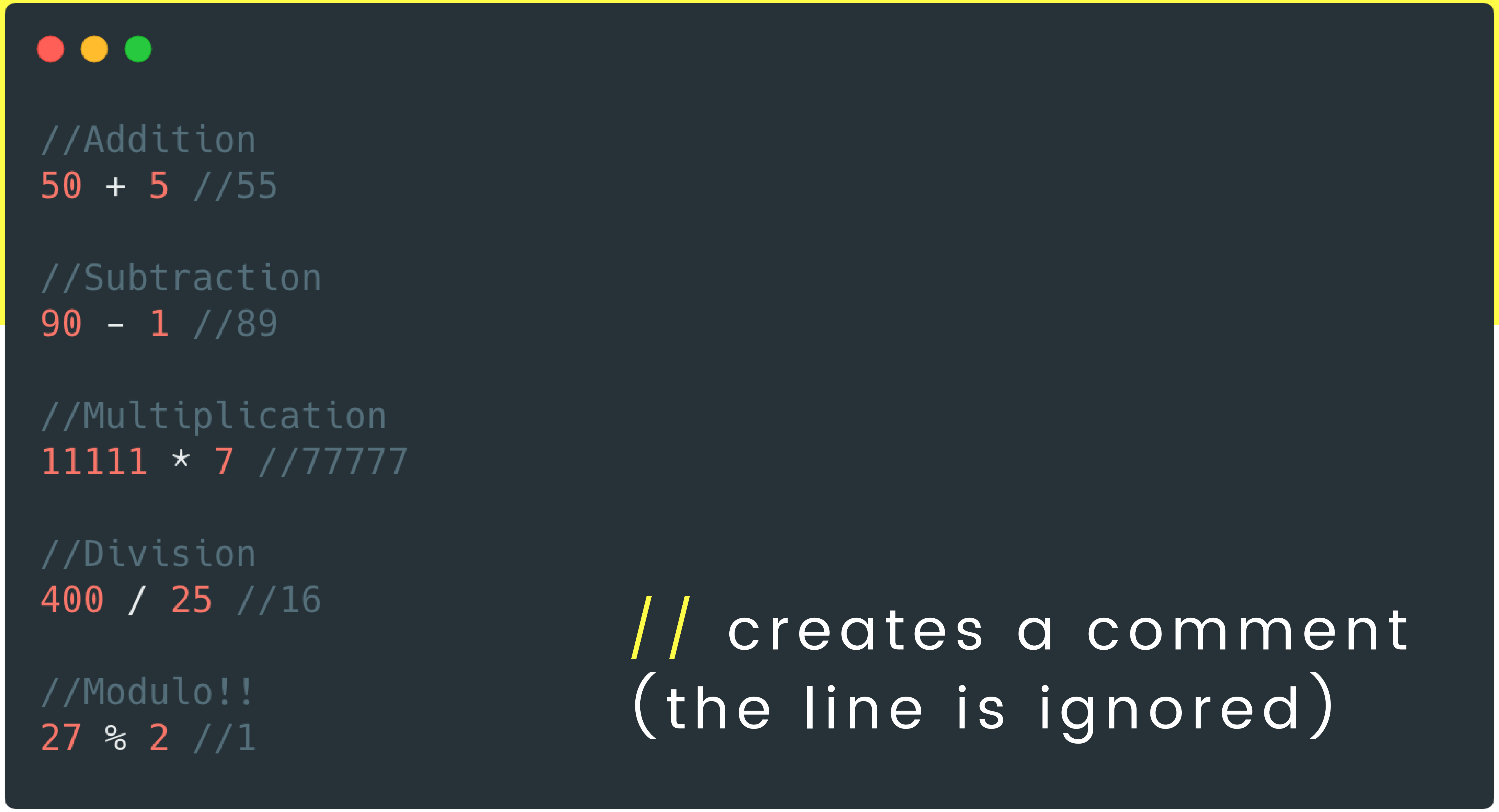
```
50
7
3.874
0.99
-45
-777.23444
```

Numbers

IN JAVASCRIPT

- JS has **one** number type
- Positive numbers
- Negatives numbers
- Whole numbers (integers)
- Decimal numbers

Math Operations



```
//Addition
50 + 5 //55

//Subtraction
90 - 1 //89

//Multiplication
11111 * 7 //77777

//Division
400 / 25 //16

//Modulo!!
27 % 2 //1
```

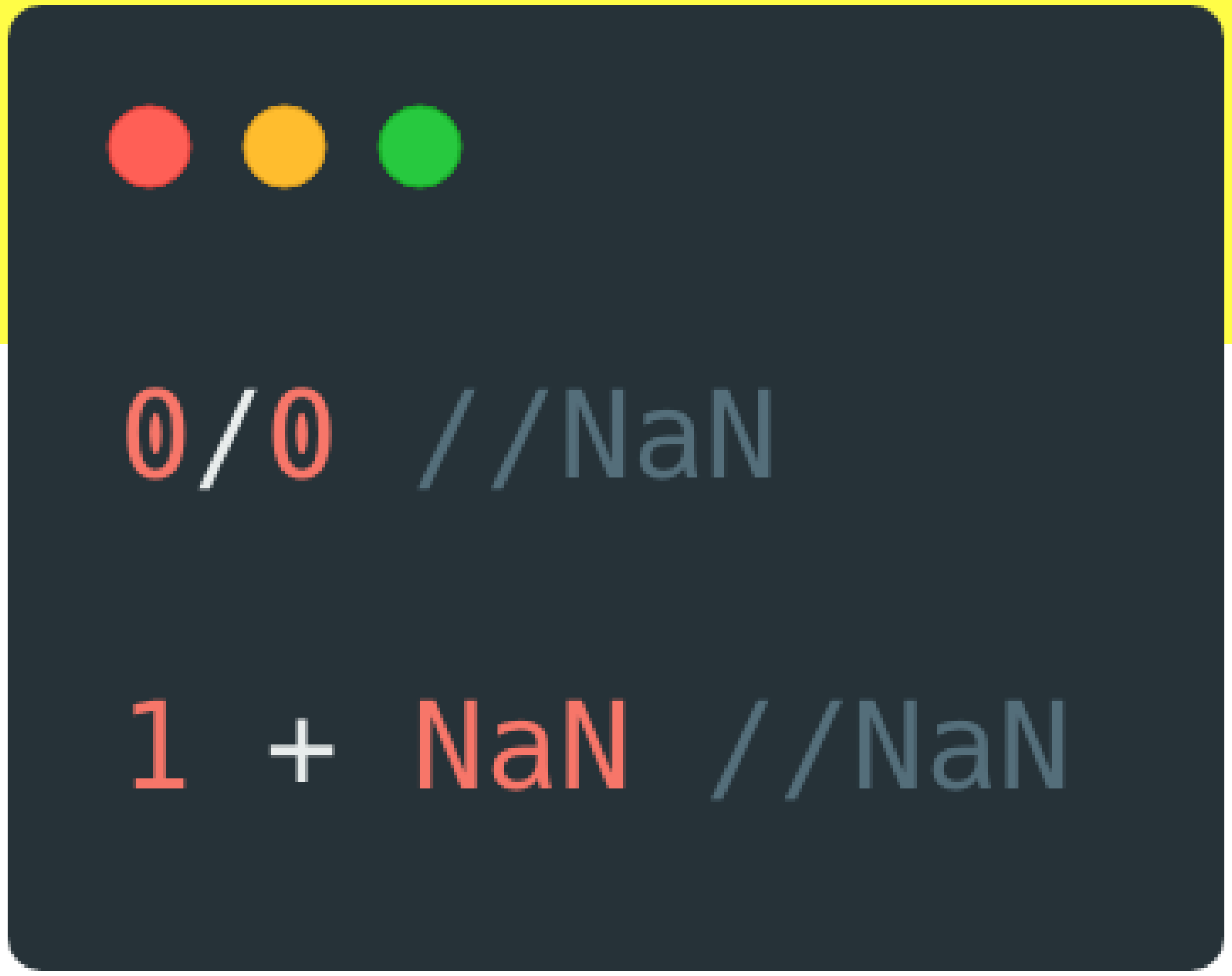
// creates a comment (the line is ignored)

NOT A NUMBER

NaN

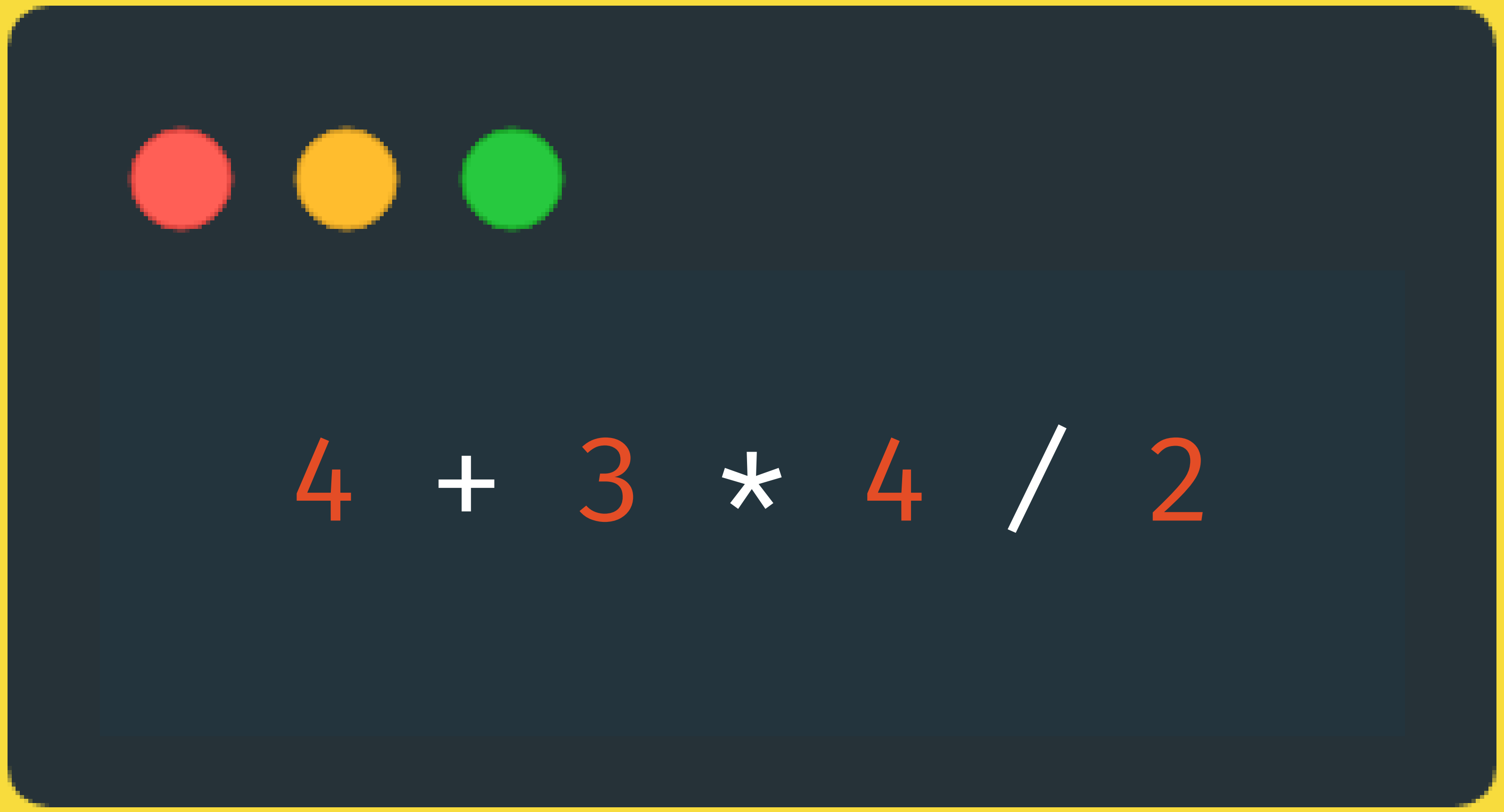
NaN is a numeric value that represents something that is...not a number

Not A Number



```
0/0 //NaN

1 + NaN //NaN
```

WHAT DOES THIS EVALUATE TO??

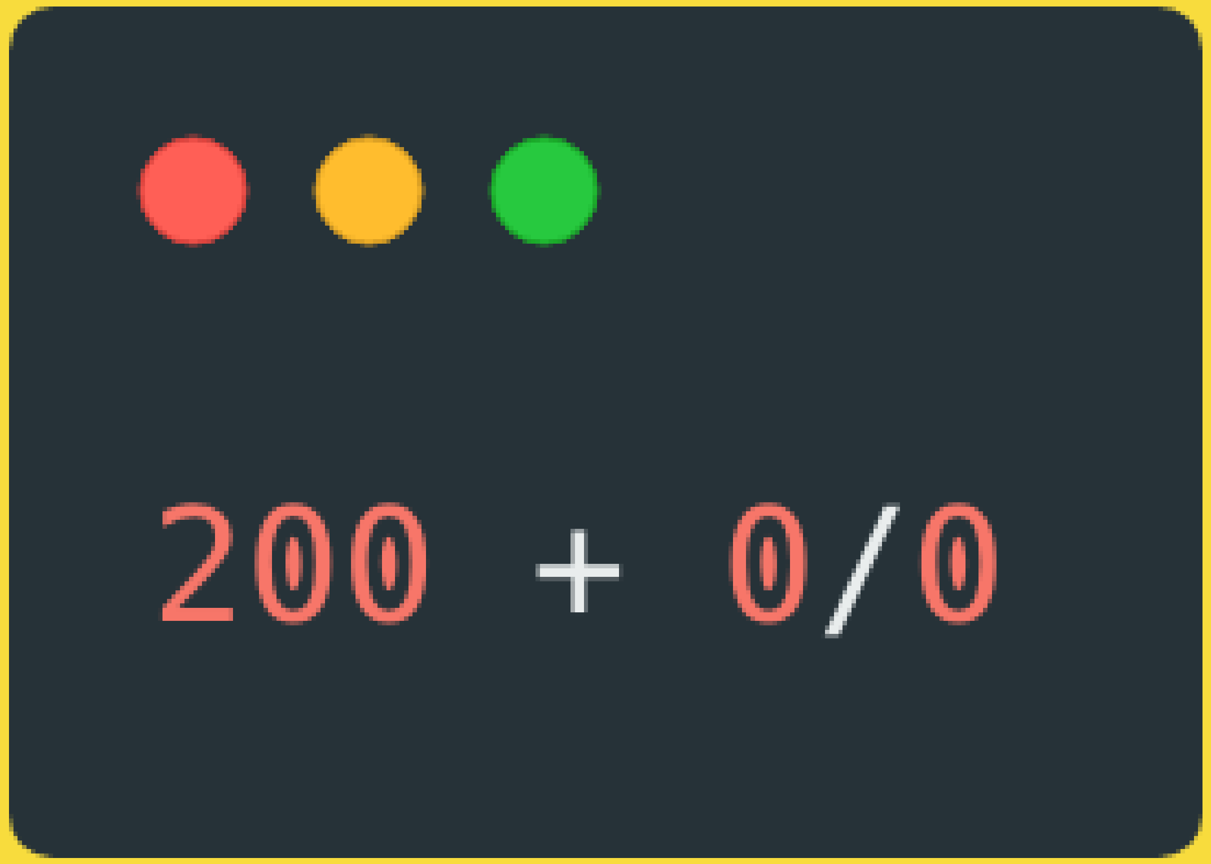


```
4 + 3 * 4 / 2
```

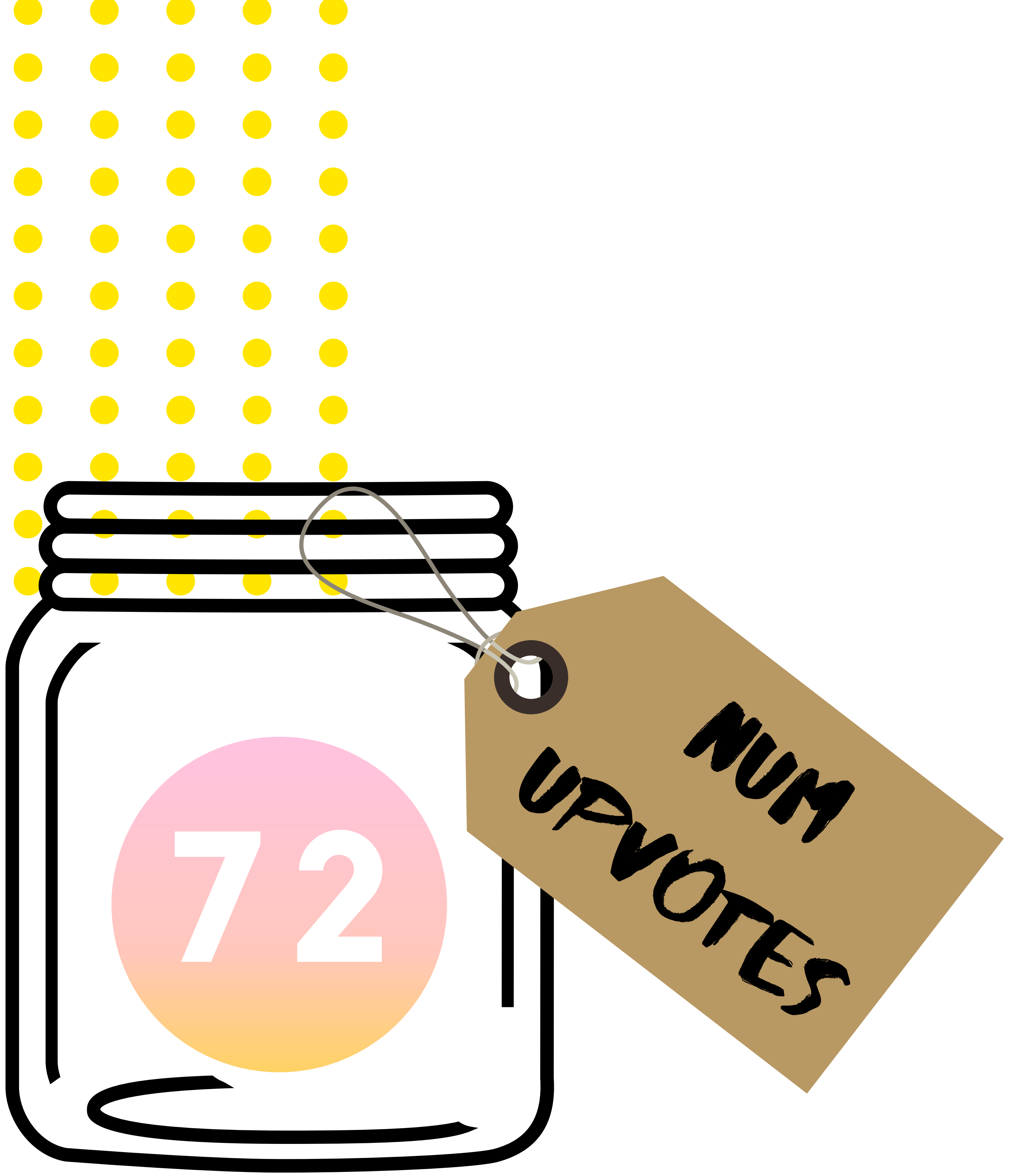
WHAT DOES THIS EVALUATE TO??

```
(13 % 5) ** 2
```

WHAT DOES THIS EVALUATE TO??



```
200 + 0/0
```



Variables

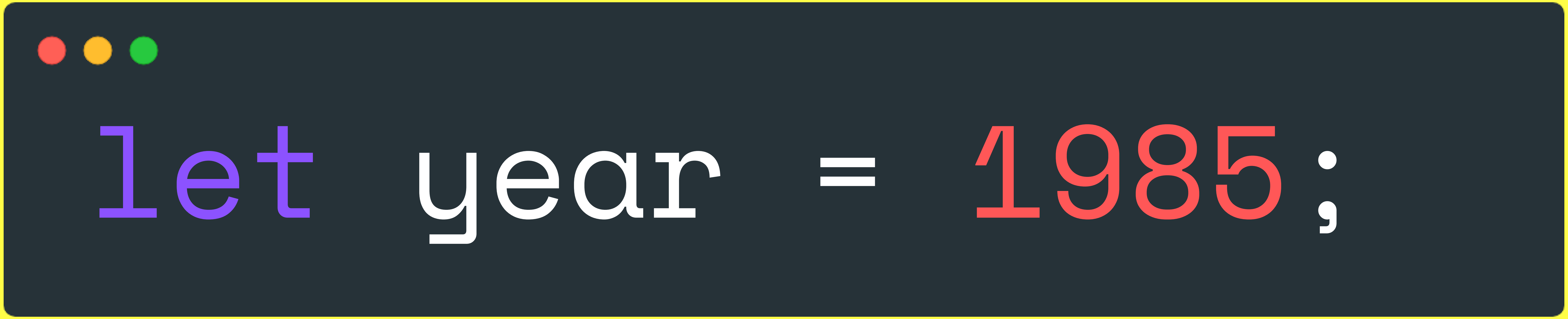
VARIABLES ARE LIKE LABELS FOR VALUES

- We can store a value and give it a name so that we can:
- Refer back to it later
- Use that value to do...stuff
- Or change it later one

BASIC SYNTAX

```
let someName = value;
```

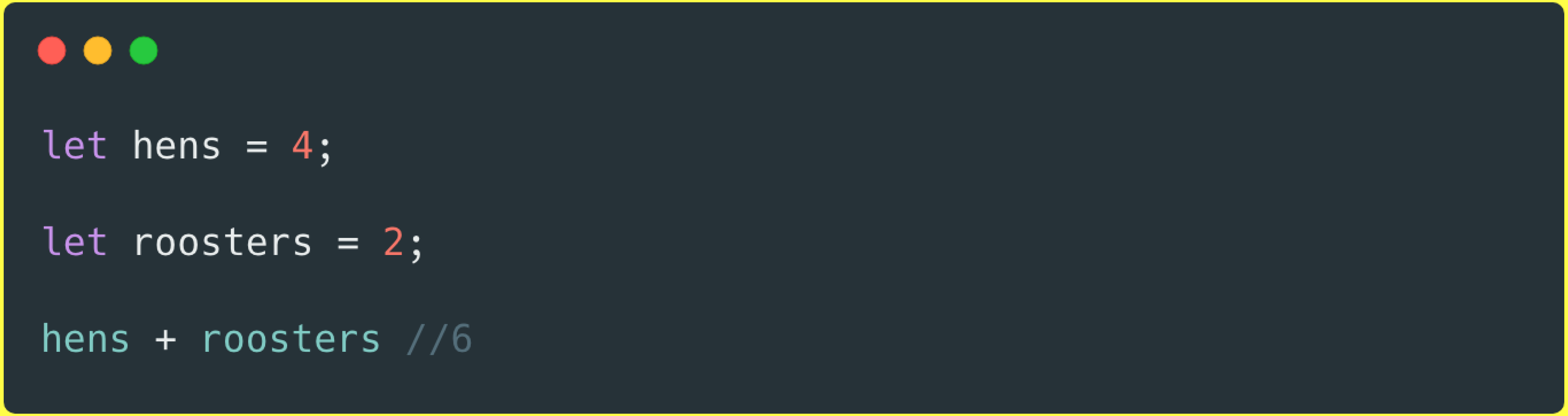
BASIC SYNTAX



```
let year = 1985;
```

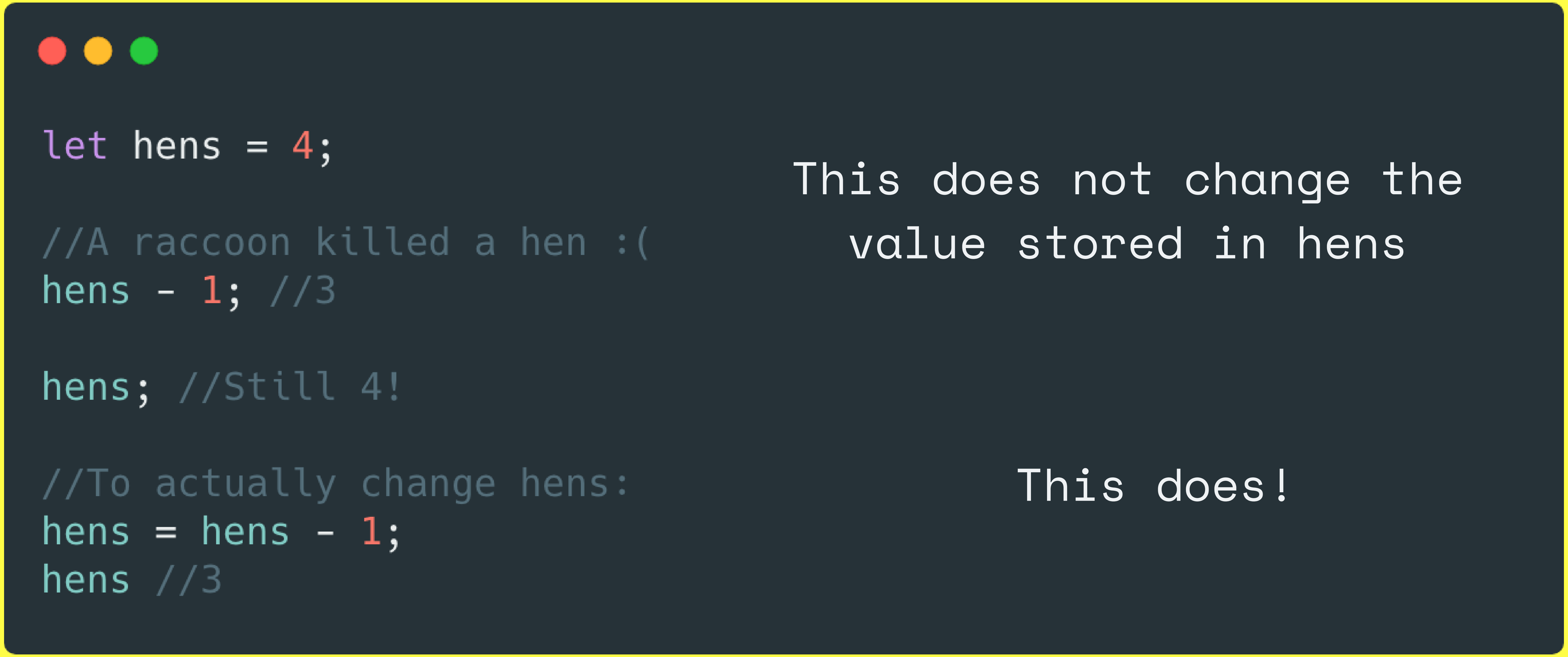
Make me a variable called "year" and give it the value of 1985

RECALL VALUES



```
let hens = 4;

let roosters = 2;

hens + roosters //6
```

RECALL VALUES



```
let hens = 4;

//A raccoon killed a hen :(
hens - 1; //3

hens; //Still 4!

//To actually change hens:
hens = hens - 1;
hens //3
```

This does not change the value stored in hens

This does!

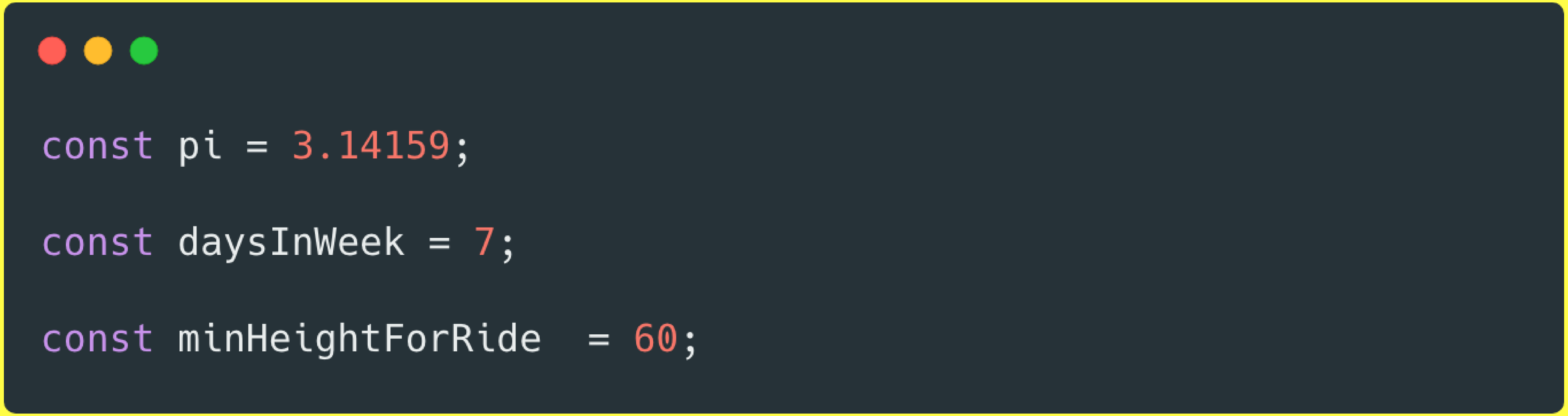
CONST

```
const hens = 4;
hens = 20;  //ERROR!


const age = 17;
age = age + 1;  //ERROR!
```

const works just like let, except you CANNOT change the value

NOT ALLOWED!
YOU'RE IN TROUBLE!!
I'M TELLING MOM!!!

WHY USE CONST?

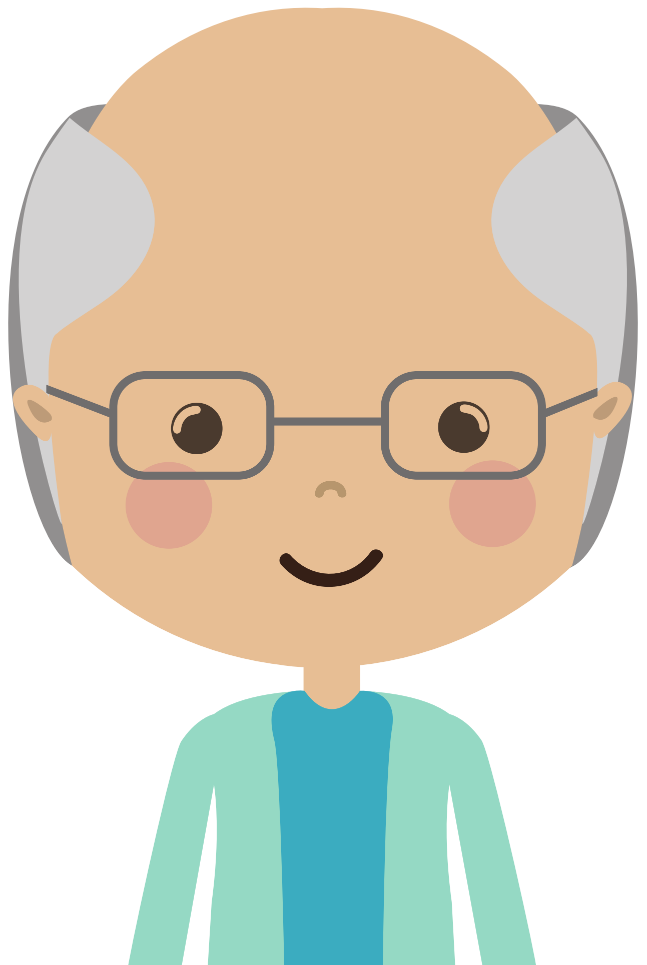


```
const pi = 3.14159;

const daysInWeek = 7;

const minHeightForRide  = 60;
```

Once we cover Arrays & Objects, we'll see other situations where *const* makes sense over *let*.

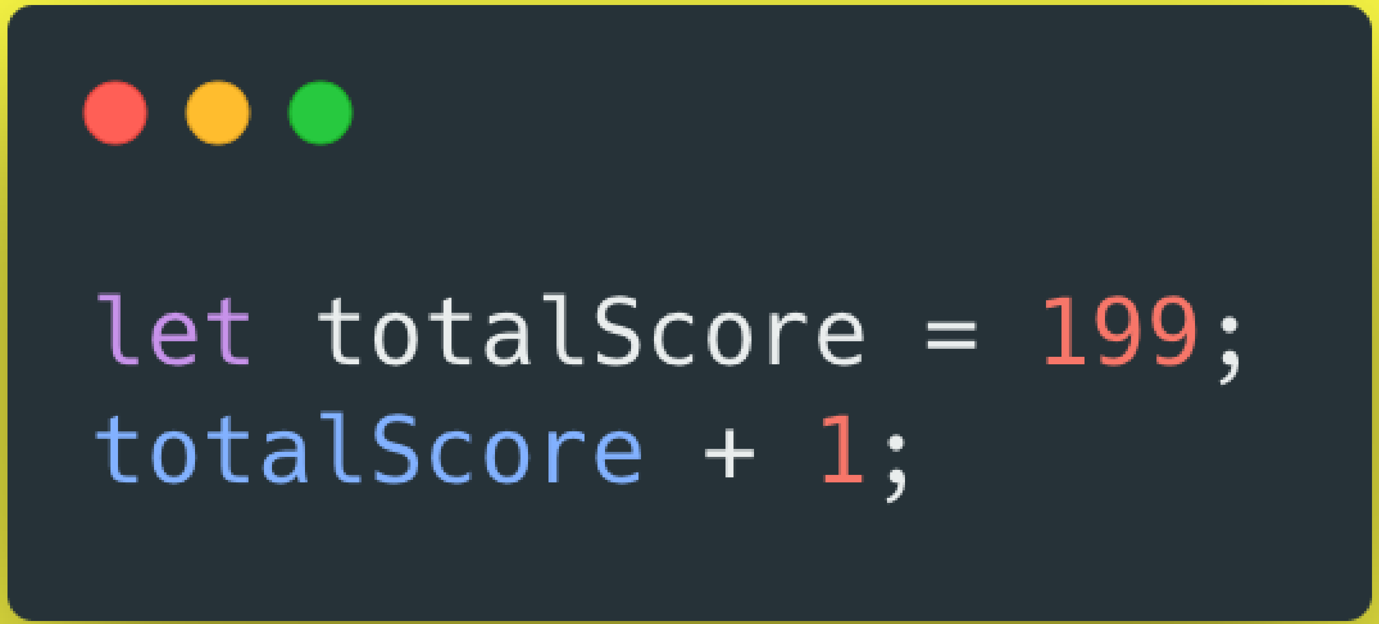
VAR

THE OLD VARIABLE KEYWORD



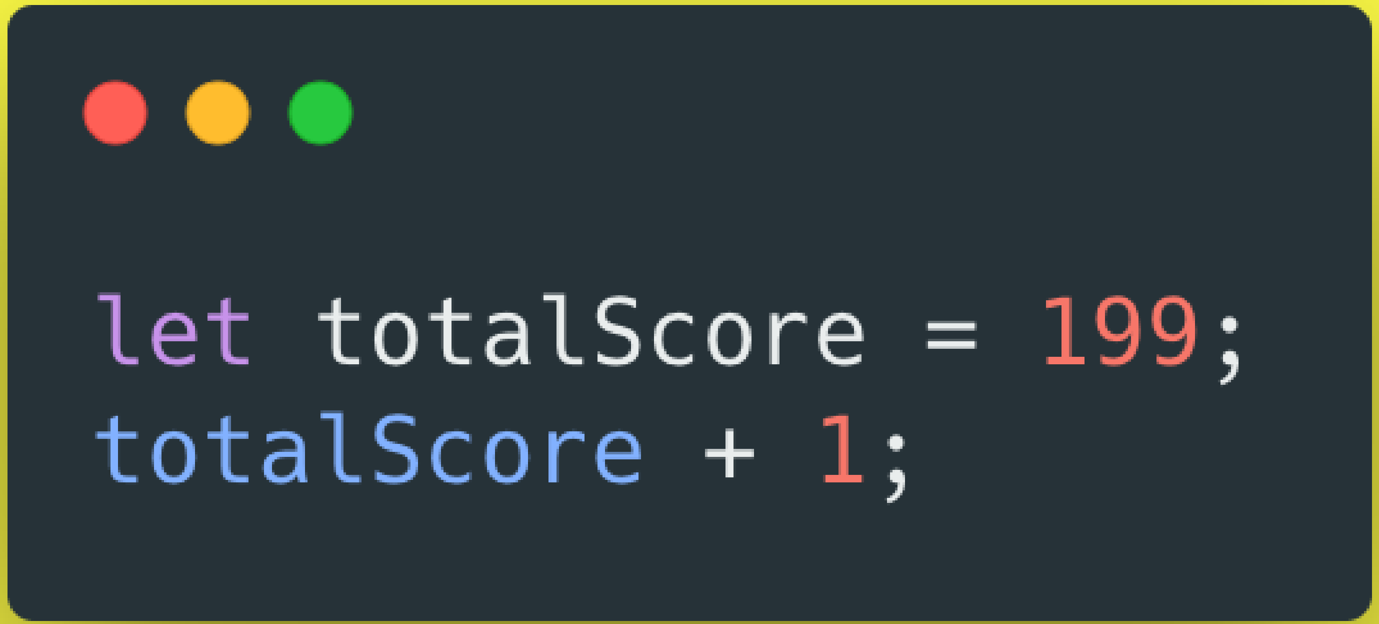
BEFORE LET & CONST, VAR WAS THE ONLY WAY OF DECLARING VARIABLES. THESE DAYS, THERE ISN'T REALLY A REASON TO USE IT.

What is the value of totalScore?



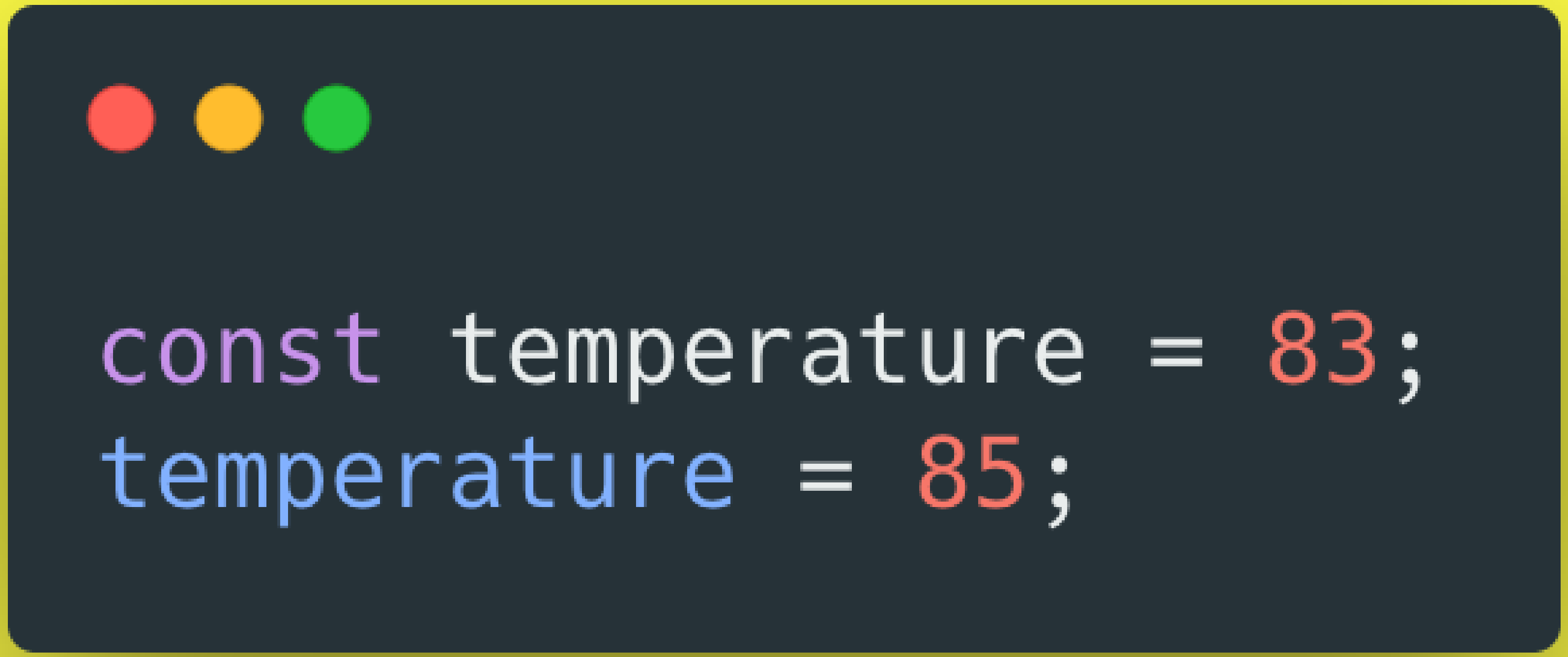
```
let totalScore = 199;
totalScore + 1;
```

What is the value of totalScore?



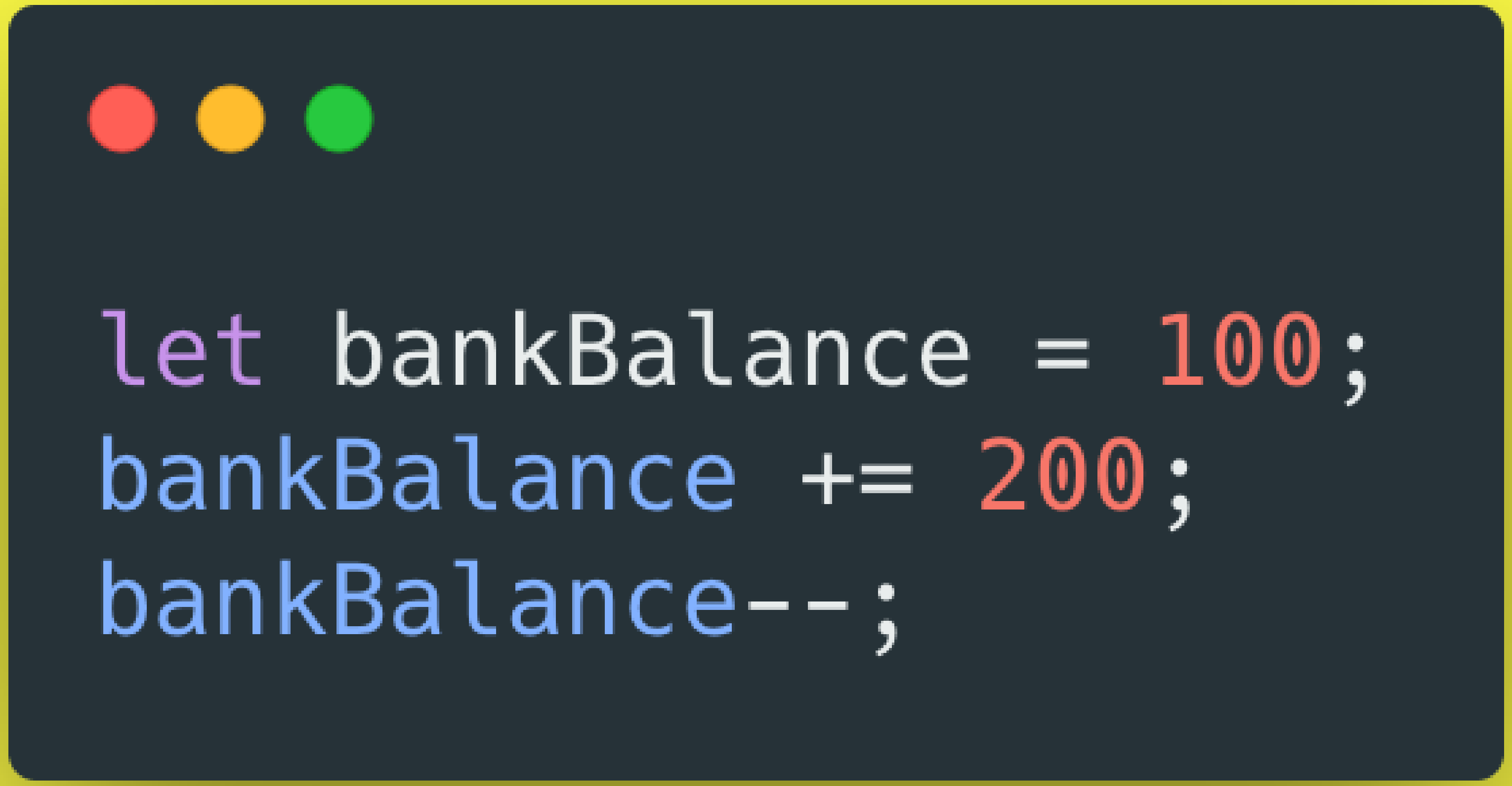
```
let totalScore = 199;
totalScore + 1;
```

What is the value of temperature?



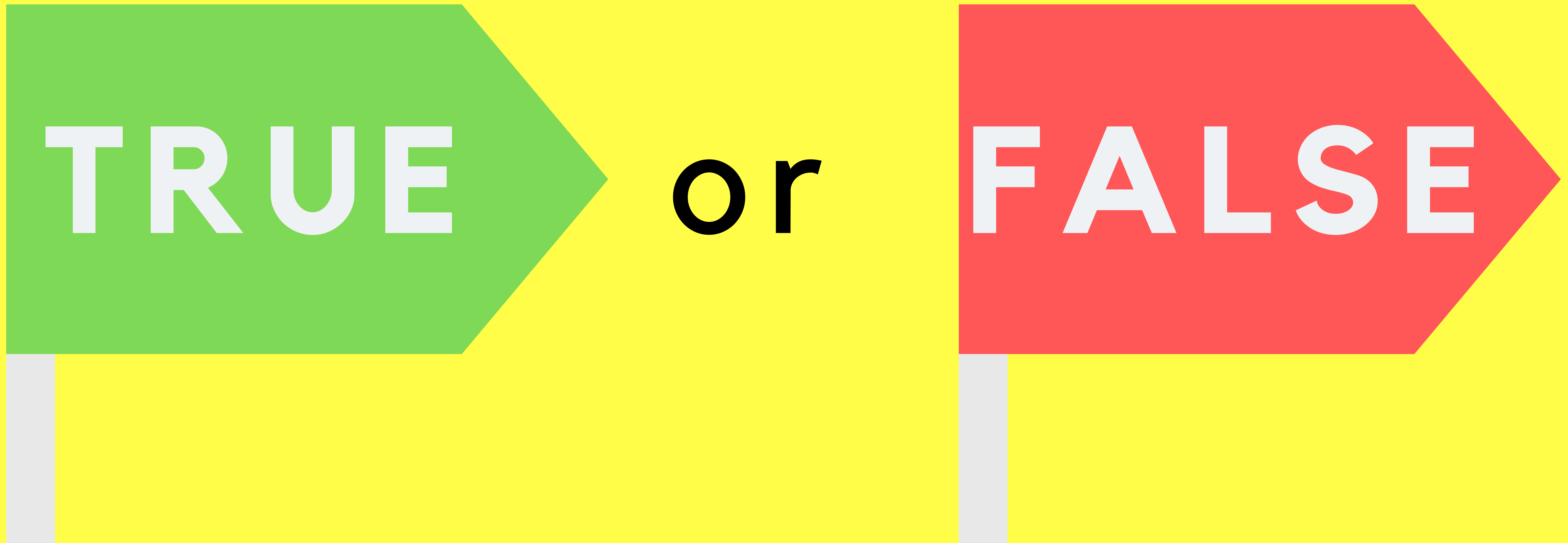
```
const temperature = 83;
temperature = 85;
```

What is the value of bankBalance?



```
let bankBalance = 100;
bankBalance += 200;
bankBalance--;
```

BOOLEANS



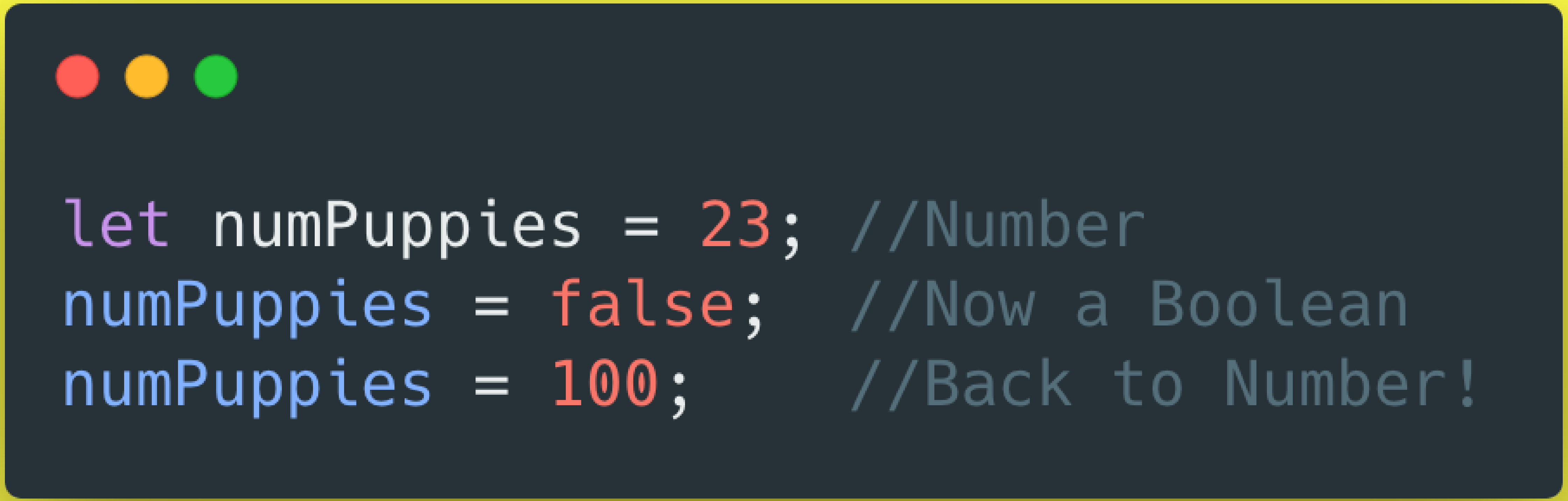
Booleans

TRUE OR FALSE

```
let isLoggedIn = true;

let gameOver = false;

const isWaterWet = true;
```

Booleans are very simple.
You have two possible options: true or false. That's it!

Variables Can Change Types



```
let numPuppies = 23; //Number
numPuppies = false;  //Now a Boolean
numPuppies = 100;    //Back to Number!
```

It doesn't really make sense to change from a number to a boolean here, but we can!