# Welcome to Bootcamp Prep!

*Fullstack Academy*

# Course Overview: Schedule

| Day | Time | Unit |
|---|---|---|
| Sat. | 10:00 am | Introduction, Tidy Code, Loops, Debugging |
| | 1:00 pm | Lunch |
| | 2:00 pm | Coercion and Truthiness |
| Sun. | 10:00 am | Scope |
| | 1:00 pm | Lunch |
| | 2:00 pm | Arrays I |
| Mon. | 6:40 pm | Arrays II |
| Tue. | 6:40 pm | Objects |

| Day | Time | Unit |
|---|---|---|
| Wed. | 6:40 pm | Object Methods |
| Thu. | 6:40 pm | PBR/PBV |
| | | |
| Sat. | 10:00 am | Higher-order Funcs I |
| | 1:00 pm | Lunch |
| | 2:00 pm | Higher-order Funcs II |
| Sun. | 10:00 am | Recursion I |
| | 1:00 pm | Lunch |
| | 2:00 pm | Recursion II |

# Course Overview: Format

1. Watch the pre-recorded lecture (if time permits)

2. Attend the live lecture (required)

3. Work on the workshop with your partner (required)

4. Watch the workshop review videos and compare your answers with the provided solutions (strongly recommended)

# Workshops

◉ **Workshops are not intended to be finished**

◉ **You will be randomly paired with a different partner or small team for each workshop**

◉ **Instructors and fellows available via the Help Desk**
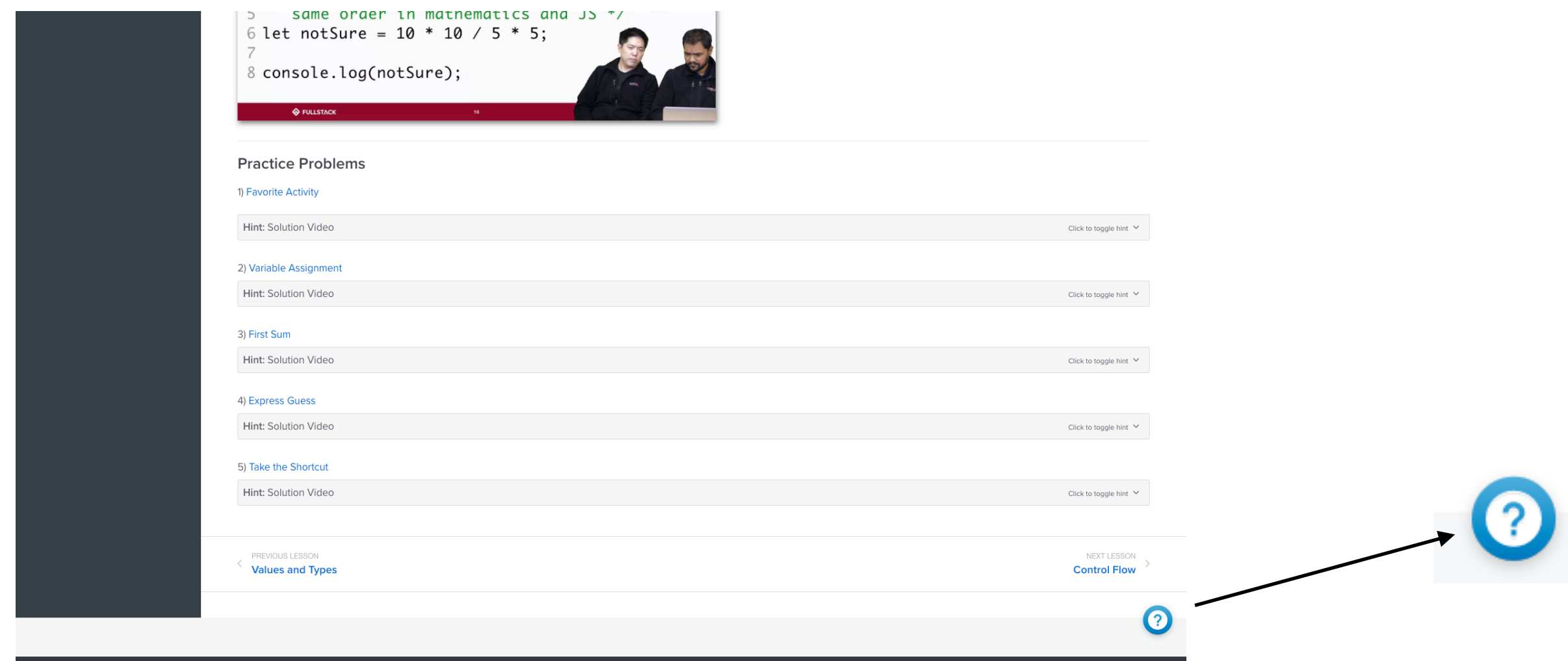
# Workshops: pair programming

- **Pair programming**
  - 2 programmers, sharing the same computer, work collaboratively on all aspects of software development (Williams and Kessler 2000).
  - "Driver" writes code
  - "Navigator" plans code and reviews it as it is written
  - Switch every 20 or 30 minutes
- **At Fullstack, you switch partners for each workshop**

# Workshops: pair programming

⦾ **Why use pair programming at Bootcamp Prep? Shouldn't I practice on my own?**

- "Teach to learn" is a central aspect of Fullstack's instructional philosophy

- Communicating about code is just as, if not more important, than actually writing it

- Solving problems with a partner is the best way to prepare for an admissions interview at a top bootcamp

- Most top bootcamps, and a growing number of employers, use pair programming every day

# Course Overview: Projects

◉ **Two projects will be made available to you on LearnDot after you finish the course**

◉ **100% optional**

◉ **Good opportunity to use what you've learnt to build larger programs**

# Course Overview: Practice Assessments

◎ There are two practice assessments that will be available to you on LearnDot after you graduate

◎ They are meant to help you prepare for the real admissions assessment

# Fullstack Community Values

◉ Be patient. With yourself and others.

◉ Ask questions (even "dumb" ones)

◉ Trust the process. Be on time.

◉ Help others. Teach others.

◉ We like to have fun, and…

# Fullstack Community Values

◉ Be mindful of:

◉ Subtle "isms"

◉ E.g. Sexism, Racism, Ageism

◉ Be Professional - "No Asshole Policy"

◉ You are each other's most valuable resource

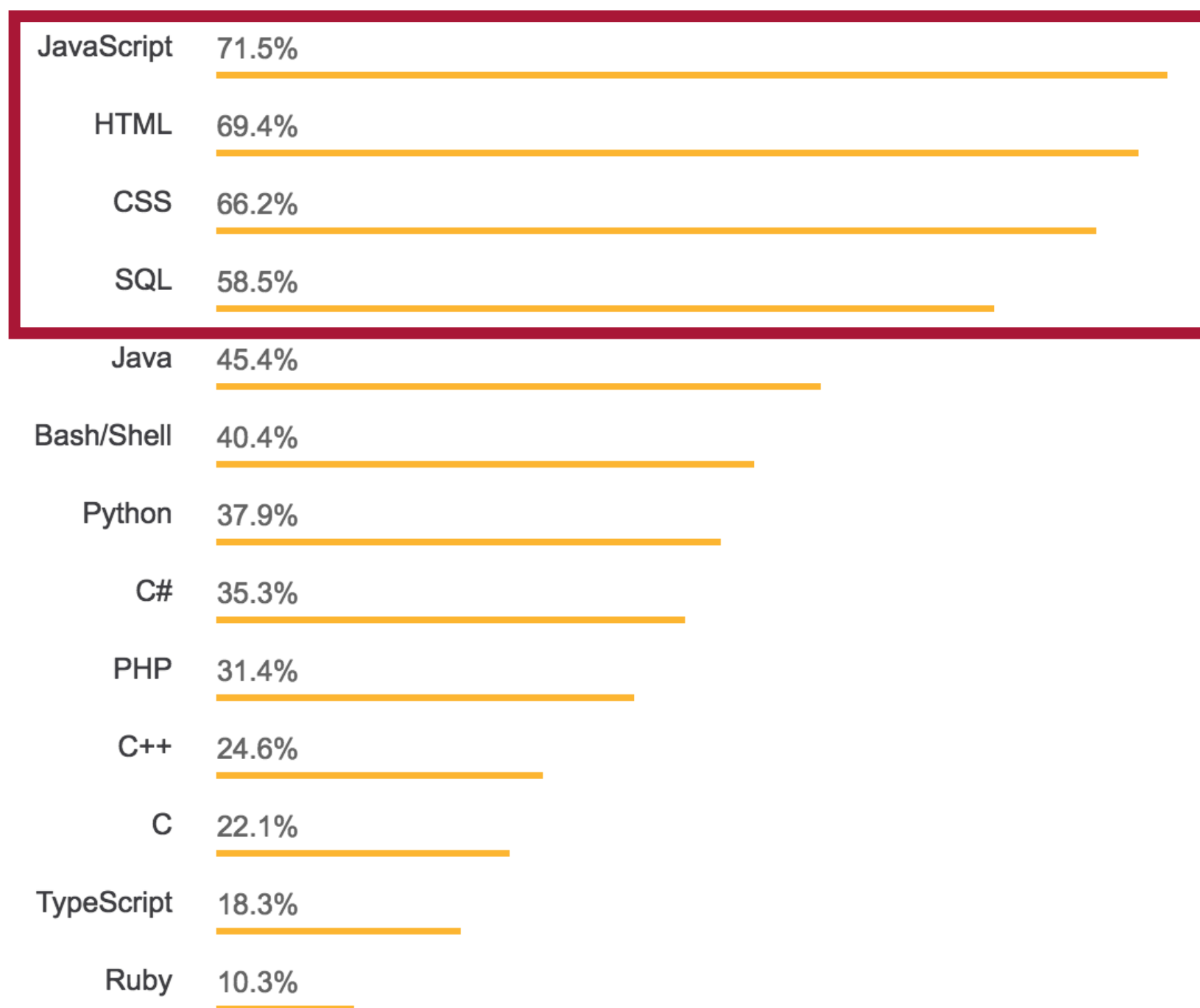◉ No NSFW content. When in doubt, leave it out.

# Why Javascript?

**Most Popular Technologies**

**Programming, Scripting, and Markup Languages**

All Respondents | Professional Developers

| | |
|---|---|
| JavaScript | 71.5% |
| HTML | 69.4% |
| CSS | 66.2% |
| SQL | 58.5% |
| Java | 45.4% |
| Bash/Shell | 40.4% |
| Python | 37.9% |
| C# | 35.3% |
| PHP | 31.4% |
| C++ | 24.6% |
| C | 22.1% |
| TypeScript | 18.3% |
| Ruby | 10.3% |

JS === most popular language in the world!

FSA/GHA curriculum

Ruby is great for beginners but much less widely used

Stack Overflow 2018 Developer Survey Results

# Why is JavaScript so Popular?

◉ **Powers the majority of web applications**

◉ **Runs almost anywhere on the "full stack": web browsers and servers**

◉ **Is relatively learner friendly**

# ES5? ES6?

- **ES5 and ES6 are different versions of JavaScript**
- **JavaScript versions are backwards compatible**
  - Older features work in newer environments
  - Newer features may not work in older environments
- **Fullstack teaches "ESNext"**
  - Immersive programs continuously updated to cover the latest JS features
  - BCP program covers some newer features too, but only those that are helpful for beginning developers — don't expect in-depth or comprehensive coverage of new JS features in this course

# Getting the most from this course

- Complete the BCP Pre-work curriculum first!
- Do the pre-readings and watch the pre-recorded lectures
- Enjoy the "productive struggle"
- Ask for help when you need it (otherwise, why are you here?)
- Don't look at solutions until you give problems an honest try
- Go back to unfinished workshop problems and finish them when time permits
- Watch the solution videos, even for problems you think you got right
- Review the solution code

# Tidy Code Tips

# Why write tidy code?

◎ **Easier and faster to debug and read your code**

  • Engineers spend a lot more time reading and debugging code than actually writing it!

◎ **Demonstrates qualities attractive to bootcamps and employers:**

  • Professionalism: no longer a "beginner"

  • Attention to detail

  • Pride in work

# Indentation

```
1  // indent one level inside every code block
2  // code blocks start with a { and end with a }
3
4  function amazingFunction() {
5    // inside the function block!
6    if (true) {
7      // inside the if block!
8      console.log('this function is amazing');
9    }
10   // outside of the if block
11 }
12
13 // outside of the function block
14 amazingFunction();
```

**FULLSTACK**

**BOOTCAMP PREP**

# Variable Naming

```
1   // use camelCase when defining variable names
2   let myFavoritePlace = 'Fullstack';
3
4   // don't use ambiguous variable names!
5   let x = 68;
6
7   // use names that describe the value they contain
8   let currentTemp = 68;
9
10  // it's ok to use short variable names as counters, like i in a for loop
11  for (let i = 1; i <= 3; i++) {
12    console.log(i);
13  }
14
```