**LECTURES**

**WHAT IS CS?**

Reasons / Details for Grade:

What are major gaps that need to be addressed?

Need to provide the instructors with a definition of “algorithm” explicitly in the notes (slide 25)

Should explain what you mean by “mobile web app” explicitly (slide 40)

Any secondary concerns?

Should provide more detailed explanation of what it means that CS is in “Code” (slide 30)

Maybe give examples of previous year demo day projects? **(NOT ADDRESSED BECAUSE I DID NOT HAVE ACCESS TO PREVISOUS YEAR DEMOS)**

If games are an option, maybe show some games that achieve these goals?

Is there sufficient places to check student comprehension? What ways can we determine

student comprehension?

I would recommend asking students explicitly for examples of CS in each of the areas in slides 30-38.

Student Engagement:

For each learning style, identify existing curriculum presentation and exercises that will work well. **(NOT ADDRESSED BECAUSE I DO NOT KNOW WHAT EXERCIESES AND EXISTING CURRICULUM COULD WORK WELL IN THIS LECTURE)**

Visual

CURRICULUM:

Suggestions:

Maybe add a visual to slide 30

Logcial / Abstract

CURRICULUM:

Suggestions:

Have students give examples of algorithms? May be too early. **(NOT ADDRESSED AS JACK SAYS HIMSELF THAT ITS TOO EARLY)**

VERBAL / AUDITORY

CURRICULUM:

Suggestions:

I would potentially flesh out the “notes” a little more where they’re empty. **(ADDRESSED ON SOME SLIDES)**

GROWTH:

Does this lesson provide guidance on how to grow and learn more about the subject area?

I would recommend adding one slide that explicitly says the breadth of careers that can exist in CS!

OTHER FEEDBACK:

The use of this symbol seems potentially inappropriate in this context.

**COMMAND LINE**

Reasons / Details for Grade:

It’s a little inaccurate to say that the command line is the same as a programming language – it’s more accurate to call it a way of interacting with your computer without using the mouse or windows.

Reasons / Details for Grade:

What are major gaps that need to be addressed?

Slide 26: “Can ask students why Developers don’t bother with a mouse? - elicit “It’s quicker!....and it looks

cool”... this isn’t accurate.

- it gives you more control

- you can do more complicated things

- you aren’t depending on someone to write a program to do what you need

Between slides 37 and 38, should explicitly introduce the term “flags”, and why you added one as a modifier. **(NOT ADDRESSED AS I AM NOT OVERLY FAMILIAR WITH ‘flags’)**

Any secondary concerns?

Does this lesson grow upon previous lessons? Are there any ways it could provide more opportunity to review previous concepts?

The lesson could explicitly call out how accessing a computer at a lower level is necessary to do CS / use the ideas from the previous lesson. **(NOT ADDRESSED AS I AM NOT 100% SURE ON HOW TO INCORPORTATE THIS INFORMATION ONTO THE LECTURE)**

Is there sufficient places to check student comprehension? What ways can we determine student comprehension?

Yes. **(NOT ADDRESSED BECAUSE JACK DOESN’T GIVE SUGGESTIONS FOR PLACES WHERE STUDENT’S COMPREHENSION COULD BE CHECKED)**

Student Engagement:

For each learning style, identify existing curriculum presentation and exercises that will work well.

Visual

CURRICULUM:

Suggestions:

As these are the students that will be the most lost in this section (command line is inherently the opposite of visual), make sure you give them justification for why the command line is useful (see suggestions above).

GROWTH:

Does this lesson provide guidance on how to grow and learn more about the subject area?

Not explicitly – it would be good to integrate some of the reasons why they’re learning this (see suggestions above).

OTHER FEEDBACK:

N/A – solid lesson, good unfolding of the content at the end in particular.

**VARIABLES**

Reasons / Details for Grade:

What are major gaps that need to be addressed?

Slide 14: Need a slide just showing visually how modulo works – it’s a complicated enough idea.

Slide 20: Before they’ve learned how strings are represented is too soon to show them this idea. Needs to be after they learn how strings are represented.

Any secondary concerns?

Slide 7: In addition to the three “basic” types, say they eventually will be creating their own types of data.

Slide 21: May be worth reinforcing the rule that double quotes are how to represent strings, as languages other than JavaScript are more stringent

Slide 27: Don’t introduce conditionals yet unless you’re going to explain them completely.

Slide 56: Tell them to put const before some, and let before others, to show that they’re interchangeable.

Does this lesson grow upon previous lessons? Are there any ways it could provide more opportunity to review previous concepts?

Does a good job linking back to the command line lesson.

Is there sufficient places to check student comprehension? What ways can we determine student comprehension?

The first quiz may be unnecessary – but maybe have the students write down trick questions to give each other after the surprise quiz?

Student Engagement:

For each learning style, identify existing curriculum presentation and exercises that will work well**(NOT ADDRESSED BECAUSE I DO NOT KNOW WHAT EXERCIESES AND EXISTING CURRICULUM COULD WORK WELL IN THIS LECTURE)**

Visual

Suggestions:

Slides 10-14 – need more examples / visual help for what each type of math we’re talking about. **(NOT ADDRESSED AS THERE IS ONE SLIDE THAT GIVES EXAMPLES ON HOW THE OPERATORS WORKS)**

Slide 14: Need a slide just explaining modulo.

Logical / Abstract

EXERCISES:

Suggestions:

Have students create problems to give to each other.

Have students write three lines of code and try to predict what their neighbor’s code will do (can introduce the idea of expressiveness.

GROWTH:

Does this lesson provide guidance on how to grow and learn more about the subject area?

Yes – maybe after slide 59, encourage students to do other explorations.

OTHER FEEDBACK:

The cover slide includes that there is a “surprise” quiz – include in the notes that this is intentional so the instructor doesn’t get caught off guard.

Slides 38-39 – the president isn’t a good example, as the students may get lost in their dislike for Trump rather than get the point you’re trying to make. He’s such a divisive figure, it’s probably not productive to learning how variables work.

Maybe don’t introduce Booleans at all until you’re introducing conditionals, especially because you don’t use them. **(NOT ADDRESSED AS BOOLEANS AS A DATA TYPE IS BEING INTRODUCED TO THE STUDENTS)**

**PROJECT**

Reasons / Details for Grade:

What are major gaps that need to be addressed?

Could include practical examples of when to use modulo.

Is there sufficient places to check student comprehension? What ways can we determine

student comprehension?

Have students create variables in a guided way – we’re making a bookstore, what kinds of variables do we need about each book?

GROWTH:

Does this project provide guidance on how to grow and learn more about the subject area?

No, but the content is too basic. **(NOT ADDRESSED AS JACK DOESN’T ELABORATE ON HOW THE “CONTECT IS TOO BASIC” AND HOW TO FIX IT)**

**CODE ALONG**

Reasons / Details for Grade:

What are major gaps that need to be addressed?

Although it’s great to have the instructor ask the students how to navigate to a folder / make a file, you

should include the correct answer for the instructor to reduce possible issues.

Also, the lesson should probably not include variables before variables are introduced. The script could

include only the line “console.log(“:-)”);”, and the variable lesson could build on how to break “:-)” into a

variable.

Does this code along grow upon the lesson(s)? Are there any ways it could provide more opportunity to review included concepts?

Good use of the information from the command line lesson.

Is there sufficient places to check student comprehension? What ways can we determine student comprehension?

Ask them to make a new file on their own, and have the console log whatever sentence they want.

GROWTH:

Does this code-along provide guidance on how to grow and learn more about the subject area?

No, but that’s appropriate for a code-along at this level.