Intro:

The following is a series of challenges for week 1, based on the material you learned in class. If you have any questions, please let me know, and I will be happy to answer them. In order to finish these challenges, you should spend approximately 20-30 minutes per day, attempting to finish 1 challenge per day.

I have also included a list of extremely simple activities to do at the end of this document to review fundamentals that we learned during the first course. Practice these to ensure you remember key ideas. I recommend spending 10 minutes a day doing these.

Finally, in between the Challenges and the Basic Practice I have included a series of questions you should consider asking yourself and trying to answer. I will send the answers before our next class, but trying to find them yourself will help you better understand the concepts.

Challenge 1:

Description:

Create a function that takes one input (commonly referred to as an argument) and converts it from Celsius to Fahrenheit. Fahrenheit is the US system of measuring temperatures. The formula for this conversion is Celsius temperature \* 9/5 + 32.

Example:

0 \* (9/5) + 32 = 32

The function input should be called celsius.

For this and all future challenges, make a new file. Write the function and use node to test it with the test case(s) I provide below. Once your result in node matches mine, make sure to:

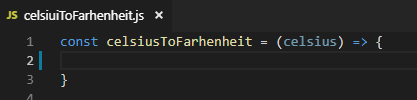
git add .

git commit –m “”

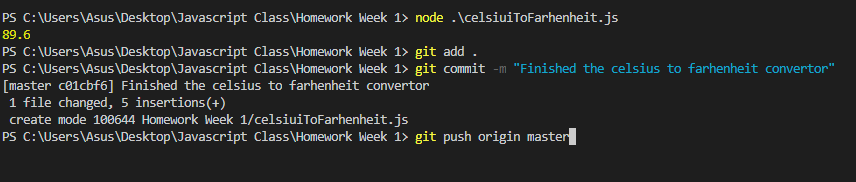
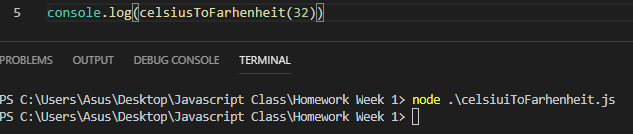
git push origin master

Here is the set up and an example of the function being used to get you started. Note that I am using const, not let, to make my function. As a general rule, when making functions it is best practice to use const, as you do not want to be able to change them.

Set Up:



Test Case:



Challenge 2:

Description:

In this challenge, you need to reverse the order of a string. For example, the string “Hello”, when entered into your function, should change to “olleH”.

For now, the best approach is to change the string into an array, reversing the order of the array, and then changing it back to a string. You’ll need to try some of the basic practice below.

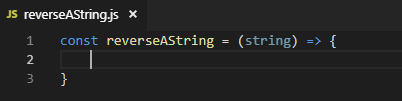
As always, create a new file, and don’t forget to:

git add .

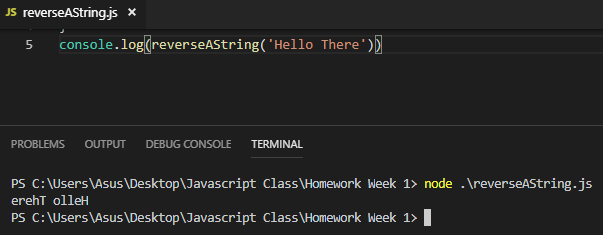
git commit –m “”

git push origin master

Set Up:



Test Case:



Hint: There is a specific set of array methods you will need to use. They can in the basic practice 21-30.

Challenge 3:

Description:

In this challenge, create function that takes an array of numbers and empties it if the first number or the last number is 1, then returns the array. Check the hints at the end of the page.

Don’t forget: Create a new file, and when you finish:

git add .

git commit –m “”

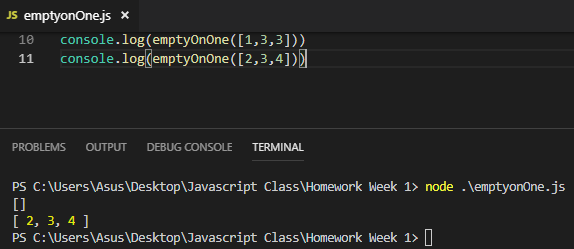
git push origin master

Set Up

:



Test Cases:



Hint: You will need to use conditionals and some array methods. See below in the basic challenges at the end of the document to try to figure out how to do this. Specifically, check out numbers 20-30.

Hint 2: Assume that your array will be 3 elements long, no more and no less. We will learn how to handle an array of any length this Saturday using a loop.

Challenge 4:

Description: Congratulations! You’ve made it a good long way, and you are doing well. It’s time for a more challenging, more life-like function. Create a simple function that emulates an elevator. You should be able to give an argument, or input, of the floor that you want to go to, and then it should check the floor and print a message that says what floor you are going to. If you enter a floor that doesn’t exist, it should tell you the floor doesn’t exist.

You will need to use conditionals for this, not because it’s actually necessary in this simple case, but because I really want you to practice using them. This Saturday I will show you how you could do this without conditionals, given the simplicity of it.

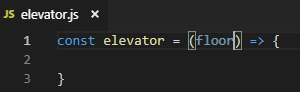
As always, make sure you create a new file, and when you finish don’t forget to:

git add .

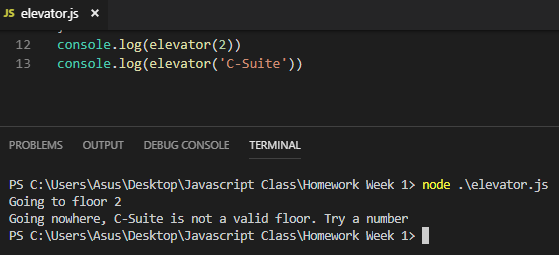
git commit –m “I finished my elevator function”

git push origin master

Set Up:



Test Case:



Challenge 5:

Great Job! If you have made it here, it’s time to take your first step towards being a big step towards being a real programmer: Doing your own research to find out what it is you need to do.

Unlike previous challenges, we haven’t learned the coding skills for this challenge yet, at least not formally. However, I have provided you with all the basic knowledge you need to complete this challenge on your own in my power point and in reference files I gave you Monday night. In order to do this challenge, you will need to read over those files and that power point. This is not unlike what we will learn this Saturday, which is how to look up more information on the internet when we get stuck. More on that when I see you again.

As always, make sure you create a new file, then, when you finish the challenge, don’t forget to:

git add .

git commit –m “”

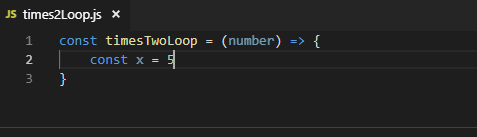
git push origin master

Here is the challenge.

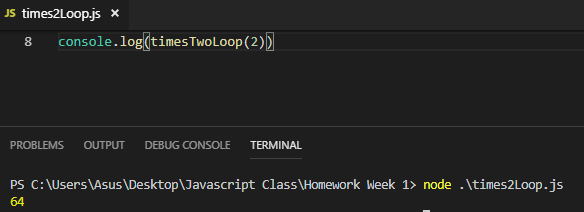
Description:

Create a function that uses a loop to multiply a number by 2 a certain number of times. You can decide how many times you want it to be when you create the loop. In the example below, let’s assume that I want it to be 5 times. I will use x to control this. Call this function timesTwoLoop.

Set Up:



Test Case:



Some Questions You Should Think About:

What is the difference between using console.log() and return?

What is the difference between let and const?

Both strings and arrays are indexed, meaning the content inside them is in an order, and each element of the content has a position, starting from 0. What is the difference between manipulating a string using its indexes and manipulating an array using indexes?

What array methods are most useful?

I have been using the word ‘methods’ in many contexts, but what is the difference between a method and a function?

Think of some basic interactions you have with machines and computers in your daily life. Can you imagine how the programs that operate these machines might work? Try to speculate a little and maybe even write out your ideas.

Basic Practice:

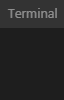
In this section, just type the following things into node in your terminal. Do not alter what you type; just type it and press enter. After you type it, come back to this word document and write a brief summary of what happened. You can think of this as your own personal reference guide, where you document how things work in Javascript.

Warning! Do not copy directly into the terminal from this file. Doing so will not work the way you would think, because Word formats certain things, especially “”, and the formatting will cause issues. Instead, type everything in. Not sure what I mean by this? Ask!

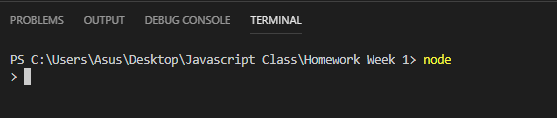
Here also is a brief refresher on getting node working.

First, open VS Code

Next, open a terminal



Next, type node into the terminal:



If you see the above, you can start typing the practice below where the > is!

1. 2 + 6

2. 5 -3

3. 4 \* 6

5. 9 / 3

6. 6 \*\* 2

7. 4 % 3

7. 7 – 3.5

8. 3.2 + .8

9. 2.5 \* 2

10. 9 / 4.5

11. let number = 1

12. number++

13. number--

14. number += 10

15. number -= 9

16. number \*= 10

17. number /= 10

18. “Hello” + “ World”

19. let myCity = “Beijing”

20. myCity = “Shanghai”

21. let myCityArray = myCity.split()

22. let myCityStringAgain = myCityArray.join(‘’)

23. let myList = [“Dogs”, “Cats”, “Rabbits”]

24. myList[0]

25.myList[2]

26. myList.reverse()

27. myList.push(“Hamsters”)

28. myList.pop()

29. myList.pop(“Dogs”)

30. myList.unshift(“Dogs”)

31. myList.shift()

32. myCity.toUpperCase()

33. let number = 5

if (number > 3) {

console.log(‘It is greater than 3’)

}  
 else if (number < 3) {

console.log(‘It is less than 3’)

}

34. let newNumber = 5

35. newNumber.valueOf()

36. newNumber.toString()

37. let decimalNumber = 3.25

38. decimalNumber.toPrecision(2)

39. let flexibleString = “ Watch me change “

40. flexibleString.trimRight()

41. flexibleString.trimLeft()

42. flexibleString.trim()

43.let wordOne = flexibleString.substring(2, 7)

44. let wordTwo = flexibleString.substring(8,10)

45. let wordThree = flexibleString.substring(11, 17)

46. flexibleString.startsWith(' ')

47. flexibleString.startsWith('W ',2)

48. flexibleString.startsWith(' m',8)

49. flexibleString.startsWith(' c',11)

50. let smallString = “short string”

51. smallString.repeat(3)

52. smallString.repeat(0)

53. smallString.indexOf(‘s’)

54. smallString.lastIndexOf(‘s’)