Task: Arrays

The purpose of this chapter is for you to write functions that do not mutate the array arguments; instead you need to make a new copy. You're not used to doing this. Ways to do this:

Using the spread operator to make a new copy of any array argument, on the first line of your function.

For example, const newArg = [...arg]

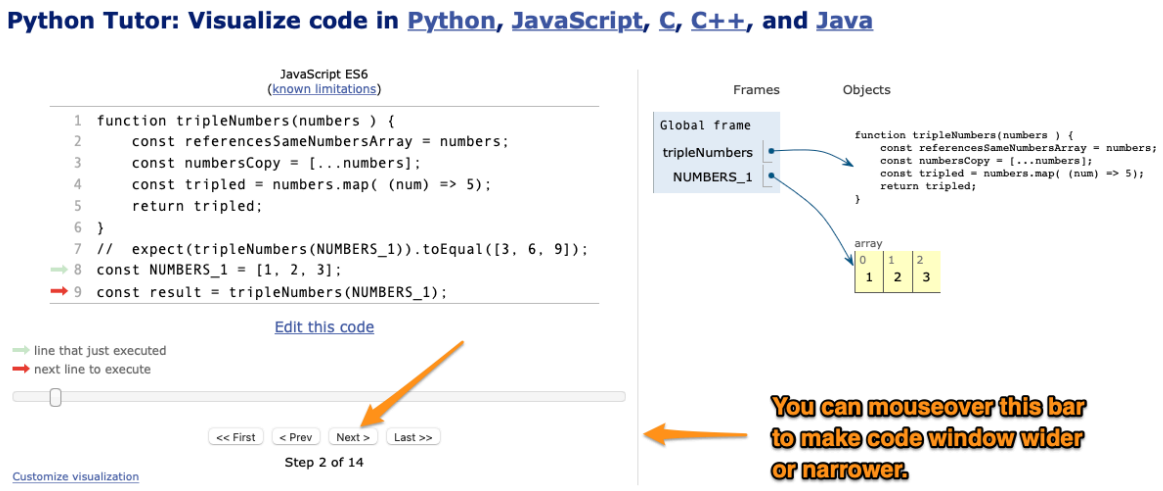
Using map, filter, every, etc.

Other things you may need to use:

.length on an array

potentially other things

Visualize this example to learn how to visualize code



To use this on your own code, you'll have to copy-paste your function code here: https://pythontutor.com/javascript.html#mode=edit **but you'll have to make changes to make it work since the tutor doesn't work with Typescript,** and you have to add a call to your function to visualize your code. See an example further below.

Now, read the textbook first. Execute all the code examples and understand them, and ask questions if you don't. It's easier to figure them out now, than trying to learn the same things with code you wrote that probably is not correct.

How to deal with common problems:

You'll probably forget to do the above. When that happens, **you will find your tests passing when run individually, but then they fail when you run all of them.** That's because the arrays used in the tests are getting mutated/changed by your functions. **Look over all your functions and make sure you're not mutating/changing any arguments. This is the only cause for that behavior.**

If you have a syntax error in your code, it may look like only one of your tests runs then fails, when you expect all of your tests to pass. Read your error messages when your tests fail, to detect this.

Sometimes the prettier extension will underline with a red squiggle something it doesn't like. Save your file and that may solve the issue. Always try that first.

When you're trying to solve a problem, look through all the examples in the textbook for functions that might be useful for you, or similar code you can copy paste then edit to fix your problem.

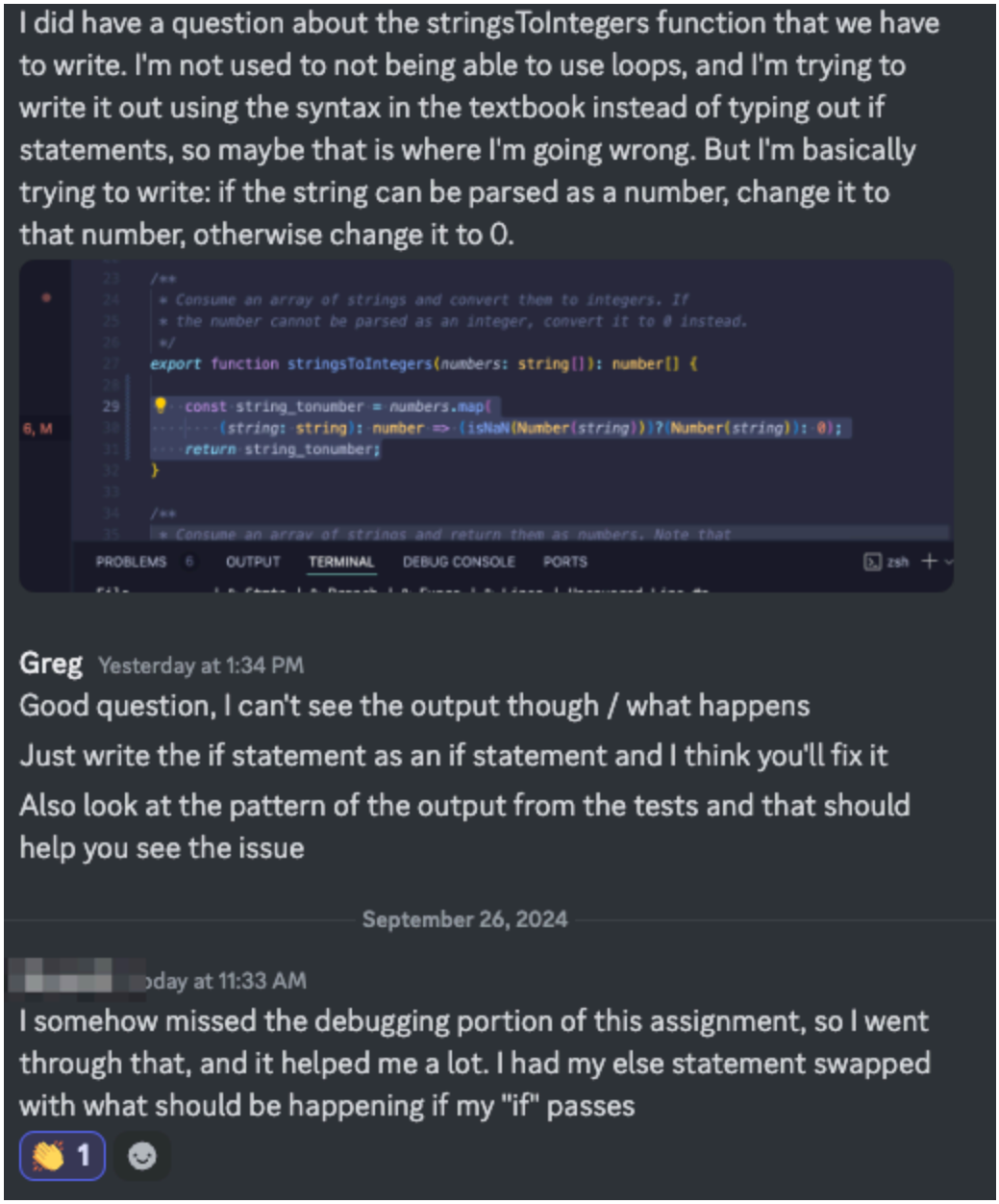
You might not fully have internalized that arrays are references. Predict the value of these variables. The answer is at the end of this document.

```
  const array1 = [1, 2]
  const array2 = [3, 4]
  const array3 = array1;


  array1[0]=10;

  function f(x) {
    x[0]=100;
    x=array2;
  }
  const array4=array3;

  array3[1] = 1000;
  f(array1);


// what are all the values of all the variables, write them below
//   array1 =
//   array2 =
//   array3 =
//   array4 =
```

Go check your answers (at the end of this document). There's even bonus material down there to help you. If your answers are different, try to fix your understanding, but practically speaking you'll take time to do that so you **will benefit from using the debugger to find errors in your code. Save yourself time by diligently using the debugger to confirm what you think the problem is / what is happening, before editing your code.**

Common problem: not using if statement(s)

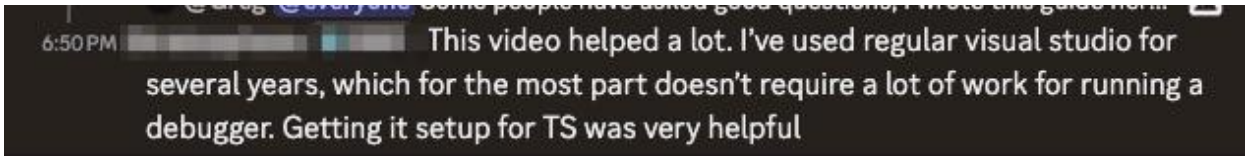
Example below. Separately, sometimes you should have an if statement with multiple cases i.e. if (...) .. else if (...) …



Using the debugger in VSCode:

Watch this video on how to set up and use a debugger in VSCode. For each line of your solution, before you execute it, think what you want it to do, then look at the variables in the debugger after it executes to see if it did what you think. See more in this guide which has been in the Brightspace assignments also: **How to do and debug assignments with automated tests** which has much more detail on setting that up.

Student testimonial to encourage you to actually do the above:



For reference, you're probably not trying to do this, but don't use the javascript console in the browser to debug typescript, it doesn't understand typescript annotations.

Potentially better more visual debugger:

Visualize this example.

To use this on your own code, you'll have to copy-paste your function code here: https://pythontutor.com/javascript.html#mode=edit **but** you'll have to make changes to make it work since the tutor doesn't work with Typescript, and you have to add a call to your function to visualize your code. **Here's an example:**

For example, to show: (this is not a correct implementation by the way) you must remove all the Typescript type annotations and the export, e.g.

```
export function tripleNumbers(numbers: number[]): number[] {
    const referencesSameNumbersArray = numbers;
    const numbersCopy = [...numbers];
    const tripled = numbers.map( (num: number): number => 5);
    return tripled;
}
```

Here's the code with that removed:

```
function tripleNumbers(numbers ) {
    const referencesSameNumbersArray = numbers;
    const numbersCopy = [...numbers];
    const tripled = numbers.map( (num) => 5);
    return tripled;
}
```

Then add a call with an example value, copy paste that from your test, i.e.


```
function tripleNumbers(numbers ) {
    const referencesSameNumbersArray = numbers;
    const numbersCopy = [...numbers];
    const tripled = numbers.map( (num) => 5);
    return tripled;
}
//  expect(tripleNumbers(NUMBERS_1)).toEqual([3, 6, 9]);
```

const NUMBERS_1 = [1, 2, 3];
tripleNumbers(NUMBERS_1);

If you're having trouble with a function, you could write the correct implementation using loops or recursion but in a new function you make, and have your function for the task call that function, have it pass the tests, then refactor the code using the new map, filter, etc array methods. You could also write new test cases to see where your changed/refactored implementation is different from your loops-based implementation.

If you're having trouble thinking in Typescript, you could write a Python solution, then translate it line by line.

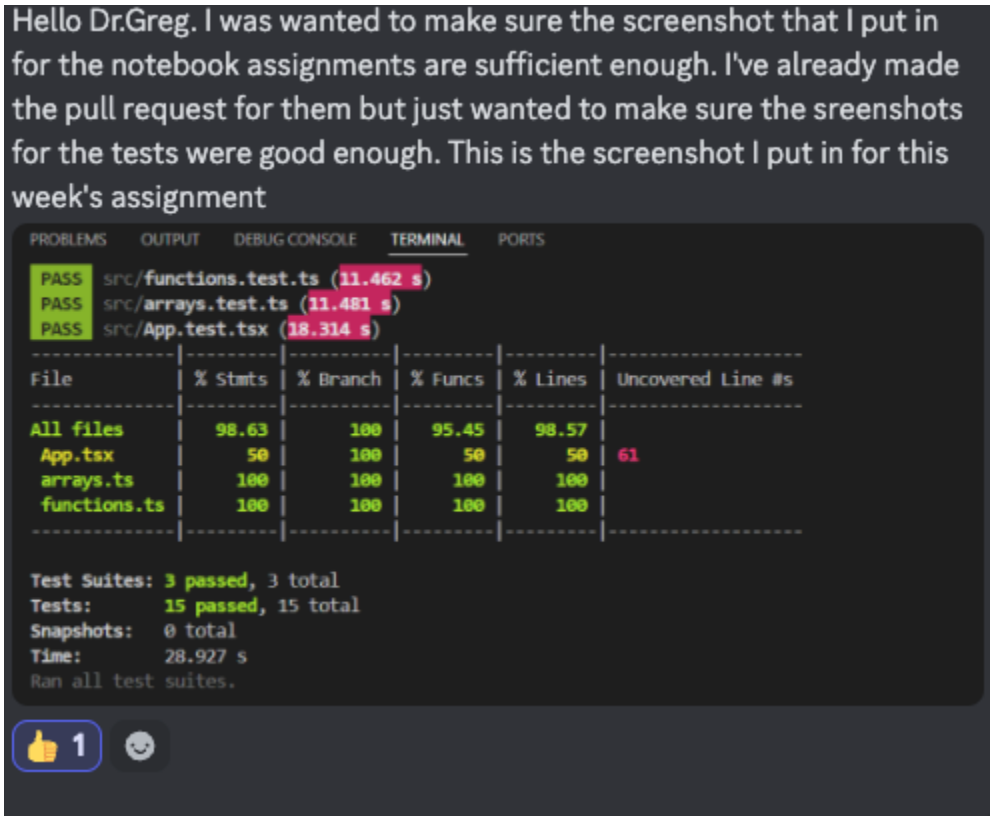
Tasks Arrays Knowledge check question answer

```
const array1 = [1, 2]
  const array2 = [3, 4]
  const array3 = array1;


  array1[0]=10;

  function f(x) {
    x[0]=100;
    x=array2;
  }
  const array4=array3;

  array3[1] = 1000;
  f(array1);

// what are all the values of all the variables, write them below
//   array1 = [100,1000]
//   array2 = [3,4]
//   array3 =  [100, 1000]
//   array4 = [100, 1000]
```

If you have the wrong answer or just want a review, copy and paste the code into https://pythontutor.com/javascript.html#mode=edit

How should the tests look when you are done:

Question: is this how the tests should look when I'm done with the arrays chapter (screenshot shows passing arrays.test.tsx file)



Answer: This is good enough for 100% credit for this week, it shows you are passing the arrays tests for the arrays chapter. However, it also shows you're not starting from your previous solved branch properly (if you're doing it properly, you should have 24 total tests, as in the screenshot below labeled "example that is fully correct" - you can also see the tests files from each task - App.test.ts text.test.tsx HtmlCss.test.tsx functions.test.ts arrays.test.ts). If you have been doing the pull requests like the textbook instructs for each chapter, you can fix easily this by 1) checkout your most recent solved-(whatever your last task was) branch 2) git pull origin main That will take all of the code you had merged into main from prior chapters, and merge that into your solved-(whatever your last task was) branch 3) run your tests to make sure they all pass Now, when you start the next textbook chapter, your local repository will already be on your prior solved branch, just run all the commands in the textbook. If you haven't been doing the pull requests, you'll need to go back and do each one of those for each task, then do the above commands.

Here is an example that is fully correct: