**Udemy**

**JavaScript Fundamentals Part 2**

**Coding Challenge #4**

**The Problem**

1. Create an array ‘bills’ containing all 10 test bill values.
2. Create empty arrays for the tips and the totals (‘tips’ and ‘totals’).
3. Use the ‘calcTip’ function to calculate tips and total values (bill + tip) for every bill value in the bills array. Use a for loop to perform the 10 calculations.

TEST DATA: 22, 295, 176, 440, 37, 105, 10, 1100, 86, 52.

**The Solution**

*const* bills = [22, 295, 176, 440, 37, 105, 10, 1100, 86, 52] ;

Here we have created our array that contains our test data.

We now need to create two arrays; tips and totals.

*const* tips = [] ;

*const* totals = [] ;

**calcTip Function**

We now need to calculate how much tip to add for each bill.

*let* a = 100

*const* tip = *function* (*a*) {

return (*a* \* 0.2) ;

} ;

console.log(`The tip is set at 20% which is; ${bill(a)} when the bill total is ${a}.`);

Now we have a function that calculates how much tip we need to add onto the bill amount.

*let* billTotal = *function* () {

return a + tip(a);

} ;

Here is a simple function that returns our bill total.

**For Loop**

We now need to create a for loop which takes the values out of the bills array and performs calculations on them.

Graphical user interface, text, application

Description automatically generatedRemembering our array from earlier. We have an array length of 10.

It is important that we do not create an infinite loop. We need to set a start and a stop point.

Because the array has a length of 10, we could use this to prevent our loop from exceeding this number. We can use the .length property. In this instance, if we used .length, it would stop at position 9. If we continued to add to the array in the future, it would automatically stop at the final position. This will prevent an infinite loop.

for (*let* i = 0; i < bills.length ; i++ ) {

console.log(bills[i]) ;

} ;

Graphical user interface, application, Teams

Description automatically generatedWe now have a simple loop that starts at position 0 which is 6.

Continues until ‘i’ is less than the length of the array, which is position 9 which is 52.

The for loop will continue in increments of 1.

Now that we have created a basic outline for a for loop, we can use it to take the data points and calculate new tip and bill totals.

//Array for Bills containing test values

*const* bills = [22, 295, 176, 440, 37, 105, 10, 1100, 86, 52] ;

console.log(bills);

*const* tips = [] ;

*const* totals = [] ;

*const* tip = *function* (*a*) {

return (*a* \* 0.2) ;

} ;

*let* billTotal = *function* (*a*) {

return *a* + tip(*a*);

} ;

for (*let* i = 0; i < bills.length ; i++ ) {

console.log(`Your bill total is $${bills[i]}. Your tip total is $${tip(bills[i])}. Your bill total is: $${billTotal(bills[i])}.`) ;

**Text

Description automatically generated**} ;

We have verified that the functions work effectively. Now we need to think about using them to push data into a new array. For this we have created two new arrays;

*const* tips = [] ;

*const* totals = [] ;

Tips and totals. There are multiple ways we can add data to an array, but for this method we are going to use the push method. We can use the push method and then implement our function we created earlier.

**For Loop & Tips**

for (*let* i = 0; i < bills.length; i++) {

tips.push(tip(bills[i])) ;

} ;

We want to start at position 0 so i = 0.

We want our for loop to continue until the end of our bill array. ‘i’ must be less than the length of the bills array, so we set it to less than the length. This is our condition. We want our loop to operate in increments of 1, so ‘i++’.

Now that we have created the for loop with a starting point, an end point, and the increments we want it to take, we need some code to run within in it.

tips.push(tip(bills[i]))

This is a simple push command that calls up our function from earlier.

*const* tip = *function* (*a*) {

return (*a* \* 0.2) ;

} ;

The condition of our push command is to run our tip function. When we run the tip function, we need to set the parameter; in this case ‘a’ to a value. We want the value to be the value from within our bills array. Hence our condition must be ‘bills[i]’. The ‘I’ donates the position.



Success! The code works.

Now we need to think about populating our totals array. We could write a simple function that adds the value from tips, to our original array bills.

Or we could simply write a new function that does everything for us.

**Function Solution**

*let* billFinal = *function* (*a*) {

return (*a* \* 0.2) + *a*;

} ;

This simple function would calculate the tip amount and then add it to create a new total.

We could also run our function from earlier;

*let* billFinal = *function* (*a*) {

return tip(*a*) + *a*;

} ;

Here we are simply calling our function ‘tip’ with the parameter ‘a’. It would produce the same result.

**Array Solution**

We could use a simple push function that takes the values of one array and adds them to the other;

We need to add the new value into a ‘totals’ array.

//for Loop to add one array to the other

for (*let* i = 0; i < bills.length; i++) {

totals.push(bills[i] + tips[i]) ;

} ;

Graphical user interface, text, application

Description automatically generated

Success! Our starting position will be the position at 0. We want the loop to continue running for the length of the bills array. We want the for loop to operate in increments of 1.

We can then add a simple push function; our totals.push will push values into our totals array. It takes the value in the array bills at position ‘I’ and then adds it to the value in the array tips, also at position ‘I’. This adds our new value into our ‘totals’ array.

**Bonus**

Write a function ‘calcAverage’ which takes an array called ‘arr’ as an argument. This function calculates the average of all numbers in the given array.

*const* calcAverage = *function*(*a*) {

*let* sum = 0;

for (*let* i = 0; i < *a*.length; i++) {

sum = sum + *a*[i] ;

} ;

return sum / *a*.length;

} ;

1. First we create a function ‘calcAverage’ which takes the parameter ‘arr’.
2. We then create a new variable ‘sum’ which should be set equal to 0.
3. Then we need a ‘for’ loop;

for (*let* i = 0; i < *a*.length; i++) {

sum = sum + *a*[i] ;

} ;

Here we are simply running a for loop that keeps adding to our sum variable.

‘sum + a[i]’ takes the value of position ‘I’ in the array ‘a’. A is an arbitrary number that we can use as a placeholder.

We then return the sum divided by the length of the array .

We can use a consle.log to check if it is working;

console.log(calcAverage(arr);)

console.log(calcAverage(totals));



**Final Code**

//JavaScript Fundamentals Coding Challenge #4

//Array for Bills containing test values

*const* bills = [22, 295, 176, 440, 37, 105, 10, 1100, 86, 52] ;

console.log(bills);

*const* tips = [] ;

*const* totals = [] ;

*const* tip = *function* (*a*) {

return (*a* \* 0.2) ;

} ;

*let* billTotal = *function* (*a*) {

return *a* + tip(*a*);

} ;

*let* billFinal = *function* (*a*) {

return tip(*a*) + *a*;

} ;

console.log(billFinal(22)) ;

//for Loop to add to our array

for (*let* i = 0; i < bills.length; i++) {

tips.push(tip(bills[i])) ;

} ;

//for Loop to add one array to the other

for (*let* i = 0; i < bills.length; i++) {

totals.push(bills[i] + tips[i]) ;

} ;

console.log(`This array is for tips; ${tips}`) ;

console.log(`This array is for totals; ${totals}`);

*let* arr = [

5,

6,

7,

9,

] ;

*const* calcAverage = *function*(*arr*) {

*let* sum = 0;

for (*let* i = 0; i < *arr*.length; i++) {

sum = sum + *arr*[i] ;

} ;

return sum / *arr*.length;

} ;

console.log(calcAverage(arr));

console.log(calcAverage(totals));