JavaScript (ES5 and earlier)

LAB GUIDE

LANGUAGE BASICS

1. Use a while loop to print the first 10 numbers in the following sequence
1, 2, 4, 8, 16, 32, 64, ..., 2^(n - 1), ...

2. Use a for loop to print the first 10 numbers in the following sequence
1, 2, 4, 8, 16, 32, 64, ..., 2^(n - 1), ...

3. Use for loops to print the following
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
Hint: Since console.log() prints a newline character at the end you will need to concatenate values before printing a line

4. Use for loops to print the following
1
2 3
4 5 6
7 8 9 10
11 12 13 14 15

5. Create an array with a few numbers (at least 4). Now,
* Print the length of the array (number of items in the array)
* Increment the value of the first item
* Re-assign the value of the last item to the sum of the last 2 items.
* Add another number after the end of the array
Example: If the array is [1, 3, 5, 7] and you need to add 9, it is added to the end of the array, and the array becomes [1, 3, 5, 7, 9]
* Declare a new array that is empty (has no items). Use a for loop to copy the array items into the new array.

6. Write a for loop that calculates sum of items in an array of numbers.
Example: For array [1, 2, 3, 4] it calculates the sum as 10

7. Write a for loop that calculates sum of squares of items in an array of numbers.
Example: For array [1, 2, 3, 4] it calculates the sum of squares as 30 (i.e. 1 + 4 + 9 + 16)

8. Write a for loop that creates a new array with squares of numbers in a supplied array.
Example: Supplied array is [1, 2, 3, 4]. You should generate a new array [1, 4, 9, 16] from it.

FUNCTIONS

9. Write a function square() that returns the square of a number passed to it. Use function declaration syntax to declare the function.

Example::

```
console.log( square( 3 ) ); // prints 9
```

10. Write a function that logs "Good morning", "Good afternoon", "Good evening", or "Good night" based on the hour of the day, and call it.

* 5 <= hour < 12: "Good morning"
* 12 <= hour < 16: "Good afternoon"
* 16 <= hour < 20: "Good evening"
* Otherwise, "Good night"

Note: The hour can be obtained by using this statement

```
var hour = (new Date()).getHours()
```

11. Write a function *sumArray* that calculates and returns the sum of items in an array of numbers that is passed to it, and call it like so

```
var result = sumArray( [ 1, 2, 3, 4 ] );
console.log( result ); // prints 10
```

12. Write a function *sumSquares* that creates a new array with squares of numbers in the array passed to it, and call it like so

```
var result = sumSquares( [ 1, 2, 3, 4 ] );
console.log( result ); // prints 30
```

13. Write a function *contains* that accepts an array, and a number and returns true if the number appears in the array, and false otherwise. Use function expression syntax to declare the function

```
console.log( contains( [ 1, 2, 3, 4 ], 3 ) ); // prints true
console.log( contains( [ 1, 2, 3, 4 ], 5 ) ); // prints false
```

14. Write a function that accepts another function and calls the accepted function

15. Write a function *sum* that accepts 2 numbers (say *x* and *y*) and another function (say, *transform*) as arguments. The transform function should be a function that accepts a number and returns another number - for example, a function *square* that accepts a number and returns the square of a number. The sum() function applies the transform function on each of each and y and returns the sum of the results of calling transform - for example, sum() would return $x_2 + y_2$ if called as sum(x, y, square);

Example:

```
function square( x ) { return x * x };

function cube( x ) { return x * x * x };
```

```
console.log( sum( 2, 3, square ) ); // prints 13
console.log( sum( 2, 3, cube ) ); // prints 35
```

16. Write a function *sumArray* that works like so.

```
console.log( sumArray( [ 1, 2, 3 ], square ) ); // prints 14
console.log( sumArray( [ 1, 2, 3 ], cube ) ); // prints 36
```

17. Write a function *exponentFactory* that accepts a number, say x. Define 2 functions *square* and *cube* within it (which accept a number each, and return the square and cube respectively). If x is 2, exponentFactory returns the square function, if 3 it returns the cube function. For any other input it returns a function that returns the number it accepts as such. Call the exponentFactory() function and then the returned function, and log the result.

Example:

```
var fn;

fn = exponentFactory( 2 );
console.log( fn( 5 ) ); // prints 25;

fn = exponentFactory( 3 );
console.log( fn( 5 ) ); // prints 125;

fn = exponentFactory( 4 );
console.log( fn( 5 ) ); // prints 5;
```

18. Write a function *addTo* that accepts a number x. addTo() returns a function that accepts a number y and returns the sum of x and y. Call the addTo() function few times, and then the returned function, and log the result.

Example:

```
var addTo10 = addTo( 10 );
console.log( addTo10( 5 ) ); // prints 15
console.log( addTo10( 7 ) ); // prints 17

var addTo20 = addTo( 20 );
console.log( addTo20( 5 ) ); // prints 25
console.log( addTo20( 7 ) ); // prints 27
```

OBJECTS

19. Create 2 objects (that represents 2 persons, say John and Jane), each with 2 properties - name (a string), and age (a number).

* Print John's age.
* Increase Jane's age and print the Jane object.
* Add an address property to John and set it to an object with "first line" and "city" as properties (the values for these properties also need to be set).
* Print John's city name

* Add a new property spouse to each object. Set John's spouse property to Jane object, and Jane's spouse property to John object
* Add an emailids property to Jane. Set it to an array with 2 strings representing Jane's email ids.
* Print the second email id of Jane.
* Change the second email id of Jane and print it.
* Add a third email id for Jane and print the Jane object.
* Add a method celebrateBirthday() on John that adds 1 to the John's age. Call it on John to increase John's age.
* Add a method celebrateBirthday() on Jane that adds 1 to the Jane's age. Call it on Jane to increase Jane's age.
* Wouldn't it be nice to have a single celebrateBirthday() method shared by both John and Jane objects? Declare celebrateBirthday() as a global function and set it up as a method on both John and Jane objects. Call it to check it increases the age.

20. Create a movie object that represents details of your favorite movie. Suggested information to have in the object - name, cast (an array of strings with cast member's names), yearOfRelease, boxOfficeCollection, addToCast(newMember) that accepts a new cast member's name and adds to the cast array, addToCollection(amount) that accepts box office collections for a week and adds it to the current boxOfficeCollection.

FUNCTION CONTEXT

21. Declare a function *foo* and log its context
* Use bind() to create a new function where the context is the object { x: 1 } instead
* Call the bound function

22. Declare a function *sum* that accepts 2 numbers *x* and *y*, and returns their sum.
* Use bind() to create a new function where x is bound to 10, and the context is unchanged. Call the bound function with a value for y, and log the result.
* Use bind() to create a new function where x is bound to 10, and y to 20 (context is unchanged). Call the bound function, and log the result.

BUILT-IN CLASSES – STRINGS, ARRAYS

23. Given the following snippet of code, solve the questions that follow.

```
var numbers = [ 5, 11, 13, 7, 2, 31, 3, 19, 23, 17, 29 ];
```

* Sort the numbers in the array in increasing order and print the array
* Sort the array in decreasing order and print the array
* Add the number 37 to the end of the array using push()
* Remove the last 2 numbers in the array
* Remove the numbers at indices 3, 4 (i.e. the 4th and 5th numbers) and insert the strings 'Seven' and 'Eleven' in their place.
* Use indexOf() to check if 23 belongs to the array or not. Also, check if 41 belongs to the array or not.

24. Given the following array, solve the questions that follow using appropriate array iterator methods (forEach, find, filter, map)

```
var days = [ 'Sunday', 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday' ];
```

* Print the name of each item in the array
* Create a new array with the number of letters in each day (example, Sunday has 6 letters). Thus the new array that should be generated would be [6, 6, 7, 9, 8, 6, 8]
* Create a new array with the days that begin with letters S - Z. Thus the new array that should be generated would be ['Sunday', 'Tuesday', 'Wednesday', 'Thursday', 'Saturday']
* Create a new array with days that have exactly 6 letters. Thus the new array that should be generated would be ['Sunday', 'Monday', 'Friday']

25. Given the following array, solve the questions that follow using appropriate array iterator methods (forEach, find, filter, map)

```
var phones = [
   { name : 'Samsung Galaxy S10+ Plus', price: 620, type: 'refurbished', manufacturer: 'Samsung' },
   { name : 'Apple iPhone 7 Plus', price: 450, type: 'refurbished', manufacturer: 'Apple' },
   { name : 'One Plus 6', price: 430, type: 'new', manufacturer: 'OnePlus' },
   { name : 'Apple iPhone Xs', price: 910, type: 'new', manufacturer: 'Apple' },
   { name : 'One Plus 7', price: 430, type: 'refurbished', manufacturer: 'OnePlus' },
   { name : 'Apple iPhone 8 Plus', price: 610, type: 'new', manufacturer: 'Apple' },
];
```

* Create a new array with the name of each phone. Thus the new array that should be generated would be ['Samsung Galaxy S10+ Plus', 'Apple iPhone 7 Plus', ...]
* Create a new array with objects representing new phones
* Find all the phones whose price is less than $440 and print them.

26. Given the following string, solve the questions that follow.

```
var quote = 'With great power comes great responsibility';
```

* Create a string from the given string (quote) by replacing 'responsibility' with 'electricity bill'
* Print the index of the first occurence of the word 'great'
* Print the first 10 letters of the string

DOM AND EVENT HANDLING

27. Create an HTML page with 3 paragraphs (give each a unique id). Now do the following by manipulating the DOM via JavaScript.
* Create a new paragraph with some text within, and add it before the first paragraph.
* Create a new paragraph with some text within, and add it to the end of the document body
* Print the HTML within the second paragraph to the console
* Replace the contents of the second paragraph with 2 spans, each with some text
* Print the id of the second paragraph
* Change the id of the second paragraph
* Set the name attribute for the second paragraph to some value. Use setAttribute() to do so.
* Set styles for the third paragraph and turn it to crimson color background with white colored text
* Define a class in CSS that sets a black border, small padding and olive colored text. Add the class to the last paragraph.

28. Create an HTML page with a paragraph, a form with text input (with minlength set to 8) and a submit button. Now do the following.
* Set up an event handler that shows an alert dialog with text of the paragraph when it is clicked

* Set up an event handler that turns the background color of the input to lightgreen when the number of characters within the input is 8 or more (this happens as the user types)
* Set up an event handler that prevents form submission when the submit button is clicked. It then checks to see if number of characters within the input is at least 8. If so, it submits the forms. Else, an appropriate error message is displayed below the input.

29. Create a Bootstrap-like panel component with a header and body. Give it appropriate styles using CSS.

* Set up an event handler so that the panel body toggles (opens/closes) when the header is clicked
* Add more such panels to your page and make sure clicking a panel's header toggles only that panel's body
* Create a function that accepts a panel DOM node and turns it into a toggleable panel

Reference: https://getbootstrap.com/docs/3.4/javascript/#collapse-example-accordion

30. Create a page with 2 inputs (both accepting number), button, and a span (initially empty).

* When the button is clicked, the two numbers within the input are added and displayed inside the span
* Now add a dropdown (select input) with options +, -, *, / (default selection is +). When user select an option, the contents of the span change to reflect the result of the new operation.

AJAX

31. Within an HTML page, write a script to fetch data about things to do from https://jsonplaceholder.typicode.com/todos. Display the results in a list in the page. Make sure to display an error message in the same page in case of any errors.

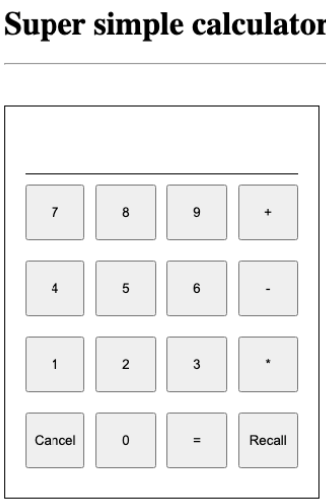
32. Within an HTML page, write a script to fetch data about products from https://awesome-store-server.herokuapp.com/products. Display the results in a card layout (gallery). The image of the product should be placed on top every card. Make sure to display an error message in the same page in case of any errors.

OBJECT PROTOTYPE

33. Create an object vehicle with name (string), averageSpeed (number) and type (string with value "air" | "water" | "land"). Create 2 other objects - car, aeroplane, with vehicle as their prototype. Add engineCapacity, typeOfFuel to each and set them. To the car object, add numAirbags. To the aeroplane add thrust.

CALCULATOR IMPLEMENTATION IN JAVASCRIPT

Define a calculator like the following



Instructions

1. Define an IIFE to avoid creating global variables.
2. Define an array to hold results, and an index to maintain for last recalled result (lastRecalled) initialized to 0.
3. Define a function onNumberPress() that is invoked on click of number buttons (".btn-number"). It reads the number (text within the button), and appends it to the display input's (".display") value.
4. Define a function onOperatorPress() that is invoked on click of operator buttons (".btn-op"). It reads the operator (text within the button), and appends it to the display input's value. But before doing so it checks if the last character displayed is an operator (an operator cannot be followed by another operator).
5. Define a function onCancelPress() that is invoked on click of cancel button (".btn-cancel"). It clears the display and resets lastRecalled to last index of results array (so that a future recall button press will start recall with latest results).
6. Define a function onEvalPress() that is invoked on click of evaluate button, i.e. the = button (".btn-eval"). It reads the display value and evaluates it using JavaScript's eval() function, and shows the result in the display. It also saves the result to the results array. Set lastRecalled to last index of results array.
7. Define a function onRecallPress() that is invoked on click of recall button (".btn-recall"). It decrements lastRecalled value, gets the value of results at lastRecalled index and sets the display to that value.