Variables and constants:

* Declare a variable with a string value.
* Declare a variable with an int value.
* Declare a variable with a floating-point number value.
* Declare a variable with a Boolean value.
* Declare a variable with a value and change the value on a new line.
* Declare two int variables and add them together in a third variable. Print the result in the console.
* Declare a constant with a string value. Print the value into the console.
* Add together 5 and “5” in a const. Print the value in the console.
* Print the data type of this const into the terminal
* Declare two strings in two separate constants. Concatenate together the two values in a new const. Print the value in the terminal.
* Declare two variables with two separate values. Swap the values of the variables without directly typing in new values.

Mathematical operations:

* Add together two numbers in a variable. Print the result in the console.
* Subtract two numbers from each other in a variable. Print the result in the console.
* Multiply two numbers with each other in a variable. Print the result in the console.
* Divide two numbers with each other in a variable. Print the result in the console.
* Add together two numbers and subtract a third one from the result of the addition. Print the result in the console.
* Find the remainder of 23784 / 458 and print it in the console.

Arrays:

* Declare an array with five integers.
* Print the first value of the array in the console.
* Print the last value of the array in the console.
* Find the length of the array and print it.
* Print the last value of the array without using a concrete int as index number.
* Add a new value to the array.
* Remove an element from the array.
* Change the third value to 15.
* Declare an int variable with a random value. Assign the second element in the array the value in the variable without typing out the number.
* Declare a const with the length of your array -2. Change the value of the array element with the index number that equals your const to a new value.

Conditionals:

* Declare a const with a floating-point number. Print the number in the console if the number is bigger than 15. Print “Lower than 15.” if not.
* Declare a const *price*. Print “Cheap.” If the value is lower than 100, and print “Expensive” if more than 150. Print affordable otherwise. Make two different logical solutions.
* Simulate the Hungarian school system in a short program. Declare a const called *age.* If *age* is lower than 6, print “The child is in kindergarten.” If the age is between 7 and 14, print “The child is going to primary school.” If the age is between 15 and 18, print “The child is going to high school.” If the age is over 18, print “Going to university or working.”

NB! This implementation will be far from ideal, but we need to learn a few more things to have a better solution for this problem.

* Create a const day = new Date().getDay() that yields a 0 if the current day is Sunday, 1 if Monday … and 6 if the current day is Saturday. Create a program that prints the current day in a proper text format. E.g.: “Today is Monday.”
* Declare two integer constants. Write a program that decides whether the square of the first number is bigger than the second number.
* Declare an array with two numbers.
  + If the first element is lower than 50, bring it to its square, change the value of the array element to it, and check if it is still lower than 50. If yes, print “First element still lower than 50.”. If not, print “Now it is bigger than 50.”
  + Else if the first element is bigger than 50, subtract the second element from it, and check if it is still bigger than 50. If yes, print “First element still bigger than 50.”. If not, print “First element is now lower than 50.”

Logical operators:

|  |  |  |  |
| --- | --- | --- | --- |
| TRUE | AND | TRUE | ? |
| FALSE | AND | FALSE | ? |
| TRUE | AND | FALSE | ? |
| FALSE | AND | TRUE | ? |

|  |  |  |  |
| --- | --- | --- | --- |
| TRUE | OR | TRUE | ? |
| FALSE | OR | FALSE | ? |
| TRUE | OR | FALSE | ? |
| FALSE | OR | TRUE | ? |

|  |  |  |
| --- | --- | --- |
| NOT | TRUE | ? |
| NOT | FALSE | ? |

|  |  |  |  |
| --- | --- | --- | --- |
| 5 < 6 | && | (10 % 2 == 1) | ? |
| !true | && | !!true | ? |
| !false | || | !!false | ? |
| 5 == “5” | && | 5 ===”5” | ? |
| !(23 == “23”) | || | 15 > 50 | ? |
| null == undefined | && | null === undefined | ? |
| !null == true | || | !!null === false | ? |
| 12 > 50 && !true | || | !!true == (45 == “45”) | ? |

Loops:

* Make a loop that prints the first 10 numbers starting from 0.
* Make a loop that prints every even number up to 10 starting from 0.
* Make a loop that prints every odd number up to 10 starting from 0.
* Declare an array of 10 random integers. Write a loop that calculates and prints the sum of all odd and even numbers separately.
* Declare an array with 5 string values and print every value in it with the help of a loop.
* Declare an array of 5 integers and change their value to the current iteration index number of your loop.
* Starting from the number 100 backwards, find and print every number that is dividable by 3.
* Starting from the value 10000 backwards, find and print the first number that is dividable by 456.
* Write a loop to calculate the sum of the first 10 natural numbers.
* Write a loop to calculate 99!.
* Declare two int variables. With the help of a loop, raise the first value to the power of the second value.
* Write a program to print the following shapes:
  + \*\*\*\*\*   
    \*\*\*\*\*   
    \*\*\*\*\*
  + \*  
    \*\*  
    \*\*\*  
    \*\*\*\*  
    \*\*\*\*\*
  + \*  
     \*\*  
     \*\*\*  
     \*\*\*\*  
    \*\*\*\*\*

Functions:

* Write a function that takes a name as a string argument and prints “Hello <name>!”
* Write a function that only greets Alice and Bob, otherwise prints “I don’t know you.”
* Write a program that decides if a number is even or odd.
* Write a function that takes two integers X and Y as arguments and prints “<X> to the power of <Y> equals: <result>“.
* Write a program that takes an array as argument and prints “The content of this array is the following:”, and then prints the content.
* Write a program that takes a number N and prints the sum of all number from 0 to N.
* Write a program that takes a number N and prints N!
* Write a function that prints the largest number in an int array.
* Write a function that reverses an array and prints the result. [1,2,3] -> [3,2,1]
* Write a function that takes two arrays and concatenates the two elements with the same index number and prints it. [1,2,3], [“a”, “b”, “c”] -> 1a, 2b, 3c
* Write a function that computes the list of the first 100 Fibonacci numbers. The first two Fibonacci numbers are 1 and 1. The n+1-st Fibonacci number can be computed by adding the n-th and the n-1-th Fibonacci number. The first few are therefore 1, 1, 1+1=2, 1+2=3, 2+3=5, 3+5=8.
* Write a function that changes the value of an array at index number i to a given value.
* Write a function that takes a string as a parameter and breaks it down into characters. The characters are then put into an array. If the character is already in the array, it should not put it in twice. Finally print the array of characters.

Functions with return values and high-order functions:

* Write a function that takes an int array as argument and filters out all the odd numbers from the list and returns a new list with only even numbers.
* Write a function that checks if a given string is a palindrome. Return a Boolean value.
* Write a function that takes two arguments. The first argument is a string value, the second argument is another function. The inner function should only return “My inner parameter was: <value>”. Print the result of the whole function in the terminal.
* Write a high-order function that takes an array of integers and applies a function on each item on the list. The result should be printed each time the inner function is applied on an item. Test your high-order function by squaring every value in an array.
* Write a function that takes an integer array and a number as arguments. Filter the array so that a new array will only contain the numbers that are dividable by the number given in the arguments. Return the new array.

Objects:

* Make a Car class that can receive the brand, fuel type, and top speed attributes of the car. The class should also implement the functionality to:
  + Print the brand of the car.
  + Print the fuel type of the car.
  + Print the top speed of the car.
* Make a Calculator class that implements the following functionality:
  + Addition of two numbers.
  + Subtraction of one number from another.
  + Multiplication of two numbers.
  + Division of two numbers.
  + Squaring one number.
  + Find the square root of a number.
* Make a morse-code generator that has the following functionality:
  + Take a random string and prints it in morse code. Make sure the morse code is separated by spaces. E.g.: S O S -> … --- …
  + Take a random morse code and turn it into a readable string and print it. E.g.: … --- … -> S O S
  + Create morse-data.js to export the data object with the letter – morse code key-value pairs:
* let data = {
* A: ".-",
* B: "-...",
* C: "-.-.",
* D: "-..",
* E: ".",
* F: "..-.",
* G: "--.",
* H: "....",
* I: "..",
* J: ".---",
* K: "-.-",
* L: ".-..",
* M: "--",
* N: "-.",
* O: "---",
* P: ".--.",
* Q: "--.-",
* R: ".-.",
* S: "...",
* T: "-",
* U: "..-",
* V: "...-",
* W: ".--",
* X: "-..-",
* Y: "-.--",
* Z: "--.."
* }
* export default data;
  + Reuse the following method to find a key based on a value:

getKeyByValue(obj) {

      return Object.keys(obj).find(key => obj[key] === value);

}

Inheritance:

* Create an Animal class with parameters of species, height, weight, diet, and class.
  + Write a function to get the species of the animal.
  + Write a function to get the weight of the animal.
  + Write a function to get the diet of the animal.
  + Write a function to get the class of the animal.
* Create a subclass Mammal with parameter of fur color.
  + Write a function to get the fur color of the animal.
* Create a subclass Bird with parameters of wingspan, and number of eggs.
  + Write a function to get the wingspan of the animal.
  + Write a function to get the number of eggs of the animal.
* Create a subclass Reptiles with the parameter of skin color.
  + Write a function to get the skin color of the animal.
* Create a class Building with the parameters name, area, and max number of people allowed inside.
  + Write a function to get the name of the building.
  + Write a function to get the area of the building.
  + Write a function to get the max number of people allowed inside of the building.
* Create a subclass TicketOffice with the parameter of:
  + A list of full price tickets to be sold.
  + A list of bird-house only tickets to be sold.

The office should have the following functionality:

* Produce not more tickets than the max number of allowed people in the Zoo.
  + If the max number of people allowed is an odd number, produce just +1 full price ticket. The rest should be only bird-house tickets.
* Sell a ticket and warn if tickets are sold out. The buyer should be able to buy a ticket to entire Zoo, or just to the bird house. Return the ticket object.
* Display current number of available tickets.
* Create a subclass Zoo with the parameters bird house, reptile park, mammal section, list of visitors.
  + Write a function to add a bird to the bird house.
  + Write a function to add a reptile to the reptile park.
  + Write a function to add a mammal to the mammal section.
  + Write a function to admit a visitor to the Zoo. Checks if the visitor has a ticket at all. Add the visitor to the list of visitors, but not more than the maximum capacity of the Zoo.
  + Write a function to admit a visitor to a specific section of the Zoo. Visitors with full tickets should be able to enter every part of the Zoo, visitors with only bird house entrance should be able to only enter the bird house and get rejected otherwise.
* Create a class Visitor with the parameter ticket.
  + Write a function to take a Ticket.
  + Write a function to show the ticket. It returns a Ticket object.
* Create a class Ticket with the parameter price and type.
  + Write a function to show the type of the ticket.