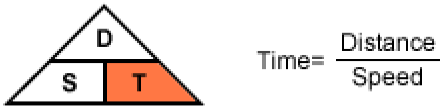
Coding Basics

Setup

- Use the same exercise-runner project for all the exercises below
- For each exercise, create its own file, named as: 01.js, 02.js…
- Copy the question as a comment at the top of your javascript file
- Make sure your projects are runnable (when ex button is clicked)
- The output should appear in the console in an easy- to-understand way

Basics

1. Read first name and last name, also declare a variable *fullName*, and then welcome the user by his full name.
2. Read two numbers and print the result of the following operations on them: (*, /, %)
3. Read temperature in Celsius, and print in Fahrenheit
4. Read distance and speed and print the time



(Now let's build waze ;))

5. Ask the user for 3 digits and print the number in full
   for example: if the user entered 3,2,6 then we should have a variable holding the value 326 and then print that variable.
   a) BONUS: In this case, working with strings is easier, try solving using numeric variables
6. Read a, b, c from the user, (for example: 2, -5, 2)
   your output should be something like:
   $2x^2 – 5x + 2 = 0$
   $x1 = 2 ; x2 = 0.5$
   Hint: To print the X^2 (X Baribua…) to the console, use this: string: 'x\u00B2'

Conditions

7. Read 3 numbers and check if the 3rd is the sum of the first two, if so, print all numbers to the console like this: 6 + 4 = 10
8. Read 3 numbers and print the smallest
9. Read 2 positive numbers and calculate the difference (absolute value)

   - If the diff is smaller from both values say that those numbers are relatively-close (i.e. – num1=5, num2=9 then diff=4 => relatively-close!)
   - Validate that you got numbers
     (hint: search for something like: *javascript check if number*)

10. Ask the user how many friends he has on FB and print out an analysis:
    - More than 500 – OMG, celebrity!
    - More than 300 (and up to 500) – Well connected!
    - 100 and more – Knows some people
    - Up to 100 – Quite picky aren't you?
    - 0 – Lets be friends?
11. Rolling Project: BankSystem
    a) Initialize a variable: currBalance with the value: 1000
    b) prompt for the user secret pin code and how much he would like to withdraw, and then print a nice message with the new balance.
    - Check that the secret pin code is '0796', if not, tell the user its wrong and don't let him withdraw
    - Add feature: don't let the user withdraw more than he has in the account
12. Guess Who
    - Alert the user to think about some actor
    - Using the *confirm* function, ask the user 2 yes-no questions:
      - Question 1: Male?
        - Yes:
          - Question 2: Blond?
            - Yes: Philip Seymour!
            - No: Tom Cruise!
        - No:
          - Question 2: English?
            - Yes: Julianne Moore!
            - No: Natalie Portman!
13. The Elevator –
    - Keep a currentFloor variable, initialize it to 0
    - Ask the user which floor he needs

- Validate its between -2 and 4
- Update the currentFloor variable accordingly
- And let the use know his current floor
- If the user goes to floor 0 add 'Bye Bye'
- If the user goes to parking (negative floors) add: 'Drive Safely'

Functions

14. Write a function that gets a name as a parameter and greets him
15. Write a function that gets 2 numbers and return their sum
16. Write a function *isEven* that gets a number, and returns *true* if the number is even otherwise *false*.
17. Write a function getBigger that receives 2 numbers and returns the bigger.
18. Write a function: *isOfAge* that gets a name and an age, it the user is younger than eighteen, alert him that he is too young, this function also returns a boolean

Loops

19. Read 10 numbers from the users and output "the number is even" if this is the case, else "the number is not even"
20. Read 10 numbers and print:
    - The maximum number
    - The minimum number
    - The average
21. Read numbers, until the number 999 is entered.

    For each number:

    a) Print if it’s divided by 3

    b) Print whether this num is much bigger (more than 10) than the previous number.
22. Write the function *myPow* that gets 2 params: *base, exp* and returns the power.
23. Write the function *factorial* that gets a number n and return n!
24. Play with the function Math.abs() , read the documentation in MDN

    a) Write your own function myAbs()

25. Write A function *getRandomInteger(min, max)* get that generates a random integer between min and max.

    Hint: Use Math.Random & Math.Floor

    a) After you're played with it enough (or done), read this page. And look at the *getRandomInt* function.

    b) Yes, it's better, now copy that function and remember to use it whenever you need in later exercises (how amazing is that?)

26. Write a program that generates 10 random numbers.

    The numbers should be generated so each number is greater than the ones generated before.

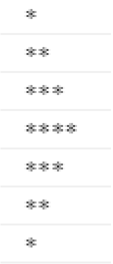
    To simplify, generate the first number n so it is between (0→1000), and each subsequent number will be in the range of (n → n+1000)

27. Asterisks!

    a) Write a function *getAsterisks(length)* that returns a string containing asterisks according the number supplied.

    for example: when the requested length is 4, it returns '****'

    b) Write a function: *drawTriangle*(height) that prints a triangle:



```
*
**
***
****
***
**
*
```

(the requested height here is 4)

Hint: use the function *getAsterisks* in a loop.

c) Write a function: drawMusicEqualizer(count) that generate 100 numbers between 1 and 10 and draws '#' rows at random lengths:

```
#####
# #
######
## ###
```

d) Write a function that prints a block of asterisks (*) by parameters: rows, cols

ex: for 4,5.

```
******
******
******
******
```

Now, print only the outline

```
*****
*     *
*     *
*****
```

e) Surprise, there is a new requirement to support any character (not necessarily asterisk), how easy it is to refactor your code so that the character will be decided by the user and the code will still make sense?

28. Write a program to compute the greatest common divisor (GCD) of two positive integers.
Example: 6, 15 => gcd: 3
Hint: we need something like a loop: i 2 -> 6 and check modulus.

29. Read a number and
    a) Calculate sum of its digits.
    b) Calculate multiply of its digits.
    c) Print it with swapped first and last digits (2731 => 1732)
    d) Check if its symmetric (such as: 95459)
    e) Print its reverse (BONUS: as number not as string).
    f) Sum its first and last digit.

g) Print if it's an Armstrong number.
i.e. 371 is an Armstrong number:
$3^{3+}7^{3+}1^{3} = 371$

h) Print if it's a perfect number.
Perfect number is number that sum of all his dividers is the number itself.
i.e. 6 is a perfect number (1 + 2 + 3).

Now read a number: *max* and prints all Armstorng and Perfect positive numbers until *max*

Strings

30. Read 2 names and print the longer one
31. read a string and print:
    - Its length
    - Its first and last chars
    - in uppercase and lowercase

Strings and Loops

32. Read a string and print it backwards using loop
33. VOWELS (aoieu)
    - write a function that gets a string and print how many times each vowel appears
    - Read a string and make the vowels lowercase, the others uppercase (GiZiM GiDoo)
    - Write a function that reads a string and double all the vowel in it

34. write a function **myIndexOf(str, searchStr)** that accepts 2 strings, and returns the location of the second string in the first, if not found return -1

35. Write the function **encrypt** that gets a string and encrypt it by replacing each character c with c+5 (i.e. 'r' will be replaced by: 'w'), now write a function to decrypt a message.

36. Write a function that gets a comma ',' delimited string that consists of names. Ex: 'igal,moshe,haim'
and returns who has the longest name, and the shortest name.
Tip: use the split function to covert a string to an array

37. Write a function: generatePass(passLength) that generates a password of a specified length. Password is made out of random single-digit numbers.

Arrays

38. Write a function **biggerThan100** that gets an array of nums and returns an array of items that are bigger than 100.

39. Write a function **countVotes(votes, candidateName)** that counts how many votes this candidate got.
i.e.: if the *votes* array looks like this: ['c1', 'c1', 'c2', 'c1']
And the *candidateName* is : 'c2', the function returns 1.

40. Write a function: **getLoremIpsum(wordsCount)** that return a sentence with made-up words (google for: lorem ipsum...)
TIP, here are the steps you may use to get there:

- first write a function getWord that returns a single word with random letters and size 3
  Tip: you can create a string or an array of all the characters in english.
- Now change the size to be a random number between 3-5.
- Now call this function in a loop to create a sentence

41. Write a function **sayNum(num)** that prints each digit in words.
For example:
123 => One Two Three.
7294 => Seven Two Nine Four

TIP: You may use Switch inside a loop OR an array digitNames.
What the heck, try them both.

42. Write a function **startWithS** that gets an array of names and returns an array of names that start with S
    a. Refactor your function to work on any letter by adding a letter argument

43. Write a function **sumArrays** that gets 2 arrays and returns a sum array:
    a. For example, for: [1, 4, 3] [2, 5, 1, 9] it returns: [3, 9, 4, 9]
       TIP: this can be done in a single loop by first identifying the big and small arrays.
    b. Read those arrays from the user (until 999 is entered)
       TIP: write a function: getArrayFromUser and call it twice

Write a function **printNumsCount(nums)** the array nums contains integers in the range 0-3 such as:

3	2	0	2	2	0	3

The function prints how many times each of these nums appear in array.
GUIDANCE: the fact that the values are in a specific range allows us to use an array where the index is actually the number itself and the value in the array is the count

44. Write a function**: findUniques(nums)** the array nums contains numbers in the range 0 -10 such as:

5	4	5	7	1	9	4

the function returns a new array where each value appear only once.
TIP: Notice that the values are in a specific range

45. Write a function: **multBy(nums, multiplier)** that returns a modified array in which each item in the array is multiplied by myltiplier.
    a. Add another param: isImmutable, when true use *array.slice()* to work on a new array and leave the original array as is.

46. Implement your own version of the split function: **mySplit(str, sep)**
    Test with different types of data, i.e. : 'Japan,Russia,Sweden'
    You can assume that the separator (delimeter) is a single character
    (BONUS: don’t assume that)

47. Write a function **getNthLargest()** to get nth largest element from an array of unique nums.
    For example: nthlargest([7, 56, 23, 89, 88, 92, 99, 11], 4)
    Result: 88
    - If we sort the array, its easy
    - BONUS: Try without sorting the array!

48. Read about how to sort an array yourself, by using the **bubble sort** algorithm, see here.
    copy the function, and make it use a normal while loop instead of a do-while loop.

49. Making **Water**! Lets imagine that we have the following atoms:

1	Hydrogen	H
5	Boron	B
6	Carbon	C
7	Nitrogen	N
8	Oxygen	O
9	Fluorine	F

- Represent that in an array with just the atom symbol
- Pick random atoms from an array to create molecules of 3 atoms

- Stop when you get water (H2O – Two Hydrogens and one Oxygen)
- Print how many molecules you had to try before randomly creating water