Basic 03 – Functions

# Exercise 01: Minimum

Write a function min that takes two arguments and returns their minimum.

|  |
| --- |
| // Your code here.  console.log(min(0, 10));  // → 0  console.log(min(0, -10));  // → -10 |

# Exercise 02: Recursion

We’ve seen that % (the remainder operator) can be used to test whether a number is even or odd by using % 2 to see whether it’s divisible by two. Here’s another way to define whether a positive whole number is even or odd:

* Zero is even.
* One is odd.
* For any other number *N*, its evenness is the same as *N* - 2.

Define a recursive function isEven corresponding to this description. The function should accept a single parameter (a positive, whole number) and return a Boolean.

Test it on 50 and 75. See how it behaves on -1. Why? Can you think of a way to fix this?

|  |
| --- |
| // Your code here.  console.log(isEven(50));  // → true  console.log(isEven(75));  // → false  console.log(isEven(-1));  // → ?? |

# Exercise 03: Bean Counting

You can get the Nth character, or letter, from a string by writing "string"[N]. The returned value will be a string containing only one character (for example, "b"). The first character has position 0, which causes the last one to be found at position string.length - 1. In other words, a two-character string has length 2, and its characters have positions 0 and 1.

Write a function countBs that takes a string as its only argument and returns a number that indicates how many uppercase “B” characters there are in the string.

Next, write a function called countChar that behaves like countBs, except it takes a second argument that indicates the character that is to be counted (rather than counting only uppercase “B” characters). Rewrite countBs to make use of this new function.

|  |
| --- |
| // Your code here.  console.log(countBs("BBC"));  // → 2  console.log(countChar("kakkerlak", "k"));  // → 4 |