ZIP

zip together several arrays based the corresponding indexes of their elements. Elements of the input arrays will be joined into new arrays based on identical index positions in their original arrays. zip should then return an array of these new arrays. HINT: You can use Math.max to find which array has the most arguments or you can count the arguments by using arguments.length and access each argument by iterating through and using arguments[i].

INTERSECTION

Find the values that several arrays have in common using intersection. This function takes an arbitrary number of arrays and produces an array containing every element shared between all the passed-in arrays.

DIFFERENCE

Now let's let differences shine. The difference function compares the values in a target array and compares it to other arrays passed in as arguments. difference then returns those values that are only in the target.

THROTTLE

throttle is used when the same function is called multiple times within some window of time, and you want it to only run once within that window. throttle creates an entirely new function that calls the original one after a certain amount of time. You can throttle a function when it is susceptible to being called more often than you want it to run. This function is a bit more complicated, so we have included some extra context for you. \_.throttle(func, wait): Wrap a function func so that it can be called at most once within a period of wait milliseconds. This is useful for throttling access to expensive APIs or to drawing routines in a video game.

Let's see how it's used:

JavaScript var counter = 0; var increment = function() { return counter += 1; };// Create a function called throttledIncrement. This function can be called at // most once every 100ms var throttledIncrement = \_.throttle(increment, 100);throttledIncrement(); // return 1; `counter` should now be 1 throttledIncrement(); // return 1; schedule `increment()` call in 100ms throttledIncrement(); // return 1; should do nothing// Wait 100 ms; `increment` is called

Arguments passed to the throttled function should be passed to the original function. The throttled function should always return the most recently returned value of the original function. If the wait period is 100ms and the function was last called 30ms ago, another call to the throttled function should schedule a call for 0ms in the future.