Coding challenges

Mohamed Boucetta

Exercise 0.1 *Build a function which take a sorted array of integers and a number and returns true if the number is in the array false otherwise. For instance, if* $L = [0, 5, 7, 10, 19]$*, the function returns true if the number is 7 and false if the number is 6.*

Solution : The naive solution is to scan the array and to compare the number to the items of the array. This solution has $O(N)$ complexity and can be used for an unsorted array (N is the number of items in the array).
The fact that the array is sorted permits to have a far more efficient algorithm known as binary search which has $O(ln_2 N)$ complexity.

1. We set three variables low = 0 and high = N and mid = (low + high) / 2.
2. We compare the number to array[mid]. If they match, we return array[mid].
3. If array[mid] < number, since the array is sorted we are certain that the number is not in [array[0],... array[mid]] and we look in [array[mid+1],...,array[high-1]] by setting low = mid + 1.
4. If array[mid] > number, since the array is sorted we are certain that the number is not in [array[mid],... array[high-1]] and we look in [array[0],...,array[mid-1]] by setting high = mid .
5. We repeat this operations while low < high.

Let us see how this algorithm works on a concrete example.

$$L = [2, 3, 4, 5, 6, 7, 8, 10, 32, 54, 78, 90, 101, 120] \quad \text{and} \quad number = 91.$$

1. low = 0, high = 14 and mid = 7.
2. L[7] = 10 < 91. low = 8, high = 14 and mid = 11.
3. L[11] = 90 < 91, low = 12, high = 14 and mid = 13.
4. L[13] = 120 > 91, low = 12, high = 13 and mid = 12.

5. L[12] = 101 > 91, low = 12, high = 12 and the while loop stop and the function returns false.