Coding Problems Patterns

1. Two pointers:

* two pointers pattern uses two pointers to iterate over an array or list until the conditions of the problem are satisfied. This is useful because it allows us to keep track of the values of two different indexes in a single iteration. Whenever there’s a requirement to find two data elements in an array that satisfy a certain condition, the two pointers pattern should be the first strategy to come to mind
* Does my problem match this pattern?
  + Yes, if all of these conditions are fulfilled:
  + The input data can be traversed in a linear fashion, that is, it’s in an array, in a linked list, or in a string of characters.
  + The input data is sorted, or else, arranged in a way that is relevant to the problem, such as numeric data sorted in ascending or descending order, or characters arranged symmetrically.
  + We are only considering the two elements in the input data that are pointed to by the two pointers rather than the whole set of elements located between the two pointers.
* Additionally, problems in this pattern usually involve comparing or swapping values pointed to by two indexes. In less common cases, each index may move along a separate array or string.
* No, if either of these conditions is fulfilled:
  + The input data cannot be traversed in a linear fashion, that is, it’s neither in an array, nor in a linked list, nor in a string of characters.
  + The problem requires an exhaustive search of the solution space, that is, eliminating one solution does not eliminate any others.
  + The problem requires us to find a subset of contiguous elements in the input data structure.
* Questions:
  + Write a function that takes a string s as input and checks whether it’s a palindrome or not.
  + Sum of Three Values: Given an array of integers, nums, and an integer value, target, determine if there are any three integers in nums whose sum equals the target. Return TRUE if three such integers are found in the array. Otherwise, return FALSE.
  + Reverse Words in a String: Given a sentence, reverse the order of its words without affecting the order of letters within a given word.
  + Valid Palindrome II: Write a function that takes a string as input and checks whether it can be a valid palindrome by removing at most one character from it.

1. Fast and Slow Pointers:
   * The fast and slow pointers pattern uses two pointers to traverse an iterable data structure at different speeds. It’s usually used to identify distinguishable features of directional data structures, such as a linked list or an array.
   * The pointers can be used to traverse the array or list in either direction, however, one moves faster than the other. Generally, the slow pointer moves forward by a factor of one, and the fast pointer moves by a factor of two in each step. However, the speed can be adjusted according to the problem statement.
   * Does my problem match this pattern?
   * Yes, if either of these conditions is fulfilled:
     + Either as an intermediate step, or as the final solution, the problem requires identifying:
       - the first x % of the elements in a linked list, or,
     + the element at the k-way point in a linked list, for example, the middle element, or the element at the start of the second quartile, etc.
     + the k th last element in a linked list
   * Solving the problem requires detecting the presence of a cycle in a linked list.
   * Solving the problem requires detecting the presence of a cycle in a sequence of symbols.
   * No, if either of these conditions is fulfilled:
     + The input data cannot be traversed in a linear fashion, that is, it’s neither in an array, nor in a linked list, nor in a string of characters.
     + The problem can be solved with two pointers traversing an array or a linked list at the same pace.