**MergeSort:**

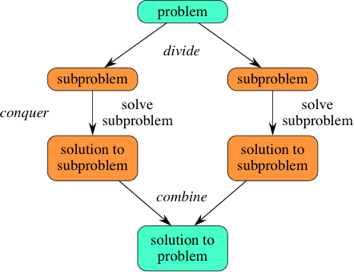
Un algoritmo de ordenamiento colocara los objetos (números o letras) en orden.

Un arreglo ordenado, es un arreglo que tiene un orden específicamente definido. Por ejemplo:

* [a, b, c, d] es un arreglo ordenado alfabéticamente
* [1, 2, 3, 4, 5] es un arreglo de números enteros ordenados ascendentemente

Tanto el ordenamiento por mezcla como el ordenamiento rápido emplean un paradigma algorítmico común que se basa en la recursividad. Este paradigma, **divide y vencerás**, separa un problema en subproblemas que se parecen al problema original, de manera recursiva resuelve los subproblemas y, por último, combina las soluciones de los subproblemas para resolver el problema original. Como divide y vencerás resuelve subproblemas de manera recursiva, cada subproblema debe ser más pequeño que el problema original, y debe haber un caso base para los subproblemas. Debes pensar que los algoritmos de divide y vencerás tienen tres partes:

1. **Divide** el problema en un número de subproblemas que son instancias más pequeñas del mismo problema.
2. **Vence** los subproblemas al resolverlos de manera recursiva. Si son los suficientemente pequeños, resuelve los subproblemas como casos base.
3. **Combina** las soluciones de los subproblemas en la solución para el problema original.

Puedes recordar fácilmente los pasos para un algoritmo de divide y vencerás como *divide, conquista, combina*. Aquí está cómo ver un paso, al suponer que cada paso de dividir crea dos subproblemas (aunque algunos algoritmos de divide y vencerás crean más de dos):

Si expandimos a dos pasos recursivos más, se ve así:

**InsertionSort:**

Aquí está otra forma de pensar acerca del ordenamiento. Imagina que estás jugando un juego de cartas. Tienes las cartas en tu mano y las cartas están ordenadas. Tomas exactamente una nueva carta del mazo. La tienes que colocar en el sitio correcto de manera que las cartas en tu mano sigan estando ordenadas. En el ordenamiento por selección, cada elemento que agregas al subarreglo ordenado es mayor o igual que los elementos que ya están en el subarreglo ordenado. Pero en nuestro ejemplo de las cartas, la nueva carta podría ser menor que algunas de las cartas que ya tienes en la mano, así que vas una por una, comparando la nueva carta con cada una de las que ya tienes en la mano, hasta encontrar el lugar donde debe ser colocada. Insertas la nueva carta en el sitio correcto y, una vez más, tienes en la mano cartas completamente ordenadas. Entonces tomas otra carta del mazo y repites el mismo procedimiento. Luego otra carta, y otra, y así sucesivamente, hasta terminar con el mazo.

Esta es la idea detrás del **ordenamiento por inserción**. Itera sobre las posiciones en el arreglo, comenzando con el índice 1.