**数据结构**

* **栈**：一种遵从先进后出 (LIFO) 原则的有序集合；新添加的或待删除的元素都保存在栈的末尾，称作栈顶，另一端为栈底。在栈里，新元素都靠近栈顶，旧元素都接近栈底。
* **队列**：与上相反，一种遵循先进先出 (FIFO / First In First Out) 原则的一组有序的项；队列在尾部添加新元素，并从头部移除元素。最新添加的元素必须排在队列的末尾。
* **链表**：存储有序的元素集合，但不同于数组，链表中的元素在内存中并不是连续放置的；每个元素由一个存储元素本身的节点和一个指向下一个元素的引用（指针/链接）组成。
* **集合**：由一组无序且唯一（即不能重复）的项组成；这个数据结构使用了与有限集合相同的数学概念，但应用在计算机科学的数据结构中。
* **字典**：以 [键，值] 对为数据形态的数据结构，其中键名用来查询特定元素，类似于 Javascript 中的Object。
* **散列**：根据关键码值（Key value）直接进行访问的数据结构；它通过把关键码值映射到表中一个位置来访问记录，以加快查找的速度；这个映射函数叫做散列函数，存放记录的数组叫做散列表。
* **树**：由 n（n>=1）个有限节点组成一个具有层次关系的集合；把它叫做“树”是因为它看起来像一棵倒挂的树，也就是说它是根朝上，而叶朝下的，基本呈一对多关系，树也可以看做是图的特殊形式。
* **图**：图是网络结构的抽象模型；图是一组由边连接的节点（顶点）；任何二元关系都可以用图来表示，常见的比如：道路图、关系图，呈多对多关系。

**算法**

**排序算法**

* **冒泡排序**：比较任何两个相邻的项，如果第一个比第二个大，则交换它们；元素项向上移动至正确的顺序，好似气泡上升至表面一般，因此得名。
* **选择排序**：每一次从待排序的数据元素中选出最小（或最大）的一个元素，存放在序列的起始位置，以此循环，直至排序完毕。
* **插入排序**：将一个数据插入到已经排好序的有序数据中，从而得到一个新的、个数加一的有序数据，此算法适用于少量数据的排序，时间复杂度为 O(n^2)。
* **归并排序**：将原始序列切分成较小的序列，只到每个小序列无法再切分，然后执行合并，即将小序列归并成大的序列，合并过程进行比较排序，只到最后只有一个排序完毕的大序列，时间复杂度为 O(n log n)。
* **快速排序**：通过一趟排序将要排序的数据分割成独立的两部分，其中一部分的所有数据都比另外一部分的所有数据都要小，然后再按此方法对这两部分数据分别进行上述递归排序，以此达到整个数据变成有序序列，时间复杂度为 O(n log n)。

**搜索算法**

* **顺序搜索**：让目标元素与列表中的每一个元素逐个比较，直到找出与给定元素相同的元素为止，缺点是效率低下。
* **二分搜索**：在一个有序列表，以中间值为基准拆分为两个子列表，拿目标元素与中间值作比较从而再在目标的子列表中递归此方法，直至找到目标元素。

**其他**

**贪心算法**：在对问题求解时，不考虑全局，总是做出局部最优解的方法。

**动态规划**：在对问题求解时，由以求出的局部最优解来推导全局最优解。

**复杂度概念**：一个方法在执行的整个生命周期，所需要占用的资源，主要包括：时间资源、空间资源。