在现代的高级语言中，我们用：

用变量来代表数据，信息 ，变量代表在一个(内存或外存)地址空间，在此空间内放置的数据则是变量的值，这个数据的表达的意义就叫信息。

用数据结构来代表数据间的关系（关系可以有并存，互斥…等等），在机器层，就相当于一系列地址空间的集合，也即一系列变量的群集，典型的数据结构比如 数组，链表等。

注意，数据结构有自提升阶的特性，自提升阶的意思为，一个概念可以由这个概念本身作为子元素来组成，形成一种更高层次（高阶）上的同意义概念。

比如，数据结构可以由数据结构组成，一个数组，如果它的每一个元素都是一个链表，那么，链表是一种一阶的数据结构，而这个数组则是一个二阶数据结构。

还需要注意的一点是，到目前为止，变量不具备自提升阶的属性，因为我们将变量的集合定义成了数据结构这种概念，不过，如果我们将数据结构看层是一种高阶变量的话（其实在程序语言中也是用一个变量来代表数据结构的），则变量也可以具备自提升阶的特性，当然，这时就不在需要数据结构这个概念了

处于简便的目的，在本文的以下部分我们使用具有自提升阶特性的变量概念，所以请注意，变量可以包含变量，不再仅仅代表一个单一的地址空间，而且也允许代表一群地址控件的集合，它也相当于数据结构这个概念。在需要使用仅代表单一地址空间的变量时，会特别说明。

算法，就相当于针对某种特定的变量做操作，得到一个结果，结果可以是变量（另外，结果会不会也可能是一种算法呢？）

在一种算法中可以调用其它算法来实现特定目标，所以我们也可以将算法组合或扩展来实现更强更多的功能。这个目标在现代程序语言中也已经实现了，典型的就是在一个方法A内调用另一个方法B并使用B得到的结果，或在一个方法S内封装了多个方法X,Y,Z, 方法S将X,Y,Z方法返回的结果组合成数据结构来进行。

在现在的程序设计语言中，我们总是需要用程序语言来描述出算法的实现方式，我们用程序语言写出的这种代码都有下面这些特点：

1. 使用哪种数据结构？
2. 如何对其进行操作。
3. 如何组合出结果。

现代程序语言都提供了函数库，功能库，类库，实际上就是把一些常用的，很一般化算法描述实现编制好了，当我们要完成我们的目的时，就选择我们需要的算法组合或扩展，构造成一个更高阶的算法，

关键元概念：对“过去，现在，未来”的概念理解。对状态”运动变化”的理解,所谓的过去现在未来其实就是以时间作为客体观察其状态的变化而已。

对相对性的理解，某些客体总是相对于另一些客体而言的，比如，“我”，”你”之类。

对“好”与“坏”，倾向于 的理解。（注意好坏一定都是相对的非绝对的）