数据（data）：对客观事物的符号表示，【在计算机科学中】指的是所有能输入到计算机中并被计算机程序处理的符号的总称；

数据元素（data element）：数据的基本单位，在计算机程序中通常作为一个整体进行考虑和处理（一个数据元素可以由若干个数据项组成）；

数据项（data item）：数据的不可分割的最小单元；

数据对象（data object）：性质相同的数据元素的集合，是数据的一个子集；

数据项—>数据元素—>数据对象—>数据

数据结构（data structure）：相互之间存在一种或者多种特定关系的数据元素的集合(数据元素相互之间的关系称为结构）；

四类基本结构：

①集合：结构中的数据元素除了同属于一个集合外别无其它关系；

②线性结构：结构中的数据元素之间存在一个对一个的关系；

③树形结构：结构中的数据元素之间存在一个对多个的关系；

④图状结构和网状结构：结构中的数据元素之间存在多个对多个的关系；

结构定义中“关系”描述的是数据元素之间的逻辑关系，因此又称为数据的逻辑结构；

物理结构（储存结构）：数据结构在计算机中的表示（映像）；

位（bit)：在计算机中表示信息的最小单位是二进制数的一位；

元素（节点）：一个由若干个位组合起来形成的一个位串表示一个数据元素（一个字长的位串表示整数，用8位二进制数表示一个字符串）；

元素或节点可以看成数据元素在计算机中的映像；

数据域：当数据元素为若干数据项组成时，位串中对应的各个数据项的子位串；

数据元素之间的关系在计算机中的表示方法：顺序映像和非顺序映像；

顺序映像的特点：借助元素在存储器中的相对位置来描述元素之间的逻辑关系；

非顺序映像的特点：借助指示元素存储地址的指针表示数据元素之间的逻辑关系；

数据类型（data type）：一个值的集合和定义在这个值集上的一组操作的总称【数据对象集，数据集相关联的操作集】；

（非结构的）原子类型：原子类型的值是不可分解的【C语言中的基本类型（整型、实型、字符型和枚举类型）、指针类型、空类型】；

结构类型：值是由若干成分按某种结构组成的，是可以分解的，并且它的成分可以是结构的也可以是非结构的；

1>在某种意义上，数据结构可以看成是“一组具有相同结构的值”，则结构类型可以看成由一种数据结构和定义在其上的一种操作组成；

2>实际上，在计算机中，数据类型的概念并非局限于高级语言中，每个处理器（计算机硬件系统【位、字节、字等原子类型，他们的操作通过计算机设计的 一套指令系统直接由电路系统完成】、操作系统、高级语言【高级程序语言提供的数据类型其操作需要通过编译器或解释器转化为底层（汇编语言或机器语 言）的数据类型来实现】、数据库等）都提供了一组原子类型或结构类型；

3>引入“数据类型”的目的，从硬件角度看，是作为解释计算机内存中信息含义的一种手段，而对使用数据类型的用户来说，实现了信息的隐蔽，即将一切 用户不必了解的细节都封装在类型中；

抽象数据类型（Abstract Data Type ADT}：指一个数学模型以及定义在该模型上的一组操作；

1>定义仅取决于它的一组逻辑特性，而与计算机内部如何表现和实现无关，无论内部结构如何变化，数学特性不变，不影响外部使用；

2>和数据类型实质上一个概念；抽象的意义在于数据类型的数学抽象；

3>范畴更广，不再局限于前述个处理器已定义并实现的数据类型（固有数据类型），还包括用户自己定义的数据类型；

4>分类（按其值的不同特性）：

原子类型；

结构类型:

固定聚合类型：其值由确定数目的成分按某种结构组成；

可变聚合类型：值的成分的数目不确定；

5>（D,S,P)，D数据对象，S是D上的关系集，P是对D的基本操作集；

多形数据类型（polymorphic data type）：其值成分不确定的数据类型；

不论其元素具有何种特性，元素之间的关系相同，基本操作也相同，从抽象数据类型角度看，具有相同的数学特性；

算法（algorithm）：对特定问题求解步骤的一种描述，它是指令的有限序列（有限的指令集），其中每一条指令表示一个或多个操作；

五个重要特性：

①有穷性：一个算法必须总是（对任何合法的输入值）在执行有穷步之后结束，且每一步都在有穷时间内完成；

②确定性：算法中的每一条指令必须有确切的含义、在任何条件下算法只有惟一的一条执行路径（对相同的输入只能得出相同的输出）；

③可行性：操作可以通过已实现的基本运算执行有限次来实现；

④输入：零个和多个的输入，这些输入取自某个特定的对象的集合；

⑤输出：一个或多个的输出，输出是同输入有着某些特定关系的量；

算法设计的要求：

①正确性（correctness)：程序对于精心选择的典型、苛刻而带有刁难性的几组输入数据能够得出满足规格说明要求的结果；

②可读性（readability)：注释不低于代码量的20%；

③健壮性（robustness)：当输入数据非法时，算法也能够适当地做出反应或进行处理，而不会产生莫名其妙的输出结果；

④效率与低存储量需求：

效率：算法执行的时间；

存储量需求：指算法执行过程中所需要的最大存储空间；

时间量度：基本操作重复执行的次数；

渐进时间复杂度：随时间规模n的增大，算法执行时间的增长率和f(n)的增长率相同，简称时间复杂度；

语句的频度：指的是该语句重复执行的次数；

空间复杂度：算法所需存储空间的量度；

程序执行时占用存储单元的长度；与输入的数据规模有关；

若额外空间相对于输入数据量来说是常数，则称次算法为原地工作；







