《01-认识数据结构与算法》

1 什么是数据结构与算法？

数据结构与算法是数据结构、算法二者的合成，本质上是二部分的内容，由于二者往往是相互依赖的关系，所有习惯上叫数据结构与算法。

1.1 数据结构

数据结构是相互之间存在一种或多种特定关系的数据元素的集合。细节上将就是数据之间根据某种关系而存储起来，所以从广义上讲，数据结构就是指一组数据的存储结构。

1.1.1 数据

数据是描述客观事物的符号，是计算机中可以操作的对象，是能被计算机识别，并输入给计算机处理的符号集合。数据不仅仅包括整型、实型等数值类型，还包括字符及声音、图像、视频等非数值类型。比如我们现在常用的搜索引擎，一般会有网页、MP3、图片、视频等分类。MP3 就是声音数据，图片当然是图像数据，视频就不用说了，而网页其实指的就是全部数据的搜索，包括最重要的数字和字符等文字数据。也就是说，我们这里说的数据，其实就是符号而且这些符号必须具备两个前提:可以输入到计算机中；能被计算机程序处理。对于整型、实型等数值类型，可以进行数值计算。对于字符数据类型，就需要进行非数值的处理。而声音、图像、视频等其实是可以通过编码的手段变成字符数据来处理的。

1.1.2 结构

结构，简单的理解就是关系，比如分子结构，就是说组成分子的原子之间的排列方式。严格点说，结构是指各个组成部分相互搭配和排列的方式。在现实世界中，不同数据元素之间不是独立的，而是存在特定的关系，这些关系称为结构。

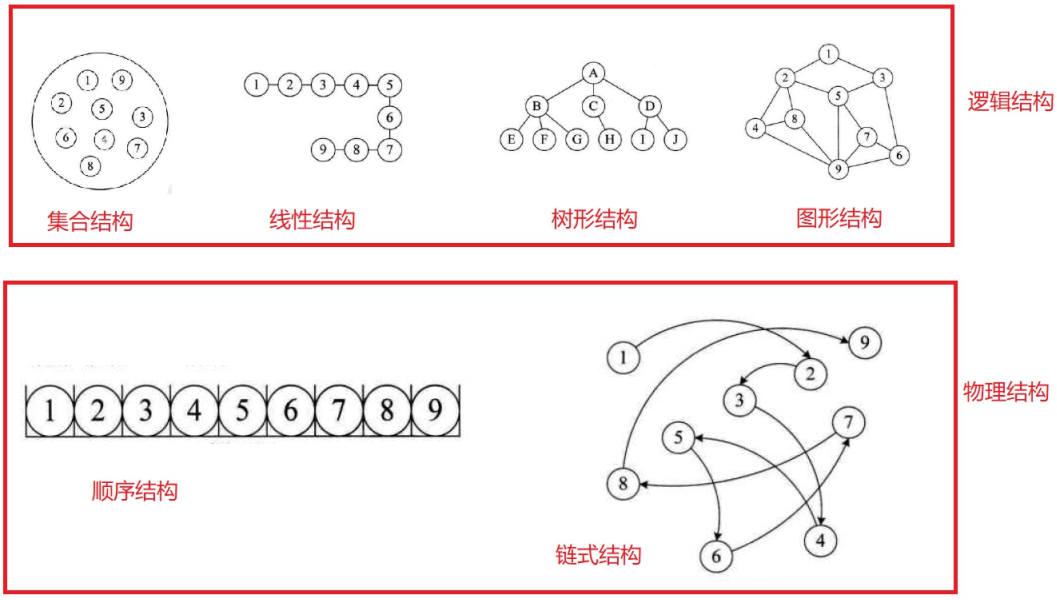
1.1.3 数据结构

数据结构是相互之间存在一种或多种特定关系的数据元素的集合。数据结构有哪些形式？逻辑结构形式和物理结构形式。

逻辑结构是指数据对象中数据元素之间的相互关系。逻辑结构有 4 种：集合结构、线性结构、树形结构、图形结构。集合结构：集合结构中的数据元素除了

同属于一个集合外，它们之间没有其他关系。线性结构：线性结构中的数据元素之间是一对一的关系。树形结构：数据元素之间存在一种一对多的层次关系。图形结构：元素之间是多对多的关系。

物理结构是指数据的逻辑结构在计算机中的存储形式。物理结构有两种：顺序存储结构和链式存储结构。物理结构是如何把数据元素存储到计算机的存储器中。存储器主要是针对内存而言的，像硬盘、软盘、光盘等外部存储器的数据组织通常用文件结构来描述。顺序存储结构:是把数据元素存放在地址连续的存储单元里，其数据间的逻辑关系和物理关系是一致的。链式存储结构:是把数据元素存放在任意的存储单元里，这组存储单元可以是连续的，也可以是不连续的。



1.2 算法

算法是解决特定问题求解步骤的描述，在计算机中表现为指令的有限序列，并且每条指令表示一个或多个操作。计算机解决问题，需要开发能够解决该问题的算法，实现相关的程序。

1.2.1 算法的性质

输入输出：算法具有零个或多个输入，至少有一个输出。

有穷性。算法在执行有限的步骤之后，自动结束而不会出现无限循环并且每个步骤在可接受的时间范围内完成。

可行性。算法的每一步都必须是可行的即每一步都能够通过有限次完成。

确定性。算法的每一步都有明确的含义，不会出现二义性。

1.2.2 算法的描述

算法的常见描述形式：采用自然语言描述。自然语言加数学符号描述。通用的计算模型描述。编程语言的形式描述。采用伪代码的形式描述。

1.2.3 算法的设计模式

枚举法。根据具体问题枚举出各种可能，从中选出有用信息或者问题的解。这种方法利用计算机的速度优势，在解决简单问题时十分有效。

贪心法。根据问题的信息尽可能做出部分的解，并基于部分解逐步扩充得到完整的解。在解决复杂问题时，这种做法未必能得到最好的解。

分治法。把复杂问题分解为相对简单的子问题，分别求解，最后通过组合起子问题的解的方式得到原问题的解。

回溯法。专指通过探索的方式求解。如果问题很复杂，没有清晰的求解路径，可能就需要分步骤进行，而每一步骤又可能有多种选择。在这种情况下，只能采用试探的方式，根据实际情况选择一个可能方向。当后面的求解步骤无法继续时，就需要退回到前面的步骤，另行选择求解路径，这种动作称为回溯。

动态规划法。在一些复杂情况下，问题求解很难直截了当地进行，因此需要在前面的步骤中积累信息，在后续步骤中根据已知信息，动态选择已知的最好求解路径，同时可能进一步积累信息。这种算法模式被称为动态规划。

分支界定法。分支限界法。可以看作搜索方法的-种改良形式。如果在搜索过程中可以得到一些信息，确定某些可能的选择实际上并不真正有用，就可以及早将其删除，以缩小可能的求解空间，加速问题求解过程。

1.2.4 算法的时间复杂度

算法的时间复杂度：在进行算法分析时，语句总的执行次数 T(n)是关于问题规模 n 的函数，进而分析 T(n)随 n 的变化情况并确定 T(n)的数量级。算法的时间复杂度，也就是算法的时间量度，记作:T(n）= O(f(n))。它表示随问题规模 n 的增大，算法执行时间的增长率和 f(n)的增长率相同，称作算法的渐近时间复杂度，简称为时间复杂度。其中 f(n）是问题规模 n 的某个数。这样用大写 O()来体

现算法时间复杂度的记法，我们称之为大 O 记法。

大 O 记法基本原则：用常数 1 取代运行时间中的所有加法常数。在修改后的运行次数函数中，只保留最高阶项。如果最高阶项存在且不是 1，则去掉与这个项相乘的常数。

执行次数函数	大 O 记法	阶数
1	$O(1)$	常数阶
$2n+1$	$O(n)$	线性阶
$3n^2+2n+1$	$O(n^2)$	平方阶
$5\log_2 n+20$	$O(\log n)$	对数阶
$2n+3n\log_2 n+3$	$O(n\log n)$	$n\log n$阶
$12n^3+2n^2+3n+1$	$O(n^3)$	立方阶
2^n	$O(2^n)$	指数阶

常用的时间复杂度所耗费的时间从小到大依次是：

$$O(1)<O(\log n)<O(n)<O(n\log n)<O(n^2)<O(n^3)<O(2^n)<O(n!)<O(n^n)$$

顺序结构的时间复杂度即单个顺序语句的执行是 O(1)常数阶。

```
1. print("hello world!")
```

分支结构中结构无论真假执行的次数都是恒定的，不包含在循环中的单纯分支结构也是 O(1)常数阶。

```
1. if a == 1:
2.     pass
```

线性阶的循环结构会复杂很多。要确定某个算法的阶次，常常需要确定某个特定语句或某个语句集运行的次数。因此，我们要分析算法的复杂度，关键就是要分析循环结构的运行情况。

```
1. # O(n)
2. for i in range(n):
3.     pass
```

```
# O(logn)
i = 1
while i<n:
    i = i*2
```

嵌套循环的阶也要根据实际问题决定。

```
# O(n**2)
for i in range(n):
for j in rang(n):
    pass
# O(n*m)
for i in range(n):
    for j in rang(m):
        pass
# 0(n**2)
for i in range(n):
    for j in rang(i,n):
        pass
```

最坏情况和平均情况

最坏情况运行时间是一种保证，那就是运行时间将不会再坏了。在应用中，这是一种最重要的需求，通常，除非特别指定，我们提到的运行时间都是最坏情况的运行间。平均运行时间是所有情况中最有意义的，因为它是期望的运行时间。也就是说，我们运行一段程序代码时，是希望看到平均运行时间的。可现实中，平均运行时间很难通过分析得到，一般都是通过运行一定数量的实验数据后估算出来的。对算法的分析，一种方法是计算所有情况的平均值，这种时间复杂度的计算方法称为平均时间复杂度。另一种方法是计算最坏情况下的时间复杂度，这种方法称为最坏时间复杂度。一般在没有特殊说明的情况下，都是指最坏时间复杂度。

1.2.5 算法的空间复杂度

算法的空间复杂度通过计算算法所需的存储空间实现，算法空间复杂度的计算公式记作:S(n)=o(f(n))，其中，n 为问题的规模，f(n)为语句关于 n 所占存储空间的函数。一般情况下，一个程序在机器上执行时，除了需要存储程序本身的指令、常数、变量和输入数据外，还需要存储对数据操作的存储单元。若输入数据所占空间只取决于问题本身，和算法无关，这样只需要分析该算法在实现时所需的辅助单元即可。若算法执行时所需的辅助空间相对于输入数据量而言是个常

数，则称此算法为原地工作，空间复杂度为 O(1)。

通常，我们都使用“时间复杂度”来指运行时间的需求，使用“空间复杂度”指空间需求。当不用限定词地使用“复杂度”时，通常都是指时间复杂度。

1.3 数据结构与算法的关系

数据结构和算法是相辅相成的。数据结构是为算法服务的，算法要作用在特定的数据结构之上。因此，无法孤立数据结构来讲算法，也无法孤立算法来讲数据结构。数据结构与算法组成程序。

2 为什么要学习数据结构与算法？

数据结构与算法是程序及编程功底的基础，任何一个想要成为高级编程者的人都不能对数据结构与算法视而不见，尽管你可以在需要时随手可以拿到想要的 API。数据结构与算法对代码的优化起到至关重要的作用，当处理大数据时体现的尤其明显。

2.1 优化代码

学习数据结构与算法，可以帮助编程者写出优化后的高性能代码。

2.2 提升编程能力

学习数据结构与算法，可以帮助编程者从基层建起大厦，如果不懂数据结构与算法，个人的能力容易遇到瓶颈，容易被行业淘汰。

3 如何学习数据结构与算法？

3.1 选择参考书籍

经典的书籍有《算法导论》、《大话数据结构》、《算法图解》、《数据结构与算法 python 语言实现》等。

3.2 理论与编程结合

学习理论之后要刷题，敲代码。