数据结构与算法（Python版）

为什么研究数据结构与算法

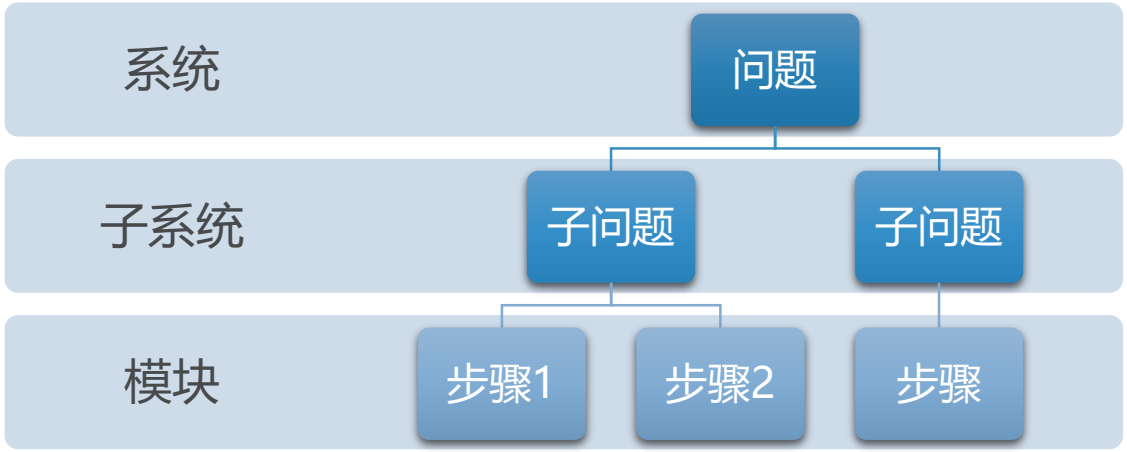
陈斌 北京大学 gischen@pku.edu.cn

清晰高效地表达算法

- 为了控制问题和问题解决过程的复杂度，利用抽象来保持问题的“**整体感**”

  而不会陷入到过多的细节中去

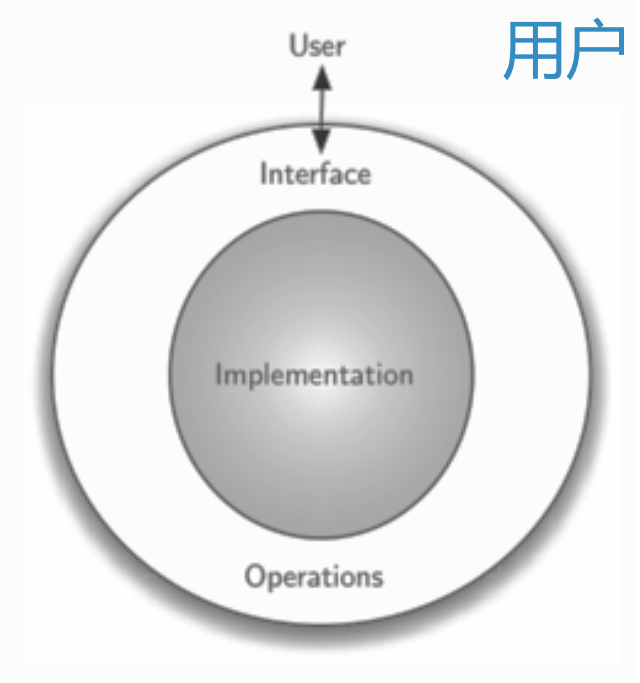
- 这要求对现实问题进行建模的时候，对算法所要处理的**数据**，也要保持与问题本身的一致性，不要有太多与问题无关的细节



数据抽象：ADT抽象数据类型

- “过程抽象”启发我们进行“**数据抽象**”
- 相对于程序设计语言中基本数据类型，抽象数据类型（ADT:Abstract Data Type）

  ADT是对数据进行处理的一种逻辑描述，并不涉及如何实现这些处理



用户

接口

实现

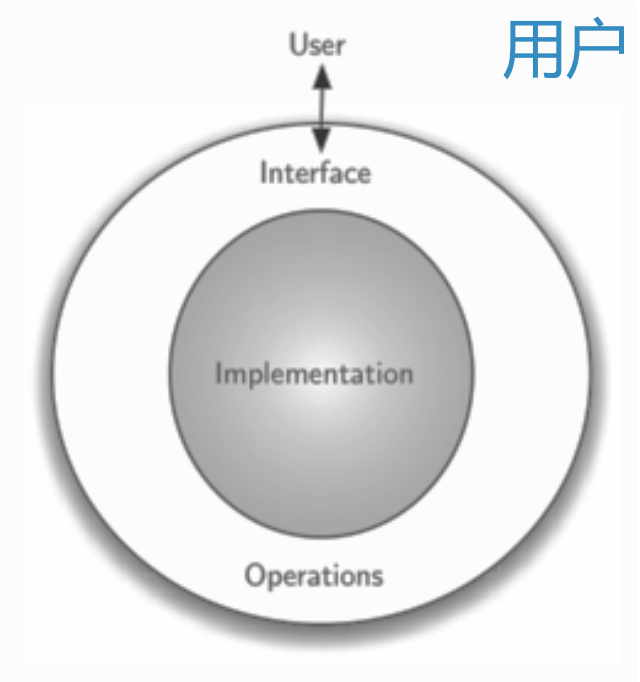
操作

数据抽象：ADT抽象数据类型

- 抽象数据类型ADT建立了一种对数据的 “**封装**” Encapsulation”

  封装技术将可能的处理实现细节隐蔽起来

  能有效控制算法的复杂度



用户

接口

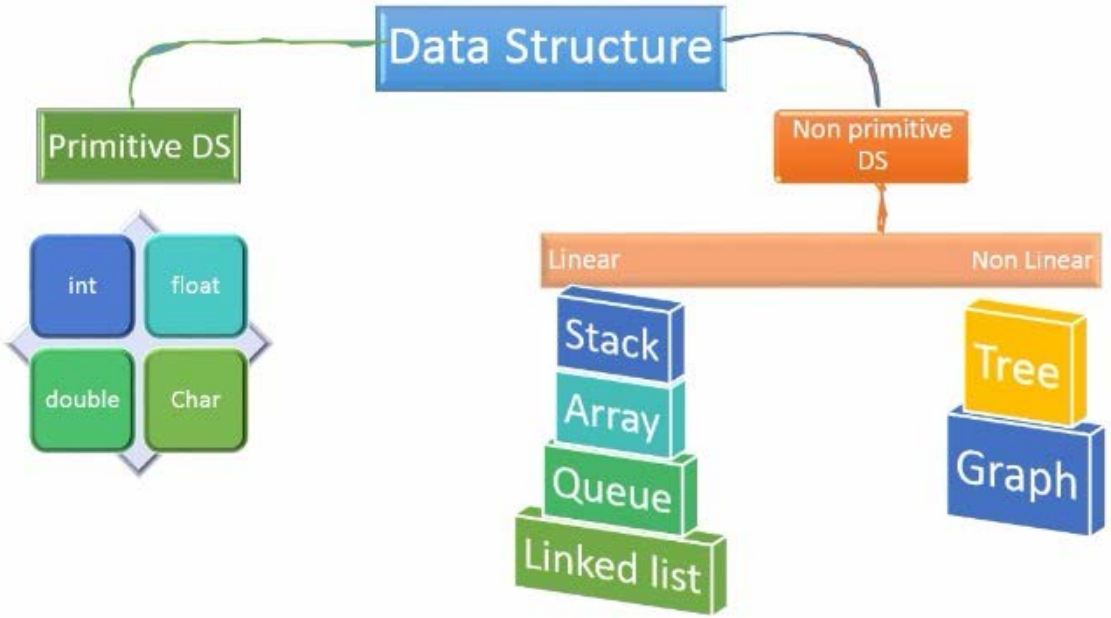
实现

操作

数据结构是对ADT的具体实现

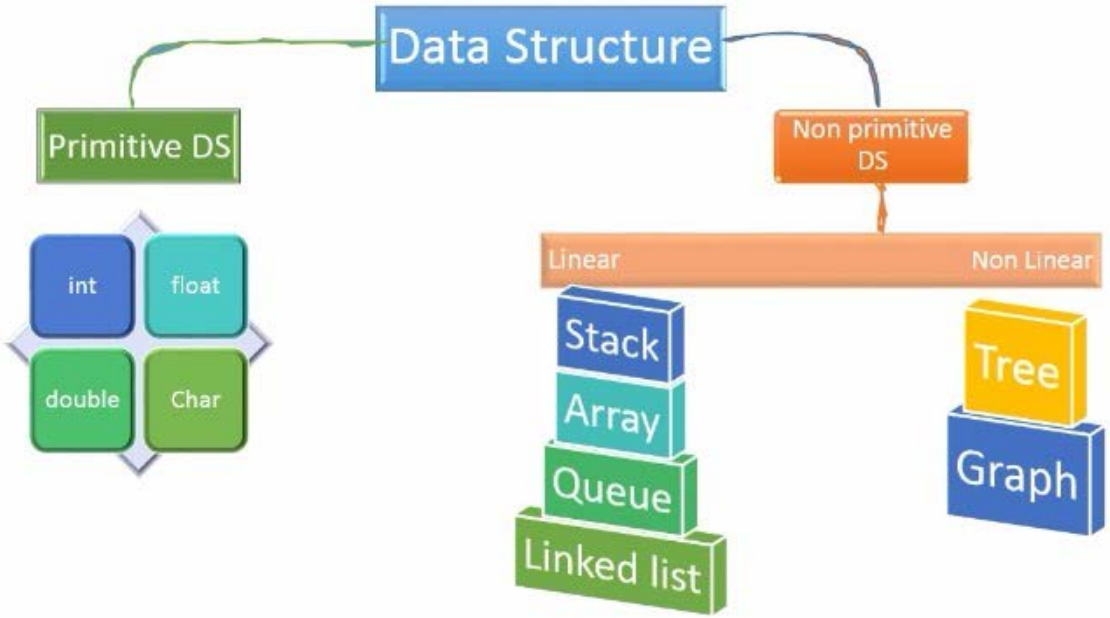
- 同一ADT可以采用不同的数据结构来实现
- 采用程序设计语言的**控制结构**和**基本数据类型**来实现ADT所提供的**逻辑接口**

  属于ADT的“物理”层次



ADT实现：数据结构Data Structure

- 对数据实现“逻辑”层次和“物理”层次的**分离**，可以定义复杂的数据模型来解决问题，而不需要立即考虑此模型如何实现



接口的两端：抽象与实现

- 如电动车与汽油车
  - 底层动力、能源都不同
  - 但开车的操作接口（方向盘、油门、刹车、档位）基本都是相同的
  - 司机无需重新考驾照，而车厂可以持续改进实现技术

接口的两端：抽象与实现

- 由于对抽象数据类型可以有多种实现方案
- 独立于实现的数据模型

  让底层开发程序员专注于实现和优化数据处理，又不改变数据的使用接口

  让用户专注于用数据接口来进行问题的解决，而无需考虑如何具体实现这些接口

- 通过层层抽象，降低问题解决过程的复杂度

为什么要研究和学习算法

- **首先，学习各种不同问题的解决方案**

  有助于我们在面对未知问题的时候，能够根据类似问题的解决方案来更好解决

- **其次，各种算法通常有较大差异**

  我们可以通过算法分析技术来评判算法本身特性

  而不仅仅根据实现算法的程序在特定机器和特定数据上运行的表现来评判它

  即使同一个程序，在不同的运行环境和输入数据的情况下，其表现的差异可能也会很大

为什么要研究和学习算法

- **在某些情况下，当我们碰到棘手的难题**
  - 得能区分这种问题是根本不存在算法
  - 还是能找到算法，但需要耗费大量的资源
- **某些问题解决需要一些折衷的处理方式**
  - 我们需要学会在不同算法之间进行选择，以适合当前条件的要求