**草稿（中文）**

**（未来需转成英文）**

参考书籍：数据结构与算法Python语言实现（Data structures and algorithms in Python）Michael T Goodrich， Roberto Tamassia，Michael H Goldwasser著

**Parsons problem在数据结构与算法课程的应用重点：**

作为直接上手写代码的预备过程，衔接过程，降低难度，帮助学生一步步解决问题

横向比较区分相似的不同算法或数据结构

培养良好的思维习惯，比如设计递归算法该从哪里入手

**介绍类编程课和数据结构与算法课程的区别（概括）：**

介绍类编程课：

简单，直接，具体，相似概念较少，需要比较的部分较小，知识点与知识点之间相对独立，几乎不涉及算法分析（时间复杂度），没有面向对象编程这个概念，几乎不需要用到伪代码

数据结构与算法课程：

抽象，普适性，不再针对单一情况，较多的相似概念，需要比较区分的部分也比较多，知识点环环相扣，涉及算法分析（时间复杂度），面向对象编程，伪代码

**汇总数据结构与算法知识点草稿：**

面向对象编程类，算法分析类，递归类，数据结构类，算法类

面向对象编程的设计原则：第二章P38

模块化，抽象化，封装

其中重点抽象化，从一个复杂的系统中提炼出最基础的部分。通常，描述系统的各个部分涉及给这些部分命名和解释它们的功能。抽象数据类型：Abstract Data Types，ADT。**题干中使用ADT来引导学生涉及数据结构。（原先无面向对象编程，不涉及）**

伪代码：第二章P41

设计实现前的中间步骤，专门为人准备的方法来描述算法。不是计算机程序，但是比平常文章更加结构化。伪代码是自然语言和高级编程结构的混合，用于描述隐藏在数据结构和算法实现之后的主要编程思想。交流复杂的思想而不是低层具体细节的实现。**预备步骤，在复杂问题中可以让学生先排序伪代码，再进一步操作真实的代码，便于理解（原先无伪代码）；用伪代码显示数据结构的一些共性？比如队列和栈？**

类：第二章P45

面向对象编程的问题，class，method，attribute。方法与方法间位置即使互换对整体程序也无大影响，也不会造成程序失败（因为它相当于是个一个部件工具集合箱，按需调用，不需要考虑工具的摆放顺序，能拿出来就行）（但是介绍类编程课程学过method。。。），并且方法内部的部分比较简短，有些部分甚至可以通过方法名猜出，**因此大大地削弱了Parsons Problem在此的实用性（Parsons Problem以排序见长）。可以通过ADT对应的方式强制限定方法之间的顺序，让学生知道方法的意思、ADT与编程的对应关系（不过可能不是很有用，毕竟ADT有名字，当然也可以去掉啦，只保留方法功能），看似是个排序题实则是个选择题。。。（后面详述）；另外一个思路是侧重于横向对比，比如有哪些结构是栈和队列共同拥有的（len，is\_empty），哪些是栈或者队列单独的（FILO，FIFO）**

Text

Description automatically generated

运算符重载：第二章49页

**介绍类课程不会涉及（介绍类课程只会涉及怎么用运算符或内置方法，不会涉及到怎么个性化构建），这些运算符重载的工作在不同的数据结构中可能指向一些共同的结构。**

迭代器：第二章51页

**介绍类课程没有的，不知道怎么用，感觉好像只能简单地应用以前的模式**

Text

Description automatically generated

Text, letter

Description automatically generated

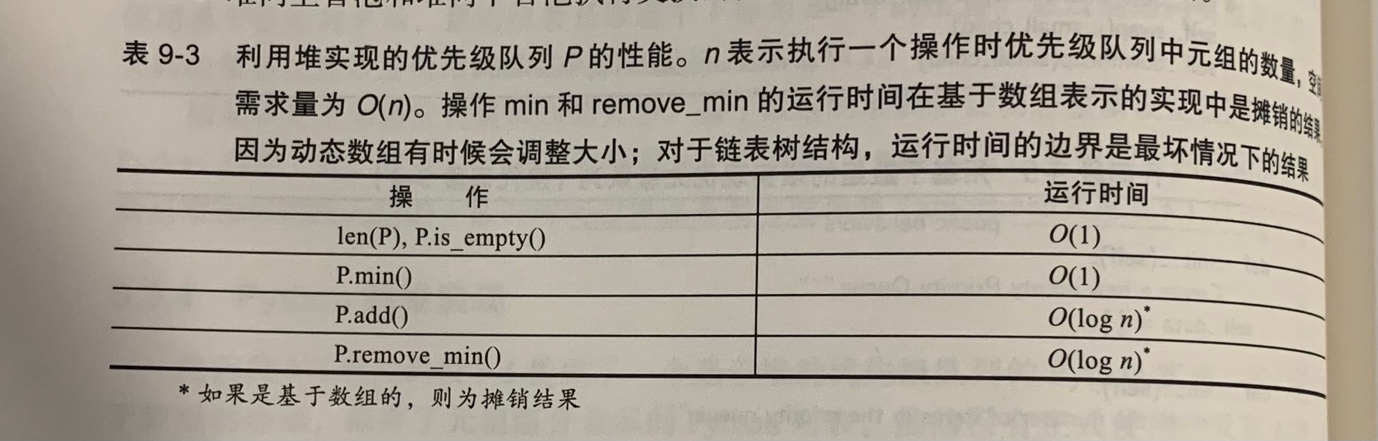
继承：第二章53页

**面向对象编程，介绍类课程中没有的，用作步骤中的一环？用作比较共同点不同点？**

算法分析：7种本书中常用的函数：第三章75页

常数函数，对数函数，线性函数，nlogn函数，二次函数，三次函数和其他多项式，指数函数

**介绍类课程中没有的（因为大部分就是常数函数和二次、三次函数，且区分的指标很简单，如果有嵌套循环就是二次，三次函数，在介绍类课程中没有太大意义），但在数据结构与算法中是重头戏，可以有以下两种使用方法，在多种答案的题型中选出对应的算法分析结果（拖拽对应的大O块到答题区或题干区），或者在同一个数据结构中选出不同方法的对应算法分析结果（拖拽大O块到答题区对应的方法前）（比如树的搜索操作和增删操作时间复杂度是不同的）以体会不同数据结构的特点（比如树搜索简便快捷，但增删困难慢）。（待补充图）；最后提出根据大O比较快慢的问题？**

****

证明技术：第三章89页

反证法归纳法

**这个介绍类的确实没有，但是太偏数学了，完全不知道该怎么融入进去。。。**

递归：第四章101页，115页

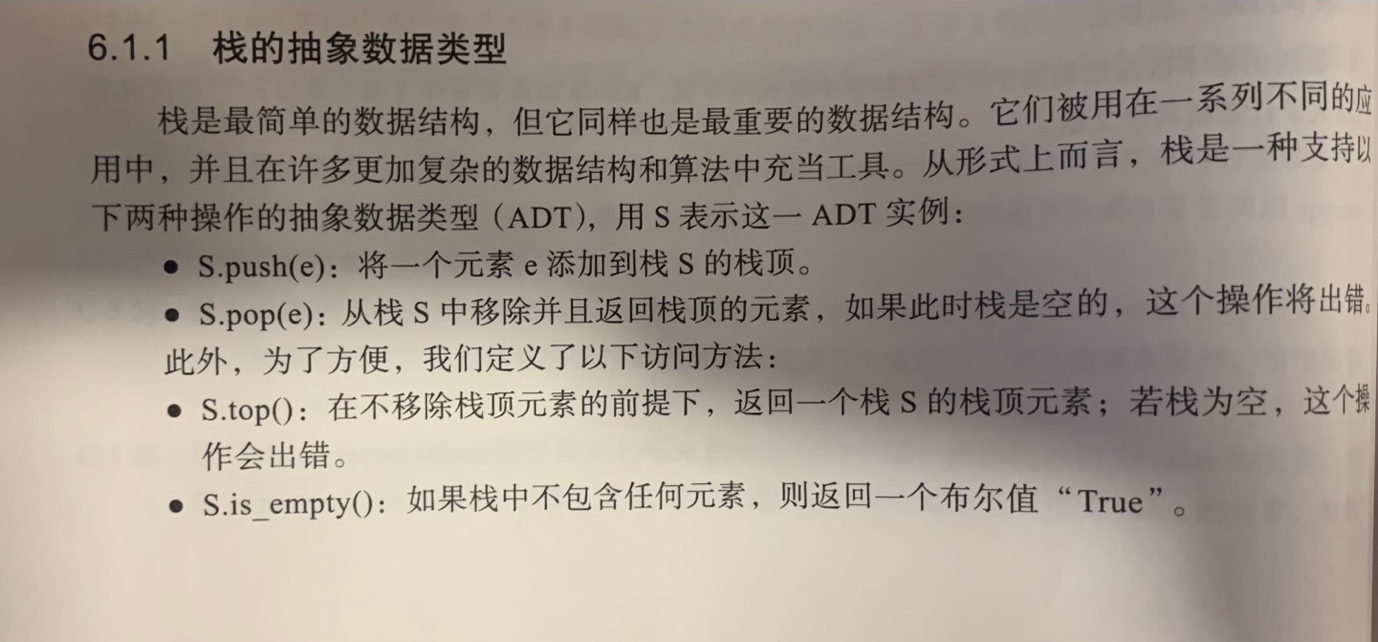
**抽象，普适性，不再针对单一情况，一个代码综合所有情况的代言人。简洁利用重复结构呈现诸多问题通过使算法描述以递归的方式利用重复结构（第四章115页）。即使完整给出所有代码，学生也很难看懂代码到底在干什么。因此，原先Parsons problem能透过代码阅读提供给学生的帮助大大减少，跟直接上手写难度差距不大。根据设计递归算法的方法（第四章114页）（定义与原始问题有相同的总体结构的子问题，再总结出普适性的递归），重新定义pre-scaffolded（需要与之前的pre-scaffolded作对比，需要补充新的材料），将一个递归问题分解为若干子步骤（而不是像之前一样分成一些for，if结构），层层递进。比如举个例子，对于一个递归来说，先让学生排序伪代码，知晓大概步骤，再让学生排序递归中一个具体的只针对于某一个轮次的程序（子问题），再让学生确定递归截止点及下一轮递归的参数，最后再让学生排序普适的递归方式，每一个步骤都能显示之前（若有）的步骤结果作为提示；另外一种方式就是选取递归的关键点，比如恰当的递归截止点和下一轮递归参数，插入到已排序的代码中的相应位置（不知道能不能把这个融入到上一个问题中，进一步降低难度）。需要教导学生消除尾递归，distractor（第四章115页）**

递归跟踪(第四章117页)

应该要用图（表示树状结构的变化）或者文字结果表示，很难用代码表示（失去了递归的意义），不是fyp的核心问题，后面再解决

摊销（amortization）第五章127页

栈，抽象数据类型：第六章148页



Text

Description automatically generated with medium confidence

基于list实现的栈：第6章150页

可能下面这个可以作为一个预步骤，或者与链表实现的栈表作对比

Letter

Description automatically generated

代码实现

Text

Description automatically generated

Text

Description automatically generated

第6章课件



应用栈的程序：第6章153页



看不出来和介绍类课程parsons problem有什么区别。。。

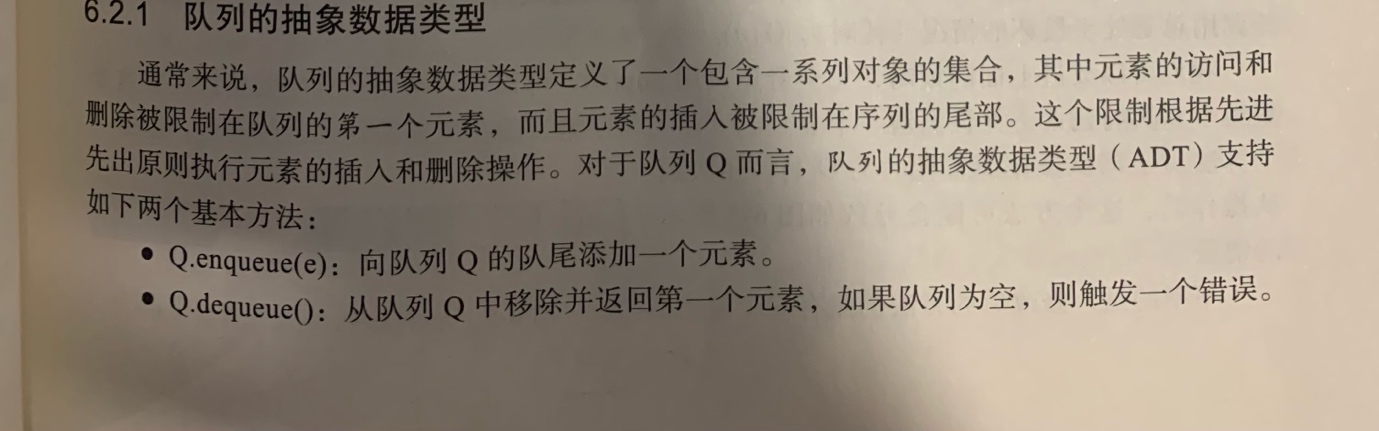
Ex6Q3

Text

Description automatically generated with medium confidence

pre-scaffolded

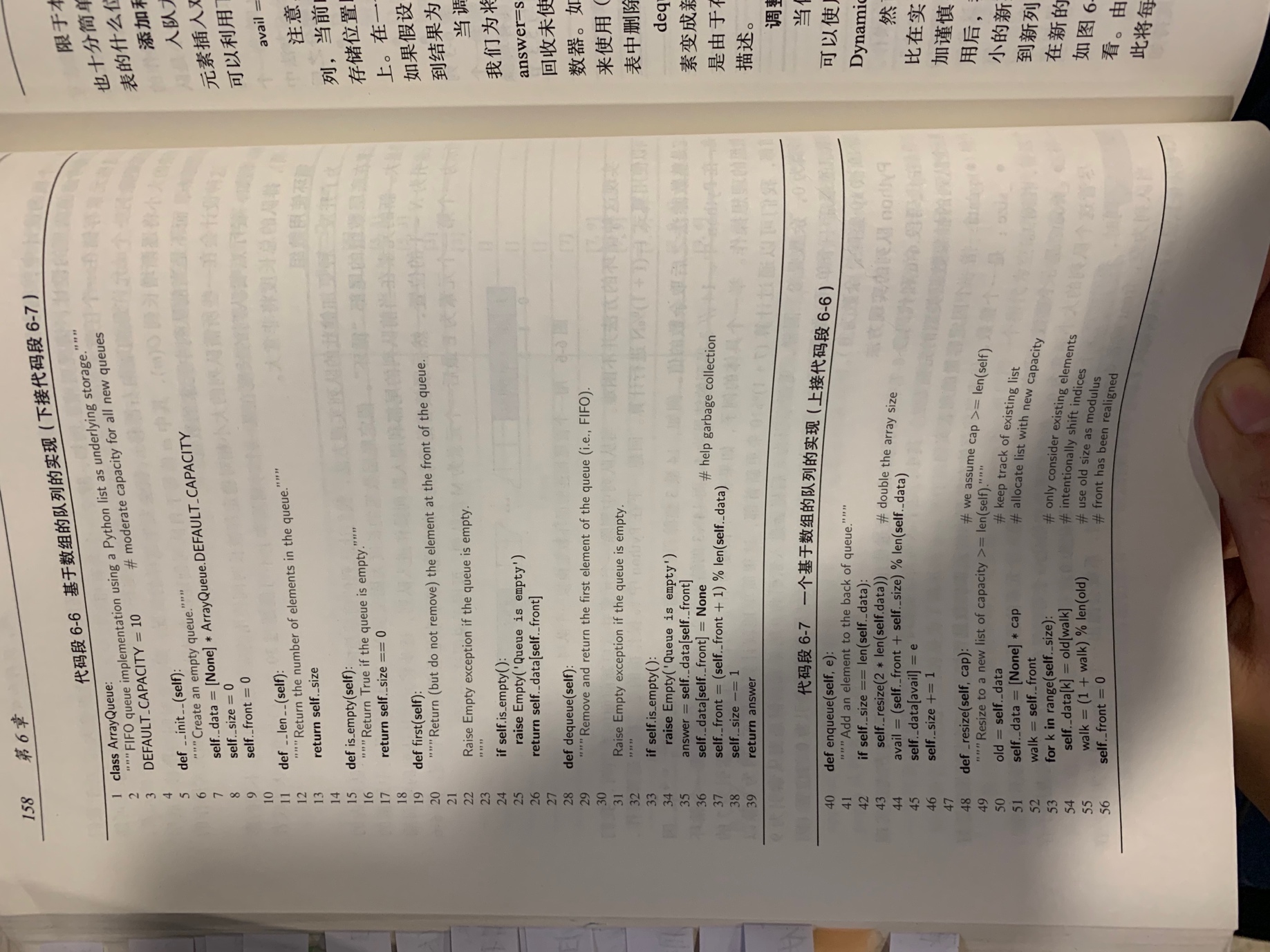
队列的抽象数据类型：第6张155页





基于数组的队列实现：第6章158页

和基于数组的栈实现差得太多了。。很难做横向对比。。。因为循环使用数组。。。



Text

Description automatically generated

Text

Description automatically generated

应用队列的题：ex6\_q5

A picture containing text

Description automatically generated

Pre-scaffold

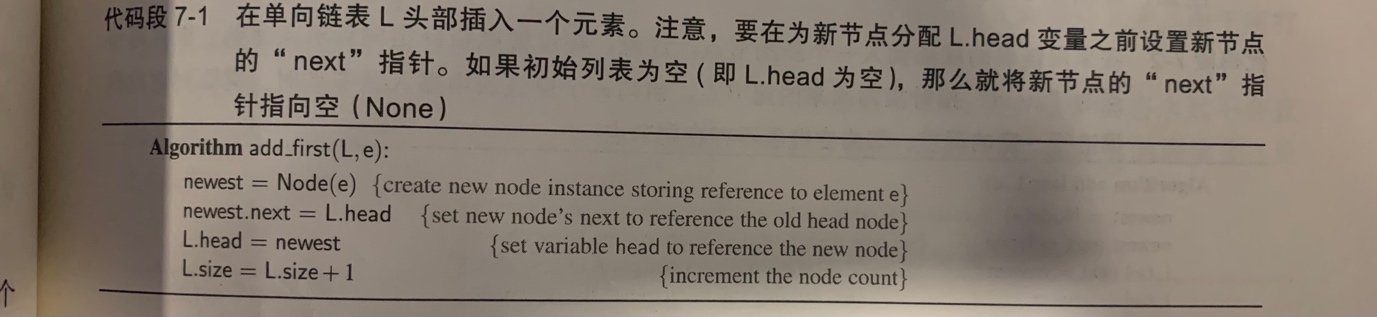
双端队列抽象数据类型：第6章161页

Text, letter

Description automatically generated

链表：第7章167页

**给出一组图（展示变化）或者伪代码作为题干来提示**（**原先没有这种数据结构图或者伪代码题干）**



Diagram

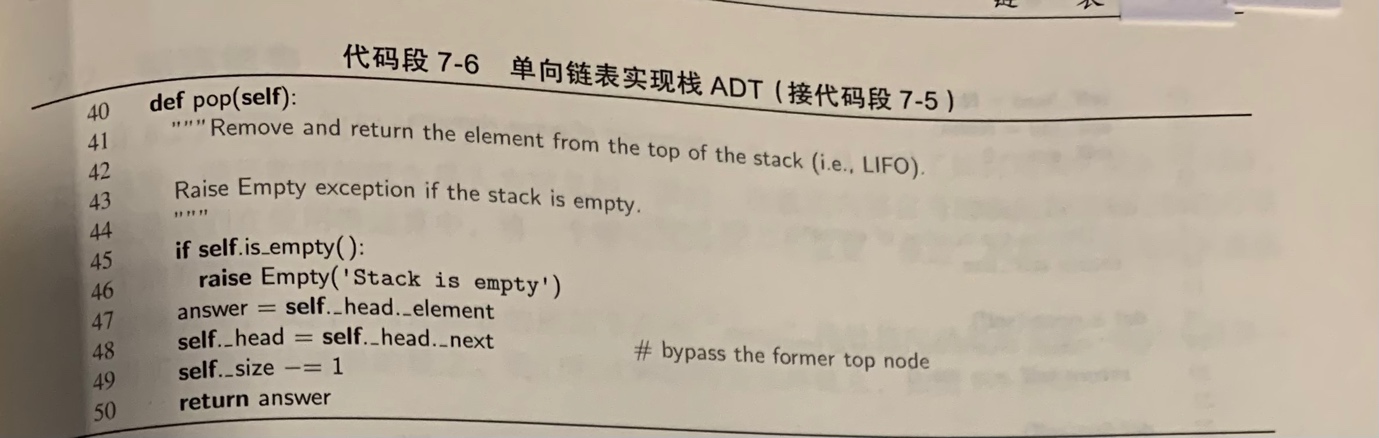
Description automatically generated

单向链表实现栈：第7章170页

**伪代码之后选择实现方式，在以某种方式实现之后，可以再替换另外一种。可以在实现列表实现栈之后让学生替换某些部分（或者翻面）（摘掉原先的，替换重新排序新的部分，但维持一些结构不变）从而转换成单向链表实现栈，让学生巩固栈的特征又横向比较不同实现方法的不同之处**

Text

Description automatically generated



单向列表实现队列：第7章

**这个不仅可以和数组实现队列对比还可以和单向列表实现栈对比，采取同样的方式，替换式**

Text, letter

Description automatically generated

Text, letter

Description automatically generated

循环列表实现循环队列类：第7章175页

**同样可以加入对比，可能和数组构建队列更像**

双向链表及双端队列：第7章179页

同

树：第8章199页

**分为好几个步骤，每一个步骤都会显示之前的步骤**：**树抽象数据类型（199页）->树抽象基类（201页）->在理解数的基础上理解二叉树的某些特性->二叉树抽象基类（205页）->链式二叉树python实现（208页）（pre-scaffold结构，每个方法一个块）**

**伪代码之后选择实现方式**

树的遍历算法：（第8章215页）

先序遍历，后序遍历，广度优先遍历，中序遍历，题目给出图，由图给伪代码排序，然后再是具体实现

Diagram

Description automatically generated

堆：第9章243页

**从二叉树到堆、二叉搜索树，AVL。。，不同特性，可以作为不同分支，也可在抽象类或具体实现方式中混在一起，也可以像是栈和队列一样横向比较**

**在堆中增加、删除一个元组：第9章243页**

**递归遍历，以图的形式排序更好，次要目标**

哈希表：第10章269页

**前面介绍类课程没有的东西，但我也不太清楚该怎么弄**

算法类：第12章351页

**快速排序，归并排序，插入排序，堆排序，由步骤图开始，不同方法混在一起，选出正确的方法，辅以伪代码？**

**深搜广搜：第14章413页，同**

**Dijkstra算法：第14章428页**

**数据结构与算法课程题型总结，之前用于介绍类编程课的Parsons problem的局限性，新的思路：**