数据结构实质：分析问题，提取操作对象，找出操作对象之间的关系，用数学语言描述=>数据结构

是一门研究非数值计算的程序设计中计算机的操作对象以及他们之间的关系和操作的学科

人机对弈问题：树形结构，非线性关系，比如文件系统

图问题：网状结构

综上所述：非数值计算的程序设计问题，如表，树，图之类的具有逻辑关系的数据

基本概念：

数据：是能被输入计算机且能被计算机处理的各种符号的集合

信息的载体

是对客观事物符号化的表示

能被计算机识别，储存和加工

包括：

数值型的数据：整数，实数等

非数值型的数据：文字，图像，图形，声音等

数据元素：是数据的一个基本单位，在计算机程序中通常作为一个整体进行考虑和处理，也被简称为元素，记录，结点或者顶点

数据元素又可以由若干个数据项组成

数据项：构成数据元素的不可分割的最小单位

数据对象：是性质相同的数据元素的集合，是数据的一个子集

那数据元素和数据对象又有什么区别呢？一个代表行，一个代表列

数据结构：

数据元素不是孤立存在的，他们之间存在着某种关系，数据元素相互之间的关系称为结构

是指相互之间存在一种或多种特定关系的数据元素集合

或者说，数据结构是带结构的数据元素的集合

包括以下三个方面的内容

1. 数据元素之间的逻辑关系，也称为逻辑结构
2. 数据元素及其关系在计算机内存中的表示（又称为映像），称为数据的物理结构或数据的存储结构
3. 数据的运算和实现，即对数据元素可以施加的操作以及这些操作在相应的存储结构上的实现。

逻辑结构

描述数据元素之间的逻辑关系

与数据的储存无关，独立于计算机

是从具体问题抽象出来的数学模型

物理结构（存储结构）

数据元素及其关系在计算机ROM中的结构（存储结构）

是数据结构在计算机中的表示

逻辑结构与存储结构的关系

存储结构是逻辑关系的映像与元素本身的映像

逻辑结构是数据结构的抽象，存储结构数据结构的实现

两者综合起来建立二零数据元素之间的结构关系

逻辑结构的种类

划分方法一

线性结构：有且仅有一个开始和一个终端结点，并且所有结点都最多只有一个直接前趋和一个直接后继，例如：线性表，栈，队列，串

非线性结构：一个结点可能由多个直接前趋和直接后续，例如：树（人机对弈），图（最短路径）

划分方法2

集合结构：结构中的数据元素之间除了同属于一个集合的关系外，无任何其他关系。

线性结构：结构中的数据元素之间存在着一对一的线性关系

树形结构：结构中的数据元素之间存在着一对多的层次关系

图状结构：结构中的数据元素之间存在着多对多的任意关系

存储结构

四种基本的存储结构：

**顺序存储结构：用内存中一片连续存储单元依次存储数据元素，数据元素之间的逻辑关系由元素的存储位置来表示。C语言中用数组来实现顺序存储机构**

**链式存储结构：非连续，元素之间的逻辑关系用指针来表示。**

索引存储结构：在储存节点信息的同时还建立附加的索引表（目录，Index）

散列存储结构：根据结点的关键字直接计算出该结点的存储地址

数据类型和抽象数据类型

使用高级程序设计语言编写程序时，必须对程序中出现的每个变量，常量或表达式，明确说明他们所属的数据类型。对于一些基本的数据结构可以用数据类型来实现。比如数组，字符串

而对于另一些常用的数据结构，如栈，队列，树，图等，不能直接用数据类型来表示

抽象数据类型：是指一个数学模型以及定义在此数学模型上的一组操作，还包括定义在数据模型上的一组抽象运算

不考虑计算机内的具体存储结构与运算的具体实现算法

形式定义：

抽象数据类型可用D，S，P三元组表示

D是数据对象；S是D上的关系集；P是对D的基本操作集

一个抽象数据类型的定义格式如下：

ADT 抽象数据类型名{

数据对象:<数据对象的定义>

数据关系:<数据关系的定义>

基本操作:<基本操作的定义>

} ADT抽象数据类型名

其中数据对象吗，数据关系的定义用伪代码描述

赋值参数只为操作提供输入值

引用参数以&打头，除了壳提供输入值外，还将返回操作结果

抽象数据类型定义举例：Circle定义

ADT 抽象数据类型名{

Data

数据对象的定义

数据元素之间逻辑关系的定义

Operator

操作1

初始条件

操作结果描述

操作2

……

操作N

……

} ADT抽象数据类型名

ADT Circle{

数据对象：D={r.x.y|均为实数}

数据关系：R={<r,x,y>|r半径，xy圆心坐标}

Circle(&C,r,x,y)

操作结果：构造一个圆

Double Area（C）

初始条件：圆已经存在

操作结果：计算面积

……

} ADT Circle



图片包含 日程表

描述已自动生成

抽象数据类型的表示与实现

抽象数据类型可以通过固有的数据类型（如整形，实型，字符型等来表示和实现

就是利用处理器中已存在的数据类型来说明新的结构，用已经实现的操作来组合新的操作。

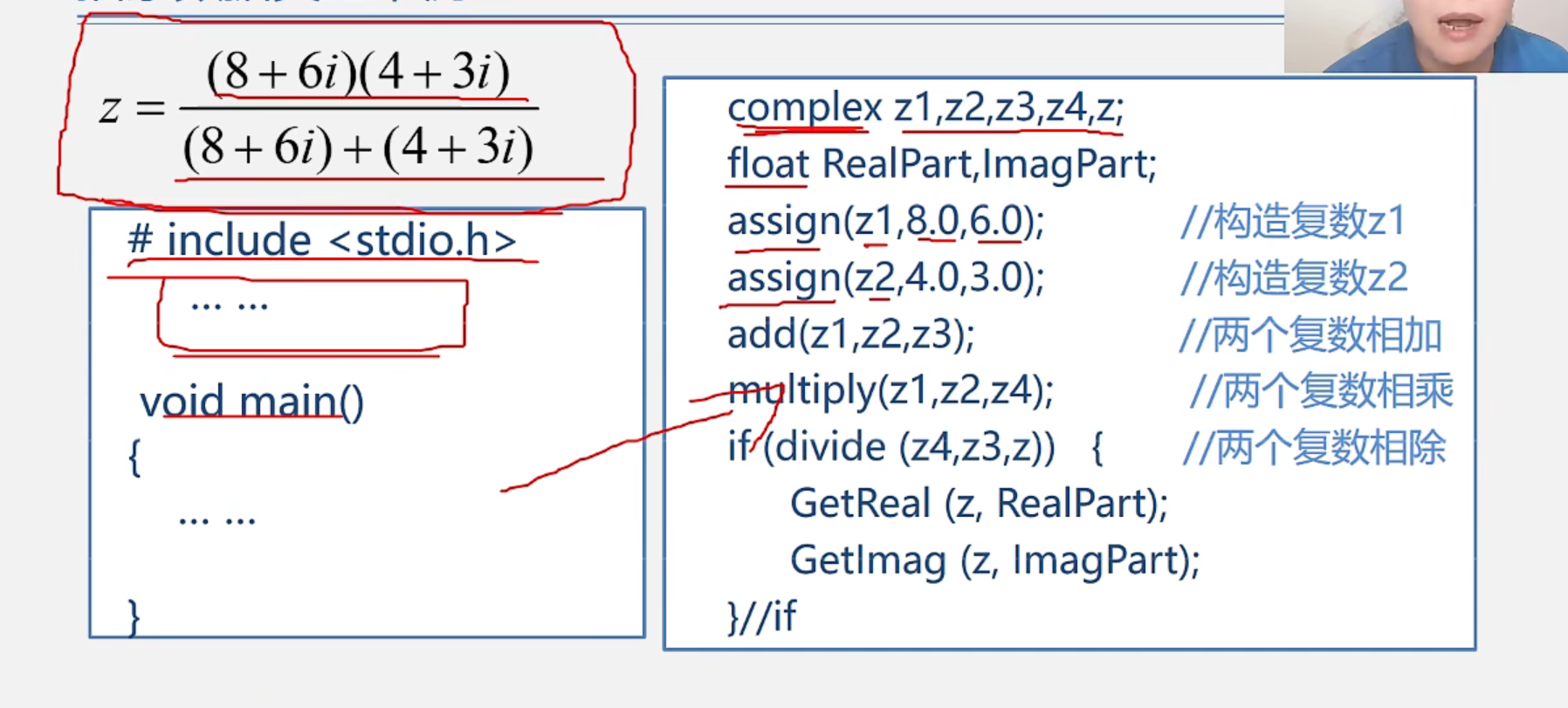
类C语言去描述

文本

描述已自动生成

文本

中度可信度描述已自动生成



算法

对特定问题求解的方法或步骤的一种描述，他是指令的有限序列。其中每个指令表示一个或多个操作=>解决问题的方法或步骤

算法的描述：自然语言：英文，中文

流程图：传统流传图（框图），NS流程图

PCode：类C语言=>这门课的描述办法

程序代码

算法和程序

算法是解决问题的一种方法或一个过程，考虑如何将输入转换成输出，一个问题可以有多种算法

程序=数据结构+算法

算法特性：一个算法必须具备以下五个重要特性

有穷性

确定性

可行性

输入性（可选）

输出，一个或多个输出

算法设计要求

正确性

可读性：给我人读的

健壮性：输入非法数据的时候不会死机或者什么的（又叫鲁棒性）

高效性：尽可能花费较少的时间和较少的RAM

算法效率

1. 时间效率
   1. 事后统计
      1. 将算法实现，测算其时间和空间开销
      2. 缺点：运行环境会对其有很大影响
   2. 事前分析
      1. 一个算法的运行时间是指一个算法在计算机上运行锁耗费的时间大致可以等于计算机执行一种简答的操作所需的时间与算法中进行的简单操作次数乘积
      2. 算法运行时间=一个简单操作所需的时间x简单操作次数
      3. 也就是算法中每条语句的执行时间之和
      4. 文本

         低可信度描述已自动生成
      5. 比较语句频度之和

For(i=1;1<=n;i++) //这段代码会执行n次，但由于有一次判断所以n+1

For(j=1;j<=n;j++){ //这段代码是最外层的结构体，会执行n次，而本身也会执行n次，加上判断就是n+1次，n(n+1)

… //n\*n次

For(k=0;k<n;k++) //n\*n\*(n+1)次，原理同第二层循环

… //n\*n\*n次

}}}

每句频度算出来了，那么频度之和就是时间小号T(n)=2n3+3n2+2n+1

为了便于比较不同算法的时间效率，我们仅比较他们的数量级

例如：两个不同的算法：Tn=10n2 Tn=5n3 如果比较数量级的话，那3大于2，自然第一个好

文本, 白板

描述已自动生成

一般来说，不必考虑所有操作的执行次数，只考虑算法中的基本操作执行次数，他就是问题规模n的某个函数，用T(n)表示，就是考虑基本操作执行次数。低情商：最高次数项（

文本

描述已自动生成

那基本操作语句怎么找？

算法中重复执行次数和算法的执行时间

对算法运行时间贡献最大

低情商：执行次数最多的语句

文本, 白板

描述已自动生成backup

图表, 散点图

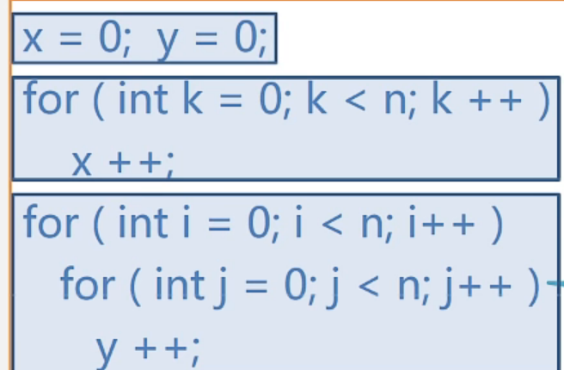
描述已自动生成

简答理解就是项数一级一级减下去的就是Tn=on

1.找出基本语句

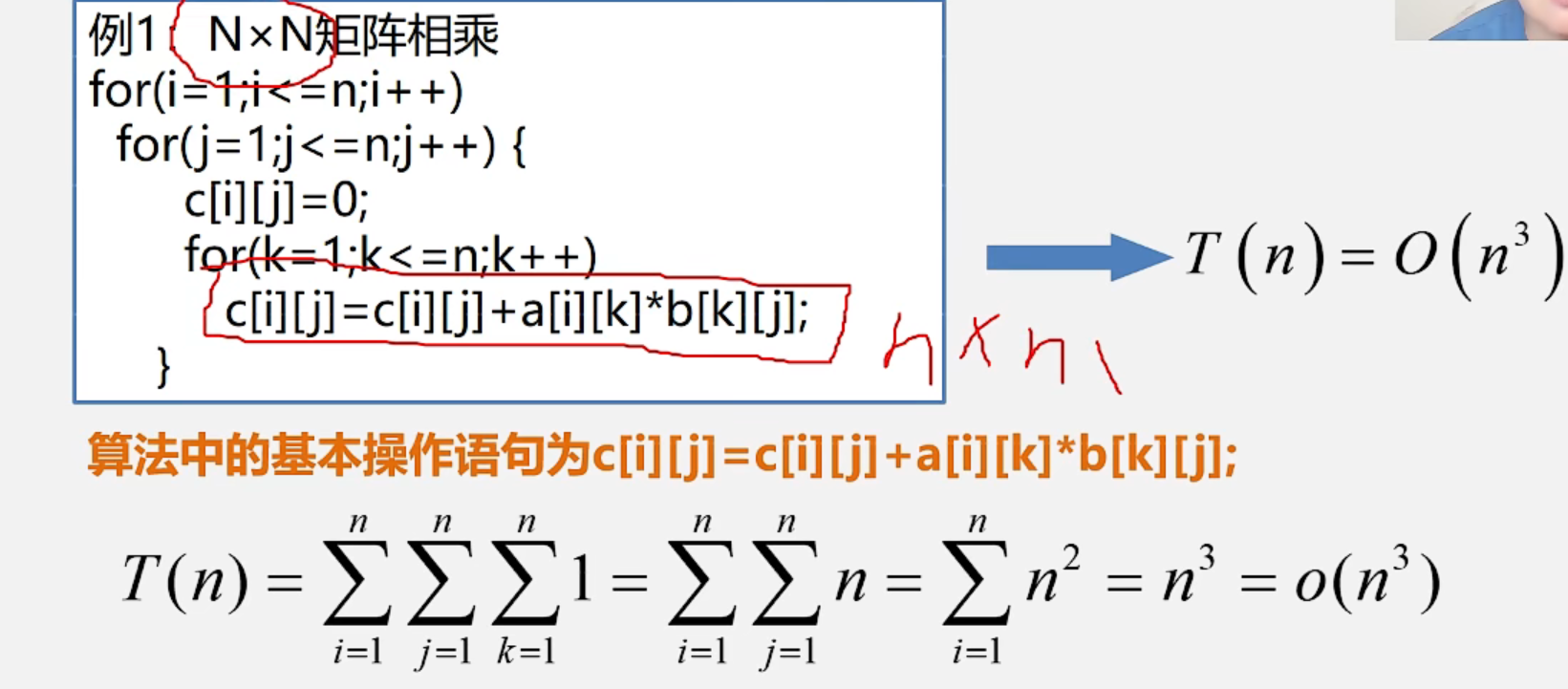
2.计算基本语句的频度得到问题规模n的某个函数Fn

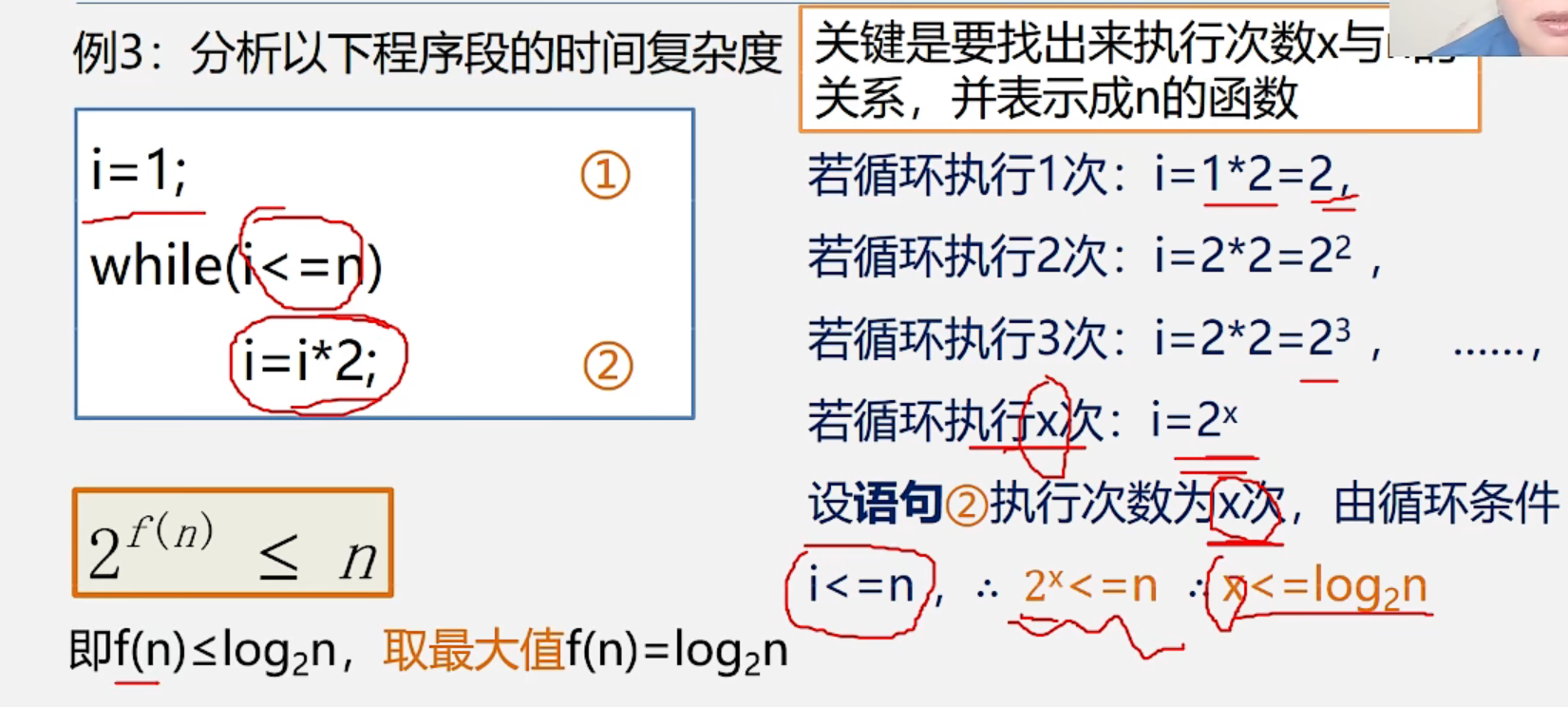
3.取其数量级用符号O表示

 //执行一次

//循环体内语句执行n次

N平方次





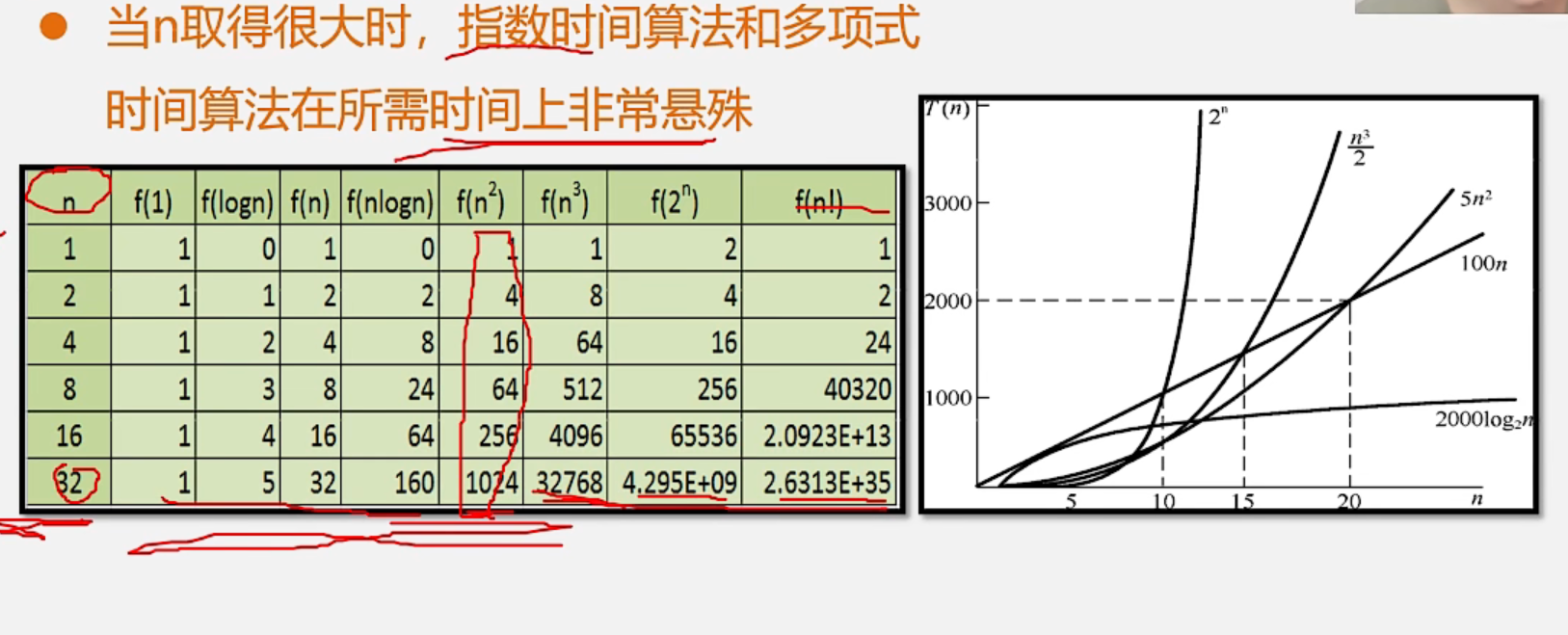
图形用户界面, 文本, 应用程序

描述已自动生成

对于复杂的算法，可以将他们分成几个容易估算的部分，然后利用大O加法法则和乘法法则，计算算法的时间复杂度

对于加法，假设有两个Tn，则取最大的

对于乘法则是将两个相乘



表格

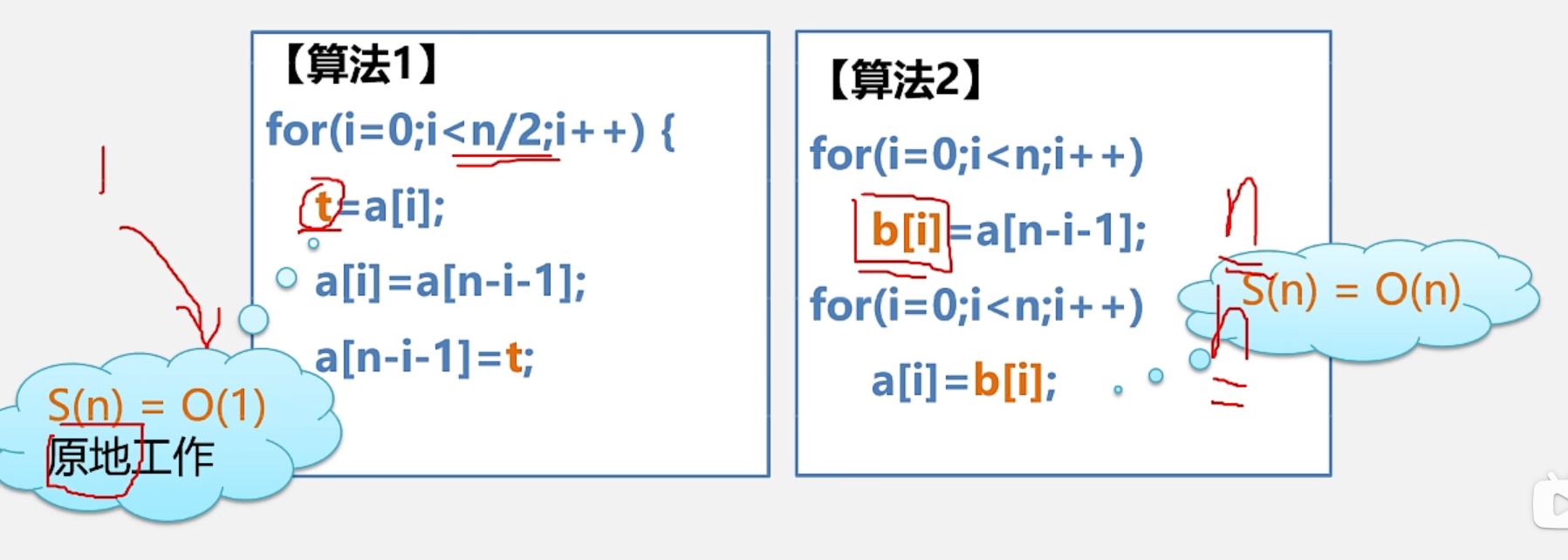
描述已自动生成

1. 空间效率

就是这个算法要占用多少空间

比如将一个int数组倒序，我可以新建一个int i，然后数组第一个放i，数组第一个=数组最后一个，数组最后一个=i

也可以新建一个int数组，循环将数组倒序存在新数组，然后再旧数组=新数组

不重要