计算导论与程序设计 复习大纲

第一部分：计算、计算机发展史、计算模型

1、什么是计算？

以数字为基础、遵循一定的计算规则进行; 如：数的加减乘除，函数的微分、积分、方程的求解、定理的证明推导;
抽象：计算就是把一个符号串 f 变换成另一个符号串 g；
广义（也是精确定义）：计算就是对信息的变换；
由于计算规则的机械化、公式化，可以借助计算工具（根据计算规则进行计算的辅助工具）来实现计算。

2、什么是计算思维？

计算思维是运用计算机科学的基础概念进行问题求解、系统设计、以及人类行为理解等涵盖计算机科学之广度的一系列思维活动，特点是抽象与自动化
抽象：忽略事物复杂的细节，抽取本质的特征，建立反映世界事物本质特征的模型；要求是充分描述（包含所要处理事物的重要元素）和充分简单（只包含重要元素）。

3、信息量度量的初步概念，信息熵，信息编码

一条消息的信息量不是由长度决定的，而是由取决于信息的不确定程度。如果消息内容已知，则消息的信息量为 0；如果消息内容发生的概率大，不确定性小，信息量小。
在信息论中，认为信源输出的消息是随机的。即在未收到消息之前，是不能肯定信源到底发送什么样的消息。而通信的目的也就是要使接收者在接收到消息后，尽可能多的解除接收者对信源所存在的疑义（不定度），因此这个被解除的不定度实际上就是在通信中所要传送的信息量；即，信息量=不确定减少的量。
先验概率=收到信息 A 的可能性；事后概率=相信程度

$$信息量 I_A = \log_2(事后概率) - \log_2(先验概率)$$

信源的熵=发送一次消息的平均信息量
时间熵=单位时间内信息量

$$信息熵 = \sum 先验概率 * 信息量 \left(\sum 先验概率 = 1\right)$$

信息编码：1bit 的熵相当于一个 2 进制位，使用其他编码只需要换算。

4、图灵机的计算模型： 组成，计算过程，状态及状态转移。

组成：一条无限长纸带+一个读写头+有限状态转换器（控制器）+程序（规则表、指令集）
计算过程：对一条两端可无限延长的纸带上的一串 0 或 1，首先从开始状态启动，每次动作都由控制器根据图灵机当前状态和读写头所对准的符号决定下一步的操作。每一步操作

包含下面三个事情：1.把某个符号写入读写头当前对准的小格内，取代原来那个符号；2。读写头向左移动一格，向右移动一格，或者不移动；3.根据控制器的命令用某个状态取代当前状态，使图灵机进入下一个新状态。经过有限步骤最后得到一个满足预先规定的符号串。

5、结合图灵机，什么是程序？ 理解程序的含义

程序是五元组<q,X,Y,R(或 L 或 N),p>形式的指令集。
每一条指令定义了机器在一个特定状态 q 下从方格中读入一个特定字符 X 时所采取的动作为在该方格中写入符号 Y, 然后向右移一格 R(或向左移一格 L 或不移动 N), 同时将机器状态设为 p 供下一条指令使用。

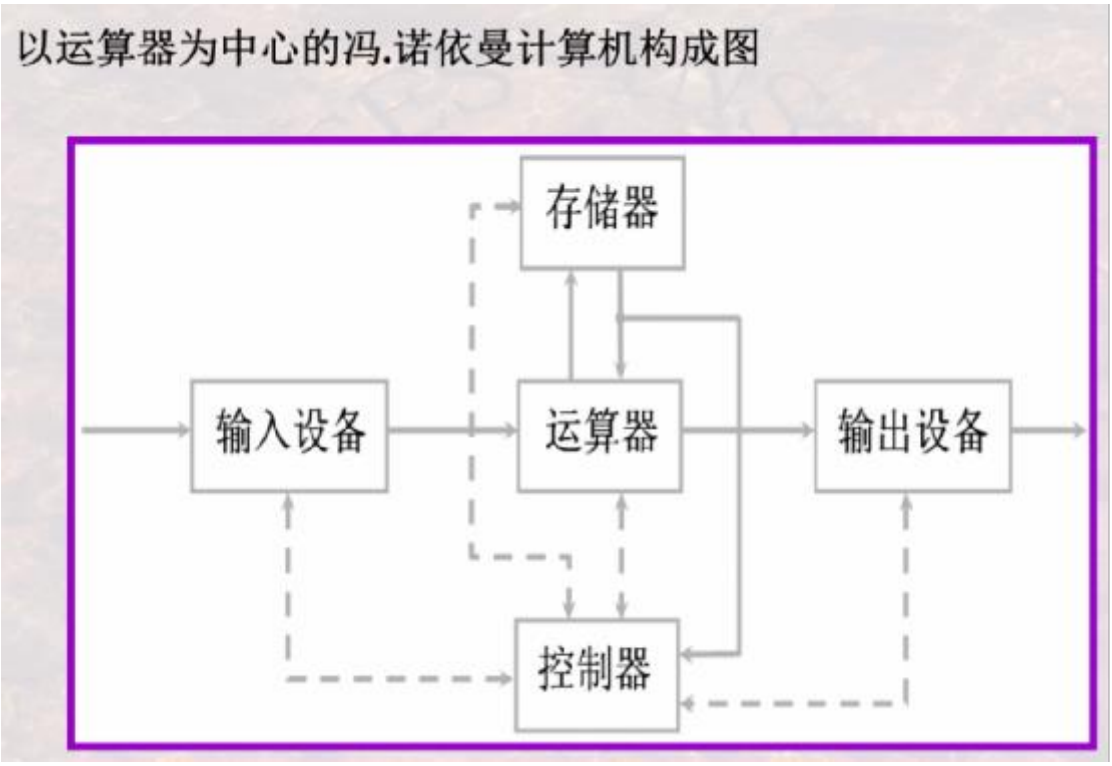
6、什么是存储程序的概念？

指令和数据以同等地位事先存于存储器, 可按地址寻访读取, 连续自动执行

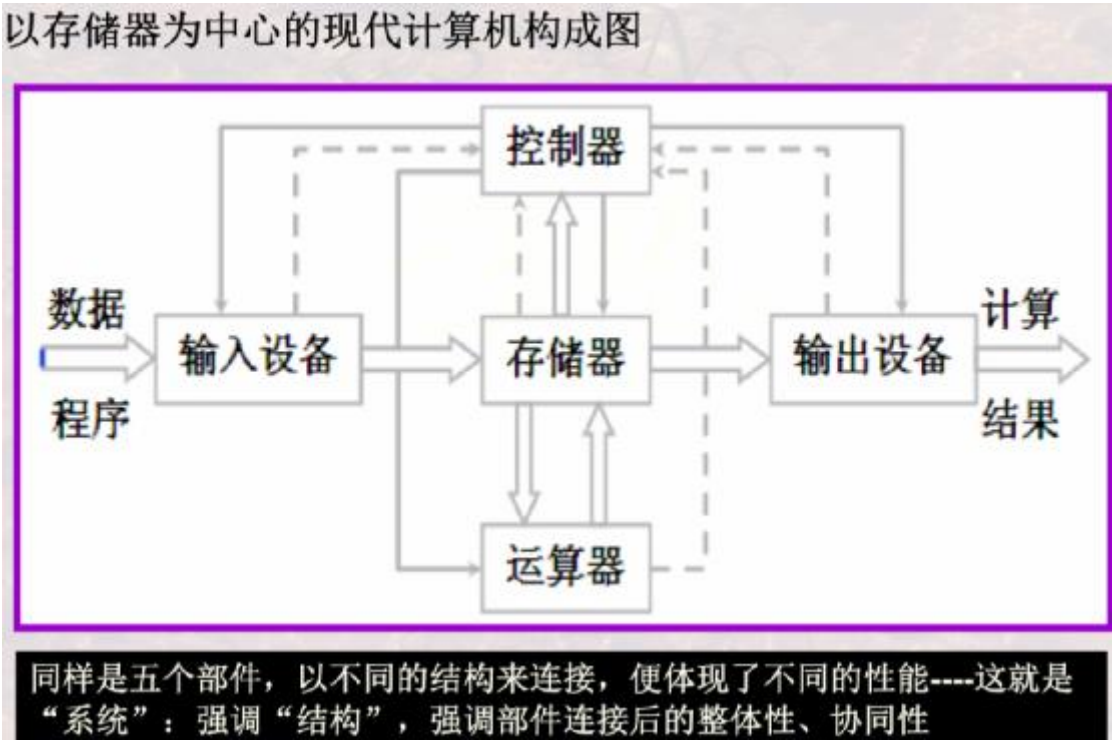
第二部分：计算机组成与原理知识点：

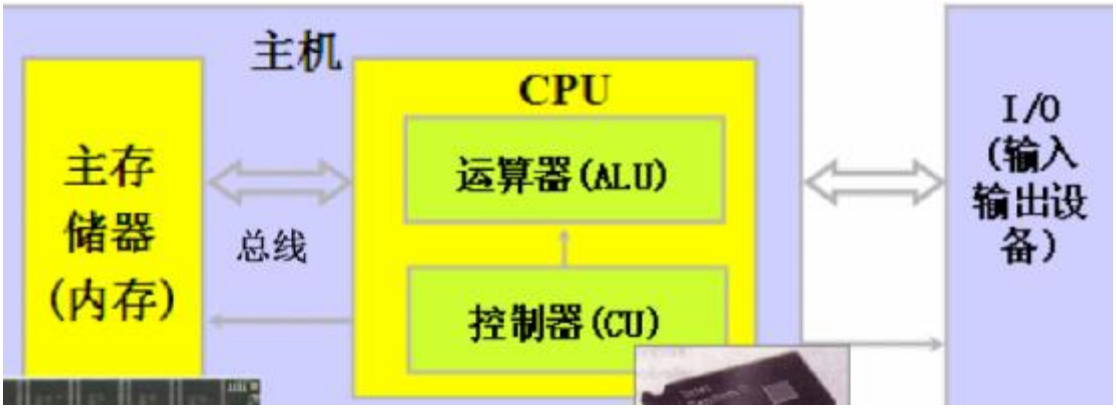
1、冯诺依曼计算机的组成结构

冯诺依曼机特点：指令和数据均采用 2 进制，均存在主存储器中，由运算器、控制器、存储器、IO 组成



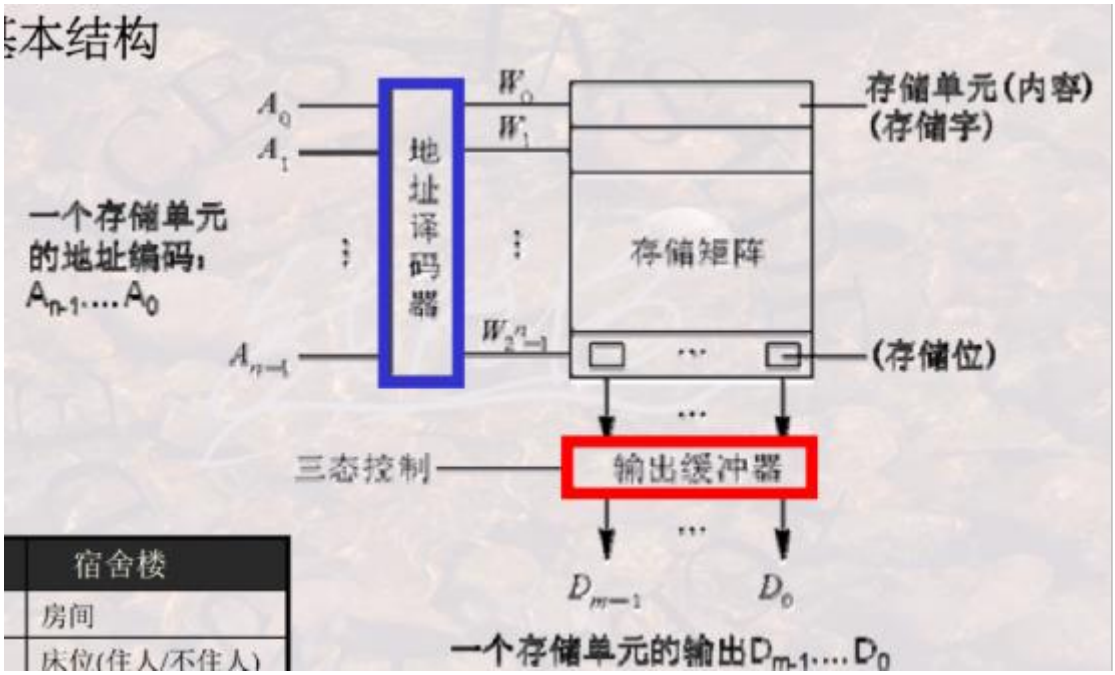
CPU、主存储器、 I/O 设备及总线成为现代计算机的四大核心部件





2、存储器与存储系统，

a)存储系统 高速缓存、内存、外存
b)存储空间，存储地址、存储单元，位与字节
存储空间=地址总数*存储字长
存储单元个数 = 2^地址总线数
存储字长 = 2^数据总线数



3、控制器及运算器

a)控制器结构
程序计数器+指令寄存器+指令译码器+微操作控制部件+时序电路
b)运算器结构结构
算术逻辑部件+数据寄存器
c)指令的执行
取指令（存储器）->分析指令（控制器）->执行指令（运算器）
d)指令计数器 PC
用来存放下一条待执行指令在主存储器中的地址
e)指令及指令系统，指令的组成
指令≈操作码+地址码

4、总线

地址总线：传送内存地址编码，给地址译码器。
数据总线：访问传送数据，用于 CPU 与内存之间、输入\输出设备与内存之间。

5、系统软件（操作系统）

a)什么是操作系统？强调对计算机资源的管理，作业（进程）的调度
由一系列具有控制和管理功能的子程序构成的大型系统软件，直接运行在裸机上，是对计算机硬件系统的第一次扩充。
b)操作系统的组成
c)通过操作系统的演变（多道批处理、分时），了解进程的概念，进程的并发，进程的同步
批处理：将作业按照它们的性质分组（或分批），然后再成组（或成批）地提交给计算机系统，由计算机自动完成后再输出结果，从而减少作业建立和结束过程中的时间浪费

多道：系统内可同时容纳多个作业。
分时：各个终端只在自己的时间片里占有 CPU。
进程：动态执行的程序，一般包括：程序 、程序控制块（PCB）。
执行中的程序，开始未停止的程序
并发：其实是分时进行的，只是时间太短，看起来像同时，不同于并行。

第三部分 程序语言及程序设计基础知识点

1、标识符

由程序员定义的单词，用来给程序中的各种对象命名

2、数据类型及数据类型的三要素（表示、存储、操作）

定义了“如何表示一个数据”、“如何存储一个数据”、“能对这些数据进行的操作”

3、变量及变量的三要素

三要素：变量名、变量类型、变量值

4、表达式

注意表达式的递归定义

5、三种基本语句：赋值、输入、输出

6、三种基本程序结构：顺序、分支、循环

第四部分 算法设计方法知识点

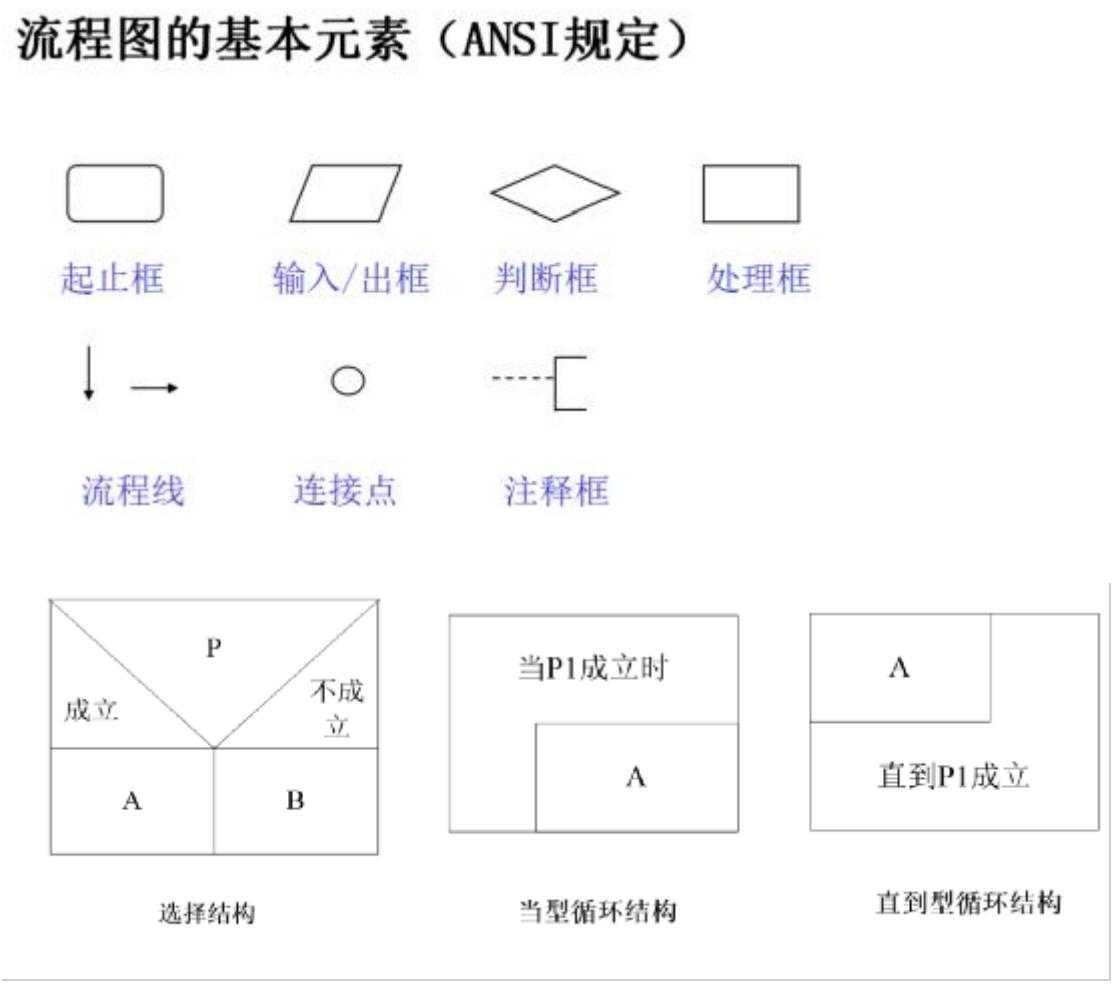
1、什么是算法？算法的五大特征

算法是解决问题的步骤序列（操作序列）
5 大特征：可执行、有穷、确定、有输入说明、有输出
（可执行性：算法中的每一个步骤都在计算机能力集范围内 ；确定性：算法中的每一个步骤必须是明确定义的，不得有任何歧义）

2、算法设计方法：自顶向下逐步求精（主函数调用子函数）、模块化（子函数）、

结构化编码（自顶向下逐步求精+顺序结构、分支结构、循环结构）
自顶向下逐步求精：明确程序与外界的数据关系，即明确输入和输出。以输入、处理和输出的顺序关系表达(顶层）；针对输入、处理和输出每一部分逐层细化。

3、算法的描述方法：自然语言、流程图、N-S 流程图、伪代码、计算机语言



4、迭代

在程序设计中，重复执行同样操作的过程称为“迭代”。程序中被重复执行的程序段称为“循环”。
①确定迭代变量 ai。在可以用迭代算法解决的问题中，至少存在一个直接或间接地不断由旧值递推出新值的变量，这个变量就是迭代变量。
②建立迭代关系式，即 ai = f(ai-1)
③对迭代过程进行控制 。迭代过程的控制通常可分为两种情况：一种是所需的迭代次数是个确定的值，可以计算出来；另一种是所需的迭代次数无法确定。

5、穷举

①明确所有的组合情况；②检查每一种组合情况是否满足条件

6、算法思路：问题抽象（数学建模），求解问题的步骤

实际问题首先抽象为数学模型（数学公式），再基于数学模型设计算法、进行求解。
钱隽朗的补充：

7、计算机科学与技术学科的根本问题是：什么能够被有效地自动化 ；

8、设计程序的根本目的：让计算机帮助人们自动地完成所要处理的复杂任务；

9、算法代表了对问题的解，程序则是算法在计算机上的特定的实现。一个算法若用程序设计语言来描述，则它就是一个程序.

10、算法设计任务结束的标准是各步骤已精细到能用语句描述，即满足算法的 5 大特征标志着算法设计任务的结束 。

第五部分 子程序（函数）知识点

1、函数三要素：函数名、参数、返回值

2、函数的定义、函数原型

子程序是封装并给以命名的一段程序代码。封装：调用者只需要关心代码能完成什么功能，如何调用代码（即子程序接口），而不需要关心代码的内部实现。
函数原型（涉及范围包括自己写的函数和库函数）：类似函数定义时的函数头，又称函数声明。
函数原型的作用：是对被调用函数的接口声明，它告诉编译器函数返回的数据类型、函数所要接收的参数个数、参数类型和参数顺序，编译器用函数原型校验函数调用是否正确。
引入子函数的意义：减少代码冗余，避免重复代码、结构化编码的需要（自顶向下逐步求精）、模块化编码的需要（封装）、更容易调试维护（因为模块化）

3、函数的调用

a)函数的参数原理，形参与实参
形参的本质是一个名字，只有在被调用时，才分配内存单元；调用结束，即刻释放。实参的本质是一个变量，已经占用了内存空间。
实参和形参占用不同的内存单元，即使同名也互不影响。实参对形参的数据传送是单向的，即只能把实参的值传送给形参，而不能把形参的值反向传送给实参。
b)函数的调用过程：堆栈、函数活动记录
函数调用过程：给被调用函数分配存储空间→计算实际参数表达式的值（就是函数括号里面的那个）→把实际参数表达式的值按赋值转换规则转换成形式参数的类型，把转换后的实参表达式的值送入形参中→运行调用函数中的语句→计算返回值表达式的值，并转换成函数的结果类型→释放被调用函数占用的存储空间→带着转换后的值返回调用函数。
传值调用：把实参的值给形参；引用调用：把实参的地址给形参。
内存主要分为以下几个区：
系统区：用于存放系统软件和运行需要的数据，如操作系统。只要机器一运行，这部分空间就必须保留给系统软件使用
用户程序代码区：存放用户程序代码
静态存储区：存放程序运行期间不释放的数据（静态局部变量、全局变量）
栈区：存放程序运行期间会被释放的数据（ 函数参数、 非静态局部变量）以及活动的控制信息

堆区：用户可以在程序运行过程中根据需要动态地进行存储空间的分配，这样的分配在堆区进行
栈使用的是
一级缓存
，由操作系统自动分配释放，他们通常都是被调用时处于存储空间中，调用完毕立即释放。堆则是存放在
二级缓存
中，一般由程序员分配释放，生命周期由虚拟机的垃圾回收算法来决定（并不是一旦成为孤儿对象就能被回收）。所以调用这些对象的速度要相对来得低一些 。
数据结构中的栈就是只有一个弹压端的“先进后出”线性表。

4、子程序设计原则

高内聚低耦合：功能相对独立、单一、完整（高内聚）；与外界（调用者）的关系尽量松散而非紧密，使其能方便地被重用（低耦合）；
参数设计：空最好要写一个 void；类型不一致的时候会被强转，以程序接头处的定义为准。
减少代码冗余；主程序（主函数）的代码控制在 10 行语句内

5、变量的作用域：仅在所处函数以及所处函数的下级函数（不能包括函数中被调用的平级函数和递归调用自身的函数）

定义在函数内部的定义部分（即任何语句之前）的变量对应函数作用域：只有在定义变量的函数内部才能使用这些变量 ；
定义在函数内部的某一个复合语句内部的变量对应块作用域：只在复合语句范围内才能引用该变量 ；
定义在所有函数之外的变量（外部变量/全局变量）对应文件作用域：用域：从变量的定义位置开始，到本文件结束为止的区域可以引用该变量。若该变量被定义成非静态外部变量， 则其也能被其他文件引用。
主函数 main()与其它函数是平行关系。main()中定义的内部变量，也只能在主函数中使用，其它函数不能使用。同时，主函数中也不能使用其它函数中定义的内部变量。
形参变量也是内部变量，属于被调用函数。
允许在不同的函数中使用相同的变量名，它们代表不同的对象，分配不同的单元，互不干扰，也不会发生混淆。在同一源文件中，允许外部变量和内部变量同名。在内部变量的作用域内，外部变量将被屏蔽而不起作用。允许函数定义部分定义的变量与该函数内部的复合语句中定义的变量同名。在复合语句执行时，函数定义部分定义的变量是“隐藏的”，直到复合语句结束。
钱隽朗的补充：

6、函数实参：函数作为另一个函数调用的实际参数。这种情况是把该函数的返回值作为实参进行传送。如：printf(” the large number is %d”, max(x, y));

第六部分 递归（函数递归）知识点

1、递归的概念，递归函数定义

从程序书写来看，在定义一个函数时，若在函数的功能实现部分又出现对它本身的调用，则称该函数是递归的或递归定义的。从函数动态运行来看，当调用一个函数 A 时，在进入函数 A 且还没有退出（返回）之前，又再一次由于调用 A 本身而进入函数 A，则称之为函数 A 的递归调用。
分为直接递归和间接递归。
任何能用递归解决的问题都能用迭代（循环）的方法去解决

2、递归过程，基于函数调用过程能够自主分析递归过程，得出结果。

递归算法的执行过程分递推和回归两个阶段。当递推时，并没有求值的计算操作，实际的计算操作是在回归过程实现的。
递推阶段是个不断简化问题的阶段：把对较复杂问题（规模为 n）的求解转化为比原问题简单一些的问题（规模小于 n）的求解。当递推到最简单的不用再简化的问题时（直接得到答案了），递推终止。
在回归阶段，当获得最简单情况的解后，逐级返回，依次得到稍复杂问题的解。

3、递归程序设计，化简为同类问题，分解直至能求解

递归的思想就是先将一个问题转化为与原问题性质相同、但规模小一级的子问题，然后再重复这样的转化，直到问题的规模减小到我们很容易解决为止。
一般来说，递归需要有边界条件、递归前进段和递归返回段。当边界条件不满足时，递归前进；当边界条件满足时，递归逐级返回。

4、递归函数的参数设计，不建议使用全局变量。

钱隽朗的补充：

5、对于有大量重复的递归求解的优化方法：记忆化

第七部分 数组知识点：

1、数组的概念：存储结构

数组是一种基于简单数据类型构造而成的复杂数据类型，引入数组是为了增强对复杂数据的抽象和表达能力，存储能力、操作能力。
数组是线性表的一种存储结构。
（存储结构）数组是一组连续的存储单元，连续的含义：这些存储单元位置相邻（一个接一个），可以容纳多个具有相同数据类型的数据元素(数据项)，这些数据元素具有相同的名字；存储位置的相邻性体现了线性表数据元素之间的相邻性。

2、数组的定义、下标运算符

3、数组的逐元素访问（Acess）

4、数组作为函数参数 （数组名是引用，传引用）

数组名是首地址，传进去就能算出其他地址

5、字符数组： 存储特征 --结束符；整体的输入与输出操作；字符串库函数

以字符'\0'作为字符串的结束符。'\0'对应的 ASCII 码为 0。

6、二维数组： 特别是作为函数参数的用法

第八部分 指针与数组知识点：

1、指针的概念，指针的定义（语法）

2、指针运算符： 间接访问（取值）* 取地址&

3、指针作为函数参数：传引用

4、指针变量指向数组，指针的算术运算、关系运算

有效的指针运算：赋值、算数运算、关系运算
变量地址：系统分配给变量的内存单元的起始地址
数组元素的地址：下标小的元素对应的内存地址小
指针的关系运算：比较两个指针所指向的存储单元的内存地址大小

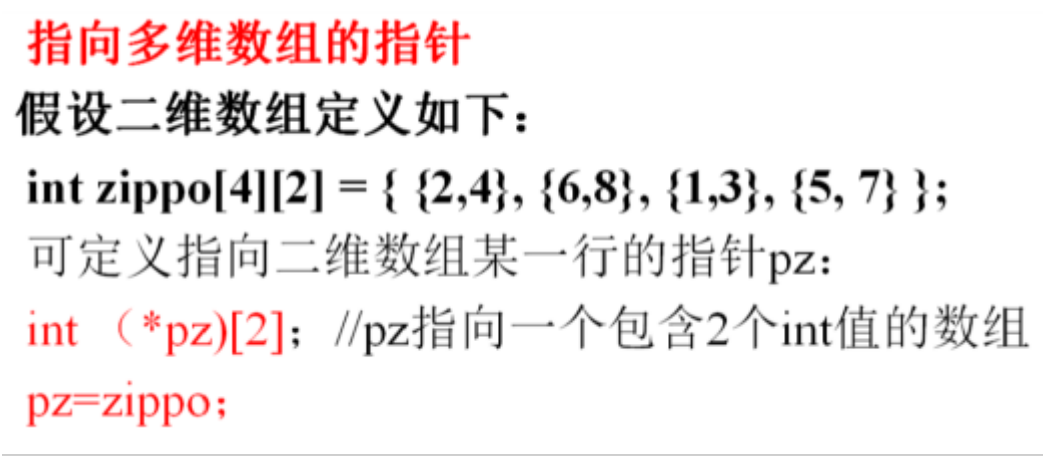
- aPtr ++, ++ aPtr：aPtr指向数组的下一元素
- aPtr--， --aPtr ：aPtr指向数组的上一元素；
- aPtr+=n： aPtr指向当前所指元素之后的第n个元素；
- aPtr-= n： aPtr指向当前所指元素之前的第n个元素；
- aPtr+n：表示aPtr当前所指元素之后的第n个元素的地址；
- aPtr-n：表示aPtr当前所指元素之前的第n个元素的地址；
- 若p1与p2指向同一数组，p1-p2=两指针间元素的个数，而非字节数
- p1+p2 无意义

注意：ptr+=i 与 ptr+i 的区别

5、指针与数组：

a)数组元素的 4 种访问方式（下标、偏移量）
b)指针与数组： 动态数组， 堆区分配存储
数组的大小是不能改变的。这种数组称为静态数组。静态数组的缺点是：对于事先无法准确估计数据量的情况，无法做到既满足处理需要，又不浪费内存空间。

动态数组：链表实现
c)指针数组，指向指针的指针，间接访问之间接访问
d)指针数组， 多个字符串的处理，排序、子串。
e)二维数组中的行指针



第九部分 自定义数据类型 结构知识点：

1、再论数据类型（三要素），用户自定义（构造）数据类型

结构是一种数据类型，是用其他类型的对象构造出来的派生数据类型

2、结构的定义，结构成员的访问，结构变量的操作（结构体的整体赋值）

3、结构作为函数参数 结构传值、传引用（值参与变参）

第十部分： 数据结构+算法=程序知识点：

1、基于数组、结构，信息与数据抽象，数据结构设计

人的信息抽象能力使计算机具有对应的信息的表示和处理能力

2、在数据结构设计的基础上进行算法设计

3、计算机学科方法论：三个形态（抽象-理论-设计），

4、算法设计的理论基础： 递归函数（迭代设计）、图灵机