CSP 初赛复习

CCF 在2019推出非专业级软件能力认证(Certified Software Professional Junior/Senior，简称CSP-J/S)，CSP-J/S分两个级别进行，分别为CSP-J（入门级，Junior）和CSP-S（提高级，Senior），两个级别难度不同，均涉及算法和编程。CSP-J/S分第一轮（初赛）和第二轮（复赛）两个阶段。初赛考察通用和实用的计算机科学知识，以笔试为主，部分省市以机试方式认证。复赛为程序设计，须在计算机上调试完成。初赛成绩优异者可进入复赛。 CSP-J/S成绩优异者，可参加NOI省级选拔（NOIP），省级选拔成绩优异者可参加NOI。

CSP-J/S第一轮有三种题目类型：单项选择题（30分）、阅读程序（40分）和完善程序（30）。单项选择题考查计算机基础知识、简单的数据结构（栈、队列、链表、树和图等）和算法（排序、递归、二分等）、简单的数学知识(排列组合等)和常识等，只要多练习多积累都不难解决；阅读程序部分有判断题和根据输入选结果两种题型，需要大家养成认真耐心的阅读习惯，结合数学、算法等内容培养良好的代码阅读和分析能力。 完善程序题目一般与经典的算法问题（高精度计算、二进制、贪心、动态规划等相关问题）有关，但是代码处理方法偶尔有些特别。

CSP-J/S第一轮成绩优异是参加第二轮的前提，所以我们必须认真备考。

1、计算机基础知识

一、计算机常识

(一) 计算机发展历程：

	年代	元件
第一代	1946 - 1958	电子管
第二代	1959 - 1964	晶体管
第三代	1965 - 1970	集成电路
第四代 (微机)	1971 - ?	大规模集成电路

(二)第一台电子计算机：

1946 年 2 月，在美国宾夕法尼亚大学诞生了世界上第一台电子计算机 ENIAC（Electronic Numerical Integrator And Computer），这台计算机占地 170 平方米，重 30 吨，用了 18000 多个电子管，每秒能进行 5000 次加法运算。

(三)冯·诺依曼理论

1944 年，美籍匈牙利数学家 冯·诺依曼 提出计算机基本结构和工作方式的设想，为计算机的诞生和发展提供了理论基础。时至今日，尽管计算机软硬件技术飞速发展，但计算机

本身的体系结构并没有明显的突破，当今的计算机仍属于冯·诺依曼架构。 其理论要点如下：

1) 计算机硬件设备由存储器、运算器、控制器、输入设备和输出设备 5 部分组成。
2) 存储程序思想——把计算过程描述为由许多命令按一定顺序组成的程序，然后把程序和数据一起输入计算机，计算机对已存入的程序和数据处理后，输出结果。

(四) 计算机的常见应用

1) 数值计算（科学计算）：弹道轨迹、天气预报、高能物理等等
2) 数据处理（信息处理）
3) 自动控制（自动加工、室内温度调节）
4) 人工智能（无人驾驶）
5) 辅助工程：
   a) CAD（计算机辅助设计）、CAM（计算机辅助制造）
   b) CAT（计算机辅助测试）、CAI（计算机辅助教学）
   c) CAE（计算机辅助教育）、CIMS（计算机集成制造系统）

注意，手机、pad等智能设备也是计算机。

(五)摩尔定律

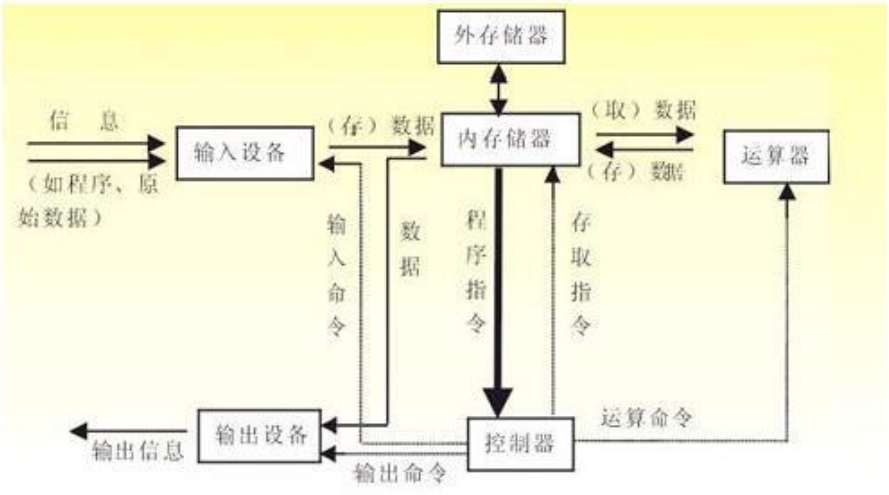
摩尔定律是英特尔创始人之一戈登·摩尔的经验之谈，其核心内容为：集成电路上可以容纳的晶体管数目在大约每经过18个月便会增加一倍。换言之，处理器的性能每隔两年翻一倍。

(六)图灵奖

图灵奖（Turing Award），全称A.M.图灵奖（ACM A.M Turing Award），是由美国计算机协会（ACM）于1966年设立的计算机奖项，名称取自艾伦·麦席森·图灵（Alan M. Turing），旨在奖励对计算机事业作出重要贡献的个人 。

二、 计算机硬件

计算机硬件由五大部分组成：运算器、控制器、存储器、输入设备、输出设备。



(一)中央处理器（CPU——Central Processing Unit）

1) 由运算器、控制器和一些寄存器组成；
2) 运算器进行各种算术运算和逻辑运算；
3) 控制器是计算机的指挥系统；
4) CPU 的主要性能指标是主频和字长。

主频是计算机主时钟在一秒钟内发出的脉冲数，即CPU的工作时钟频率。一般来说，主频越高，一个时钟周期里面完成的指令数也越多，速度也就越快。主频的单位为兆赫兹（MHz）和吉赫兹（GHz）。

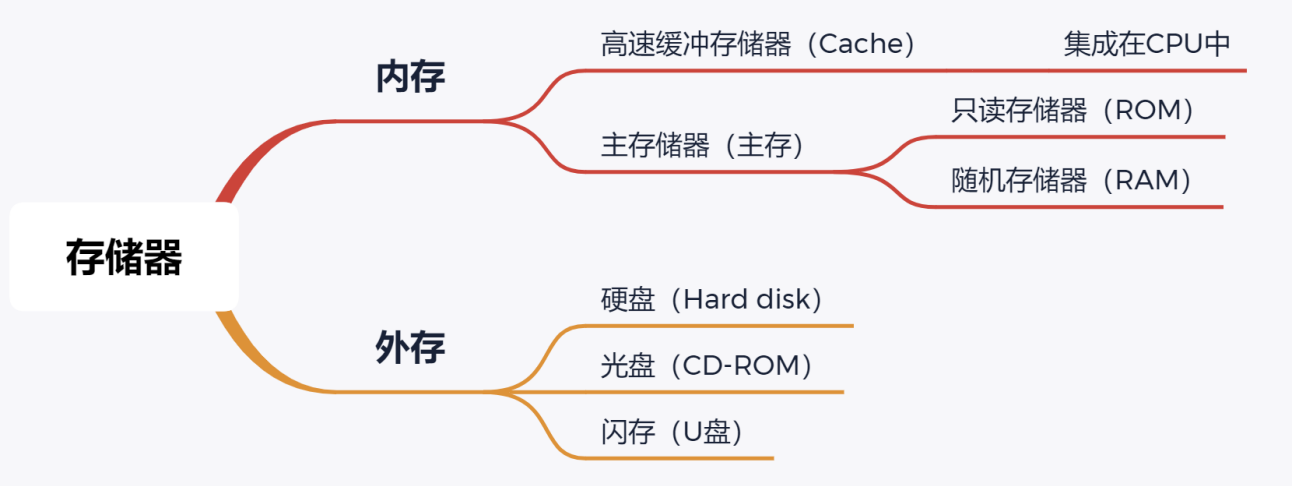
字长是计算机能够直接处理的二进制数据的位数。通用寄存器的位数和计算机字长保持一致。

指令是一组二进制代码，它规定了由计算机执行的程序的一步操作。一条指令由操作码和操作数组成，前者规定指令要完成的操作，必不可少；后者是这个操作针对的对象，可以没有。

指令系统是一种计算机所能识别并可执行的全部指令的集合。一台计算机的指令系统和它所用的CPU有直接关系。不同厂商生产的CPU，指令系统可能不同。

(二)存储器

中央处理器 (CPU) 能直接访问的存储器称为内部存储器(内存)，中央处理器不能直接访问的存储器称为外部存储器(外存)，外部存储器中的信息必须调入内存后才能被CPU处理。



内存包括高速缓冲存储器（Cache）和主存储器（主存），主存储器按读写功能，可分只读存储器（ROM）和随机存储器（RAM）两种。

CPU对寄存器和存储器的存取速度大小关系为：

寄存器 > 高速缓存 > 内存 > 外存

常见的外存有：硬盘（Hard disk）、光盘（CD-ROM）、闪存（U盘）等。

内存由半导体存储器组成，存取速度快，但容量较小。内存中存在很多的存储单元，每个单元可以存放 1 个8位二进制数，即 1 个字节（Byte，简称 B）。内存中的每个字节都各有一个固定的编号，这个编号称为地址。CPU在存取存储器中的数据时是按地址进行的。存储器（包括内存和外存）中包含的字节数称为存储器的容量，通常用KB、 MB、 GB、 TB、 PB 为单位，他们之间的换算关系为：

1KB = 1024B　　1MB = 1024KB　　1GB = 1024MB

1TB = 1024GB　　1PB = 1024TB

(三) 输入设备

常见的输入设备有：键盘（Keyboard）、鼠标（Mouse）、手写笔 、触摸屏、麦克风、扫描仪（Scanner）、视频输入设备、条形码扫描器 等。

(四)输出设备

常见的输入设备有：显示器（Monitor）、打印机（Printer）、绘图仪、音箱 等。

(五)总线结构

计算机硬件之间通过总线传输信息。总线分为数据总线（DB）、地址总线（AB）和控制总线（CB）三种。

数据总线用来传送数据信息，它主要连接CPU与各个部件，是它们之间交换信息的通路。数据总线是双向的，具体的传送方向由CPU控制。

地址总线用来传送地址信息。CPU通过地址总线中传送的地址信息访问存储器。通常地址总线是单向的。同时，地址总线的宽度决定可以访问的存储器容量大小，如20条地址总线可以控制1MB的存储空间。（2^{20}B = 1MB）。

控制总线用来传送控制信号，以协调各部件之间的操作。控制信号包括CPU对内存储器和接口电路的读写控制信号、中断响应信号，也包括其他部件传送给CPU的信号，如中断申请信号、准备就绪信号等。

(六)例题

1. 微型计算机的问世是由于(C) 的出现。

A)中小规模集成电路　B)晶体管电路　C)(超)大规模集成电路　D)电子管电路

2. 中央处理器(CPU)能访问的最大存储器容量取决于(A)。

A)地址总线　B)数据总线　C) 控制总线　D) 实际内存容量

3. 微型计算机中，(C) 的存取速度最快。

A)高速缓存　B)外存储器　C) 寄存器　D) 内存储器

4. 在计算机硬件系统中，cache 是(D)存储器。

A)只读 B)可编程只读 C)可擦除可编程只读 D)高速缓冲

5. 计算机主机是由 CPU 与（ D ）构成的。

A. 控制器 B. 输入、输出设备 C. 运算器 D.存储器

6. 计算机系统总线上传送的信号有（ B ）。

A.地址信号与控制信号 B. 数据信号、控制信号与地址信号

C.控制信号与数据信号 D. 数据信号与地址信号

7. 不同类型的存储器组成了多层次结构的存储器体系，按存取速度从快到慢的排列是（C）。

A.快存/辅存/主存 B. 外存/主存/辅存

C. 快存/主存/辅存 D. 主存/辅存/外存

8. 微机内存储器的地址是按（C）编址的。

A. 二进制位 B. 字长 C.字节 D. 微处理器的型号

9. 在微机中，通用寄存器的位数是（ C ）。

A．8 位 B．16 位 C.计算机字长 D．32 位

10. 不同的计算机，其指令系统也不同，这主要取决于（ C ）。

A 所用的操作系统 B. 系统的总体结构 C．所用的 CPU D．所用的程序设计语言

11. CPU 访问内存的速度比访问下列哪个（些）存储设备要慢（ AD ）。

A）寄存器 B）硬盘 C）软盘 D）高速缓存 E）光盘

12. 下列哪个（些）不是个人计算机的硬件组成部分（ B ）。

A）主板 B）虚拟内存 C）电源 D）硬盘 E）总线

解析：虚拟内存指的是用外存来模拟内存，是一种存储器管理的方式，而不是硬件组成部分。

13. 美籍匈牙利数学家冯·诺依曼对计算机科学发展所做出的贡献是（ C ）。

A. 提出理想计算机的数学模型，成为计算机科学的理论基础。

B. 是世界上第一个编写计算机程序的人。

C．提出存储程序工作原理，并设计出第一台具有存储程序功能的计算机 EDVAC。

D. 采用集成电路作为计算机的主要功能部件。

E. 指出计算机性能将以每两年翻一番的速度向前发展。

14. 下列说法中错误的是（ B ）。

A. CPU 的基本功能就是执行指令。

B. CPU 访问内存的速度快于访问高速缓存的速度。

C. CPU 的主频是指 CPU 在 1 秒内完成的指令周期数。

D. 在一台计算机内部，一个内存地址编码对应唯一的一个内存单元。

E. 数据总线的宽度决定了一次传递数据量的大小，是影响计算机性能的因素之一。

15. 用静电吸附墨粉后转移到纸张上，是哪种输出设备的工作方式（ C ）。

A. 针式打印机 B. 喷墨打印机 C. 激光打印机 D. 笔式绘图仪 E. 喷墨绘图仪

解析：

a) 针式打印机是通过打印头中的24根针击打复印纸，从而形成字体。医院、银行、邮局窗口一般使用此类打印机。

b) 喷墨打印机是通过加热喷嘴，使墨水产生气泡，喷到打印介质上。

c) 激光打印机是用高压静电将感光鼓表面的“墨粉图像”转印到打印纸上。

d) 笔试绘图仪是一种装有画笔的平板式绘图机。

e) 喷墨绘图仪适用于输出排料图和头版的专用宽幅单色绘图仪，打印介质是墨盒就叫喷墨绘图仪。

16. 处理器A 每秒处理的指令数是处理器B 的2 倍。某一特定程序P 分别编译为处理器A和处理器B的指令，编译结果处理器A 的指令数是处理器B的4倍。已知程序P的算法时间复杂度为O（n^2），在处理器A上执行时1个小时能完成输入规模为n的数据，那么在输入相同的情况下，程序P在处理器B上执行需要（D）小时。

A. 4 B. 2 C. 1 D. 1 / 2 E. 1 / 4

17. 以下哪个不是计算机的输出设备（D）。

A. 音箱 B. 显示器 C. 打印机 D. 扫描仪 E. 绘图仪

18. 现有一张分辨率为 2048×1024 像素的 32 位真彩色图像。请问要存储这张图像，需要多大的存储空间？（ C ）。

A. 16MB B. 4MB C. 8MB D. 2MB

19. 以下哪个奖项是计算机科学领域的最高奖？（ A ）

A. 图灵奖 B. 鲁班奖 C. 诺贝尔奖 D. 普利策奖

20. 摩尔定律（Moore's law)是由英特尔创始人之一戈登·摩尔(Gordon Moor)提出来的。根据摩尔定律，在过去几十年一级在可预测的未来纪念，单块集成电驴的集成度大约每（ C ）个月翻一番。

A. 1 B. 6 C. 18 D. 36

21. 计算机的运算速度取决于给定的时间内，它的处理器所能处理的数据量。处理器一次能处理的数据量叫字长。 已知 64 位的奔腾处理器一次能处理 64 个信息位，相当于（ A ）字节。

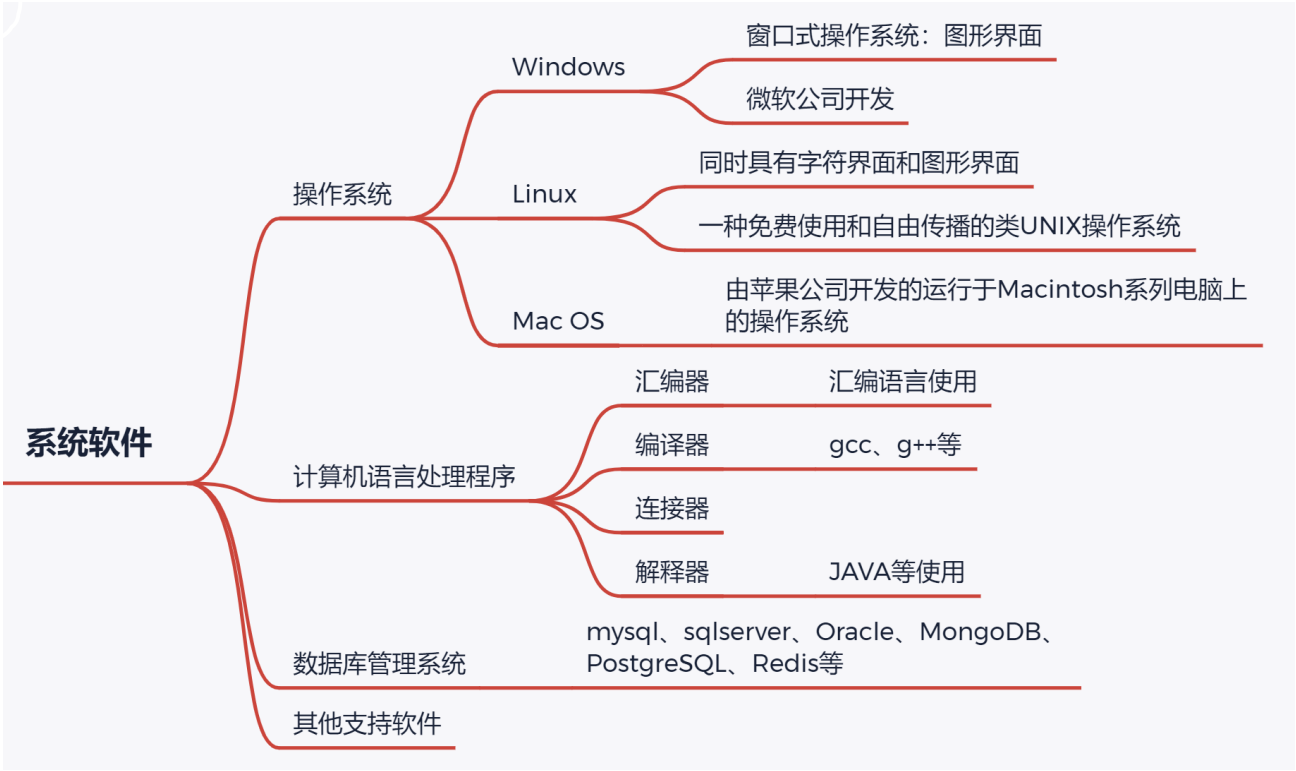
A.8 个　　B.1 个　　C.16 个　　D. 2 个

三、计算机软件系统

(一)计算机软件

计算机软件可分为系统软件和应用软件两大类。 系统软件是指控制和协调计算机及外部设备，支持应用软件开发和运行的系统，是无需用户干预的各种程序的集合，主要功能是调度、监控和维护计算机系统，负责管理计算机系统中各种独立的硬件，使得它们可以协调工作。

系统软件包括操作系统、数据库管理软件、编程语言处理程序等。



应用软件是用户为了解决各自应用领域里的具体任务而编写的各种应用程序和有关文档资料的统称。常见应用软件有：办公软件、图形图像软件、教育软件、电子游戏、通信软件等。

(二)需掌握的常用操作

1. windows 环境下新建、复制、删除、移动文件或目录
2. windows 环境下的集成开发环境的使用（如 Dev C++等）
3. Linux 系统下的集成开发环境的使用（如 Code::Blocks 等）
4. 常见编译器（gcc、g++）的基本使用

(三)例题

1. 在磁盘上建立子目录有许多优点,下列描述中不属于建立子目录优点的是(D)。

A)便于文件管理　　B) 解决根目录中目录项个数有限问题

C) 加快文件查找速度　D) 节省磁盘使用空间

2. 资源管理器的目录前图标中增加"+"号,这个符号的意思是(B)。

A)该目录下的子目录已经展开　B)该目录下还有子目录未展开

C) 该目录下没有子目录　　D) 该目录为空目录

3. 在树型目录结构中，不允许两个文件名相同主要指的是(D)

A)同一个磁盘的不同目录下　B)不同磁盘的同一个目录下

C)不同磁盘的不同目录下　D)同一个磁盘的同一个目录下

4. 以下对 Windows 的叙述中，正确的是(A)

A)从 U 盘上删除的文件和文件夹，不送到回收站

B)在同一个文件夹中，可以创建两个同类、同名的文件

C)删除了某个应用程序的快捷方式，将删除该应用程序对应的文件

D)不能打开两个写字板应用程序

5. 下列哪个软件属于操作系统软件（ E ）。

A. Microsoft Word　B. 金山词霸　C. Foxmail　D. WinRAR　E. Red Hat Linux

6. 下列哪个不是数据库软件的名称（ D ）。

A. MySQL　B. SQL Server　C. Oracle　D. 金山影霸　E. MongoDB

7. 操作系统的作用是(C)。

A. 把源程序译成目标程序　　B. 便于进行数据管理

C. 控制和管理系统资源　　D. 实现硬件之间的连接

8. 以下哪个软件不是即时通信软件（D）。

A. QQ　　B. 钉钉　　C. 微信　　D. P2P

2、数制转换与编码

一、 数制转换

常用的数制及它们之间的相互转换：

进制	基数	基数个数	进数规律
十进制	0、1、2、3、4、5、6、7、8、9	10	逢十进一
二进制	0、1	2	逢二进一
八进制	0、1、2、3、4、5、6、7	8	逢八进一
十六进制	0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F	16	逢十六进一

(一) 十进制数与为二进制数、八进制数、十六进制数相互转换的方法：

二进制数、八进制数、十六进制数转换为十进制数的方法：按权展开求和法

1．二进制与十进制间的相互转换：

（1）二进制转十进制： “按权展开求和”

例如： $(1011.01)_2 = 1\times2^3+0\times2^2+1\times2^1+1\times2^0+0\times2^{-1}+1\times2^{-2}$

$= 8+0+2+1+0+0.25$

$= 11.25$

规律：个位上的数字的次数是 0，十位上的数字的次数是 1，… …，依次递增，而十分位的数字的次数是-1，百分位上数字的次数是-2，……，依次递减。

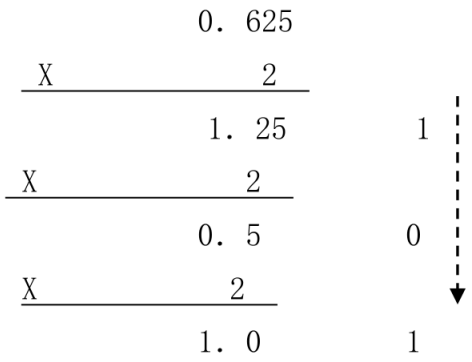
（2）十进制整数转二进制： “除以 2 取余，逆序输出” （短除反取余法）

例如： $(89)_{10} = (1011001)_2$

```
2 | 89
2 | 44   ..... 1
2 | 22   ..... 0
2 | 11   ..... 0
2 |  5   ..... 1
2 |  2   ..... 1
2 |  1   ..... 0
    0    ..... 1
```

（3）十进制小数转二进制： “乘以2取整，顺序输出”

例如： $(0.625)_{10} = (0.101)_2$



(二)八进制与二进制的转换：

二进制数转换成八进制数：从小数点开始，整数部分向左、小数部分向右，每 3 位为一组，用一位八进制数的数字表示，不足 3 位的要用“0”补足 3 位，就得到一个八进制数。

八进制数转换成二进制数：把每一个八进制数转换成 3 位的二进制数，就得到一个二进制数。

例 1：将八进制的 37.416 转换成二进制数：

3 7 . 4 1 6

011 111 . 100 001 110

即：$(37.416)_8 = (11111.10000111)_2$

例 2：将二进制的 10110.0011 转换成八进制：

0 1 0 1 1 0 . 0 0 1 1 0 0

2 6 . 1 4

即：$(10110.011)_2 = (26.14)_8$

(三) 十六进制与二进制的转换：

二进制数转换成十六进制数：从小数点开始，整数部分向左、小数部分向右，每 4 位为一组用一位十六进制数的数字表示，不足 4 位的要用“0”补足 4 位，就得到一个十六进制数。

十六进制数转换成二进制数：把每一个八进制数转换成 4 位的二进制数，就得到一个二进制数。

例 1：将十六进制数 5DF.9 转换成二进制：

5 D F . 9

0101 1101 1111 . 1001

即：$(5DF.9)_{16} = (10111011111.1001)_2$

例 2：将二进制数 1100001.111 转换成十六进制：

0110 0001 . 1110

6 1 . E

即：$(1100001.111)_2 = (61.E)_{16}$

注意：以上所说的二进制数均是无符号的数。这些数的范围如下表：

无符号位 二进制数位数	数值范围	十六进制范围表示法
8 位二进制数	0~255 （$255=2^8-1$）	00~0FFH
16 位二进制数	0~65535 （$65535=2^{16}-1$）	0000H~0FFFFH
32 位二进制数	$0\sim2^{32}-1$	00000000H~0FFFFFFFFH

二、 信息的编码

(一) 带符号数的编码

带符号二进制数用最高位的一位数来表示符号：0 表示正，1 表示负。 带符号数的编码方法有原码、反码和补码。

（1）原码表示法：

一个机器数 x 由符号位和有效数值两部分组成，设符号位为 x_0，x真值的绝对值 $|x|=x_1x_2x_3...x_n$，则 x 的机器数原码可表示为：

$[x]_{原}= x_0x_1x_2...nx$，当 x>=0 时，$x_0=0$，当 x<0 时，$x_0=1$。

例如：已知：x_a=-1011，x_b= +1001，则 x_a，x_b的原码分别是

$[x_a]_{原}$=11011，$[x_b]_{原}$=01001

规律：正数的原码是它本身，负数的原码是取绝对值后，在最高位（左端）补“1”。

（2）反码表示法：

一个负数的原码符号位不变，其余各位按位取反就是机器数的反码表示法。正数的反码与原码相同。

按位取反的意思是该位上是 1 的，就变成 0，该位上是 0 的就变成 1。即 1→0，0→1 。

例：x_a=−1011，x_b=+1001，求$[x_a]_{反}$和$[x_b]_{反}$。

解：$[x_a]_{反}$=10100 ，$[x_b]_{反}$=01001

（3）补码表示法：

a) 正数的补码表示与原码相同；

b) 负数的补码是将原码符号位保持“1”之后，其余各位按位取反，末位再加 1 便得到补码，即取其原码的反码再加“1”：$[x]_{补}=[x]_{反}+1$。

例如：

真值	原码（B）	反码（B）	补码（B）	补码（H）
+127	0 111 1111	0 111 1111	0 111 1111	7F
+39	0 010 0111	0 010 0111	0 010 0111	27
+0	0 000 0000	0 000 0000	0 000 0000	00
-0	1 000 0000	1 111 1111	0 000 0000	00
-39	1 010 0111	1 101 1000	1 101 1001	D9
-127	1 111 1111	1 000 0000	1 000 0001	81
-128	无法表示	无法表示	1 000 0000	80

从上可看出，真值+0 和-0 的补码表示是一致的，但在原码和反码表示中具有不同形式。

8 位补码机器数可以表示-128，但不存在+128 的补码与之对应，由此可知，8 位二进制补码能表示数的范围是-128——+127。还要注意，不存在-128 的 8 位原码和反码形式。

(二) 字符的表示

（1）ASCII 码

（American Standard Code for Information Interchange）美国标准信息交换代码将每个字符用 7 位的二进制数来表示，共有 128 种状态 。这里的“字符”包括大小字母、0...9、其它符号、控制符 。常用字符的ASCII码为：‘0’ — 48 、‘A’ — 65 、‘a’ — 97 。

三、 例题

1. 十进制数 11/128 可用二进制数码序列表示为(D)。

   A)1011/1000000 B)1011/100000000 C) 0.001011 D) 0.0001011

2. 算式$(2047)_{10}$ - $(3FF)_{16}$ + $(2000)_8$ 的结果是(A)。

   A)$(2048)_{10}$ B)$(2049)_{10}$ C) $(3746)_8$ D) $(1AF7)_{16}$

3. 已知 x=(0.1011010)2,则[x/2] =(C)$_2$。

   A) 0.1011101. B) 11110110 C) 0.0101101 D) 0.100110

4. [x]补码=10011000，其原码为(B)

   A)011001111 B)11101000 C)11100110 D)01100101

5. 十进制算术表达式：3*512+7*64 + 4*8 + 5 的运算结果，用二进制表示为（B）.

   A. 10111100101 B.11111100101 C.111110100101 D.11111101101

6. 十进制数 2004 等值于八进制数（B）。

   A. 3077　B. 3724　C. 2766　D. 4002　E. 3755

7. $(2004)_{10}$ + $(32)_{16}$ 的结果是（D）。

   A. $(2036)_{10}$　B. $(2054)_{16}$　C. $(4006)_{10}$　D. $(100000000110)_{2}$　E. $(2036)_{16}$

8. 十进制数 100.625 等值于二进制数（B）。

   A. 1001100.101　B. 1100100.101　C. 1100100.011　D. 1001100.11

   E. 1001100.01

9. 以下二进制数的值与十进制数 23.456 的值最接近的是（D）。

   A. 10111.0101　B. 11011.1111　C. 11011.0111　D. 10111.0111

   E. 10111.1111

3、计算机安全

计算机安全(computer security)是指防范与保护计算机系统及其信息资源在生存过程中免受蓄意攻击、人为失误和自然灾害等引起的损失和破坏。

计算机病毒是一种特殊的程序，有着与生物病毒极为相似的特点：

一是寄生性，它们大多依附在别的程序上面。

二是隐蔽性，它们是悄然进入系统的，人们很难察觉。

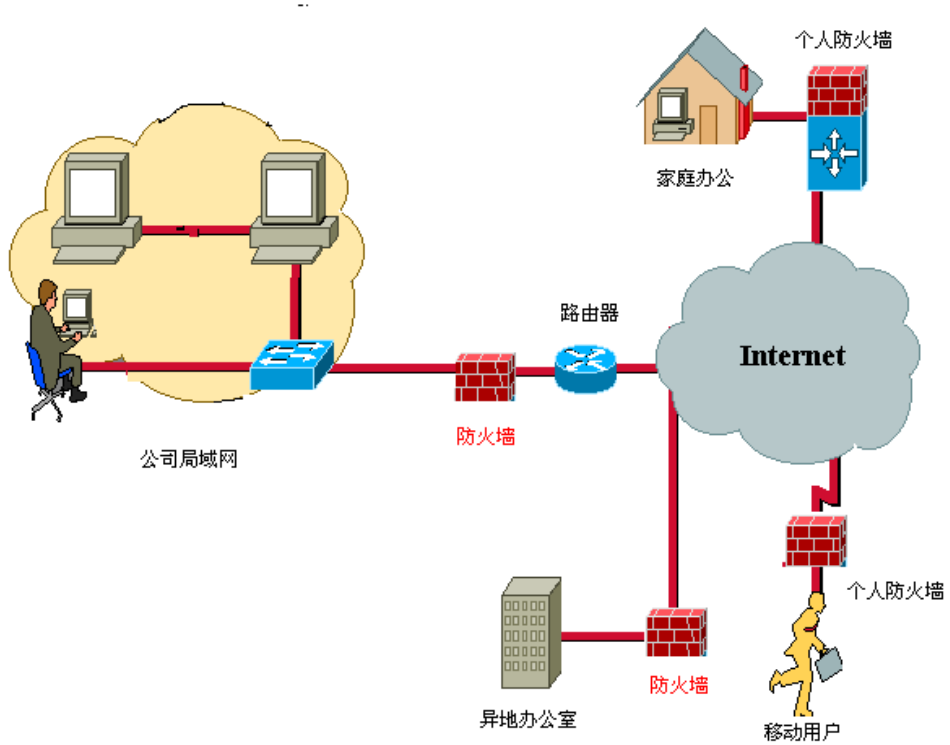
三是潜伏性，它们通常是潜伏在计算机程序中，只在一定条件下才发作的。

四是传染性，它们能够自我复制繁殖，通过传输媒介蔓延。

五是破坏性，轻则占用一定数量的系统资源，重则破坏整个系统。

对于计算机病毒，我们不必谈虎变色，而应采取积极的防治态度。首先，要防止“病从口入”，不随意下载和安装软件。另外，要用优秀的防杀病毒软件，对外来的软件和资料要进行严格的检查和杀毒。注意，防杀病毒软件需要及时更新(主要是其中的数据文件)，一般每周一次，不更新基本上等于没有防杀毒功能。

要防止“黑客”攻击，主要方法是加强安全措施，例如设置防火墙。防火墙是一种计算机设备，它设置在内部网络与外部网络之间，起一个隔离的作用，既可以阻止外部信息非法进入内部系统，也可以阻止内部人员非法访问外部系统。



例题

1. 计算机病毒传染的必要条件是(B)。

A)在内存中运行病毒程序　　B)对磁盘进行读写操作

C)在内存中运行含有病毒的程序　D) 复制文件

解析：只有B是外部信息进入计算机的必要操作。

2. 计算机病毒是(B)

A)通过计算机传播的危害人体健康的一种病毒

B)人为制造的能够侵入计算机系统并给计算机带来故障的程序或指令集合

C)一种由于计算机元器件老化而产生的对生态环境有害的物质

D)利用计算机的海量高速运算能力而研制出来的用于疾病预防的新型病毒

3. 计算机病毒的特点是（ C ）

A. 传播性、潜伏性、易读性与隐蔽性　B. 破坏性、传播性、潜伏性与安全性

C. 传播性、潜伏性、破坏性与隐蔽性　D. 传播性、潜伏性、破坏性与易读性

4、计算机网络

一、 计算机网络概述

计算机网络是指将地理位置不同的具有独立功能的多台计算机及其外部设备，通过通信线路连接起来，在网络操作系统，网络管理软件及网络通信协议的管理和协调下，实现资源共享和信息传递的计算机系统。

1. 网络的主要功能

1）资源共享（最主要） 2）信息传输 3）分步处理 4）综合信息服务

2. 计算机网络的分类

从地理范围的角度可以把各种网络类型划分为局域网、城域网和广域网。

1) 局域网 LAN(Local Area Network)

局域网地理范围一般几百米到 10km 之内,属于小范围内的连网。如一个建筑物内、一个学校内、一个工厂的厂区内等。

2) 城域网 MAN(Metropolitan Area Network)

城域网地理范围可从几十公里到上百公里,可覆盖一个城市或地区。

3) 广域网 WAN(Wide Area Network)

广域网地理范围一般在几千公里左右,属于大范围连网。如几个城市,一个或几个国家,是网络系统中的最大型的网络,能实现大范围的资源共享,如国际性的 Internet 网络。

3. 常见的计算机网络的性能指标

（1）速率

计算机发送出的信号都是数字形式的。比特是计算机中数据量的单位，一个比特就是二进制数字中的一个1或0。速率是计算机网络中最重要的一个性能指标。速率的单位是bit/s（比特每秒）、Kb/s(千比特每秒)、Mb/s（兆比特每秒）。

（2）带宽

网络带宽表示在单位时间内从网络中的某一点到另一点所能通过的“最高数据率”。 下方图示中网络的带宽只有1~2Mb/s 。

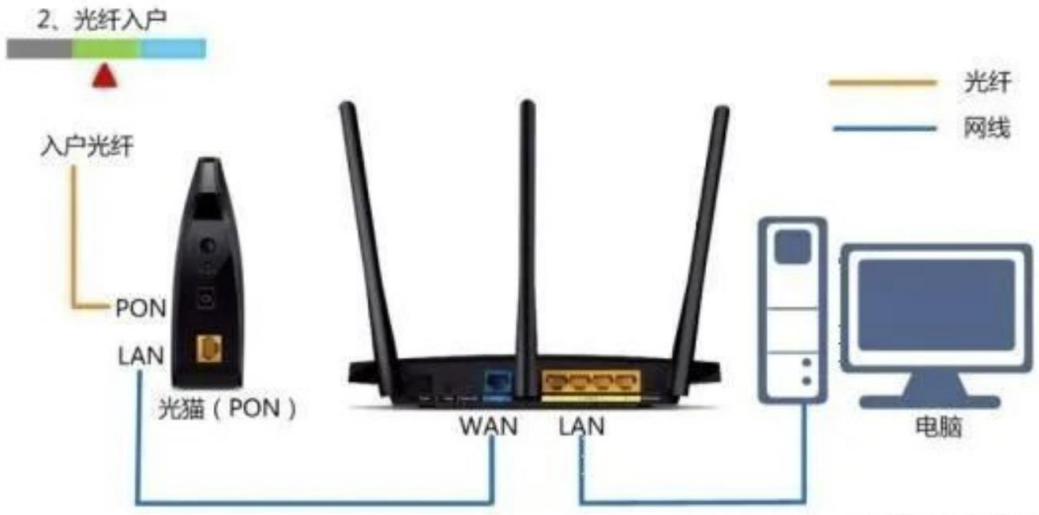
4. 网络的传输介质

1) 有线网

常用的有线传输介质有双绞线、同轴电缆和光导纤维（光纤）。

- ✧ 双绞线俗称网线，市场上主流有五类和6类，6类比较贵，当然屏蔽性较好。这种网线在塑料绝缘外皮里面包裹着8根信号线，它们每两根为一对相互缠绕，总共形成四对，双绞线也因此得名。双绞线这样互相缠绕的目的就是利用铜线中电流产生的电磁场互相作用抵消邻近线路的干扰并减少来自外界的干扰。双绞线传输速率在10Mbps 到600Mbps 之间,双绞线电缆的连接器一般为 RJ-45。

- ✧ 同轴电缆的优点是可以在相对长的无中继器的线路上支持高带宽通信，而其缺点也是显而易见的：一是体积大，细缆的直径就有3/8英寸粗，要占用电缆管道的大量空间；二是不能承受缠结、压力和严重的弯曲，这些都会损坏电缆结构，阻止信号的传输；三是成本高。而所有这些缺点正是双绞线能克服的，因此在现在的局域网环境中，基本已被基于双绞线的以太网物理层规范所取代。
- ✧ 光纤（Fiber Optic Cable）以光脉冲的形式来传输信号，因此材质也以玻璃或有机玻璃为主。光纤的优点是不会受到电磁的干扰,传输的距离也比电缆远，传输速率高。光缆的安装和维护比较困难,需要专用的设备。



2) 无线网

无线网络上网可以简单的理解为无线上网，几乎所有智能手机、平板电脑和笔记本电脑都支持Wi-Fi上网，是当今使用最广的一种无线网络传输技术。但是Wi-Fi信号也是由有线网提供的，比如家里的ADSL，小区宽带等，只要接一个无线路由器，就可以把有线信号转换成Wi-Fi信号。

3) 5G与Wi-Fi的区别

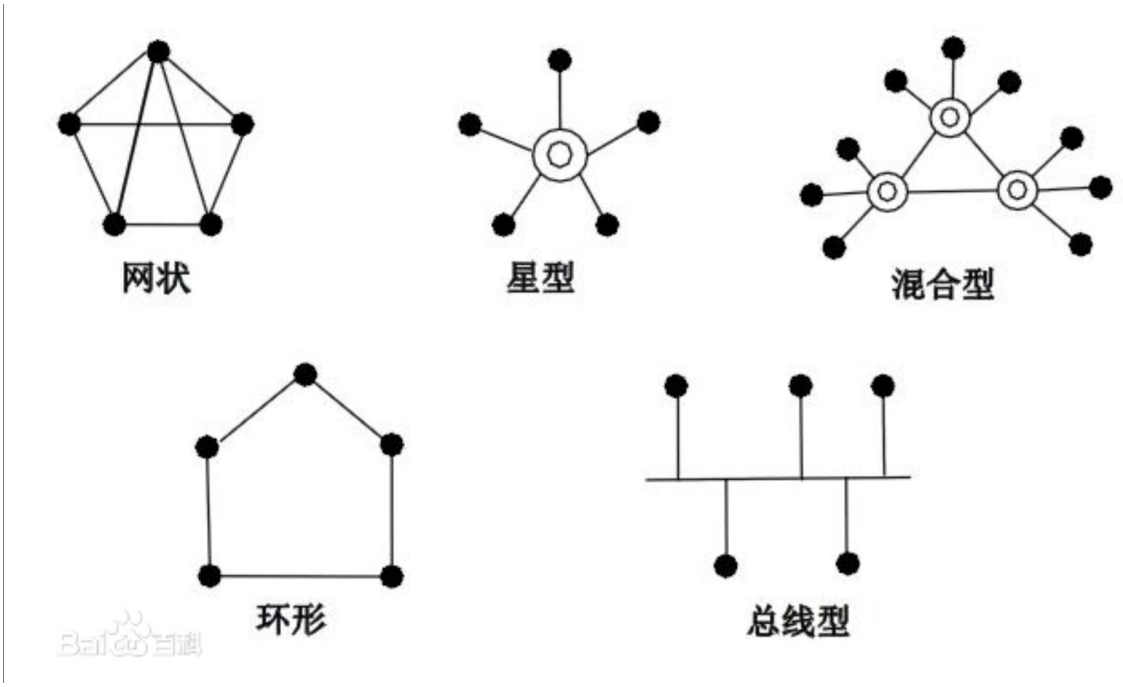
5G和WIFI的最根本的区别：5G是广域网技术，而Wi-Fi是局域网技术。

5G是第五代移动通信技术，5G之前有1G、2G、3G、4G。5G技术广域覆盖由宏基站来完成，室内部分由小基站和5G室内分布系统完成。小基站是5G里对抢占WiFi市场的一个利器，但是价格比较贵。

WiFi是无线局域网技术，无法有效地解决移动性的问题，也无法获得授权频谱去进行广域覆盖，所以应用场景也主要是一些室内场景，优点是价格便宜。

5. 网络的拓扑结构

计算机网络中常用的拓扑结构有总线型、星型、环型等。

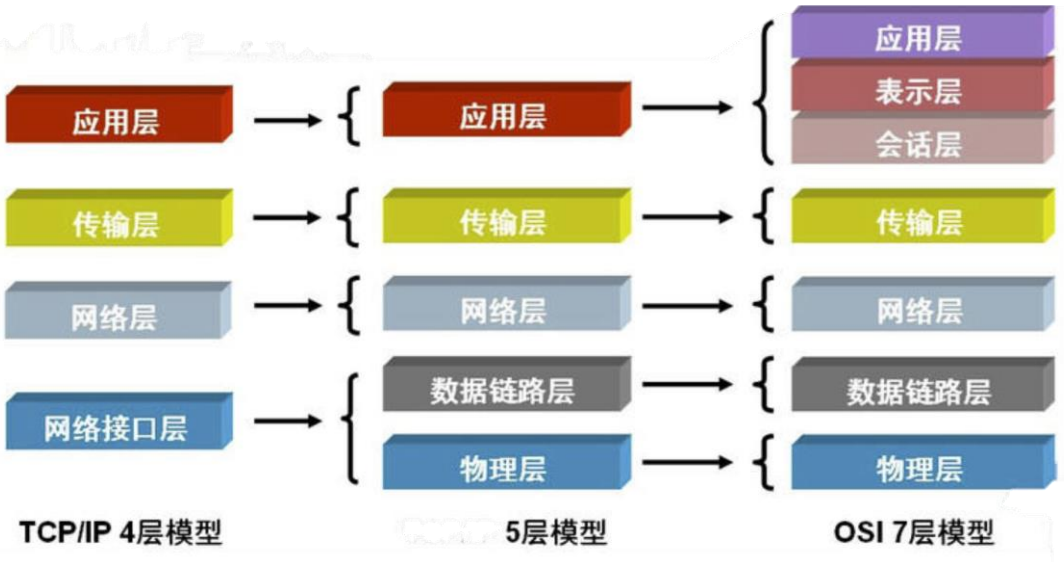


二、 网络的体系结构和协议

1. 网络的体系结构

为了使不同计算机厂家生产的计算机能够相互通信，以便在更大的范围内建立计算机网络，国际标准化组织（ISO）在1978年提出了"开放系统互联参考模型"，即著名的OSI/RM模型（Open System Interconnection/Reference Model）。

除了标准的OSI七层模型以外，常见的网络层次划分还有TCP/IP四层协议以及TCP/IP五层协议，它们之间的对应关系如下图所示。



2. 常见网络协议

1) 网络通信协议：TCP/IP，包括IP(Internet Protocol)和TCP(Transmission Control Protocol)

2) 常用的电子邮件协议有SMTP（Simple Mail Transfer Protocol，即简单邮件传输协议）、POP3（Post Office Protocol 3的简称，即邮局协议的第3个版本）、IMAP（Internet Mail Access Protocol，即交互式邮件存取协议），其中发邮件用SMTP，收邮件用POP3或IMAP 。

3) 文件传输协议（FTP）

3. IP地址

IP地址是IP协议提供的一种统一的地址格式。在目前使用的IPv4协议下，IP地址是一个32位二进制码。 Internet的IP地址分五类ABCDE，常用类有ABC。

类别	最大网络数	IP地址范围	单个网段最大主机数	私有IP地址范围
A	126(2^7-2)	1.0.0.1-127.255.255.254	16777214	10.0.0.0-10.255.255.255
B	16384(2^14)	128.0.0.1-191.255.255.254	65534	172.16.0.0-172.31.255.255
C	2097152(2^21)	192.0.0.1-223.255.255.254	254	192.168.0.0-192.168.255.255

初赛中常让我们判断IP地址的正确性，注意：IP地址中不能以十进制“127”作为开头，该类地址中数字127．0．0．1到127．255．255．255用于回路测试，如：127.0.0.1可以代表本机IP地址，用“http://127.0.0.1”就可以测试本机中配置的Web服务器。后面的三个数字可以为0，最大数值255。C类地址第一个数字最大只到223。

未来的IPV6协议中，IP地址长度为128位。

4. 域名

由于IP地址具有不方便记忆并且不能显示地址组织的名称和性质等缺点，人们设计出了域名，并通过网域名称系统（DNS，Domain Name System）来将域名和IP地址相互映射。

- 顶级域名有三类：
  - 国家顶级域名,如 cn（中国）、us（美国）、uk（英国）；
  - 国际顶级域名—— int，国际性组织可在 int 下注册；
  - 通用顶级域名，如：com、net、edu、gov、org、……

5. 常见名字翻译
   1) WWW（World Wide Web）：万维网。
   2) URL（Uniform Resource Locator）：统一资源定位器。
   3) HTTP（Hypertext Transfer Protocol）：超文本传输协议。
   4) FTP（File Transfer Protocol）：文件传输协议。
   5) TCP（Transfer Control Protocol）：传输控制协议。
   6) Internet：因特网

三、 例题

1. Ipv4 地址是由(B) 位二进制数码表示的。
   A)16 B)32 C) 24 D) 8
2. TCP/IP 协议共有(B)层协议
   A) 3 B) 4 C) 5 D) 6
3. Internet 的规范译名应为（ B ）
   A. 英特尔网 B. 因特网 C. 万维网 D. 以太网
4. 计算机网络是一个（ D ）
   A.管理信息系统 B.管理数据系统
   C.编译系统 D. 在协议控制下的多机互连系统
5. 下面哪些计算机网络不是按覆盖地域划分的（ D ）
   A.局域网 B. 都市网 C.广域网 D. 星型网
6. 下列网络上常用的名字缩写对应的中文解释错误的是（ D ）。
   A、WWW（World Wide Web）：万维网。
   B、URL（Uniform Resource Locator）：统一资源定位器。
   C、HTTP（Hypertext Transfer Protocol）：超文本传输协议。
   D、FTP（File Transfer Protocol）：快速传输协议。
   E、TCP（Transfer Control Protocol）：传输控制协议。
7. 常见的邮件传输服务器使用（B）协议发送邮件。
   A. HTTP B. SMTP C. TCP D. FTP
8. 不能在 Linux 上使用的网页浏览器是（A ）。
   A. Internet Explore B. Netscape C. Opera D. Firefox E. Mozilla
9. 中国的国家顶级域名是（ A ）
   A. .cn

B. .ch

C. .chn

D. .china

5、基本数据类型

表B.1 常用的基本数据类型

数据类型	类型标识符	所占字节数	取值范围
短整型	short[int]	2	-32768~32767
无符号短整型	unsigned short [int]	2	0~65535
整型	int	4	-2147483648~2147483647
无符号整型	unsigned int	4	0~4294967295
长整型	long long	8	-2^{63}~$2^{63}-1$
无符号长整型	unsigned long long	8	0~$2^{64}-1$
单精度浮点数	float	4	-3.4E+38~3.4E+38(7位有效数字)
双精度浮点数	double	8	-1.79E+308~1.79E308(15位有效数字)
高精度浮点数	long double	12	3.4E-4932~1.1E+4932(19位有效数字)
字符型	char	1	-128~127
	signed char	1	0~255
布尔型	bool	1	0或1

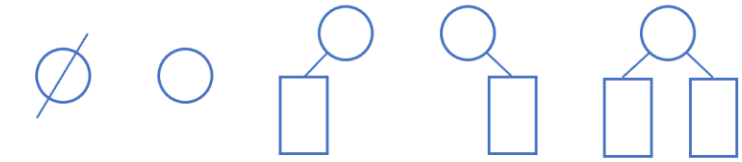
在运算过程中超出对应数据类型的数值范围，会造成数据的溢出（overflow），数据的溢出在编译和运行时并不会报错，但是会造成程序运行结果出错。所以一定要特别细心和认真对待数据类型。

6、数据结构与算法

一、 基本数据结构

1. 线性表：数组、vector， 数据项的物理存放位置与其逻辑次序完全吻合。
2. 链表 ： 数据项的物理位置可连续也可不连续
3. 栈：先进后出
4. 队列：先进先出
5. 二叉树：
   a) 二叉树的基本概念

      二叉树是度为2的树，每个结点最多有两个子结点，分别称为左孩子、右孩子。它的两棵子树分别称为左子树、右子树。二叉树有5种基本形态：



空二叉树　仅有根结点的二叉树　右子树为空的二叉树　左子树为空的二叉树　左、右子树均非空的二叉树

b) 二叉树的性质

1）在二叉树的第i层上最多有2^{i-1} 个节点 。 （i>=1）

2）高度为k的二叉树最多有2^k-1个节点。(k>=1)

3）若用n0表示叶子节点数，n2表示度数为2的节点数，则有n0=n2+1 。

4）在完全二叉树中，具有n个节点的完全二叉树的深度为[$\log_2 n$]+1，其中[log2n]是向下取整。

5）若对含 n 个结点的完全二叉树从上到下且从左至右进行 1 至 n 的编号，则对完全二叉树中任意一个编号为 i 的结点有如下特性：

(1) 若 i=1，则该结点是二叉树的根，无父亲, 否则，编号为 [i/2] 的结点为其父亲结点;

(2) 若 2i>n，则该结点无左孩子， 否则，编号为 2i 的结点为其左孩子结点；

(3) 若 2i+1>n，则该结点无右孩子结点， 否则，编号为2i+1 的结点为其右孩子结点。

c) 二叉树的常见分类：

1）完全二叉树：若设二叉树的高度为h，除第 h 层外，其它各层 (1～h-1) 的结点数都达到最大个数，第h层有叶子结点，并且叶子结点都是从左到右依次排布，这就是完全二叉树。

一维数组可以作为完全二叉树的存储结构，堆排序使用的数据结构就是完全二叉树。

2）满二叉树：除了叶结点外每一个结点都有左右孩子且叶子结点都处在最底层的二叉树。

d) 二叉树的遍历：

1）前序遍历：root -> left -> right

2）中序遍历：left -> root -> right

3）后续遍历：left ->right -> root

4）层序遍历：按照层次遍历

6. 图

a) 图的概念

有向图：图的边有方向，只能按箭头方向从一点到另一点。

无向图：图的边没有方向，可以双向。

结点的度：无向图中与结点相连的边的数目

结点的入度：在有向图中，以这个结点为终点的有向边的数目。

结点的出度：在有向图中，以这个结点为起点的有向边的数目。

权值：边的“费用”

连通：如果图中结点U、V之间存在一条从U通过若干条边、点到达V的通路，则称U、V是连通的。

回路：起点和终点相同的路径，称为回路，或“环”。

完全图：一个含有n个结点的完全无向图含有 n*(n-1)/2 条边；一个含有n个结点的完全有向图含有 n*(n-1) 条边。

稠密图：一个边数接近完全图的图。

稀疏图：一个边数远远少于完全图的图。

强连通分量：有向图中任意两点都连通的最大子图。

b) 图的存储

稠密图：邻接矩阵

稀疏图：邻接表

c) 图的遍历

深度优先遍历和广度优先遍历

7. hash表

采用散列技术将记录存储在一块连续的存储空间中，这块连续空间称为散列表或哈希表(Hash-Table)。常用处理冲突的思路：

1) 换个位置: 开放地址法

2) 同一位置的冲突对象组织在一起: 链地址法

线性探测法属于第1个思路，处理冲突的方式为：以增量序列1、2、 … 、(tablesize – 1) 循环试探下一个存储地址。

二、 算法

1. 算法的概念

算法（algorithm）是指在解决问题时，按照某种机械的步骤一定可以得到问题的结果（有的问题有解，有的没有）的处理过程。算法就是解决这个问题的方法和步骤的描述。

2. 算法的控制结构

顺序结构：各操作依次进行。

选择结构：由条件是否成立来决定选择执行。

循环结构：有些操作要重复执行，直到满足某个条件才结束。

3. 算法的基本性质

1) 有穷性

一个算法所包含的计算步骤是有限的，即算法的每个步骤都能在有限的时间内完成。

2) 确定性

对于每种情况下所应执行的操作，在算法中都有确切的规定，使算法的执行者和阅读者都能明确其含义及如何执行。并且在任何条件下，算法都只有一条执行路径。

3) 可行性

算法中描述的操作都可以通过已经实现的基本操作运算有限次地实现。

4) 输入（算法有零个或者多个输入）

5) 输出（算法有一个或者多个输出）

4. 递推、递归

a) 递推算法

递推算法能通过已知某个条件，利用特定的关系得出中间推论，然后逐步递推，直到得到结果为止。能已知数量关系写递推公式。

b) 递归算法

下面的故事与递归算法有着异曲同工之妙。从前有座山，山里有座庙，庙里有个老和尚在给小和尚讲故事:“从前有座山，山里有座庙，庙里有个老和尚在给小和尚讲故事:‘从前有座山，山里有座庙，庙里有个老和尚给小和尚讲故事.....’”

递归的特点是：函数调用它自己本身。

经典问题：Hanoi汉诺塔问题

5. 排序:

常见的排序算法有：插入排序、选择排序、冒泡排序、归并排序、快速排序、桶排序。

1) 冒泡排序

a) 基本思想：每一趟都依次比较相邻的数据，逆序则交换

```
for(int i=n-1;i>=1;i--)//第i轮排序
{
   for(int j=1;j<=i;j++)
      if(a[j]>a[j+1])//比较相邻元素大小
         swap(a[j],a[j+1]);
}
```

b) 优化:

问题：数据的顺序排好之后，冒泡算法仍然会继续进行下一轮的比较，直到n-1次，浪费时间。

方案：设置标志位flag，如果发生了交换，flag设置为true；如果没有交换就设置为false。这样当一轮比较结束后如果flag仍为false，即：这一轮没有发生交换，说明数据的顺序已经排好，没有必要继续进行下去。

```
for(int i=n-1;i>=1;i--)//第i轮排序
{
   flag=false;
   for(int j=1;j<=i;j++)
      if(a[j]>a[j+1])//比较相邻元素大小
      {
         flag=true;
```

```
            swap(a[j],a[j+1]);
        }
    if(!flag)//没有发生交换，说明已经排好序
        break;
}
```

2) 选择排序

基本思想：每一趟从待排序的数据元素中选出最小（或最大）的一个元素，顺序放在待排序的数列的最前，直到全部待排序的数据元素排完。

```
int MIN, MINA; //最值下标，最值
for(int i=1;i<=n-1;i++)
{
    MIN=i, MINA=a[i];
    for(int j=i+1; j<=n; j++)
        if(a[j]<MINA)
        {
            MIN=j;//找出后面的最小值和最小值的坐标
            MINA=a[j];
        }
    swap(a[i],a[MIN]);//把最小值放在前面
}
```

3) 插入排序

基本思想：当读入一个元素时，在已经排序好的序列中，搜寻它正确的位置，再放入读入的元素。但不该忽略一个重要的问题：在插入这个元素前，应当先将将它后面的所有元素后移一位，以保证插入位置的原元素不被覆盖。

```
for(int i=1; i<=n; i++)
{
    int j;
    for(j=i-1; j>=1; j--)
        if(a[j]<a[i])//找到第一个比a[i]小的元素位置j，插入位置为j+1
            break;
    if(j!=i-1)
    {
        tmp=a[i];//存储插入元素的值
        for(int k=i-1;k>=j+1;k--)//j+1到i-1的元素后移
            a[k+1]=a[k];
        a[j+1]=tmp;//插入
    }
```

```
}
```

4) 桶排序

基本思想：若待排序的值在一个明显有限范围内(整型)时，可设计有限个有序桶，待排序的值装入对应的桶（当然也可以装入若干个值），桶号就是待排序的值，顺序输出各桶的值，将得到有序的序列。

```
for(int i=1;i<=n;i++)
{
    cin>>num;
    a[num]++;//把相对应的整数放在对应的桶中
    MAXA=max(MAXA, num);//求出所有桶中数的最大值，排除空桶
}
for(int i=0;i<=MAXA;i++)
{
    while(a[i]!=0)
    {
        cout<<i <<" ";
        a[i]--;
    }
}
```

5) 快速排序

基本思想：快速排序是对冒泡排序的一种改进。它的基本思想是，通过一趟排序将待排记录分割成独立的两部分，其中一部分记录的关键字均比另一部分记录的关键字小，则可分别对这两部分记录继续进行排序，以达到整个序列有序。

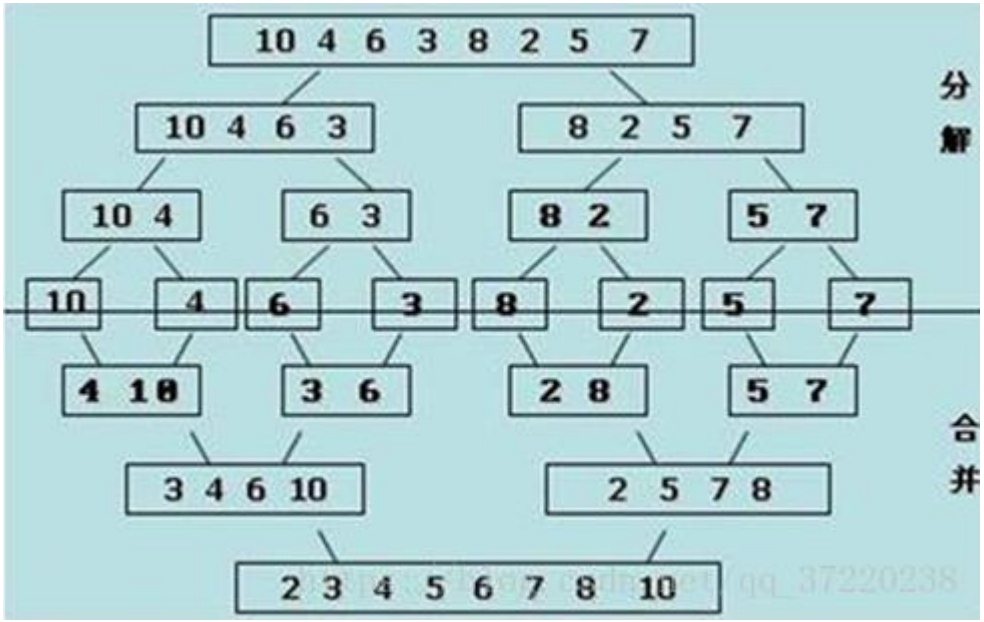
```
void quicksort(int left, int right)
{
    int i,j,t,temp;
    if(left>right)   return;
    temp=a[left]; //temp中存的就是基准数
    i=left;
    j=right;
    while(i!=j)
    {
                //顺序很重要，要先从右边开始找
                while(a[j]>=temp && i<j)    j--;
                //再找左边的
                while(a[i]<=temp && i<j)    i++;
```

```
            //交换两个数在数组中的位置
            if(i<j)  swap(a[i], a[j]);
        }
        //最终将基准数归位
        a[left]=a[i];
        a[i]=temp;

        quicksort(left,i-1);//继续处理左边的，这里是一个递归的过程
        quicksort(i+1,right);//继续处理右边的 ，这里是一个递归的过程
    }
```

6) 归并排序

基本思想：采用分治法（Divide and Conquer）将未排序序列分解成两部分，先使每个子序列有序，再将两个有序子序列二路归并成一个有序表。




```
void msort(int l,int r)
{
    if(l==r)//递归出口
        return;
    int mid=(l+r)/2;
    msort(l,mid);//左边序列
    msort(mid+1,r);//右边序列
    int i=l,j=mid+1,k=l;
    while(i<=mid&&j<=r)//左右序列进行合并
    {
        if(a[i]<=a[j])//把小的元素放在前面
            b[k]=a[i++];
```

```
            else
                b[k]=a[j++];
            k++;
        }
        while(i<=mid)//如果有剩余序列，直接加在辅助数组最后面
            b[k++]=a[i++];
        while(j<=r)
            b[k++]=a[j++];
        for(int i=l;i<=r;i++)//赋值已经排好序的序列给a数组
            a[i]=b[i];
    }
```

7) 各种排序算法的比较

排序算法	平均时间复杂度	最好情况	最坏情况	空间复杂度	排序方式	稳定性
冒泡排序	$O(n^2)$	$O(n)$	$O(n^2)$	$O(1)$	In-place	稳定
选择排序	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(1)$	In-place	不稳定
插入排序	$O(n^2)$	$O(n)$	$O(n^2)$	$O(1)$	In-place	稳定
希尔排序	$O(n \log n)$	$O(n \log^2 n)$	$O(n \log^2 n)$	$O(1)$	In-place	不稳定
归并排序	$O(n \log n)$	$O(n \log n)$	$O(n \log n)$	$O(n)$	Out-place	稳定
快速排序	$O(n \log n)$	$O(n \log n)$	$O(n^2)$	$O(\log n)$	In-place	不稳定
堆排序	$O(n \log n)$	$O(n \log n)$	$O(n \log n)$	$O(1)$	In-place	不稳定
计数排序	$O(n + k)$	$O(n + k)$	$O(n + k)$	$O(k)$	Out-place	稳定
桶排序	$O(n + k)$	$O(n + k)$	$O(n^2)$	$O(n + k)$	Out-place	稳定
基数排序	$O(n \times k)$	$O(n \times k)$	$O(n \times k)$	$O(n + k)$	Out-place	稳定

说明：

- ✓ n：数据规模
- ✓ k："桶"的个数
- ✓ In-place：不占用额外内存
- ✓ Out-place：占用额外内存
- ✓ 稳定性：排序后 2 个相等键值的顺序和排序之前它们的顺序相同

6. 高精度计算

基本思想：用字符串/数组存储大数，按照数学竖式的计算过程进行处理。注意处理负号、前导零和进位

1) 高精度数读入

```
int a[N+1],i;
string s1;
cin>>s1;//数s1
memset(a,0,sizeof(a)); //数组清0
a[0]=s1.length(); //位数
for(i=1;i<=a[0];i++) a[i]=s1[a[0]-i]-'0';//将字符转为数字并倒序存储
```

2) 高精度数比较

```
//比较a和b的大小关系，若a>b则为1，a<b则为-1,a=b则为0
int compare(int a[],int b[])
{
    int i;
    if (a[0]>b[0]) return 1;//a的位数大于b则a比b大
    if (a[0]<b[0]) return -1;//a的位数小于b则a比b小
    for(i=a[0]; i>0; i--) //从高位到低位比较
    {
            if (a[i]>b[i])return 1;
            if(a[i]<b[i]) return -1;
    }
    return 0;//各位都相等则两数相等
}
```

3) 高精度加法

```
int plus(int a[],int b[]) //计算a=a+b
{
    int i,k;
    k=a[0]>b[0]?a[0]:b[0]; //k是a和b中位数最大的一个的位数
    for(i=1; i<=k; i++)
    {
            a[i+1]+=(a[i]+b[i])/10;  //若有进位，则先进位
            a[i]=(a[i]+b[i])%10; //计算当前位数字,注意：这条语句与上一条不能交换
    }
    if(a[k+1]>0) a[0]=k+1;  //修正新的a的位数（a+b最多只能的一个进位）
    else a[0]=k;
    return 0;
}
```

4) 高精度减法

```
int gminus(int a[],int b[]);//计算 a=a-b，返加符号位 0:正数 1:负数
{
    int flag,i
```

```
        flag=compare(a,b); //调用比较函数判断大小
        if (falg==0)// 若 a=b，则 a=0,也可在 return 前加一句 a[0]=1,表示是 1 位数 0
        {
                memset(a,0,sizeof(a));
                return0;
        }
        if(flag==1) //大于
        {
                for(i=1; i<=a[0]; i++) //若不够减则向上借一位
                {
                        if(a[i]<b[i])
                        {
                                a[i+1]--;
                                a[i]+=10;
                        }
                        a[i]=a[i]-b[i];
                }
                while(a[a[0]]==0) a[0]--; //修正 a 的位数
                return 0;
        }
        if (flag==-1)//小于  则用 a=b-a,返回-1
        {
                for(i=1; i<=b[0]; i++)
                {
                        if(b[i]<a[i])
                        {
                                b[i+1]--;    //若不够减则向上借一位
                                b[i]+=10;
                        }
                        a[i]=b[i]-a[i];
                }
                a[0]=b[0];
                while(a[a[0]]==0)a[0]--; //修正 a 的位数
                return -1;
        }
}
```

5) 高精度乘法

```
cin>>str1;//输入乘数str1
cin>>str2;//输入乘数str2
int i;
lena=strlen(str1);
lenb=strlen(str2);
for(i=0;i<=lena-1;i++)//乘数str1存入数组a
    a[lena-i]=str1[i]-'0';
for(i=0;i<=lenb-1;i++)//乘数str2存入数组b
    b[lenb-i]=str2[i]-'0';

for(i=1;i<=lenb;i++)
{
    x=0;//用于存放进位
    for(j=1;j<=lena;j++)//对乘数每一位进行处理
    {
        c[i+j-1]=a[j]*b[i]+x+c[i+j-1];//当前乘积+上次乘积进位+原数
        x=c[i+j-1]/10;
        c[i+j-1]%=10;
    }
    c[i+lena]=x;//进位
}
lenc=lena+lenb;
while((c[lenc]==0)&&(lenc>1))//删除前导0
    lenc--;
for(i=lenc;i>=1;i--)//倒序输出
    cout<<c[i];
```

6) 高精度除法（使用减法实现除法运算）

```
//减法
void jian(int a[],int b[])
{
    int flag=compare(a,b);
    if(flag==0)
    {
            a[0]=0;
            return;
    }
    if(flag==1)
```

```
        {
                for(int i=1; i<=a[0]; i++)
                {
                        if(a[i]<b[i])
                        {
                                a[i+1]--;
                                a[i]+=10;
                        }
                        a[i]-=b[i];
                }
                while(a[0]>0&&a[a[0]]==0)        a[0]--;
                return;
        }
        return;
}
//复制函数
void numcpy(int p[],int q[],int det)
{
        for(int i=1; i<=p[0]; i++)
        {
                q[i+det-1]=p[i];
        }
        q[0]=p[0]+det-1;
        return;
}
//除法
void chugao(int a[],int b[],int c[])
{
        int tmp[1001];
        c[0] = a[0]-b[0]+1;// m-n+1
        for(int i=c[0]; i>0; i--)
        {
                memset(tmp,0,sizeof(tmp));
                numcpy(b,tmp,i);
                while(compare(a,tmp)>=0)
                {
                        jian(a,tmp);
                        c[i]++;
```

```
            }
        }
        while(c[0]>0 && c[c[0]]==0)                c[0]--;
        return;
    }
```

7. 搜索与回溯

1) 什么是搜索?

从问题的初始状态出发，根据其中的约束条件，按照一定的策略，有序推进，不断深入，对达到的所有目标状态一一验证，找到符合条件的解（可行解），或者找出所有可行解中的最优解。

常见的搜索算法有深度优先搜索和广度优先搜索。其中深度优先搜索会使用回溯法控制搜索过程。

2) 什么是回溯

“回溯法”也称“试探法”。它是从问题的某一状态出发，不断“试探”着往前走一步，当一条路走到“尽头”，不能再前进（拓展出新状态）的时候，再倒回一步或者若干步，从另一种可能的状态出发，继续搜索，直到所有的“路径（状态）”都一一试探过。这种不断前进、不断回溯，寻找解的方法，称为“回溯法”。

3) 搜索跟回溯的关系

深度优先搜索需要控制如何实现状态之间的转移（拓展），回溯法就是深度优先搜索的一种控制策略。

4) 深度优先搜索

深度优先搜索是一个针对图和树的遍历算法，英文缩写为DFS即Depth First Search。深度优先搜索是图论中的经典算法，利用深度优先搜索算法可以产生目标图的相应拓扑排序表，利用拓扑排序表可以方便的解决很多相关的图论问题，如最大路径问题等等。其过程简要来说是对每一个可能的分支路径深入到不能再深入为止，而且每个节点只能访问一次。

5) 深度优先搜索算法框架

a) 框架1

```
int dfs(int k)
{
    for (i=1; i<=算符种数; i++)
        if (满足条件)
        {
            保存结果
            if (到目的地) 输出解;
            else dfs(k+1);
            恢复：保存结果之前的状态{回溯一步}
```

```
                }

        }
```

b) 框架2

```
int dfs(int k)
{
        if (到目的地) 输出解;
        else
                for (i=1; i<=算符种数; i++)
                        if (满足条件)
                        {
                                保存结果;
                                dfs(k+1);
                                恢复：保存结果之前的状态{回溯一步}
                        }

        }
```

6) 广度优先搜索

广度优先搜索是连通图和树的一种遍历算法。其别名又叫BFS，属于一种盲目搜寻法，目的是系统地展开并检查图(树)中的所有节点，以找寻结果。基本过程，BFS是从根节点开始，沿着树(图)的宽度遍历树(图)的节点。如果所有节点均被访问，则算法中止。一般用队列数据结构来辅助实现BFS算法。Dijkstra单源最短路径算法和Prim最小生成树算法都采用了和宽度优先搜索类似的思想。

```
void BFS()
{
        队列初始化;
        初始结点入队;

        while (队列非空)
        {
                队头元素出队，赋给current;
                while (current 还可以扩展)
                {
                        由结点current扩展出新结点new;
                        if (new 重复于已有的结点状态) continue;
                        new结点入队;
                        if (new结点是目标状态)
```

```
                    {
                        置flag= true;
                        break;
                    }
                }
            }
        }
```

8. 贪心

1) 基本概念

所谓贪心算法是指在对问题求解时，总是做出在当前看来是最好的选择。也就是说，不从整体最优上加以考虑，他所作出的仅是在某种意义上的局部最优解。

注意：贪心策略适用的前提是局部最优策略能导致产生全局最优解。所以，要先证明贪心策略的正确性。一般用反例法证明。

2) 基本思路

a) 建立数学模型来描述问题。

b) 把求解的问题分成若干个子问题

c) 对每一个子问题求解，得到子问题的局部最优解。

d) 把子问题的局部最优解组合成原问题的一个可行解。

9. 分治

所谓分治就是分而治之，即将较大规模的问题分解成几个较小规模的问题，通过对较小规模问题的求解达到对整个问题的求解。当将问题分解成两个较小问题求解时，称为二分法。二分查找、归并排序、快速排序都是二分法的应用。

较大幂次数的取余运算也采用了分治法：比如求 b^p mod k的值。

$b^p = b^{(2*p/2+m)}$ ，当p是偶数时，m = 0， 当 p是奇数时， m = 1

10. 动态规划

1) 什么是动态规划?

动态规划是解决多阶段决策最优化问题的一种思想方法。所谓“动态”，指的是在问题的多阶段决策中，按某一顺序，根据每一步所选决策的不同，将随即引起状态的转移，最终在变化的状态中产生一个决策序列。动态规划就是为了使产生的决策序列在符合某种条件下达到最优。

2) 动态规划中的主要概念，名词术语

a) 阶段：把问题分成几个相互联系的有顺序的几个环节，这些环节即称为阶段。

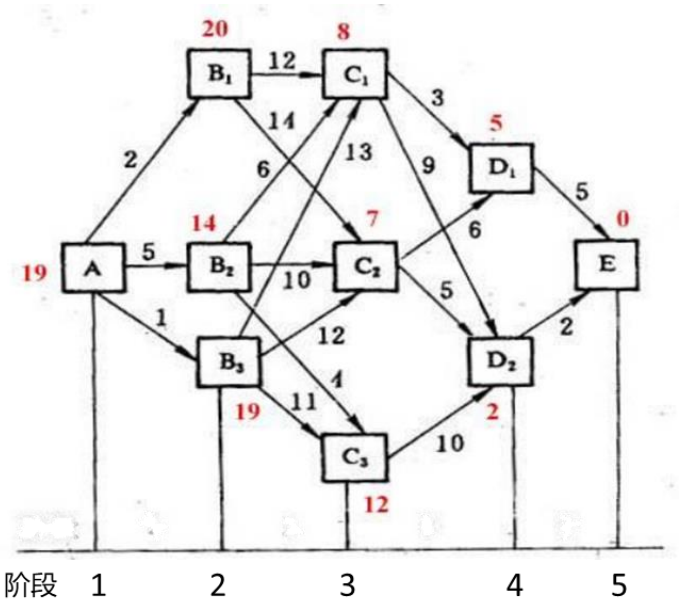
b) 状态：某一阶段的出发位置称为状态。通常一个阶段包含若干状态。如下图中，阶段3就有三个状态结点C_1、C_2、C_3。

c) 决策：从某阶段的一个状态演变到下一个阶段某状态的选择。

d) 策略：由开始到终点的全过程中，由每段决策组成的决策序列称为全过程策略，简称策略。

e) 状态转移方程：前一阶段的终点就是后一阶段的起点，前一阶段的决策选择导出了后一阶段的状态，这种关系描述了由k阶段到k+1阶段状态的演变规律，称为状态转移方程。

f) 目标函数与最优化概念：目标函数是衡量多阶段决策过程优劣的准则。最优化概念是在一定条件下找到一个途径，经过按题目具体性质所确定的运算以后，使全过程的总效益达到最优。



3) 运用动态规划需符合的条件

任何思想方法都有一定的局限性，超出了特定条件，它就失去了作用。同理，动态规划也并不是万能的。那么使用动态规划必须符合什么条件呢？必须满足最优化原理和无后效性。

a) 最优化原理：一个最优化策略的子策略总是最优的。

b) 无后效性

“过去的步骤只能通过当前状态影响未来的发展，当前的状态是历史的总结”。这条特征说明动态规划只适用于解决当前决策与过去状态无关的问题。状态，出现在策略任何一个位置，它的地位相同，都可实施同样策略，这就是无后效性的内涵。

4) 动态规划的计算方法

对于一道题，怎样具体运用动态规划方法呢？

（1）首先，分析题意，考察此题是否满足最优化原理与无后效性两个条件。

（2）接着，确定题中的阶段，状态，及约束条件。

（3）推导出各阶段状态间的状态转移方程，进行计算。

动态规划的常见实现方式：记忆化搜索、递推+填表

5) 经典问题：

a) 01背包

- ✧ 题目描述：有N件物品和一个容量为V的背包，第i（i从1开始）件物品的重量为w[i]，价值为v[i]。物品不可以被分割。在总重量不超过背包承载上限M的情况下，能够装入背包的最大价值是多少?
- ✧ 代码模板：

```
for(int i=1; i<=n; ++i)
{
        for(int j=m; j>=w[i]; j--)
                f[j]=max(f[j],f[j-w[i]]+v[i]);
}
```

b) 完全背包

- ✧ 题目描述：有N种物品和一个容量为V的背包，每种物品都有无限件。第i（i从1开始）种物品的重量为w[i]，价值为v[i]。物品不可以被分割。在总重量不超过背包承载上限M的情况下，能够装入背包的最大价值是多少?
- ✧ 代码模板：

```
for(int i=1; i<=n; ++i)
{
        for(int j=w[i]; j<=m; j++)
                f[j]=max(f[j],f[j-w[i]]+v[i]);
}
```

c) 多重背包

- ✧ 题目描述：有N种物品和一个容量为V的背包。第i（i从1开始）种物品有n[i]件、重量为w[i]，价值为v[i]。物品不可以被分割。在总重量不超过背包承载上限M的情况下，能够装入背包的最大价值是多少?
- ✧ 思路：读入数据时，对物品数量 c 进行二进制优化，转化为01背包问题
- ✧ 代码模板：

```
//二进制优化：多重背包转换为01背包
int t=1;
while(c>=t)
{
  v[++tot] = a*t;
  w[tot] = b*t;
  c -= t;
  t *= 2;
}
if(c>0) //处理s-2^k-1+1
{
```

```
        v[++tot] = a*c;
        w[tot] = b*c;
    }
```

d) 最长上升子序列

✧ 问题描述：设有由n个不相同的整数组成的数列，记为:b(1)、b(2)、……、b(n) 若存在i1<i2<i3<…<ie 且有b(i1)<b(i2)<…<b(ie) 则称为长度为e的不下降序列。程序要求，当原数列出之后，求出最长的上升序列。

例如13，7，9，16，38，24，37，18，44，19，21，22，63，15。

例中13，16，18，19，21，22，63就是一个长度为7的上升子序列，同时也有7 ，9，16，18，19，21，22，63组成的长度为8的上升子序列。

✧ 代码模板：

```
int a[100005]; //存储原始数据
int dp[100005];//dp[i]存储长度为i的不升子序列的最小数字;
int n=1,lt,rt,mid; //lt,rt,mid用于二分查找
int max;//max:当前最大上升子序列的长度

while(cin>>a[n]) n++; //读入数据
max = 1;
dp2[max] = a[1];
for(int i=2; i<n; i++)
{
    //可以直接接到最长子序列
    if(a[i]>dp[max])
    {
            max++;
            dp[max] = a[i];
    }
    else //二分查找第一个最大数字大于等于a[i]的子序列
    {
            lt = 1;
            rt = max;
            while(lt<=rt)
            {
```

```
                mid = (lt+rt)/2;
                if(a[i]>dp[mid]) //找右边界
                {
                        lt = mid+1;
                }
                else rt = mid - 1;
            }
            dp[lt] = a[i]; //直接覆盖原先的 dp[lt]
        }
    }
```

e) 最长公共子序列

✧ 问题描述：一个给定序列的子序列是在该序列中删去若干元素后得到的序列。确切地说，若给定序列X=<x1,x2,…,xm>，则另一序列Z=<z1，z2，…，zk>是X的子序列是指存在一个严格递增的下标序列<i1,i2,…,ik>，使得对于所有j=1,2,…,k有： Xij=Zj

例如，序列Z=<B,C,D,B>是序列X=<A,B,C,B,D,A,B>的子序列,相应的递增下标序列为<2,3,5,7>。给定两个序列X和Y，当另一序列Z既是X的子序列又是Y的子序列时，称Z是序列X和Y的公共子序列。例如，若X=<A,B,C,B,D,A,B>和Y=<B,D,C,A,B,A>，则序列<B,C,A>是X和Y的一个公共子序列,序列 <B,C,B,A>也是X和Y的一个公共子序列。而且，后者是X和Y的一个最长公共子序列．因为X和Y没有长度大于4的公共子序列。给定两个序列X=<x1，x2，…，xm>和Y=<y1,y2….yn>．要求找出X和Y的一个最长公共子序列。

✧ 代码模板：

```
#include<iostream>
#include<cstring>
using namespace std;
const int N = 1010;
string s1,s2;
int len1,len2,f[N][N];
int main()
{
  cin>>s1>>s2;
  len1 = s1.length();
```

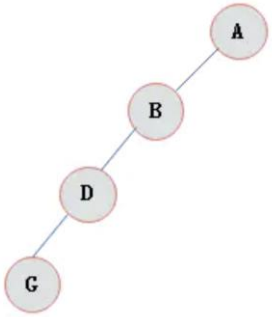
```
        len2 = s2.length();
        //边界条件：f[0][j]和f[i][0] 都是0 （0<i<=len1; 0<j<=len2）
        //边界条件和数组元素默认值相同，可不用设置
        //按照状态转移方程递推
        for(int i=1; i<=len1; i++)
                for(int j=1; j<=len2; j++)
                        //字符下标是从0开始的，所以，比较s1[i-1]==s2[j-1]
                        if(s1[i-1]==s2[j-1]) f[i][j] = f[i-1][j-1]+1;
                        else f[i][j] = max(f[i-1][j],f[i][j-1]);
        cout<<f[len1][len2];
        return 0;
    }
```

三、 【例题】

1. 一个高度为 h 的二叉树最少结点数目是（ B ）。

   A） 2h+1 B） h C） 2h-1 D） 2h

解析：当二叉树的所有结点都只有左孩子或右孩子时，树中的结点数最少，如下图。当高度为h时，只有h个结点。



2. 一个向量第一个元素的存储地址是 100,每个元素的长度是 2,则第 5 个元素的地址是(B)。

   A)110 B)108 C) 100 D) 109

解析：向量可看作是数组的增强版。在向量中，所有数据项的物理存放位置与其逻辑次序完全吻合。因此，当第一个元素的存储地址是 100，每个元素的长度是 2时，第 5 个元素的地址为：100 + （5-1）*2 = 108 。

3. 设有一个含有 13 个元素的 Hash 表(0~12),Hash 函数是:H(key)=key % 13,其中%是求余数运算。用线性探查法解决冲突,则对于序列(2、 8 、31、20、19、18、53、27)，18应放在第几号格中(B)。

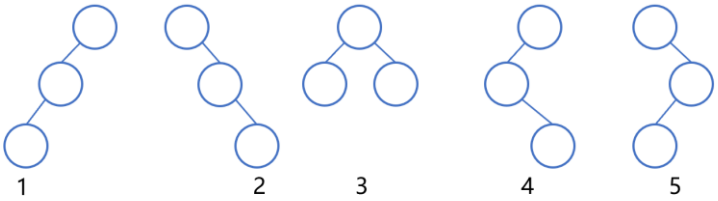
   A) 5 B) 9 C) 4 D) 0

2%13 = 2，8%13 = 8，31%13 = 5，20%13 = 7，19%13 = 6，18%13 = 5。18通过hash函数得到的地址和31的相同，发生冲突。依次往后查找可存放位置，找到位置9 。

4. 按照二叉树的定义,具有 3 个结点的二叉树有(C) 种。

A) 3　　B) 4　　C) 5　　D) 6

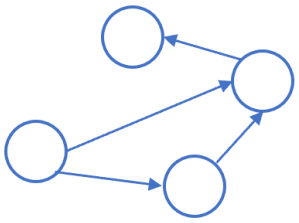
解析：二叉树是n(n>=0)个结点的有限集合，该集合或者为空集（称为空二叉树），或者由一个根结点和两棵互不相交的、分别称为根结点的左子树和右子树组成。具有 3 个结点的二叉树的形状有5种：



5. 在一个有向图中,所有顶点的入度之和等于所有顶点的出度之和的(B) 倍。

A) 1/2　　B)1　　C) 2　　D) 4

解析：有向图中每条边的一端为某个顶点的出度，另一端为另一个顶点的入度。所以，所有顶点的入度之和与所有顶点的出度之和相等。

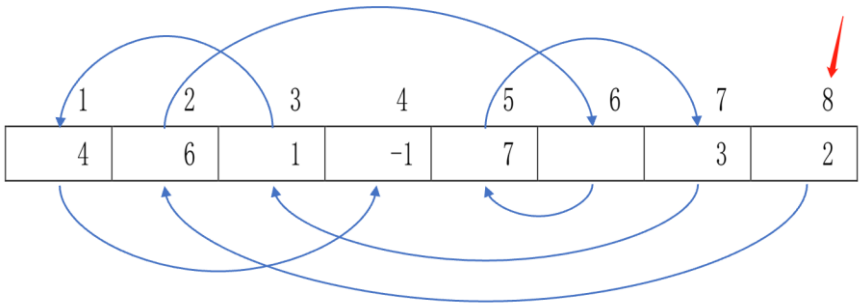


6. 要使 1...8 号格子的访问顺序为: 8、2、6、5、7、3、1、4,则下图中的空格中应填入(C) 。

1	2	3	4	5	6	7	8
4	6	1	-1	7		3	2

A) 6　　B) 0　　C) 5　　D) 3

解析：观察图中数据，可知数组元素存储的是下一个要访问的数组元素下标，即用数组模拟了链表。因此，空格中应填入6。



7. 设栈 S 和队列 Q 的初始状态为空,元素 e1,e2,e3,e4,e5,e6 依次通过栈 S,一个元素出栈后即进入队列 Q,若出队的顺序为 e2,e4,e3,e6,e5,e1,则栈 S 的容量至少应该为(B)。

A) 2　　B) 3　　C) 4　　D) 5

解析：栈的特点是先进后出，队列的特点是先进先出，所以元素的出队顺序即是其出栈顺序。由出栈顺序 e2,e4,e3,e6,e5,e1可知，栈中最多同时存在3个元素，因此，栈 S 的容量至少为 3 。

8. 若已知一个栈的入栈顺序是 1，2，3，…，n，其输出序列为 P1，P2，P3，…，Pn，若 P1 是 n，则 Pi 是(C)

A)i　　B)n-1　　C)n-i+1　　D)不确定

9. 以下哪一个不是栈的基本运算(B)

A)删除栈顶元素　　B)删除栈底的元素

C)判断栈是否为空　　D)将栈置为空栈

10. 下面关于算法的错误说法是(B)

A)算法必须有输出　　B)算法必须在计算机上用某种语言实现

C)算法不一定有输入　　D)算法必须在有限步执行后能结束

解析：算法是对特定问题求解步骤的一种描述。具有 有穷性、确定性、可行性、输入（零个或多个输入）、输出（一个或多个输出）。在计算机上可以用某种语言实现算法。

11. 在顺序表(2，5，7，10，14，15，18，23，35，41，52)中，用二分法查找 12，所需的关键码比较的次数为(　C　)。

A)2　　B)3　　C)4　　D)5

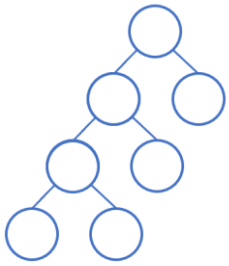
解析：分查找的基本思想是将n个元素分成大致相等的两部分，取a[n/2]与x做比较，如果x=a[n/2],则找到x,算法中止；如果x<a[n/2],则只要在数组a的左半部

分继续搜索x,如果x>a[n/2],则只要在数组a的右半部搜索x。题目的查找的范围为11个数字，12并不存在于查找范围，所以本质上是计算二分查找一个数字时的最多比较次数。$2^3 = 8$， $2^4 = 16$ ，因此需要比较4次。

12. 一棵二叉树的高度为 h，所有结点的度为 0，或为 2，则此树最少有(B)个结点。

A)2h-1　　B)2h-1　　C)2h+1　　D)h+1

解析：二叉树的所有结点的度为 0，或为 2。想要树的结点最少，除根结点外，每层最多只能包含2个结点。如下图。当树的高度为h时，结点数为 2h – 1。

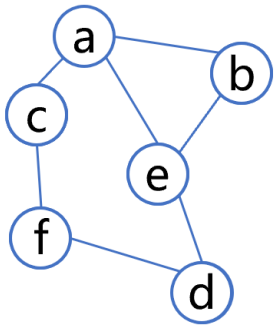


13. 无向图 G=(V，E)，其中 V={a, b, c, d, e, f} ， E={(a, b), (a, e), (a, c), (b, e), (c, f), (f, d), (e, d)}，对该图进行深度优先遍历,得到的顶点序列正确的是(D)

A)a,b,e,c,d,f　　B)a,c,f,e,b,d

C)a,e,b,c,f,d　　D)a,b,e,d,f,c

解析：题目描述的无向图如下所示。从a点开始深搜可得到的序列有：a,b,e,d,f,c 、a,e,b,d,f,c、a,c,f,d,e,b 。



14. 已知一棵二叉树的结点名为大写英文字母，其中序与后序遍历的顺序分别为：CBGEAFHDIJ 与 CGEBHFJIDA，则该二叉树的先序遍历的顺序为：ABCEGDFHIJ

解析：根据后序遍历序列可知根结点为A，根据A的位置确定左子树的中序序列为CBGE，右子树的中序序列为FHDIJ。将左子树和右子树看作一棵二叉树，结合中序和后序序列分别得到其根结点为B和D。依次类推即可构造出完整的二叉树，写出其先序序列即可。

15. 在有 N 个叶子节点的哈夫曼树中，其节点总数为 （ B ）

A.不确定　B. 2N-1　C. 2N+1　D. 2N

解析：哈夫曼树是一棵没有一度结点的二叉树。n0表示叶子结点数，n1表示度为1的结点数，n2表示度为2的结点数，则存在关系：n0 = n2 + 1 。哈夫曼树中，n1=0，所以有N个叶子结点的哈夫曼树中，结点总数为 2N-1 。

16. 线性表若采用链表存贮结构，要求内存中可用存贮单元地址（ D ）

A.必须连续 B. 部分地址必须连续 C. 一定不连续 D. 连续不连续均可

17. 下列叙述中，正确的是（ D ）

A.线性表的线性存贮结构优于链表存贮结构

B.队列的操作方式是先进后出

C.栈的操作方式是先进先出

D. 二维数组是指它的每个数据元素为一个线性表的线性表

18. 根据 Nocomachns 定理，任何一个正整数 n 的立方一定可以表示成 n 个连续的奇数的和。

例如：

$1^3 = 1$

$2^3 = 3+5$

$3^3 = 7+9+11$

$4^3 = 13+15+17+19$

在这里，若将每一个式中的最小奇数称为 X，那么当给出 n 之后，请写出 X 与 n 之间的关系表达式：$X = n^2 - (n-1)$

19. 设循环队列中数组的下标范围是 1～n，其头尾指针分别为 f 和 r，则其元素个数为（ D ）

A.r-f B.r-f+1 C.(r-f) MOD n+1 D.(r-f+n) MOD n

20. 有 2×n 的一个长方形方格，用一个 1×2 的骨牌铺满方格。例如 n=3 时，为 2×3 方格。 此时用一个 1×2 的骨牌铺满方格，共有 3 种铺法：

试对给出的任意一个 n(n>0)，求出铺法总数的递推公式。

答案：f(1)=1 f(2)=2 f(n)=f(n-1)+f(n-2), n>=3

21. 表达式(1+34)*5-56/7 的后缀表达式为（ C ）。

A) 1+34*5-56/7 B) -*+1 34 5/56 7 C) 1 34 +5*56 7/-

D） 1 34 5* +56 7/- E） 1 34+5 56 7-*/

22. 已知元素（8，25，14，87，51，90，6，19，20），问这些元素以怎样的顺序进入栈，才能使出栈的顺序满足：8 在 51 前面；90 在 87 的后面；20 在 14 的后面；25 在 6 的前面；19 在 90 的后面。（ D ）。（题意是全部进栈，再依次出栈）

A）20，6，8，51，90，25，14，19，87
B）51，6，19，20，14，8，87，90，25
C）19，20，90，7，6，25，51，14，87
D）6，25，51，8，20，19，90，87，14
E）25，6，8，51，87，90，19，14，20

23. 假设我们用 d=(a1,a2,...,a5),表示无向图 G 的 5 个顶点的度数，下面给出的哪（些）组 d 值合理（ BE ）。

A）{5，4，4，3，1} B）{4，2，2，1，1} C）{3，3，3，2，2}
D）{5，4，3，2，1} E）{2，2，2，2，2}

24. 下列哪个程序设计语言不支持面向对象程序设计方法（ C ）。

A. C++ B. Object Pascal C. C D. Smalltalk E. Java

25. 二叉树 T，已知其前序遍历序列为 1 2 4 3 5 7 6，中序遍历序列为 4 2 1 5 7 3 6，则其后序遍历序列为（ B ）。

A. 4 2 5 7 6 3 1 B. 4 2 7 5 6 3 1 C. 4 2 7 5 3 6 1 D. 4 7 2 3 5 6 1

26. 满二叉树的叶结点个数为 N，则它的结点总数为（ C ）。

A. N B. 2 * N C. 2 * N – 1 D. 2 * N + 1

27. 某大学计算机专业的必修课及其先修课程如下表所示：

课程代号	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7
课程名称	高等数学	程序设计语言	离散数学	数据结构	编译技术	操作系统	普通物理	计算机原理
先修课程			C_0，C_1	C_1，C_2	C_3	C_3，C_7	C_0	C_6

请你判断下列课程安排方案哪个是不合理的（ D ）。

A. C_0, C_6, C_7, C_1, C_2, C_3, C_4, C_5 B. C_0, C_1, C_2, C_3, C_4, C_6, C_7, C_5
C. C_0, C_1, C_6, C_7, C_2, C_3, C_4, C_5 D. C_0, C_1, C_6, C_7, C_5, C_2, C_3, C_4

28. 完全二叉树的结点个数为 4 * N + 3，则它的叶结点个数为（E ）。

A. 2 * N B. 2 * N - 1 C. 2 * N + 1 D. 2 * N - 2
E. 2 * N + 2

29. 二叉树 T 的宽度优先遍历序列为 A B C D E F G H I，已知 A 是 C 的父结点，D 是 G 的父结点，F 是 I 的父结点，树中所有结点的最大深度为 3（根结点深度设为 0），可知 F的父结点是（C）。

A. 无法确定　　B. B　　C. C　　D. D　　E. E

30. 设栈 S 的初始状态为空，元素 a, b, c, d, e, f, g 依次入栈，以下出栈序列不可能出现的是（E）。

A. a, b, c, e, d, f, g　　B. b, c, a, f, e, g, d　　C. a, e, d, c, b, f, g

D. d, c, f, e, b, a, g　　E. g, e, f, d, c, b, a

31. 将数组{32, 74, 25, 53, 28, 43, 86, 47}中的元素按从小到大的顺序排列，每次可以交换任意两个元素，最少需要交换___5___次。

32. 取火柴游戏的规则如下：一堆火柴有 N 根，A、B 两人轮流取出。每人每次可以取 1 根或 2 根，最先没有火柴可取的人为败方，另一方为胜方。如果先取者有必胜策略则记为 1，先取者没有必胜策略记为 0。当 N 分别为 100，200，300，400，500 时，先取者有无必胜策略的标记顺序为 __11011__（回答应为一个由 0 和/或 1 组成的字符串）

6、数学

一、 排列组合

1. 什么是排列组合?

排列组合是组合学最基本的概念。

排列就是指从给定个数的元素中取出指定个数的元素进行排序。

组合则是指从给定个数的元素中仅仅取出指定个数的元素，不考虑排序。

2. 两个基本原理

加法原理和乘法原理是排列和组合的基础

加法原理：做一件事，完成它可以有 n 类办法，在第一类办法中有 m_1 种不同的方法，在第二类办法中有 m_2 种不同的方法，……，在第 n 类办法中有 m_n 种不同的方法，那么完成这件事共有 $N=m_1+m_2+m_3+...+m_n$ 种不同方法。

乘法原理：做一件事，完成它需要分成 n 个步骤，做第一步有 m_1 种不同的方法，做第二步有 m_2 种不同的方法，……，做第 n 步有 m_n 种不同的方法，那么完成这件事共有 $N=m_1\times m_2\times m_3\times...\times m_n$ 种不同的方法。

- 排列的定义及公式

排列的定义：从 n 个不同元素中，任取 m(m≤n,m 与 n 均为自然数,下同）个元素按照一定的顺序排成一列，叫做从 n 个不同元素中取出 m 个元素的一个排列；从 n 个不同元素中取出 m(m≤n) 个元素的所有排列的个数，叫做从 n 个不同元素中取出 m 个元素的排列数，用符号 A(n,m）表示。

计算公式:

$$A_n^m=n*(n-1)*(n-2)...(n-m+1)\ =\frac{n!}{(n!+m!)}$$

注：n!=n*(n-1)*(n-2)*2*1，即 6！=6*5*4*3*2*1，规定 0！=1

3. 组合的定义及公式

组合的定义：从 n 个不同元素中，任取 m(m≤n）个元素并成一组，叫做从 n 个不同元素中取出 m 个元素的一个组合；从 n 个不同元素中取出 m(m≤n）个元素的所有组合的个数，叫做从 n 个不同元素中取出 m 个元素的组合数。用符号 C(n,m) 表示。

计算公式:

$$C_n^m=\frac{A_n^m}{m!}=\frac{n!}{m!(n-m)!}$$

4. 解题方法

a) 捆绑法

把相邻的若干特殊元素“捆绑”为一个“大元素”，然后与其他“普通元素”全排列，然后再“松绑”，将这些特殊元素在这些位置上全排列。

【题目描述】

7 名学生站成一排，甲、乙必须站在一起有多少不同排法?

【思路解析】

6 组人进行全排列：A（6,6)；

甲和乙的顺序有两种情况，所以结果要再乘 2；

最终答案：A(6,6)*2=6*5*4*3*2*1*2=1440

b) 插空法

某些元素不能相邻或某些元素在特殊位置时可采用插空法。即先安排好没有限制条件的元素，然后将有限制条件的元素按要求插入到排好的元素之间。

【题目描述】

7 名学生站成一排，甲、乙互不相邻有多少不同排法?

【思路解析】

7 名学生，有 6 个空隙可以分离甲，乙同学，所以是 A(5,5)*A(6,2)

A(5,5)*A(6,2)=5*4*3*2*1*6*5*4*3*2*1/(4*3*2*1)=3600

c) 排除法

有些问题用直接法考虑，比较难比较复杂，或分类不清，而它的反面往往比较简捷，可考虑用“排除法”，先求出它的反面，再从整体中排除。

【问题描述】

NOIP2015 真题。重新排列 1234 使得每一个数字都不在原来的位置上，一共有_____种排法。

【思路解析】

答案 = 4 个数字全排列的情况 – 有数字位置不变的情况

四个数字都在原来位置：1234

三个数字都在原来位置：没有

两个数字都在原来位置：1243 1432 1324 4231 3214 2134

一个数字在原来位置：1423 1342 4213 3241 4132 2431 3124 2314

共 15 种情况;

A(4,4)-15=4*3*2*1-15=9

d) 先选后排法

对于排列组合的混合应用题，可采取先选取元素，后进行排列的策略。

【题目描述】

从萝卜，白菜，黄瓜，西红柿四种蔬菜品种中选出 3 种，分别种在不同土质的三块土地上，其中萝卜必须种植，不同的种植方法共有（ ）。

【思路解析】

萝卜必须种植：可以先从 3 块地里选出一块种萝卜，方法数是 A(3,1);

还缺少 2 种蔬菜，需要从剩余 3 种蔬菜里面选出 2 种，方法数是 C(3,2);

选出来的两种蔬菜种植在两块不同的土地方法数是 A(2,2)

所以总种植方法是：A(3,1)*C(3,2)*A(2,2)=18

e) 插板法

插板法就是在 n 个元素间的（n-1）个空中插入 若干个（b）个板，可以把 n 个元素分成（b+1）组的方法

【题目描述】

NOIP2008。书架上有 21 本书，编号从 1 到 21 从中选 4 本，其中每两本的编号都不相邻

的选法一共有(　　)种

【思路解析】

先在总数上把要拿的 4 本书减去，剩下的 17 本书的前后我们一共要插入 4 本书，也就是说有 18 个位置等待插书。一共要插入 4 本书，也不能有两本书插入同一个位置，所以最后的插法个数为 C(18,4)，很显然，插书的位置其实就是我们要挑选书的位置。