全国青少年信息学奥林匹克竞赛教程

第四版

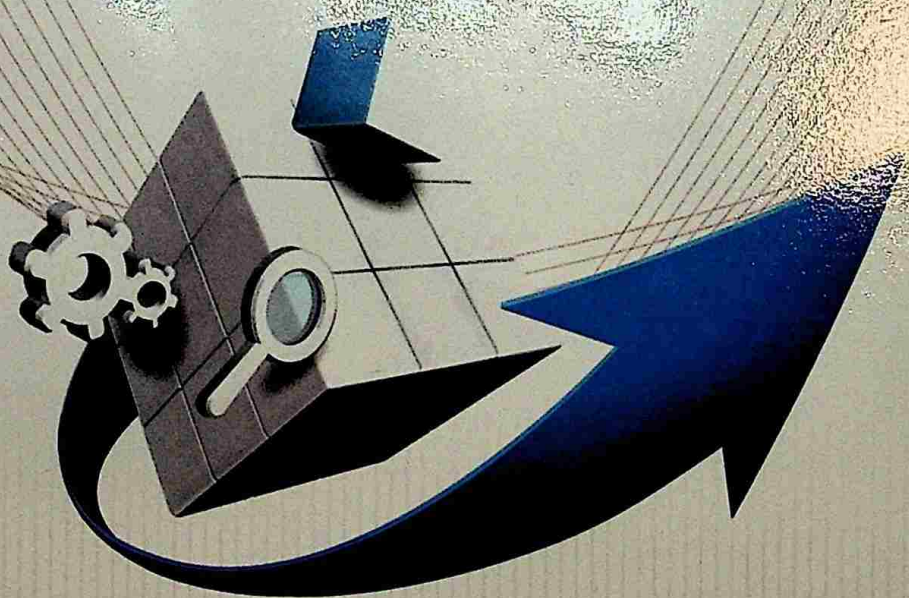
（C++ 版）

信息学奥赛一本通

XINXIXUE AOSAI YIBENTONG

初赛篇

◆ 曹文 [illegible] 吴涛 编著



南京大学出版社

目　录

第一章　计算机基础知识

第 1 节　计算机常识

一、发展史

1. 计算机发展代别划分：

代别	年代	逻辑(电子)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	1971—至今	大规模、超大规模集成电路

2. 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，质量 30 吨，用了 18000 多个电子管，每秒能进行 5000 次加法运算。

3. 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：

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。

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。

二、计算机的分类

根据计算机的性能指标，如机器规模的大小、运算速度的高低、主存储容量的大小、指令系统性能的强弱以及机器的价格等，可将计算机分为巨型机、大型机、中型机、小型机、微型机和工作站。

巨型机：具有很强的计算和处理数据的能力，主要特点表现为高速度和大容量，配有多种外部和外围设备及丰富的、高功能的软件系统。主要用来承担重大的科学研究、国防尖端技术和国民经济领域的大型计算课题及数据处理任务。如大范围天气预报，整理卫星照片，原子核物的探索，研究洲际导弹、宇宙飞船等。“天河一号”为我国首台千万亿次超级计算机。2010 年 9 月开始进行系统调试与测试，并分步提交用户使用。

大、中型机：大型机使用专用的处理器指令集、操作系统和应用软件，大量使用冗余等技

术确保其安全性及稳定性，擅长非数值计算（数据处理），主要用于商业领域，如银行和电信。

小型机是指采用精简指令集处理器，性能和价格介于PC服务器和大型主机之间的一种高性能64位计算机。

小型机与普通服务器相比具有：

（1）高可靠性（Reliability）：计算机能够持续运转，从来不停机。

（2）可用性（Availability）：重要资源都有备份；能够检测到潜在要发生的问题，并且能够转移其上正在运行的任务到其他资源，以减少停机时间，保持生产的持续运转；具有实时在线维护和延迟性维护功能。

（3）高服务性（Serviceability）：能够实时在线诊断，精确定位根本问题所在，做到准确无误的快速修复。

微型机：通常作为个人计算机，由硬件系统和软件系统组成，是一种能独立运行，完成特定功能的设备。个人计算机不需要共享其他计算机的处理、磁盘和打印机等资源也可以独立工作。从台式机（或称台式计算机、桌面电脑）、笔记本电脑到上网本和平板电脑以及超级本等都属于个人计算机的范畴。

工作站是一种高端的通用微型计算机。它是为了单用户使用并提供比个人计算机更强大的性能，尤其是在图形处理能力，任务并行方面的能力。通常配有高分辨率的大屏、多屏显示器及容量很大的内存储器和外部存储器，并且具有极强的信息和高性能的图形、图像处理功能的计算机。另外，连接到服务器的终端机也可称为工作站。

三、计算机的应用

计算机的快速性、通用性、准确性和逻辑性等特点，使它不仅具有高速运算能力，而且还具有逻辑分析和逻辑判断能力。如今，计算机已渗透到人们生活和工作的各个层面中，主要体现在以下几个方面的运用。

1. 科学计算

科学计算（或数值计算）是指利用计算机来完成科学研究和工程技术中提出的数学问题的计算。在现代科学技术工作中，科学计算问题是大量的和复杂的。利用计算机的高速计算、大存储容量和连续运算的能力，可以实现人工无法解决的各种科学计算问题。

2. 信息处理

信息处理（数据处理）是指对各种数据进行收集、存储、整理、分类、统计、加工、利用、传播等一系列活动的统称。据统计，80%以上的计算机主要用于数据处理，这类工作量大、面宽，决定了计算机应用的主导方向。

3. 自动控制

自动控制（过程控制）是利用计算机及时采集检测数据，按最优值迅速地对控制对象进行自动调节或自动控制。采用计算机进行自动控制，不仅可以大大提高控制的自动化水平，而且可以提高控制的及时性和准确性，提高产品质量及合格率。目前，计算机过程控制已在机械、冶金、石油、化工、纺织、水电、航天等部门得到了广泛的应用。

4. 计算机辅助技术

计算机辅助技术是指利用计算机帮助人们进行各种设计、处理等过程，它包括计算机辅助设计（CAD）、计算机辅助制造（CAM）、计算机辅助教学（CAI）和计算机辅助测试（CAT）等。另外，计算机辅助技术还有辅助生产、辅助绘图和辅助排版等。

5. 人工智能

人工智能(Artificial Intelligence,AI)又可称为智能模拟,是计算机模拟人类的智能活动,诸如感知、判断、理解、学习、问题求解和图像识别等。人工智能的研究目标是使计算机更好地模拟人的思维活动,那时的计算机将可以完成更复杂的控制任务。

6. 网络应用

随着社会信息化的发展,通信业也发展迅速,计算机在通信领域的作用越来越大,特别是促进了计算机网络的迅速发展。目前,全球最大的网络(Internet,即国际互联网)已把全球的大多数计算机联系在一起。除此之外,计算机在信息高速公路、电子商务、娱乐和游戏等领域也得到了快速的发展。

【课堂练习】

1.【NOIP2001】计算机软件保护法是用来保护软件(　　)的。

A. 编写权　　B. 复制权　　C. 使用权　　D. 著作权

【答案】D

【分析】我国1991年6月4日发布、2001年1月1日修订实施的《计算机软件保护条例》第2条规定:本条例所称计算机软件(以下简称软件),是指计算机程序及其有关文档。

2.【NOIP2002】微型计算机的问世是由于(　　)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

【答案】C

【分析】计算机发展大致可分为四代:

第一代,电子管计算机时代(1946—1958年);

第二代,晶体管计算机时代(1959—1964年);

第三代,集成电路计算机时代(1965—1970年);

第四代,大规模和超大规模集成电路计算机时代(1971年至今),其特点:大规模或超大规模集成电路作为逻辑元件和存储器,体积更小,可靠性更高,速度为每秒几千万至数亿次,也直接导致了微型计算机的问世。

3.【NOIP2003】图灵(Alan Turing)是(　　)。

A. 美国人　　B. 英国人　　C. 德国人　　D. 匈牙利人　　E. 法国人

【答案】B

【分析】艾伦·麦席森·图灵(Alan Mathison Turing,1912年6月23日—1954年6月7日),英国数学家。

艾伦·图灵:1931年,图灵进入剑桥大学国王学院,毕业后到美国普林斯顿大学攻读博士学位,二战爆发后回到剑桥,后曾协助军方破解德国著名密码系统Enigma,帮助盟军取得了二战胜利。

图灵对于人工智能的发展有着诸多贡献,例如:图灵曾写过一篇名为《机器人会思考吗?》(Can Machine Think?)的论文,其中提出了一种用于判定机器是否具有智能的试验方法,即图灵试验。

此外,图灵提出的著名图灵机模型为现代计算机的逻辑工作方式奠定了基础。

4.【NOIP2003】第一个给计算机写程序的人是(　　)。

A. Alan Mathison Turing　　B. Ada Lovelace　　C. John von Neumann
D. John McCarthy　　E. Edsger Wybe Dijkstra

【答案】B

【分析】某种意义上，程序设计的出现甚至早于电子计算机的出现。英国著名诗人拜伦的女儿 Ada Lovelace 曾设计了巴贝奇分析机上解伯努利方程的一个程序。她甚至还建立了循环和子程序的概念。由于她在程序设计上的开创性工作，Ada Lovelace 被称为世界上第一位程序员。

5.【NOIP2004 普及组】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(　　)。

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。

【答案】C

【分析】A. 英国数学家 Turing(图灵)(1912—1954)，1936 年提出了一种理想计算机的数学模型(图灵机)，1950 年提出了图灵试验，发表了“计算机与智能”的论文，成为计算机科学理论基础第一人。

B. Ada 是一种表现能力很强的通用程序设计语言，它是美国国防部为克服软件开发危机，耗费巨资，历时近 20 年研制成功的，为了纪念奥左斯特·艾达·洛夫莱斯伯爵夫人(Augusta Ada Lovlace 1815—1852)，她是英格兰诗人拜伦勋爵的女儿，曾对现代计算机技术之父查尔斯·巴贝奇(Charles Babage)的笔记手稿进行了整理和修正，她是世界上第一位计算机程序员。

C. EDVAC(Electronic Discrete Variable Automatic Computer)。离散变量自动电子计算机。

1945 年，冯·诺依曼以“关于 EDVAC 的报告草案”为题，起草了长达 101 页的总结报告。报告广泛而具体地介绍了制造电子计算机和程序设计的新思想。EDVAC 方案明确奠定了新机器由五个部分组成，包括：运算器、逻辑控制装置、存储器、输入和输出设备，并描述了这五部分的职能和相互关系。

E. 摩尔定律是由英特尔(Intel)创始人之一戈登·摩尔(Gordon Moore)提出来的。其内容为：当价格不变时，集成电路上可容纳的晶体管数目，约每隔 18 个月便会增加一倍，性能也将提升一倍。

6.【NOIP2004 普及组】彩色显示器所显示的五彩斑斓的色彩，是由红色、蓝色和(　　)色混合而成的。

A. 紫　　B. 白　　C. 黑　　D. 绿　　E. 橙

【答案】D

【分析】三基色是指红、绿、蓝三色，人眼对红、绿、蓝最为敏感，大多数的颜色可以通过红、绿、蓝三色按照不同的比例合成产生。同样，绝大多数单色光也可以分解成红、绿、蓝三种色光。

7.【NOIP2006】在下面各个世界顶级的奖项中，为计算机科学与技术领域作出杰出贡献的科学家设立的奖项是(　　)。

A. 沃尔夫奖　　B. 诺贝尔奖　　C. 菲尔兹奖　　D. 图灵奖　　E. 南丁格尔奖

【答案】D

【分析】沃尔夫奖(人类科学和艺术文明)、诺贝尔奖(生理医学、文字、物理、化学、经济、和平)、菲尔兹奖(数学)、图灵奖(计算机,2000 年,姚期智是目前获得图灵奖的唯一华裔计算机科学家)、南丁格尔奖(护理)。

沃尔夫奖主要是奖励对推动人类科学与艺术文明做出杰出贡献的人士,每年评选一次,分别奖励在农业、化学、数学、医学和物理领域,或者艺术领域中的建筑、音乐、绘画、雕塑四大项目之一中取得突出成绩的人士,其中以沃尔夫数学奖影响最大,因为诺贝尔奖中没有数学奖;菲尔兹奖是据加拿大数学家约翰·查尔斯·菲尔兹的要求设立的,被视为数学界的诺贝尔奖;图灵奖(A. M. Turing Award,又译"社林奖"),由美国计算机协会(ACM)于 1966 年设立,又叫"A. M. 图灵奖",专门奖励那些对计算机事业作出重要贡献的个人,其名称取自计算机科学的先驱、英国科学家艾伦·麦席森·图灵。他是计算机界最负盛名、最崇高的一个奖项,有"计算机界的诺贝尔奖"之称。

8.【NOIP2007 普及组】IT 的含义是(　　)。

A. 通信技术　　B. 信息技术　　C. 网络技术　　D. 信息学

【答案】B

【分析】IT 是"Information Technology"(信息技术)的缩写,指计算机、通信及相关技术。

第2节　计算机系统的基本结构

计算机系统由硬件和软件两部分组成。硬件系统是计算机的“躯干”，是物质基础。而软件系统则是建立在这个“躯干”上的“灵魂”。

- 计算机系统
  - 硬件系统
    - 主机
      - 中央处理器 CPU
        - 控制器
        - 运算器
      - 内存
        - ROM(只读存储器)
        - RAM(读写存储器)
        - Cache(高速缓冲存储器)
    - 外设
      - 输入设备——键盘、鼠标、扫描仪、麦克风等
      - 输出设备——显示器、打印机、音响等
      - 外存——硬盘、软盘、光盘、闪存(优盘)等
  - 软件系统
    - 系统软件
      - 操作系统
      - 编译程序和解释程序
      - 数据库管理系统
    - 应用软件
      - 各种字处理系统
      - 各种软件包

一、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。

1. 中央处理器(CPU——Central Processing Unit)

由运算器、控制器和一些寄存器组成；

运算器进行各种算术运算和逻辑运算；

控制器是计算机的指挥系统；

CPU 的主要性能指标是主频和字长。

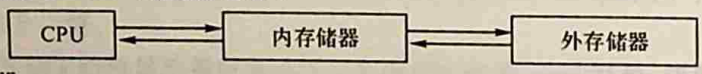
2. 存储器

存储器的主要功能是用来保存各类程序的数据信息。

存储器可分为主存储器和辅助存储器两类。

①主存储器(也称为内存储器)，属于主机的一部分。用于存放系统当前正在执行的数据和程序，属于临时存储器。

②辅助存储器(也称外存储器)，属于外部设备。用于存放暂不用的数据和程序，属于永久存储器。存储器与 CPU 的关系表示：



(1)内存储器

内存又称为主存，它和 CPU 一起构成了计算机的主机部分，它存储的信息可以被 CPU 直接访问。内存由半导体存储器组成，存取速度较快，但一般容量较小。内存中含有很多的存储单元，每个单元可以存放 1 个 8 位的二进制数，即 1 个字节(Byte，简称“B”)。内存中的每个字节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。所谓存储器容量，即指存储器中所包含的字节数，通常用 KB、MB、GB、TB 和 PB 作为存储器容量单位。它们之间的关系为：

1 KB=1024 B　1 MB=1024 KB　1 GB=1024 MB　1 TB=1024 GB　1 PB=1024 TB

内存储器通常可以分为随机存储器 RAM、只读存储器 ROM 和高速缓冲存储器 Cache 三种。

①RAM 是一种读写存储器，其内容可以随时根据需要读出，也可以随时重新写入新的信息。当电源电压去掉时，RAM 中保存的信息都将全部丢失。

②ROM 是一种内容只能读出而不能写入和修改的存储器，其存储的信息是在制作该存储器时就被写入的。在计算机运行过程中，ROM 中的信息只能被读出，而不能写入新的内容。计算机断电后，ROM 中的信息不会丢失。它主要用于检查计算机系统的配置情况并提供最基本的输入/输出(I/O)控制程序。

③由于 CPU 速度的不断提高，RAM 的速度很难满足高速 CPU 的要求，所以，在读/写系统内存时都要加入等待的时间，这对高速 CPU 来说是一种极大的浪费。Cache 是指在 CPU 与内存之间设置的一级或两级高速小容量存储器，称之为高速缓冲存储器，固化在主板上。在计算机工作时，系统先将数据由外存读入 RAM 中，再由 RAM 读入 Cache 中，然后 CPU 直接从 Cache 中取数据进行操作。

(2)外存储器

外存储器又称为辅助存储器，它的容量一般都比较大，而且大部分可以移动，便于在不同计算机之间进行信息交流。在微型计算机中，常用的外存有软盘、硬盘、闪存和光盘 4 种。

①软盘存储器

软盘存储器由软盘、软盘驱动器和软盘适配器三部分组成。软盘是活动的存储介质，软盘驱动器是读写装置，软盘适配器是软盘驱动器与主机连接的接口。软盘驱动器安装在主机箱内，软盘驱动器插槽暴露在主机箱的前面板上，可方便地插入或取出软盘。

②硬盘存储器

硬盘存储器是由电机和硬盘组成的，一般置于主机箱内。硬盘是涂有磁性材料的磁盘组件，用于存放数据。硬盘的机械转轴上串有若干个盘片，每个盘片的上下两面各有一个读/写磁头，与软盘磁头不同，硬盘的磁头不与磁盘表面接触，它们“飞”在离盘片面百万分之一英寸的“气垫”上。硬盘是一个非常精密的机械装置，磁道间只有百万分之几英寸的间隙，磁头传动装置必须把磁头快速而准确地移到指定的磁道上。

③闪存

闪存又名优盘，是在存储速度与容量上介于软盘与硬盘之间的一种外部存储器。

④光盘

光盘的存储介质不同于磁盘，它属于另一类存储器。由于光盘的容量大、存取速度较快、不易受干扰等特点，其应用越来越广泛。光盘根据其制造材料和记录信息方式的不同一般分为三类：只读光盘、一次写入型光盘和可擦写光盘。

3. 输入设备

输入设备是外界向计算机传送信息的装置。在微型计算机系统中，最常用的输入设备是键盘和鼠标。此外，还有光电笔、数字化仪、图像扫描仪、触摸屏、麦克风、视频输入设备、条形码扫描器等，也可以用磁盘和磁带进行输入。

4. 输出设备

输出设备的作用是将计算机中的数据信息传送到外部媒介，并转化成某种为人们所认识的表示形式。在微型计算机中，最常用的输出设备有显示器和打印机。此外，还有绘图仪等，也可以通过磁盘和磁带输出。

二、总线结构

按照总线上传输信息的不同，总线可以分为数据总线(DB)、地址总线(AB)和控制总线(CB)三种。

①数据总线：用来传送数据信息，它主要连接了 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，而具体的传送方向由 CPU 控制。

②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 1 MB 的存储空间。

③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。

三、主要的性能指标

计算机的常用指标有：

1. 字长

字长是指一台计算机所能处理的二进制代码的位数。计算机的字长直接影响它的精度、功能和速度。字长越长，能表示的数值范围就越大，计算出的结果的有效位数也就越多；字长越长，能表示的信息就越多，机器的功能就更强。目前常用的是 16 位、32 位、64 位字长。

2. 运算速度

运算速度是指计算机每秒钟所能执行的指令条数，一般用 MIPS(Million of Instructions Per Second，即每秒百万条指令)为单位。由于不同类型的指令执行时间长短不同，因而运算速度的计算方法也不同。

3. 主频

主频是指计算机 CPU 的时钟频率，它在很大程度上决定了计算机的运算速度。一般时钟频率越高，运算速度就越快。主频的单位一般是 MHz(兆赫)或 GHz(吉赫)，如微处理器 Pentium4/2.0 GHz 的主频为 2 * 1000 MHz。

4. 内存容量

内存容量是指内存储器中能够存储信息的总字节数，一般以 GB 为单位。内存容量反映内存储器存储数据的能力。目前计算机的内存容量有 2 GB、4 GB、8 GB 等。

【课堂练习】

1.【NOIP1999】 微机内的存储器的地址是以(　　)编址的。

A. 二进制位　　B. 字长　　C. 字节　　D. 微处理器的型号

【答案】 B

【分析】 字长表示一个存储单元由多少位二进制数组成，八位机的一个字长就是一个字节，十六位机的一个字长是两个字节，三十二位机的一个字长可以表示四个字节。字节位的多少，表明可访问存储器的地址多少。

2.【NOIP2000】 某种计算机的内存容量是 640 K，这里的 640 K 容量是指(　　)个字节。

A. 640　　B. 640 * 1000　　C. 640 * 1024　　D. 640 * 1024 * 1024

【答案】 C

【分析】 1 KB=1024 B，640 K=640 * 1024 B。

3.【NOIP2000】在外部设备中，绘图仪属于(　　)。

A. 输入设备　　B. 输出设备　　C. 辅(外)存储器　　D. 主(内)存储器

【答案】B

【分析】能按照人们要求自动绘制图形的设备，将计算机的输出信息以图形的形式输出。

4.【NOIP2000 普及组】RAM 中的信息是(　　)。

A. 生产厂家预先写入的　　B. 计算机工作时随机写入的

C. 防止计算机病毒侵入所使用的　　D. 专门用于计算机开机时自检用的

【答案】B

【分析】RAM 表示的是读写存储器，可对其中的任一存储单元进行读或写操作，计算机关闭电源后其内的信息将不再保存。

5.【NOIP2000】计算机主机是由 CPU 与(　　)构成的。

A. 控制器　　B. 运算器　　C. 输入、输出设备　　D. 内存储器

【答案】D

【分析】计算机包括运算器、控制器、存储器、输入设备、输出设备五个部分，计算机主机指的是除去输入、输出设备的计算机，CPU 包括了运算器和控制器，所以答案应该是存储器，而主机里面可以有外存储器，也可以没有。因此，D 是唯一的正确答案。

6.【NOIP2000 提高组】计算机系统总线上传送的信号有(　　)。

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

【答案】B

【分析】数据总线、地址总线、控制总线分别传送数据信号、地址信号、控制信号。数据总线用来在两个逻辑部件之间传送数据，数据总线通常是双向的。控制总线用来完成控制和监视功能，一般是单向的。地址总线不仅传送地址，还用来选择将要进行信息传输的设备。

7.【NOIP2001】在计算机硬件系统中，Cache 是(　　)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

【答案】D

【分析】计算机硬件系统中，通常采用三级存储器结构，即使用快速缓冲存储器、主存储器和外存储器(也称辅助存储器或辅存)。中央处理器能直接访问的存储器称为内存储器，它包括快速缓冲存储器和主存储器。主存储器和外存储器相比，前者速度快、容量小，后者速度慢、容量大，快存的含义有些基础教材上没有介绍，它是高速缓冲存储器的简称，即 Cache，是为了解决 CPU 和主存储器之间速度匹配问题而设置的，和主存储器相比，速度更快，容量更小。

8.【NOIP2002 提高组】微型计算机中，(　　)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

【答案】C

【分析】各种存储器的存取速度，其中以高速缓冲存储器存取速度最快，寄存器是中央处理器内部运算器的存储单位，尽管中央处理器种类繁多，但其中至少有指令寄存器、程序计数器、地址寄存器、缓冲寄存器、累加寄存器、状态条件寄存器等。因其是中央处理器内部存储单位，其速度自然比中央处理器外部的高度缓冲存储器等要快得多。

9.【NOIP2003】下列计算机设备中，既是输入设备又是输出设备的是(　　)。

A. 键盘　　B. 触摸屏　　C. 扫描仪　　D. 投影仪　　E. 数字化仪

【答案】B

【分析】触摸屏与 PLC 是利用通信方式工作的，通信属于全双工通信，触摸屏既可以显示运行数据，又可以发出指令给 PLC 控制运行，所以既是输入设备，也是输出设备。

10.【NOIP2003 提高组】下列哪个不是个人计算机的硬件组成部分(　　)。

A. 主板　　B. 虚拟内存　　C. 电源　　D. 总线

【答案】B

【分析】虚拟内存指的是用外存(如硬盘)来模拟内存，是一种存储器管理的方式，而不是硬件组成。

11.【NOIP2004】下面哪个部件对于个人桌面电脑的正常运行不是必需的(　　)。

A. CPU　　B. 图形卡(显卡)　C. 光驱　　D. 主板　　E. 内存

【答案】C

【分析】五大部件不能少。

12.【NOIP2004 普及组】下列哪个不是计算机的存储设备(　　)。

A. 文件管理器　B. 内存　　C. 高速缓存　　D. 硬盘　　E. U 盘

【答案】A

【分析】文件管理器一般指软件。

13.【NOIP2004】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(　　)。

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

【答案】C

【分析】A. 针式打印机是一种特殊的打印机，和喷墨、激光打印机都存在很大的差异。

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。对于一些医院窗口、银行窗口、邮局窗口等行业用户来说，针式打印机是他们的必备产品之一，因为只有通过针式打印机才能快速地完成各项单据的复写，为用户提供高效的服务，而且还能为这些窗口行业用户存底。

B. 喷墨打印机是通过加热喷嘴，使墨水产生气泡，喷到打印介质上的。

C. 激光打印机是用高压静电将感光鼓表面的“墨粉图像”转印到普通纸上。

D. 笔式绘图机(pen plotter)是一种装有画笔的平板式绘图机。

E. 喷墨绘图仪是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。

14.【NOIP2005】一位艺术史学家有 20000 幅 1024 * 768 的真彩色图像，如果将这些图像以位图形式保存在 CD 光盘上(一张 CD 光盘的容量按 600M 计算)，大约需要(　　)张 CD 光盘。

A. 1　　B. 10　　C. 100　　D. 1000　　E. 10000

【答案】C

【分析】(1024 * 768 * 32 * 20000)/(8 * 1024 * 1024 * 600)＝100(张)。

15.【NOIP2005】下列设备不具有计算功能的是(　　)。

A. 笔记本电脑　B. 掌上电脑　　C. 智能手机　　D. 电子计算器　E. 液晶显示器

【答案】E

【分析】输出设备，没有 CPU 或者计算元器件。

16.【NOIP2005 普及组】以下哪个不是计算机的输出设备(　　)。

A. 音箱　　B. 显示器　　C. 打印机　　D. 扫描仪　　E. 绘图仪

【答案】D

【分析】扫描仪是标准的输入设备，它只能把相关信息“输入”到电脑里。

17.【NOIP2006 普及组】以下断电之后仍能保存数据的有(　　)。

A. 寄存器　　B. ROM　　C. RAM　　D. 高速缓存

【答案】B

【分析】ROM 表示的是只读存储器，只能读出信息，不能写入信息，计算机关闭电源后其内的信息仍旧保存，一般用它存储固定的系统软件和字库等。

18.【NOIP2006 提高组】BIOS(基本输入输出系统)是一组固化在计算机(　　)上一个 ROM 芯片上的程序。

A. 控制器　　B. CPU　　C. 主板　　D. 内存条　　E. 硬盘

【答案】C

【分析】BIOS(基本输入输出系统)是一组固化在计算机主板 ROM 芯片上的程序。

19.【NOIP2007 普及组】一个完整的计算机系统应包括(　　)。

A. 系统硬件和系统软件　　B. 硬件系统和软件系统

C. 主机和外部设备　　D. 主机、键盘、显示器和辅助存储器

【答案】B

【分析】一个完整的计算机系统应包括计算机的硬件系统和软件系统。

20.【NOIP2007 普及组】以下断电之后仍能保存数据的有(　　)。

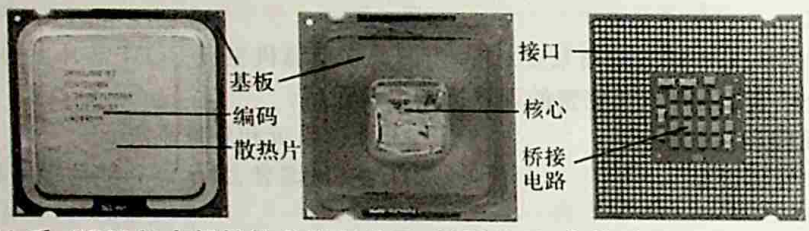
A. 硬盘　　B. 高速缓存　　C. 显存　　D. RAM

【答案】A

【分析】其他会丢失。

第3节　中央处理器CPU

CPU(中央处理单元)是微机的核心部件,是决定微机性能的关键部件。20世纪70年代微型机的CPU问世,微型计算机的核心部件微处理器从Intel 4004,80286,80386,80486发展到Pentium Ⅱ/Ⅲ和Pentium 4,数位从4位、8位、16位、32位发展到64位,主频从几MHz到今天的数GHz以上(1 GHz=1000 MHz),CPU芯片里集成的晶体管数由2万个跃升到1000万个以上。CPU的发展和技术的进展直接推动了微型计算机的发展,也是微机各个发展阶段的主要标志。



从原理上看,CPU的内部结构分控制单元、逻辑单元、存储单元三部分。从组成器件上看,CPU的内部是由成千上万个晶体管组成,晶体管实质上就是一双位开关:即"开"和"关"。

CPU的主要性能指标包括时钟主频、字长、高速缓存容量、指令集合和动态处理技术、制造工艺、封装方式和工作电压等。

主频是指CPU的工作时钟频率,是CPU内核电路的实际运行频率。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,速度也越快。主频的单位为兆赫兹(MHz)和吉赫兹(GHz)。我们通常所说的2.8 GHz、3.0 GHz就是指CPU的主频。

字长(word size)指的是微处理器CPU能够同时处理的二进制位数的个数。字长的大小取决于ALU中寄存器的容量和连接着这些寄存器的电路性能。例如,8位字长的微处理器有8位的寄存器,每次能处理8位的数据,因此,被称为"8位处理器"。有更大字长的处理器能够在每个处理器周期内处理更大的数据,因此,字长越长计算机性能越好。目前的个人计算机通常都带有32位或64位的处理器。

高速缓存(Cache)也称为"RAM缓存"或"缓冲存储器"。它是一种具有很高速度的特殊内部存储器,与安装在主板上其他位置的内存相比,它能够使微处理器更快地获得数据。

字节和字长的区别:常用的英文字符用8位二进制就可以表示,所以通常就将8位称为一个字节,字节是一种存储容量单位。而字长是CPU处理能力的一种标准,字长的长度是不固定的,对于不同的CPU字长的长度也不一样。8位的CPU一次只能处理一个字节,而32位的CPU一次就能处理4个字节。同理,字长为64位的CPU一次可以处理8个字节。

1971年,英特尔公司推出了世界上第一款微处理器4004,字长4位,是4位微处理器。

1978年,英特尔公司生产的8086是第一个16位的微处理器。

1985年,英特尔生产出32位字长处理器80386。

目前市场上主流的CPU的字长几乎都达到了64位。

【课堂练习】

1.【NOIP1999】在微机中,通用寄存器的位数是(　　)。

A. 8位　　B. 16位　　C. 计算机字长　　D. 32位

【答案】C

【分析】通用寄存器的位数跟CPU型号有关,它取决于计算机的字长。

2.【NOIP1999】不同的计算机,其指令系统也不相同,这主要取决于(　　)。

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

【答案】C

【分析】计算机指令系统取决于中央处理器中的控制器，所有的控制和运算操作，均由控制器中的微指令系统进行操作。

3.【NOIP2001 普及组】CPU 处理数据的基本单位是字，一个字的字长(　　)。

A. 为 8 个二进制位　　B. 为 16 个二进制位
C. 为 32 个二进制位　　D. 与芯片的型号有关

【答案】D

【分析】CPU 处理数据的基本单位是字，一个字的字长通常与微处理器芯片的型号有关。

4.【NOIP2001】若我们说一个微机的 CPU 是用的 PII300，此处的 300 确切指的是(　　)。

A. CPU 的主时钟频率　　B. CPU 产品的系列号
C. 每秒执行 300 百万条指令　　D. 此种 CPU 允许最大内存容量

【答案】A

【分析】300 指的是 CPU 的主时钟频率，以 MHz(兆赫兹)为单位，300 即 300 MHz。

5.【NOIP2001 提高组】中央处理器 CPU 能访问的最大存储器容量取决于(　　)。

A. 地址总线　　B. 数据总线　　C. 控制总线　　D. 内存容量

【答案】A

【分析】地址总线主要用来传输内存地址，地址线的条数越多，CPU 能访问存储器的范围越大。如果地址条数不够，寻址能力也就有限，内存容量再大也用不上，因此不能选 D。

6.【NOIP2004 普及组】下列说法中错误的是(　　)。

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

【答案】B

【分析】A. CPU 包括运算逻辑部件、寄存器部件和控制部件。CPU 从存储器或高速缓冲存储器中取出指令，放入指令寄存器，并对指令译码。

B. CPU 访问存储器的速度：Cache＞内存＞外存，Cache 是 CPU 内部的高速缓存，容量很小，当然速度也是最快。

C. CPU 的主频，即 CPU 内核工作的时钟频率(CPU Clock Speed)。

D. 计算机在使用存储器时，要给这些存储器进行编号，这个编号就是地址。

E. 数据总线负责计算机中数据在各组成部分之间的传送，数据总线宽度是指在芯片内部数据传送的宽度，而数据总线宽度则决定了 CPU 与二级缓存、内存以及输入/输出设备之间一次数据传输的信息量。

地址总线宽度决定了 CPU 可以访问的物理地址空间，简单地说，就是 CPU 到底能够使用多大容量的内存。16 位的微机我们就不用说了，但是对于 486 以上的微机系统，地址线的宽度为 32 位，最多可以直接访问 4096 MB(4 GB)的物理空间。

7.【NOIP2004 普及组】下列哪个不是 CPU(中央处理单元)(　　)。

A. Intel Itanium　　B. DDR SDRAM　　C. AMD Athlon64
D. AMD Opteron　　E. IBM Power 5

【答案】B

【分析】A. Intel 安腾处理器应该说是大多数人不是很了解的处理器之一。

B. DDR SDRAM 是 Double Data Rate SDRAM 的缩写，是双倍速率同步动态随机存储器的意思。DDR 内存是在 SDRAM 内存基础上发展而来的，SDRAM 在一个时钟周期

内只传输一次数据，它是在时钟的上升期进行数据传输；而 DDR 内存则是一个时钟周期内传输两次数据，它能够在时钟的上升期和下降期各传输一次数据，因此称为双倍速率同步动态随机存储器。DDR 内存可以在与 SDRAM 相同的总线频率下达到更高的数据传输率。

C. Athlon 是由美国超微(AMD)公司生产的一个 CPU 系列。

D. AMD Opteron™(皓龙)处理器专为服务器、工作站而设计。

E. IBM POWER 是 RISC 处理器架构的一种，由 IBM 设计，全称为“Performance Optimization With Enhanced RISC”

8.【NOIP2005 普及组】 Intel 的首颗 16 位处理器是(　　)。

A. 8088　　B. 80386　　C. 80486　　D. 8086　　E. Pentium

【答案】 D

【分析】 比较生冷，1978 年英特尔公司生产的 8086 是第一个 16 位的微处理器。

9.【NOIP2005 提高组】 Intel 的首颗 64 位处理器是(　　)。

A. 8088　　B. 8086　　C. 80386　　D. 80486　　E. Pentium

【答案】 E

【分析】 Intel 公司于 1993 年又推出了 80586，其为 64 位微处理器，正式名称为 Pentium。Pentium 含有 310 万个晶体管，时钟频率最初为 60 MHz 和 66 MHz。

10.【NOIP2005 提高组】 处理器 A 每秒处理的指令数是处理器 B 的两倍。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 $O(n^2)$，如果处理器 A 执行程序 P 时能在一小时内完成的输入规模为 n，则处理器 B 执行程序 P 时能在一小时内完成的输入规模为(　　)。

A. 4 * n　　B. 2 * n　　C. n　　D. n / 2　　E. n / 4

【答案】 B

【分析】 每秒指令数 2∶1，编译后指令量 4∶1，综合速度 1∶2，无论在哪一个处理器上，该程序 P 的算法复杂度都是一样的，因此，产生影响的主要是指令数及两个处理器的速度。综合起来看，处理器 B 有两倍的速度优势，因此，在同样的一小时内，B 可以处理多一倍的数据，即输入数据的规模可以是 2 * n。

11.【NOIP2006 提高组】 在以下各项中，(　　)不是 CPU 的组成部分。

A. 控制器　　B. 运算器　　C. 寄存器　　D. ALU　　E. RAM

【答案】 E

【分析】 CPU 由控制器、运算器和寄存器组成。运算器的基本操作包括加、减、乘、除四则运算，与、或、非、异或等逻辑操作，以及移位、比较和传送等操作，亦称算术逻辑部分(ALU)。而 RAM 是随机储存器(内存)，不是 CPU 的组成部分。

第 4 节　计算机软件系统

软件是计算机的灵魂。没有安装软件的计算机称为“裸机”，无法完成任何工作。硬件为软件提供运行平台。软件和硬件相互关联，两者之间可以相互转化、互为补充。计算机的软件分成系统软件和应用软件两大类。

- 软件系统
  - 系统软件
    - 操作系统
      - Windows：窗口式操作系统
      - Dos：磁盘操作系统
      - UNIX：网络操作系统
    - 各种计算机语言处理程序
      - 机器语言
      - 汇编语言
      - 高级语言
    - 数据库管理系统
    - 支持软件
  - 应用软件
    - 程序库
    - 软件包
    - 套装软件
    - 用户程序

一、系统软件

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。系统软件使得计算机使用者和其他软件将计算机当作一个整体而不需要顾及到底层每个硬件是如何工作的。

常用的操作系统：

1. 桌面操作系统从软件上可主要分为两大类，分别为类 Unix 操作系统和 Windows 操作系统。

Unix 和类 Unix 操作系统：Mac OS X，Linux 发行版（如 Debian，Ubuntu，Linux Mint，openSUSE，Fedora，Mandrake，Red Hat，Centos）；

微软公司 Windows 操作系统 [5]：Windows 98，Windows 2000，Windows XP，Windows Vista，Windows 7，Windows 8，Windows 8.1，Windows10 等。

2. 服务器操作系统。

服务器操作系统主要集中在三大类：

Unix 系列：SUNSolaris，IBM－AIX，HP－UX，FreeBSD，OS X Server [6]等；

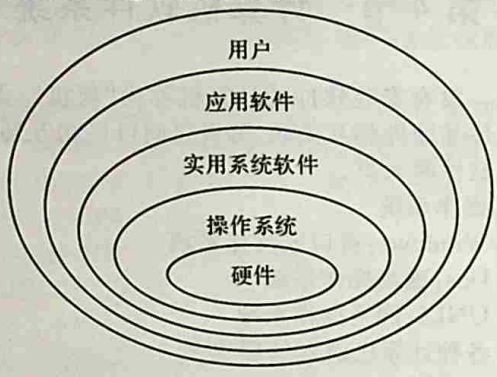
Linux 系列：Red Hat Linux，CentOS，Debian，UbuntuServer 等；

Windows 系列：Windows NT Server，Windows Server 2003，Windows Server 2008，Windows Server 2008 R2，windows server 2012，windows server technical 等。

二、应用软件

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。这类软件能解决特定问题。应用软件与系统软件的关系是：系统软件为应用软件提供基础和平台，没有系统软件应用的软件是无源之本，反过来应用软件又为系统软件服务。

常用的应用软件有以下几类:(1)字处理软件;(2)电子制表软件;(3)计算机辅助设计软件;(4)图形软件;(5)教育软件;(6)电子游戏软件。



三、计算机的指令

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。例如,80386 的指令系统共有 123 种指令,可分为 9 类指令操作:数据传递、算术运算、逻辑运算、串操作、位操作、程序控制、高级语言指令、保护模式、处理器控制指令。

程序是计算机为了执行某种操作任务而将一条条指令按照一定的顺序排列起来的指令集。

【课堂练习】

1.【NOIP1999】计算机能直接执行的指令包括两部分,它们是(　　)。

A. 源操作数与目标操作数　　B. 操作码与操作数

C. ASCII 码与汉字代码　　D. 数字与字符

【答案】B

【分析】计算机的指令系统是由操作码和操作数组成的。

2.【NOIP1999】计算机的软件系统通常分为(　　)。

A. 系统软件与应用软件　　B. 高级软件与一般软件

C. 军用软件与民用软件　　D. 管理软件与控制软件

【答案】A

【分析】本题是软件系统基本知识题。

3.【NOIP2001 普及组】Word 是一种(　　)。

A. 操作系统　　B. 文字处理软件　　C. 多媒体制作软件　　D. 网络浏览器

【答案】B

【分析】Word 是 Microsoft 公司推出的办公自动化套装软件 Office 中的字处理软件。

4.【NOIP2001 普及组】应用软件和系统软件的相互关系是(　　)。

A. 后者以前者为基础　　B. 前者以后者为基础

C. 每一类都以另一类为基础　　D. 每一类都不以另一类为基础

【答案】B

【分析】系统软件是基础,没有它系统无法运行。

5.【NOIP2003 普及组】下列哪个软件不是操作系统软件的名字(　　)。

A. Windows XP　　B. DOS　　C. Linux　　D. OS/2　　E. Arch/Info

【答案】E

【分析】Windows XP 是微软公司的操作系统，DOS 是微软公司早期的操作系统，Linux 是自由操作系统，OS/2 是 IBM 公司的操作系统。

6.【NOIP2003 普及组】数字图像文件可以用下列哪个软件来编辑(　　)。

A. 画笔(Paintbrush)　　B. 记事簿(Notepad)

C. Recorder　　D. WinRAR

【答案】A

【分析】画笔(Paintbrush)是 Windows 操作系统自带的绘图软件，Photoshop 是美国 Adobe 软件公司的专业图像处理软件，两者都可以编辑图像文件。记事簿(Notepad)只能处理文本文件，WinRAR 是 Eugene Roshal 的共享压缩软件。

7.【NOIP2004 普及组】下列哪个不是数据库软件的名称(　　)。

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

【答案】D

【分析】A. MySQL 是一个小型关系型数据库管理系统，开发者为瑞典 MySQL AB 公司。在 2008 年 1 月 16 日被 Sun 公司收购。

B. SQL(Structured Query Language)，结构化查询语言。SQL Server 是一个关系数据库管理系统。它最初是由 Microsoft、Sybase 和 Ashton-Tate 三家公司共同开发的，于 1988 年推出了第一个 OS/2 版本。

C. Oracle 是殷墟(Yin Xu)出土的甲骨文(oracleboneinscriptions)的英文翻译的第一个单词，在英文里是“神谕”的意思。Oracle 数据库产品为财富排行榜上的前 1000 家公司所采用，许多大型网站也选用了 Oracle 系统。

E. Visual FoxPro 原名 FoxBase，最初是由美国 Fox Software 公司于 1988 年推出的数据库产品。

8.【NOIP2005 普及组】以下哪个软件不是即时通信软件(　　)。

A. 网易泡泡　　B. MSN Messenger　　C. Google Talk

D. 3DS Max　　E. QQ

【答案】D

【分析】这是 3D 造型软件。

9.【NOIP2005 提高组】不能在 Linux 上使用的网页浏览器是(　　)。

A. Internet Explore　　B. Netscape　　C. Opera

D. Firefox　　E. Mozilla

【答案】A

【分析】微软的，未跨平台。

10.【NOIP2006 普及组】Linux 是一种(　　)。

A. 绘图软件　　B. 程序设计语言　　C. 操作系统　　D. 网络浏览器

【答案】C

【分析】Linux 是一种自由和开放源码的类 UNIX 操作系统。目前存在着许多不同的 Linux，但它们都是用了 Linux 内核。Linux 可安装在各种计算机硬件设备中，从手机、平板电脑、路由器和视频游戏控制台，到台式计算机、大型机和超级计算机。Linux 是一个领先的操作系统，世界上运算最快的 10 台超级计算机运行的都是 Linux 操作系统。严格来讲，Linux 这个词本身只表示 Linux 内核，但实际上人们已经习惯了用 Linux 来形容整个基于 Linux 内核，并且使用 GNU 工程各种工具和数据库的操作系统。Linux 操作系统是 UNIX 操作系统的一个克隆版本。UNIX 操作系统是美国贝尔实验室的肯·汤普逊(Ken Thompson)和丹尼斯·里奇(Dennis Ritchie)于 1969 年夏在 DEC PDP-7 小型计算机上开发的一个分时操作系统。

第5节　计算机语言

程序就是一系列的操作步骤，计算机程序就是由人事先规定的计算机完成某项工作的操作步骤。每一步骤的具体内容由计算机能够理解的指令来描述，这些指令告诉计算机“做什么”和“怎样做”。编写计算机程序所使用的语言称为程序设计语言。

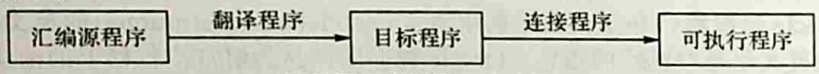
通常分为三类：机器语言、汇编语言和高级语言。

1. 机器语言

计算机最早的语言处理程序是机器语言，它是计算机能直接识别的语言，而且速度快。机器语言是用二进制代码来编写计算机程序的，因此又称二进制语言。例如用机器语言来表示“8+4”这个算式，是一串二进制码“00001000 00000100 00000100”。机器语言书写困难、记忆复杂，一般很难掌握。

2. 汇编语言

由于机器语言的缺陷，人们开始用助记符编写程序，用一些符号代替机器指令所产生的语言称为汇编语言。但是，用汇编语言编写的源程序不能被计算机直接识别，必须使用某种特殊的软件将用汇编语言写的源程序翻译和连接成能被计算机直接识别的二进制代码。其示意图如图所示。



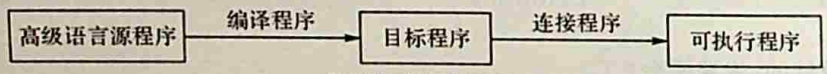
汇编源程序翻译连接过程

汇编语言虽然采用了助记符来编写程序，比机器语言简单，但是汇编语言仍属于低级语言，它与计算机的体系结构有关，在编写程序前要花费相当多的时间和精力去熟悉机器的结构。因此工作量大、繁琐，而且程序可移植性差。

3. 高级语言

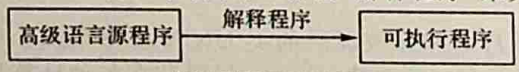
计算机并不能直接接受和执行用高级语言编写的源程序，源程序在输入计算机时，通过“翻译程序”翻译成机器语言形式的目标程序，计算机才能识别和执行。这种“翻译”通常有两种方式，即编译方式和解释方式。

编译方式是：编译方式的翻译工作由“编译程序”来完成，它是先将整个源程序都转换成二进制代码，生成目标程序，然后把目标程序连接成可执行的程序，以完成源程序要处理的运算并取得结果。



高级语言编译过程

解释方式是：源程序进入计算机时，解释程序边扫描边解释，对源程序的语句解释一条、执行一条，不产生目标程序。解释方式的翻译工作由“解释程序”来完成。



高级语言解释过程

编译性语言有 C/C++、Pascal/Object Pascal(Delphi)等。

解释性语言有 ASP、PHP、Java、JavaScript、VBScript、Perl、Python、Ruby、MATLAB 等。

使用编译语言程序将整个源程序编译连接为可执行的文件，这种方式效率高、可靠性高、可移植性好。不过，当源程序修改后，必须重新编译。

面向对象语言借鉴了 20 世纪 50 年代的人工智能语言 LISP，引入了动态绑定的概念和交互式开发环境的思想；始于 20 世纪 60 年代的离散事件模拟语言 Simula67，引入了类的要领和继承，成形于 20 世纪 70 年代的 Smalltalk。

面向对象语言的发展有两个方向：一种是纯面向对象语言，如 Smalltalk、EIFFEL 等；另一种是混合型面向对象语言，即在过程式语言及其他语言中加入类、继承等成分，如 C++、Objective-C 等。

【课堂练习】

1.【NOIP2001 普及组】解释程序的功能是(　　)。

A. 将高级语言程序转换为目标程序　　B. 将汇编语言程序转换为目标程序

C. 解释执行高级语言程序　　D. 解释执行汇编语言程序

【答案】C

【分析】解释程序是高级语言翻译程序的一种，它将源语言(如 Basic)书写的源程序作为输入，解释一句后就提交计算机执行一句，并不形成目标程序。

2.【NOIP2002 普及组】下列哪一种程序设计语言是解释执行的(　　)。

A. Pascal　　B. Gwbasic　　C. C++　　D. Fortran

【答案】B

【分析】GWBasic 是高级程序设计语言 Basic 的一个方言版本，GWBasic 输入一句执行一句，不需要编译。最近比较火的解释性语言应该是 python，有跨平台的优点。

3.【NOIP2003 普及组】下列关于程序语言的叙述，不正确的是(　　)。

A. 编写机器代码不比编写汇编代码容易

B. 高级语言需要编译成目标代码或通过解释器解释后才能被 CPU 执行

C. 同样一段高级语言程序通过不同的编译器可能产生不同的可执行程序

D. 汇编代码可被 CPU 直接运行

E. 不同的高级语言语法略有不同

【答案】D

【分析】汇编语言是一种最简单的编程语言，CPU 是不能直接识别的，需要转换成机器语言才能执行。

4.【NOIP2004 普及组】下列哪个程序设计语言不支持面向对象程序设计方法(　　)。

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

【答案】C

【分析】A. 美国 AT&T 贝尔实验室的本贾尼·斯特劳斯特卢普(Bjarne Stroustrup)博士在 20 世纪 80 年代初期发明并实现了 C++(最初这种语言被称作“C with Classes”)。

B. Object Pascal 语言是 Pascal 之父在 1985 年于 apple macintosh 机器上实现的。后来 Borland 公司也在它的 Pascal 产品 Turbo Pascal/Delphi 中实现了 Object Pascal。Object Pascal 是一种高级编译语言，具有强类型(对数据类型的检查非常严格)特性，支持结构化和面向对象编程。

C. C 语言是一种面向过程的计算机程序设计语言，最初为 UNIX 而生。它既有高级语言的特点，又具有汇编语言的特点。它可以作为系统设计语言，编写工作系统应用程序，也可以作为应用程序设计语言，编写不依赖计算机硬件的应用程序。

D. Smalltalk 被公认为历史上第二个面向对象的程序设计语言和第一个真正的集成开发环境(IDE)。

E. Java 是一种可以撰写跨平台应用软件的面向对象的程序设计语言，是由 Sun Microsystems 公司于 1995 年 5 月推出的 Java 程序设计语言和 Java 平台(即 JavaSE，JavaEE，JavaME)的总称。

5.【NOIP2006 普及组】在下列关于计算机语言的说法中，不正确的是(　　)。

A. Pascal 和 C 都是编译执行的高级语言

B. 高级语言程序比汇编语言程序更容易从一种计算机移植到另一种计算机上

C. C++是历史上的第一个支持面向对象的计算机语言

D. 与汇编语言相比，高级语言程序更容易阅读

【答案】C

【分析】历史上的第一个支持面向对象的计算机语言是 1967 年挪威计算中心的 Kisten Nygaard 和 Ole Johan Dahl 开发的 Simula67 语言，它提供了比子程序更高一级的抽象和封装，引入了数据和类的概念，它被认为是第一个面向对象的语言，高级语言是面向开发人员的，所以看起来很容易明白，但是转化为机器语言后，通常会占用很大的内存。

汇编通常是针对芯片来的，也需要转化为机器语言，机器才能识别，但是它占用的内存很低，执行的稳定性高。

高级语言和低级语言的区别：如果两者都是 32 位应用程序，那么汇编程序由 CPU 指令直接汇编而成，且指令助记码与机器码之间一一对应，故而精简。

高级语言需要先通过编译器将高级语言源程序汇编成汇编程序，然后再由汇编器和连接器生成最终程序，编译时高级语言将被转化为数倍于自身的汇编语言，虽然有编译器的优化，但还是没有手动写出的汇编程序精简，这就造成了代码量（容量）的扩大，更多的代码通常意味着要消耗更多的 CPU 周期去执行，这样，单位时间的执行速率也就相应延缓，汇编只适合开发小型软件、接口程序，不宜用来开发大型软件，反之，高级语言适用于大型软件的开发。

Smalltalk 是历史上第二个面向对象的程序设计语言。

6.【NOIP2007 普及组】在下列关于计算机语言的说法中，正确的有（　　）。

A. 高级语言比汇编语言更高级，是因为它的程序的运行效率更高

B. 随着 Pascal、C 等高级语言的出现，机器语言和汇编语言已经退出了历史舞台

C. 高级语言程序比汇编语言程序更容易从一种计算机移植到另一种计算机上

D. C 是一种面向对象的高级计算机语言

【答案】C

【分析】高级语言比汇编语言更高级，是针对计算机语言的发展阶段讲的。人们使用高级语言编写程序，要比汇编语言容易得多。优秀的程序设计人员用汇编语言编写的程序，往往效率更高一些。机器语言和汇编语言并没有退出历史舞台。一些和硬件操作（特别是设计外部设备的操作）关系十分密切的程序，往往还需要用汇编语言编写。

第 6 节　数制转换

一、进位计数制的基本概念

将数字符号按序排列成数位，并遵照某种由低位到高位的进位方式计数表示数值的方法，称作进位计数制。

1. 十进制

十进制计数制由 0、1、2、3、4、5、6、7、8、9 共 10 个数字符号组成。相同数字符号在不同的数位上表示不同的数值，每个数位计满十就向高位进一，即“逢十进一”。

2. 八进制

八进制计数制由 0、1、2、3、4、5、6、7 共 8 个数字符号组成。相同数字符号在不同的数位上表示不同的数值，每个数位计满八就向高位进一，即“逢八进一”。

3. 二进制

二进制计数制由 0 和 1 共两个数字符号组成。相同数字符号在不同的数位上表示不同的数值，每个数位计满二就向高位进一，即“逢二进一”。

4. 其他进制

在日常生活和工作中还会使用其他进制数。如：十二进制数、十六进制数、百进制数和千进制数等。无论哪种进制数，表示的方法都是类似的。如：十六进制数由 0、1、2、3、4、5、6、7、8、9、A、B、C、D、E 和 F 共 16 个符号组成，“逢十六进一”。不同的是，用 A、B、C、D、E 和 F 分别表示 10、11、12、13、14 和 15 六个数字符号。

5. 基数与权

某进制计数制允许选用的基本数字符号的个数称为基数。一般而言，J 进制数的基数为 J，可供选用的基本数字符号有 J 个，分别为 0 到 J－1，每个数位计满 J 就向高位进一，即“逢 J 进一”。

某进制计数制中各位数字符号所表示的数值表示该数字符号值乘以一个与数字符号所处位置有关的常数，该常数称为“位权”（简称“权”）。位权的大小是以基数为底、数字符号所处的位置的序号为指数的整数次幂。

十进制数允许使用十个基本数字符号，所以基数为 10，每位数字符号代表的位数的大小是以 10 为底，数字符号所处位置的序号为指数的整数次幂。

十进制数的百位、十位、个位和十分位的权分别为 10^2、10^1、10^0、10^{-1}。故 $(555.5)_{10}$ 可表示成 $(555.5)_{10}=5*10^2+5*10^1+5*10^0+5*10^{-1}$。J 进制数相邻两位数相差 J 倍，若小数点向左移 n 位，则整个数值就缩小 J^n。反之，小数点向右移 n 位，数值就放大 J^n。

如下给出了任意进制数（K^2、K^1、K^0、K^{-1}、K^{-2}），当 J 分别为 2、8、10 和 16 时各位权值对照。

数位 / 枚 / 进位制 J	K^2	K^1	K^0	小数点位置	K^{-1}	K^{-2}
J=2	$2^2=4$	$2^1=2$	$2^0=1$		$2^{-1}=0.5$	$2^{-2}=0.25$
J=8	$8^2=64$	$8^1=8$	$8^0=1$		$8^{-1}=0.125$	$8^{-2}=0.015625$
J=10	$10^2=100$	$10^1=10$	$10^0=1$		$10^{-1}=0.1$	$10^{-2}=0.01$
J=16	$16^2=256$	$16^1=16$	$16^0=1$		$16^{-1}=0.0625$	$16^{-2}=0.00390625$

二、数制之间的转换

计算机内部使用的数字符号只有“0”和“1”两个。也就是说，计算机内部使用的是二进制数，所有的数值数据和非数值数据，都是由“0”和“1”这两个数字符号加以组合而成的，我们称之为“二进制代码”。

计算机只用二进制的两个数码“0”和“1”来实现算术和逻辑运算，而人们仍然用十进制的形式向计算机中输入原始数据，并让计算机也用十进制形式显示和打印运算结果。所以，必须有一种自动转换方法，即让数据输入计算机后，将十进制转换成对应的二进制数，并在处理完毕后，再自动将二进制结果转换为十进制数。

为了表达方便起见，常在数字后加一缩写字母后缀作为不同进制数的标识。各种进制数的后缀字母分别为：

B：二进制数。　　O：八进制数。

D：十进制数。　　H：十六进制数。

对于十进制数，通常不加后缀，也即十进制数后的字母 D 可省略。

1. 二进制与十进制的转换

(1)二进制转十进制

方法：“按权展开求和”。

例：$(1011.01)_2=(1*2^3+0*2^2+1*2^1+1*2^0+0*2^{-1}+1*2^{-2})_{10}$

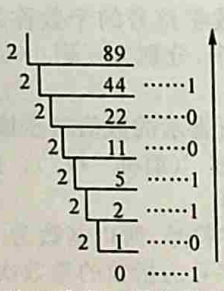
$=(8+0+2+1+0+0.25)_{10}$

$=(11.25)_{10}$

(2)十进制转二进制

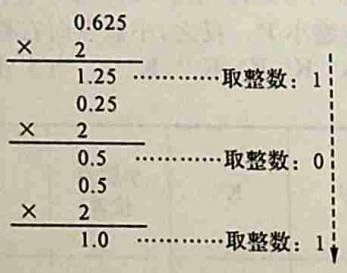
· 十进制整数转二进制数：“除以 2 取余，逆序输出”。

例：$(89)_{10}=(1011001)_2$



· 十进制小数转二进制数：“乘以 2 取整，顺序输出”。

例：$(0.625)_{10}=(0.101)_2$



2. 八进制与二进制的转换

例：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：将二进制的 10110.0011 转换成八进制。

$\frac{010}{2}\ \frac{110}{6}.\frac{001}{1}\ \frac{100}{4}$

即：$(10110.0011)_2=(26.14)_8$。

3. 十六进制与二进制的转换

例：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：将二进制数 1100001.111 转换成十六进制。

0110　0001　.1110

6　1　.　E

即：$(1100001.111)_2=(61.E)_{16}$。

4. 把一个八进制转换成十进制

把这个八进制的最后一位乘上 8^0，倒数第二位乘上 8^1，……，一直到最高位乘上 8^n，然后将各项乘积相加，结果即为它的十进制表达式。

例：把八进制 36 转换为十进制。

$(36)_8=3*8^1+6*8^0=24+6=(30)_{10}$

5. 把一个十六进制转换成十进制

把这个十六进制的最后一位乘上 16^0，倒数第二位乘上 16^1，……，一直到最高位乘上 16^n，然后将各项乘积相加，结果即为它的十进制表达式。

例：把十六制 1E 转换为十进制。

$(1E)_{16}=1*16^1+14*16^0=16+14=(30)_{10}$

【课堂练习】

1.【NOIP1998】用十六进制、八进制和十进制写了如下一个等式：52－19＝33，式中三个数是各不相同进位制的数。则 52、19、33 分别为(　　)。

A. 八进制，十进制，十六进制　　B. 十进制，十六进制，八进制

C. 八进制，十六进制，十进制　　D. 十进制，八进制，十六进制

【答案】B

【分析】不需要死算，恒等变形为 52＝19＋33，十进制时成立，假设 52 改为八进制，19 和 33 中的一个改为十六进制，则等式左边变小，等式右边变大，等式不再平衡，所以 A 和 C 都不对。D 选项中，19 不可能是八进制数，所以 D 肯定不对。

2.【NOIP1999】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(　　)。

A. 10111100101　　B. 11111100101　　C. 11110100101　　D. 11111101101

【答案】B

【分析】十进制表达式 3 * 512＋7 * 64＋4 * 8＋5 可以看成：

$3*8^3+7*8^2+4*8^1+5*8^0=(3745)_8=(011111100101)_2=(11111100101)_2$

3.【NOIP2000】下列无符号数中，最小的数是(　　)。

A. $(11011001)_2$　　B. $(75)_{10}$　　C. $(37)_8$　　D. $(2A)_{16}$

【答案】C

【分析】将其转换成同一进制数进行比较即可，如全部转换成二进制。

4.【NOIP2001 普及组】与二进制数 101.01011 等值的十六进制数为(　　)。

A. A.B　　B. 5.51　　C. A.51　　D. 5.58

【答案】D

【分析】二进制转化为十六进制：4 位换一位，不足四位补 0(整数部分是从个位向大位数 4

位，不足在最前面补0；小数部分就是按顺序4位一个，不足4位在最后补0）。

101.01011＝0101.0101 1000，则转化为5.58。

5.【NOIP2002 普及组】$(0.5)_{10}=(\quad)_{16}$。

A. 0.1　　B. 0.75　　C. 0.8　　D. 0.25

【答案】C

【分析】十进制下的0.5等于十六进制下的0.8，因为5/10＝8/16（考虑“半斤八两”）。

6.【NOIP2002】算式$(2047)_{10}-(3FF)_{16}+(2000)_8$的结果是（　　）。

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

【答案】A

【分析】2047－3＊16＊16－15＊16－15＋2＊8＊8＊8＝2047－16＊48－16＊15－15＋16＊64＝2047＋1＝2048。

7.【NOIP2002 提高组】十进制数11/128可用二进制数码序列表示为（　　）。

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

【答案】D

【分析】用乘2取整法。

11/128＝0.0859375 .....整数部分

0.0859375＊2＝0.171875 0

0.171875＊2＝0.34375 0

0.34375＊2＝0.6875 0

0.6875＊2＝1.375 .....1 去掉整数的1

0.375＊2＝0.75 0

0.75＊2＝1.5 .....1 去掉整数的1

0.5＊2＝1 .....1 去掉整数的1，去掉后为0，结束

然后从上往下排，前面加“0”，所以结果为0.0001011。

8.【NOIP2003】十进制数2003等值于二进制数（　　）。

A. 11111010011　B. 10000011　　C. 110000111　　D. 010000011　　E. 1111010011

【答案】A

【分析】用除2法取余法。

9.【NOIP2004 提高组】十进制数100.625等值于二进制数（　　）。

A. 1001100.101　B. 1100100.101　C. 1100100.011　D. 1001100.11　　E. 1001100.01

【答案】B

【分析】正常是乘2取整，即0.625＝0.5＋0.125＝1/2＋1/8，对应二进制0.1和0.001。

10.【NOIP2005 提高组】以下二进制数的值与十进制数23.456的值最接近的是（　　）。

A. 10111.0101　B. 11011.1111　C. 11011.0111　D. 10111.0111　E. 10111.1111

【答案】D

【分析】求最接近，整数部分容易确定，小数部分按位权展开，$(0.11)_2=(0.4375)_{10}$。

11.【NOIP2006 提高组】与十进制数1770.625对应的八进制数是（　　）。

A. 3352.5　　B. 3350.5　　C. 3352.1161　　D. 3350.1151

【答案】A

【分析】1770转换为八进制是除以8的余数反序，即3352。小数部分是乘以8的整数部分正序，即0.625＊8＝5.000（取5）。结果为3352.5。

第 7 节　信息编码表示

一、基本概念

1. 编码

计算机要处理的数据除了数值数据以外，还有各类符号、图形、图像和声音等非数值数据。而计算机只能识别两个数字。要使计算机能处理这些信息，首先必须将各类信息转换成“0”和“1”表示的代码，这一过程称为编码。

2. 数据

能被计算机接受和处理的符号的集合都称为数据。

3. 比特

比特(Bit:——二进制数位)是指 1 位二进制的数码(即 0 或 1)。比特是计算机中表示信息的数据编码中的最小单位。

4. 字节

字节表示被处理的一组连续的二进制数字。通常用 8 位二进制数字表示一个字节，即一个字节由 8 个比特组成。

字节是存储器系统的最小存取单位。

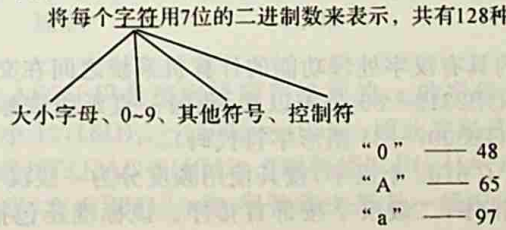
二、字符的表示

字符是人与计算机交互过程中不可缺少的重要信息。要使计算机能处理、存储字符信息，首先必须用二进制 “0”和 “1”代码对字符进行编码。

下面以西文字符和汉字字符为例，介绍常用的编码标准。

三、ASCII 编码

ASCII 编码是由美国国家标准委员会制定的一种包括数字、字母、通用符号和控制符号在内的字符编码集，全称为美国国家信息交换标准代码(American Standard Code for Information Interchange)。ASCII 码是一种 7 位二进制编码，能表示 $2^7=128$ 种国际上最通用的西文字符，是目前计算机中，特别是微型计算机中使用最普遍的字符编码集。



ASCII 编码包括 4 类最常用的字符。

①数字“0”～“9”。ASCII 编码的值分别为 0110000 B ～ 0111001 B，对应十六进制数为 30 H～39 H。

②26 个英文字母。大写字母 “A” ～ “Z” 的 ASCII 编码值为 41 H～5AH，小写字母 “a” ～ “z”的 ASCII 编码值为 61 H ～ 7AH。

③通用符号。如“+”“-”“=”“*”和“/”等共 32 个。

④控制符号。如空格符和回车符等共 34 个。

ASCII 码是一种 7 位编码，它存储时必须占全一个字节，也即占用 8 位：b7、b6、b5、b4、b3、b2、b1、b0，其中 b7 恒为 0，其余几位为 ASCII 码值。

人们可以通过键盘输入和显示器显示不同的字符，但在计算机中，所有信息都是用二进

制代码表示的。n 位二进制代码能表示 2^n 个不同的字符，这些字符的不同组合就可表示不同的信息。为使计算机使用的数据能共享和传递，必须对字符进行统一的编码。ASCII 码（美国标准信息交换码）是使用最广泛的一种编码。ASCII 码由基本的 ASCII 码和扩充的 ASCII 码组成。在 ASCII 码中，把二进制位最高位为 0 的数字都称为基本的 ASCII 码，其范围是 0～127；把二进制位最高位为 1 的数字都称为扩展的 ASCII 码，其范围是 128～255。

四、内码和外码

内码：对于输入计算机的文本文件，机器是存储其相应的字符的 ASCII 码（用一个 ASCII 码存储一个字符需 8 个二进制位，即一个字节），这些可被计算机内部进行存储和运算使用的数字代码称内码。如输入字符“A”，计算机将其转成内码 65 后存于内存。

外码：计算机与人进行交换的字形符号称为外码，如字符“A”的外码是“A”。

通常一个西文字符占一个字节（半角），一个中文字符占两个字节。

十进制	字符	十进制	字符
48	0	97	a
49	1	98	b
50	2	99	c
…	…	…	…
57	9	122	z
65	A		
66	B		
67	C		
…			
90	z		

五、汉字信息编码

1. 汉字交换码

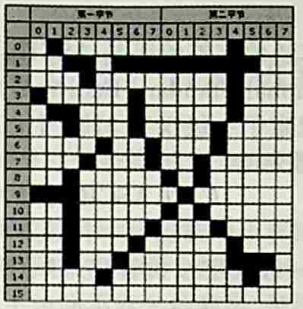
汉字交换码是指不同的具有汉字处理功能的计算机系统之间在交换汉字信息时所使用的代码标准。自国家标准 GB2312—80 公布以来，我国一直延用该标准所规定的国标码作为统一的汉字信息交换码（GB5007—85 图形字符代码）。

GB2312—80 标准包括了 6763 个汉字，按其使用频度分为一级汉字 3755 个和二级汉字 3008 个。一级汉字按拼音排序，二级汉字按部首排序。该标准还包括标点符号、数种西文字母、图形、数码等符号 682 个。

区位码的区码和位码均采用从 01 到 94 的十进制，国标码采用十六进制的 21 H 到 73 H（数字后加 H 表示其为十六进制数）。区位码和国标码的换算关系是：区码和位码分别加上十进制数 32。如“国”字在表中的 25 行 90 列，其区位码为 2590，国标码是 397AH。

2. 字形存储码

字形存储码是指供计算机输出汉字（显示或打印）用的二进制信息，也称字模。通常，采用的是数字化点阵字模。



16×16点表示

一般的点阵规模有 16 * 16、24 * 24 等，每一个点在存储器中用一个二进制位(bit)存储。在 16 * 16 的点阵中，需 8 * 32 bit 的存储空间，每 8 bit 为 1 字节，所以，需 32 字节的存储空间。在相同点阵中，不管其笔画繁简，每个汉字所占的字节数相等。

为了节省存储空间，普遍采用字形数据压缩技术。所谓矢量汉字，是指用矢量方法将汉字点阵字模进行压缩后得到的汉字字形的数字化信息。

【课堂练习】

1.【NOIP1999】在 24 * 24 点阵的字库中，汉字“一”与“编”的字模占用字节数分别是(　　)。

A. 32、32　　B. 32、72　　C. 72、72　　D. 72、32

【答案】C

【分析】24 * 24 点阵的字模需要 24 * 24 个二进制位来存储，每个字节有 8 个二进制位。所以，一个汉字字模占用字节数为 24 * 24/8＝72 个字节。汉字“一”与“编”尽管简繁不一样，但所需的空间是一样的。

2.【NOIP1999 普及组】在计算机中，ASCII 码是(　　)位二进制代码。

A. 8　　B. 7　　C. 12　　D. 16

【答案】A

【分析】在计算机中一个字符是用一个字节(8 位二进制位)来表示的，低 7 位是这个字符的 ASCII 码，最高位通常恒为“0”，所以 ASCII 码是用 7 位二进制数来表示一个字符的。

3.【NOIP1999】已知小写字母“m”的十六进制的 ASCII 码值是 6D，则小写字母“c”的十六进制数的 ASCII 码值是(　　)。

A. 98　　B. 62　　C. 99　　D. 63

【答案】D

【分析】小写字母的 ASCII 码是按字母顺序排列的，c 的字母排列位置在 m 的前 10 个，ASCII 码也就比 m 小 10，$(6D)_{16}-(10)_{10}=(63)_{16}$，因此答案是 D。

4.【NOIP1999】组成“教授”(JIAO SHOU)、“副教授”(FU JIAO SHOU)与“讲师”(JIANG SHI)这三个词的汉字，在 GB2312—80 字符集中都是一级汉字，对这三个词排序的结果是(　　)。

A. 教授、副教授、讲师　　B. 副教授、教授、讲师

C. 讲师、副教授、教授　　D. 副教授、讲师、教授

【答案】D

【分析】GB2312—80 是我国于 1981 年颁布的《信息交换用汉字编码字符集》，字符集中共收录 6763 个汉字，其中一级字库 3755 个，按拼音排序，二级字库 3008 个，按偏旁部首排序，另外还有 682 个图文符号。因副教授、讲师、教授三个词的汉字都在一级字库，所以按拼音排序是副教授、讲师、教授。

5.【NOIP1999 提高组】在计算机，字符编码通常采用(　　)。

A. 原码　　B. 反码　　C. ASCII 码　　D. 补码

【答案】C

【分析】在计算机系统中，应用最为广泛的字符编码是 ASCII 码。美国信息交换标准码是由美国国家标准学会(American National Standard Institute，ANSI)制定的。

6.【NOIP2000 普及组】GB2312—80 规定了一级汉字 3755 个，二级汉字 3008 个，其中二级汉字字库中的汉字是以(　　)为序排列的。

A. 以笔画多少　　B. 以部首　　C. 以 ASCII 码　　D. 以机内码

【答案】B

【分析】国标 GB2312—80 中收集的一级汉字按拼音字母顺序排序，而二级汉字按偏旁部首排序。

7.【NOIP2001 普及组】在计算机内部，一切信息存取、处理和传递的形式是(　　)。

A. ASCII 码　　B. BCD 码　　C. 二进制　　D. 十六进制

【答案】C

【分析】输入计算机的任何信息最终都要转化为二进制，目前通用的是 ASCII 码，最基本的单位为 bit。

8.【NOIP2001】2 KB 的内存能存储(　　)个汉字的机内码。

A. 1024　　B. 516　　C. 2048　　D. 218

【答案】A

【分析】一个汉字机内码占 2 个字节，2 KB 是 2 * 1024 个字节，因此能存储 1024 个汉字机内码。

9.【NOIP2001 提高组】64 KB 的存储器用十六进制表示，它的最大的地址码是(　　)。

A. 10000　　B. FFFF　　C. 1FFFF　　D. EFFFF

【答案】B

【分析】64 KB 即 64 * 1024 字节，由 $64=2^6$、$1024=2^{10}$ 得 $64*1024=2^{16}$，要表示 2^{16} 个存储单元，需 16 根地址线，最大地址码为 1111111111111111，用十六进制表示 FFFF。

10.【NOIP2003 普及组】下列说法中，正确的是(　　)。

A. 在内存中，可执行程序用二进制码表示，源程序用八进制表示

B. 程序和数据在内存中都是用二进制码表示的

C. 内存中数据的存取是以二进制位为单位的

D. 中央处理器 CPU 执行的每条指令的长度都不同

E. 一般来说，在计算机内部，中文信息用十六进制表示，英文信息用八进制表示

【答案】B

【分析】在计算机内部，所有数据都是以二进制编码形式表示的。

11.【NOIP2007】ASCII 码的含义是(　　)。

A. 二、十进制转换码　　B. 美国信息交换标准代码

C. 数字的二进制编码　　D. 计算机可处理字符的唯一编码

【答案】B

【分析】American Standard Code for Information Interchange，美国信息交换标准代码，ASCII 是缩写，开始是 128 个。

第 8 节　计算机安全知识

计算机安全中最重要的是存储数据的安全，其面临的主要威胁包括：计算机病毒、非法访问、计算机电磁辐射、硬件损坏等。

计算机病毒是附在计算机软件中的隐蔽的小程序，它和计算机其他工作程序一样，但会破坏正常的程序和数据文件。恶性病毒可使整个计算机软件系统崩溃，数据全毁。要防止病毒侵袭主要是加强管理，不访问不安全的数据，使用杀毒软件并及时升级更新。

由于计算机硬件本身就是向空间辐射的强大的脉冲源，和一个小电台差不多，频率在几十千周到上百兆周。盗窃者可以接收计算机辐射出来的电磁波，进行复原，获取计算机中的数据。为此，计算机制造厂家增加了防辐射的措施，从芯片、电磁器件到线路板、电源、转盘、硬盘、显示器及连接线，都全面屏蔽起来，以防电磁波辐射。更进一步，可将机房或整个办公大楼都屏蔽起来，如没有条件建屏蔽机房，可以使用干扰器，发出干扰信号，使接收者无法正常接收有用信号。

计算机存储器硬件损坏，使计算机存储数据读不出来也是常见的事。防止这类事故的发生有几种办法，一是将有用数据定期复制出来保存，一旦机器有故障，可在修复后把有用数据复制回去。二是在计算机中使用 RAID 技术，同时将数据存在多个硬盘上；在安全性要求高的特殊场合还可以使用双主机，一台主机出问题，另外一台主机照样运行。

计算机硬件安全

计算机在使用过程中，对外部环境有一定的要求，即计算机周围的环境应尽量保持清洁、温度和湿度应该合适，电压稳定，以保证计算机硬件可靠地运行。计算机安全的另外一项技术就是加固技术，经过加固技术生产的计算机防震、防水、防化学腐蚀，可以使计算机在野外全天候运行。

从系统安全的角度来看，计算机的芯片和硬件设备也会对系统安全构成威胁。比如 CPU，电脑 CPU 内部集成有运行系统的指令集，这些指令代码都是保密的，我们并不知道它的安全性如何。据有关资料透漏，国外针对中国所用的 CPU 可能集成有陷阱指令、病毒指令，并设有激活办法和无线接收指令机构。他们可以利用无线代码激活 CPU 内部指令，造成计算机内部信息外泄、计算机系统灾难性崩溃。如果这是真的，那我们的计算机系统在战争时期有可能全面被攻击。

硬件泄密甚至涉及了电源。电源泄密的原理是通过市电电线，把电脑产生的电磁信号沿电线传出去，利用特殊设备可以从电源线上就把信号截取下来还原。

计算机里的每一个部件都是可控的，所以叫做可编程控制芯片，如果掌握了控制芯片的程序，就控制了电脑芯片。只要能控制，那么它就是不安全的。因此，我们在使用计算机时首先要注意做好电脑硬件的安全防护，把我们所能做到的全部做好！

常用防护策略：

(1)安装杀毒软件。

对于一般用户而言，首先要做的就是为电脑安装一套杀毒软件，并定期升级所安装的杀毒软件，打开杀毒软件的实时监控程序。

(2)安装个人防火墙。

安装个人防火墙(Fire Wall)以抵御黑客的袭击，最大限度地阻止网络中的黑客来访问你的计算机，防止他们更改、拷贝、毁坏你的重要信息。防火墙在安装后要根据需求进行详细配置。

(3)分类设置密码并使密码设置尽可能复杂。

在不同的场合使用不同的密码，如网上银行、E-Mail、聊天室以及一些网站的会员等。

应尽可能使用不同的密码，以免因一个密码泄露而导致所有资料外泄。对于重要的密码（如网上银行的密码）一定要单独设置，并且不要与其他密码相同。

设置密码时要尽量避免使用有意义的英文单词、姓名缩写以及生日、电话号码等容易泄露的字符作为密码，最好采用字符、数字和特殊符号混合的密码。建议定期修改自己的密码，这样可以确保即使原密码泄露，也能将损失减小到最少。

(4)不下载不明软件及程序。

应选择信誉较好的下载网站下载软件，将下载的软件及程序集中放在非引导分区的某个目录，在使用前最好用杀毒软件查杀病毒。

不要打开来历不明的电子邮件及其附件，以免遭受病毒邮件的侵害，这些病毒邮件通常都会以带有噱头的标题来吸引你打开其附件，如果下载或运行了它的附件，就会受到感染。同样也不要接收和打开来历不明的QQ、微信等发过来的文件。

(5)防范流氓软件。

对将要在计算机上安装的共享软件进行甄别选择，在安装共享软件时，应该仔细阅读各个步骤出现的协议条款，特别留意那些有关安装其他软件行为的语句。

(6)仅在必要时共享。

一般情况下不要设置文件夹共享，如果共享文件则应该设置密码，一旦不需要共享时立即关闭。共享时访问类型一般应该设为只读，不要将整个分区设定为共享。

(7)定期备份。

数据备份的重要性毋庸讳言，无论你的防范措施做得多么严密，也无法完全防止“道高一尺，魔高一丈”的情况出现。如果遭到致命的攻击，操作系统和应用软件可以重装，而重要的数据就只能靠你日常的备份了。所以，无论你采取了多么严密的防范措施，也不要忘了随时备份你的重要数据，做到有备无患！

计算机病毒的特性

1. 隐蔽性

计算机病毒程序是人为制造的小巧玲珑的经过精心编制的程序，这就是病毒的源病毒。这种源病毒是一个独立的程序体，源病毒经过扩散生成的再生病毒，往往采用附加或插入的方式隐蔽在可执行程序或数据文件中，可以在几周或几个月内不被人发现，这就是所谓的隐蔽性。

2. 潜伏性

所谓潜伏性，指病毒具有依附于其他媒体的“寄生”能力。

3. 传播性

所谓传播性，指病毒具有极强的再生和扩散能力，潜伏在计算机系统中的病毒，可以不断进行病毒体的再生和扩散，从而使病毒很快扩散到磁盘存储器和整个计算机系统中。

4. 激发性

所谓病毒的激发性，指病毒在一定条件刺激下，病毒程序迅速活跃起来的特性。

5. 破坏性和危害性

计算机病毒程序，从本质上来说，它是一个逻辑炸弹。一旦满足条件要求被激活并发起攻击，就会迅速扩散，使整个计算机系统无法正常运行，所以它具有极大的破坏性和危害性。

【课堂练习】

1.【NOIP2000】计算机病毒的特点是(　　)。

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

【答案】C

【分析】计算机病毒的特点有：寄生性、传染性、潜伏性、隐蔽性、破坏性、可触发性等。

2.【NOIP2001】计算机病毒是(　　)。

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

【答案】B

【分析】病毒指“编制或者在计算机程序中插入的破坏计算机功能或者破坏数据,影响计算机使用并且能够自我复制的一组计算机指令或者程序代码”。

3.【NOIP2006 普及组】在计算机中,防火墙的作用是(　　)。

A. 防止火灾蔓延　　B. 防止网络攻击

C. 防止计算机死机　　D. 防止使用者误删除数据

【答案】B

【分析】防火墙(英文:firewall)是一项协助确保信息安全的设备,会依照特定的规则,允许或限制传输的数据通过。防火墙可以是一台专属的硬件也可以是架设在一般硬件上的一套软件。

4. 大部分计算机病毒会主要造成计算机(　　)的损坏

A. 软件和数据　　B. 硬件和数据　　C. 硬件、软件和数据　D. 硬件和软件

【答案】A

【分析】计算机病毒通常是导致正常的程序无法运行,把计算机内的文件删除或受到不同程度的损坏。只有少数特别厉害的病毒会破坏硬盘引导扇区及 BIOS,从而导致硬件环境的破坏。

5. 计算机病毒主要是通过(　　)传播的。

A. 磁盘与网络　　B. 微生物“病毒体”　C. 人体　　D. 电源

【答案】A

【分析】计算机病毒的扩散方式有多种:移动载体,如硬盘、U 盘、文件等,受感染文件传播,邮件链接传播,直接安装方式等。

6. 发现计算机病毒后,较为彻底的清除方法是(　　)。

A. 删除磁盘文件　　B. 格式化磁盘　　C. 用查毒软件处理　D. 用杀毒软件处理

【答案】B

【分析】A 错,因为病毒是隐藏传染的,删了一个文件也没用。C、D 错,有些杀毒软件只能查到不能杀,有时还查不到。格式化是最彻底的,格式化后磁盘上没有东西了。

第9节　原码　补码　反码

一、数的原码、补码和反码表示

(1)机器数与真值

在计算机中，表示数值的数字符号只有0和1两个数码，我们规定最高位为符号位，并用0表示正数符号，用1表示负数符号。这样，机器中的数值和符号全"数码化"了。为简化机器中数据的运算操作，人们采用了原码、补码、反码及移码等几种方法对数值位和符号位统一进行编码。为区别起见，我们将数在机器中的这些编码表示称为机器数(如10000001)，而将原来一般书写表示的数称为机器数的真值(如－0000001)。

(2)原码表示法

原码表示法是一种简单的机器数表示法，即符号和数值表示法。设x为真值，则$[x]_{原}$为机器数表示。

例：设 x＝1100110，则$[x]_{原}$＝01100110；

x＝－1100111，则$[x]_{原}$＝11100111。

(3)反码表示法

正数的反码就是真值本身；负数的反码，只须对符号位以外各位按位"求反"(0变1，1变0)即可。

例：设 x＝1100110，则$[x]_{反}$＝01100110；

x＝－1100111，则$[x]_{反}$＝10011000。

(4)补码表示法

负数用补码表示时，可以把减法转化成加法。正数的补码就是真值本身；负数的补码是符号位为1，数值各位取反(0变为1、1变为0)，最低位加1。

例：设 x＝1100110，则$[x]_{补}$＝01100110；

x＝－1100111，则$[x]_{补}$＝10011001。

从上面关于原码、反码、补码的定义可知：一个正数的原码、反码、补码的表示形式相同，符号位为0，数值位是真值本身；一个负数的原码、反码、补码的符号位都为1，数值位原码是真值本身，反码是各位取反，补码是各位取反，最低位加1。真值0的原码和反码表示不唯一，而补码表示是唯一的，即

$[+0]_{原}$＝000…0，$[-0]_{原}$＝100…0；

$[+0]_{反}$＝000…0，$[-0]_{反}$＝111…1；

$[+0]_{补}$＝$[-0]_{补}$＝000…0。

注：不同编码表示的整数的范围是这样的(以N位二进制位)：

原码：$0\sim2^n-1$(无符号)，$-2^{n-1}-1\sim2^{n-1}-1$(有符号)；

反码：$-2^{n-1}-1\sim2^{n-1}-1$(不存在无符号的情况)；

补码：$-2^{n-1}\sim2^{n-1}-1$(不存在无符号的情况)。

大家很明显看出，补码表示的范围最大。现以8位二进制位为例说明如下：

原码：00000000～11111111，即0～255(无符号)，

11111111～01111111，即－127～＋127(有符号)；

反码：10000000－01111111，即

－127～＋127(因为10000000的值为－127，11111111的值为－0)；

补码：10000000～01111111，即

－128～＋127(因为10000000的值为－128，11111111的值为－1)。

如果没说具体编码形式，则计算机中N位二进制无符号数的范围是$0\sim2^n-1$；有符号

数的范围是$-2^{n-1}\sim 2^{n-1}-1$。

以 8 位有符号整数为例子：

原码：若 X 为正数，则最高位(符号位)为 0，其余按照二进制数排列；

若 X 为负数，则最高位为 1，后面和正数原码一样。

例：+7：00000111，−7：10000111；

+0：00000000，−0：10000000。

反码：若 X 为正数，则反码与原码相同；

若 X 为负数，则将原码除符号位取反。

例：−7：11111000。

补码：反码+1。

例：−7：1111001。

二、数的定点表示和浮点表示

在计算机中，小数点一般有两种表示法：一种是小数点固定在某一位置的定点表示法；另一种是小数点的位置可任意移动的浮点表示法。相应于这两种表示的计算机分别称为定点计算机和浮点计算机。

(1)定点表示法

机器中所有数的小数点位置是固定不变的，因而小数点就不必使用记号表示出来。实际上，小数点可固定在任意一个位置上。

(2)浮点表示法

在数的定点表示法中，由于数的表示范围较窄，常常不能满足各种数值问题的需要。为了扩大数的表示范围，方便用户使用，有些计算机常采用浮点表示法。表示一个浮点数，要用两部分：尾数和阶码。尾数用以表示数的有效数值；阶码用以表示小数点在该数中的位置。

计算机多数情况下采用浮点数表示数值，它与科学计数法相似，把一个二进制数通过移动小数点位置表示成阶码和尾数两部分：

$$N=2^{E}*S$$

其中：E 为 N 的阶码(Expoent)，是有符号的整数；S 为 N 的尾数(Mantissa)，是数值的有效数字部分，一般规定取二进制定点纯小数形式。

例：1011101 B$=2^{+7}*0.1011101$，101.1101 B$=2^{+3}*0.1011101$，

0.01011101 B$=2^{-1}*0.1011101$。

【课堂练习】

1.【NOIP1999】 计算机中的数有浮点数与定点数两种，其中用浮点数表示的数，通常由(　　)这两部分组成。

A. 指数与基数　　B. 尾数与小数　　C. 阶码与尾数　　D. 整数与小数

【答案】C

【分析】计算机中的数如 1.701412E+09，叫指数计数法，也叫浮点计数法，其一般形式为$\pm N^{E}\pm nn$，$\pm N$ 叫尾数，可以是一位非零整数或整数部分是一位非零数字的小数，小数点后面最多可有 6 位数字。表示正数时，正号省略掉。E 是英文 exponent 的第一个字母的大写形式，表示指数。后面$\pm nn$ 叫阶码，阶码有两整数位，表示尾数所乘 10 的幂的指数。绝对值小于 10 的阶码，输出时十位补 0，阶码为正时“+”不省略。

2.【NOIP2001 提高组】$[x]_{补码}=10011000$，其原码为(　　)。

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

【答案】B

【分析】对于正数来说，$[x]_{原码}=[x]_{反码}=[x]_{补码}$。对于负数来说，$[x]_{补码}=[x]_{反码}+1$、

$[x]_{反码}$等于$[x]_{原码}$除符号位外逐位取反。所以，$[x]_{原码}$等于$[x]_{补码}-1$且除符号位逐位取反，得11101000。

3.【NOIP2002提高组】已知$x=(0.1011010)_2$，则$[x/2]_{补}=(\quad)_2$。

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

【答案】C

【分析】因$[x/2]_{原码}=0.0101101$，$[x/2]_{补}=[x/2]_{原码}$，所以答案为C。

4. 用十六位机器码1111000101000000来表示定点整数(最高位为符号位)，当它是原码时表示的十进制真值为−25216；当它是补码时表示的十进制真值是(　　)。

A. −12608　　B. −7551　　C. −7552　　D. −25216

【答案】C

【分析】当它是补码时，求真值应该是除最高位符号位之外减1求反。

5. 已知x的原码表示为11110111，下列(　　)是x的补码表示。

A. $[x]_{补}=01010011$ B　　B. $[x]_{补}=10001001$ B

C. $[x]_{补}=11111111$ B　　D. $[x]_{补}=11000000$ B

【答案】B

【分析】已知x的原码表示为11110111，最高位1表示负数，求补码的方法是除最高位符号位外求反加1。

6. 十进制数−103的补码是(　　)。

A. 10011001　　B. 11100111　　C. 10110011　　D. 00011001

【答案】A

【分析】十进制数−103的原码是11100111，反码是10011000，补码时10011001。

7. 关于“零”的原码、反码和补码，下列说法正确的是(　　)。

A. 零的原码表示只有一种　　B. 零的反码表示只有一种

C. 零的补码表示只有一种　　D. 零的原码、反码和补码的表示都有两种

【答案】C

【分析】零的补码表示只有一种，就是二进制全0；如果最高位符号位是1，则为当前字节表示下的负数最大值，比如单字节二进制10000000的十进制真值为−128。

8. 下列关于十进制数100的正确说法是(　　)。

A. 原码为01100100 B　　B. 反码为64 H　　C. 反码为9B H

D. 补码为64 H　　E. 补码为9B H

【答案】ABD

【分析】十进制数100是正数，其单字节表示的原码、反码、补码分别是01100100 B，64 H，64 H(其中B表示二进制、H表示十六进制)。

第 10 节　计算机网络

一、网络的定义

所谓计算机网络，就是利用通信线路和设备，把分布在不同地理位置上的多台计算机连接起来。

计算机网络是现代通信技术与计算机技术相结合的产物。

网络中计算机与计算机之间的通信依靠协议进行。协议是计算机收、发数据的规则。

TCP/IP：用于网络的一组通信协议。包括 IP(Internet Protocol)和 TCP(Transmission Control Protocol)

二、网络的发展

计算机网络的发展过程大致可以分为三个阶段：

远程终端联机阶段：主机—终端

计算机网络阶段：{计算机—计算机 / Internet 阶段：Internet}

三、网络的主要功能

(1)资源共享；(2)信息传输；(3)分布处理；(4)综合信息服务。

四、网络的分类

(一)按网络的地理范围进行分类：局域网(LAN)、城域网(MAN)、广域网(WAN)。

1. 局域网(Local Area Network)

一般局限在 1 km 的范围内，局域网内传输速率较高，误码率低，结构简单、容易实现，具体标准是美国电气工程师协会制定的 IEEE802 系列标准。

2. 城域网(Metropolitan Area Network)

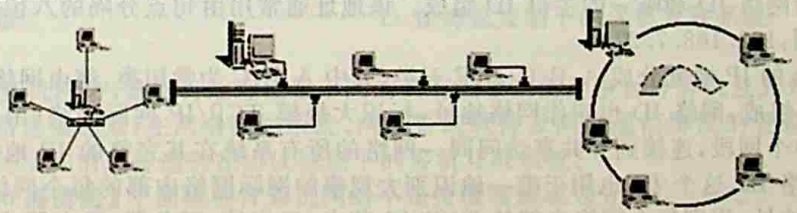
城域网的范围为几千米到几十千以内。

3. 广域网(Wide Area Network)

广域网的范围在几十千米到几千千米以上。

注：MAN 和 WAN 一般都是由多个 LAN 构成的。

(二)按网络的拓扑结构进行分类：星形、总线形、环形、树形、网状形。



星形　　总线形　　环形

1. 星形

通信协议简单，对外围站点要求不高，单个站点故障不会影响全网，电路利用率低，连线费用大，网络性能依赖中央结点，每个站点需要一个专用链路。

2. 总线形

结构简单，可靠性高；布线容易，连线总长度小于星形结构；总线任务重，易产生瓶颈问题；总线本身的故障对系统是毁灭性的。

3. 环形

传输速率高，传输距离远；各节点的地位和作用相同；各节点传输信息的时间固定；容易实现分布式控制；站点的故障会形起整个网络的崩溃。

4. 树形

分级结构，又称为分级的集中式网络。

5. 网状形(不规则形)

计算机之间无规则地连接，一般广域网属于不规则形。

网络拓扑结构是指计算机网络节点和通信链路所组成的几何形状。

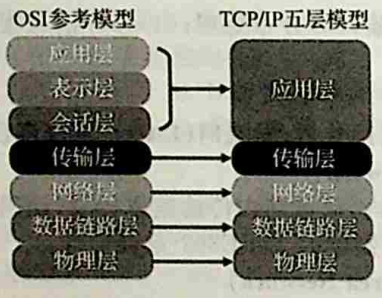
Internet 网是当今世界上规模最大、用户最多、影响最广泛的计算机互联网络。Internet 网上联有大大小小成千上万个不同拓扑结构的局域网、城域网和广域网。因此，Internet 网本身的拓扑只是一种虚拟拓扑结构，无固定形式。

按采用的交换技术进行分类：电路交换、报文交换、分组交换。

五、网络的体系结构

国际标准化组织(International Standardization Organization，ISO)提出的开放式系统互联(Open System Interconnection，OSI)参考模型。它将数据从一个站点到达另一个站点的工作按层分割成七个不同的任务。

开放性是指任何遵循 OSI 标准的系统，只要物理上连接起来，它们之间都可以相互通信。OSI 参考模型并不是网络体系结构。OSI 只是描述了每一层的功能，并没有确定一个层的协议。而网络体系结构是网络层次结构和相关协议的集合。

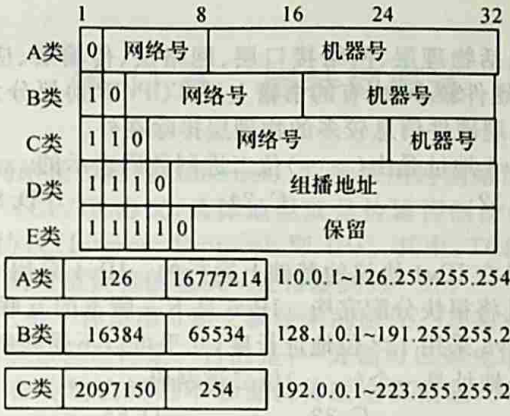


六、IP 地址

在 TCP/IP 体系中，IP 地址是一个最重要的概念，什么叫 IP 地址？

所谓 IP 地址，是用于标识 Internet 网络上节点的 32 位地址(以后可能使用的 V6 版本是 128 位的，分 8 组，每组 16 位)。对于 Internet 网络上的每个节点，都必须指派一个唯一的地址，它由网络 ID 和唯一的主机 ID 组成。该地址通常用由句点分隔的八位字节的十进制数表示(例：192.168.7.27)。

Internet 的 IP 地址分成 A、B、C、D、E 五类，其中 A、B、C 为常用类，都由网络 ID 和主机 ID 两个部分组成，网络 ID 也叫作网络地址，标识大规模 TCP/IP 网际网络(由网络组成的网络)内的单个网段，连接到并共享访问同一网络的所有系统在其完整的 IP 地址内都有一个公用的网络 ID，这个 ID 也用于唯一地识别大规模的网际网络内部的每个网络；主机 ID，也叫作主机地址，识别每个网络内部的 TCP/IP 节点(工作站、服务器、路由器或其他 TCP/IP 设备)，每个设备的主机 ID 唯一地识别所在网络内的单个系统。



注：

①尽管 IP 地址的主机号的每个域的取值范围是 0～255，但主机 ID 所有域不能都为 0 或 255。例如：如果网络 ID 为 10，那么就不能把 10.0.0.0 和 10.255.255.255 两个 IP 地址分配给任何主机；如果网络 ID 为 192.114.31，那么就不能把 192.114.31.0 和 192.114.31.255 两个 IP 地址分配给任何主机。

②专用网络寻址选项：如果某个连接仅存在于群集节点之间，并且不支持其他网络的客户机，可以为它配置私有 IP 网络地址，而不是企业的正式 IP 网络地址。经过 Internet 编号指派机构（IANA）协商同意，几个 IP 网络被保留下来以用作企业内部私有。这些保留的号码是：

10.0.0.0 到 10.255.255.255(A 类)

172.16.0.0 到 172.31.255.255(B 类)

192.168.0.0 到 192.168.255.255 (C 类)

IP 地址的表示方法：采用点分十进制记法(Dotted Decimal Notation)，即将 32 bit 的 IP 地址中的每 8 位用。等效的十进制表示，并每 8 位之间加上一个点，即 4 段二进制位组成，“.”隔开，每组数字的取值范围只能是 0～255。

例如：10000001 00001011 00000011 00011111

129　　11　　3　　31

对应的 IP 地址为：129.11.3.31。

【课堂练习】

1.【NOIP2000 提高组】计算机网络是一个(　　)系统。

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联系统

【答案】D

【分析】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在协议控制下的多机互联系统。

2.【NOIP2000 提高组】下面哪些计算机网络不是按覆盖地域划分的(　　)。

A. 局域网　　B. 都市网　　C. 广域网　　D. 星形网

【答案】D

【分析】计算机网络按覆盖地域可分为局域网(Local Area Network，LAN)、城域网(即都市网，Metropolitan Area Network，MAN)、广域网(Wide Area Network，WAN)。按拓扑结构可分为总线形、星形、环形等。

3.【NOIP2001 提高组】TCP/IP 协议共有(　　)层协议

A. 3　　B. 4　　C. 5　　D. 6

【答案】C

【分析】TCP/IP 协议包括物理层、网络接口层、网络层、传输层、应用层五层协议。因为 TCP、IP 核心技术涉及硬件较少，也有的书籍上把 TCP/IP 协议分为接口层、网络层、传输层、应用层四层，而将处理硬件信息较多的物理层排除在外。

4.【NOIP2002 普及组】IPv4 地址是由(　　)位二进制数码表示的。

A. 16　　B. 32　　C. 24　　D. 8

【答案】B

【分析】现有的互联网是在 IPv4 协议的基础上运行的。IPv4 采用 32 位地址长度，只有大约 43 亿个地址，IP 地址将很快分配完毕。IPv6 是下一版本的互联网协议，也可以说是下一代互联网的协议。IPv6 采用 128 位地址长度，几乎可以不受限制地供应地址。

5.【NOIP2003 普及组】IP 地址是一个(　　)位二进制码。

A. 8　　B. 16　　C. 32　　D. 64　　E. 12

【答案】C

【分析】IP 地址的 32 位二进制数，用 4 个字节存储，每个字节是 8 位二进制位，4 个数总共 32 位，而每个字节表现成十进制数，范围只有 0 到 255。形如：10. 250. 36. 78。

6. 计算机网络最主要的优点是(　　)。

A. 运算速度快　　B. 共享资源　　C. 精度高　　D. 存储容量大

【答案】B

【分析】计算机网络的最大优点是资源共享。资源共享指的是网上用户能部分或全部地享受这些资源，包括软件、硬件及数据资源，提高了系统资源的利用率。

7. OSI 的七层协议中，最底层是(　　)。

A. 会话层　　B. 数据链路层　　C. 物理层　　D. 网络层

【答案】C

【分析】OSI 七层协议从上到下依次是：应用层、表示层、会话层、传输层、网络层、数据链路层、物理层。

8. 192. 168. 0. 1 是属于(　　)。

A. A 类地址　　B. B 类地址　　C. C 类地址　　D. D 类地址

【答案】C

【分析】C 类网络 IP 地址范围从 192. 0. 0. 0 到 223. 255. 255. 254 的单址广播 IP 地址。192. 168. 0. 0～192. 168. 255. 255 是 C 类私有地址。

9. 下列 IP 地址中错误的是(　　)。

A. 202. 300. 12. 4　　B. 192. 168. 0. 3　　C. 100. 128. 35. 91　　D. 19. 255. 0. 1

【答案】A

【分析】IP 地址的最大数值为 255，所以 202. 300. 12. 4 是错误的。

10. 下列地址中，属于 B 类 IP 地址的是(　　)。

A. 27. 33. 119. 2　　B. 192. 97. 32. 121　　C. 133. 201. 189. 32

D. 126. 33. 82. 107　　E. 10. 11. 12. 13

【答案】C

【分析】B 类地址范围：128. 0. 0. 0 到 191. 255. 255. 254。

第 11 节　因特网概述

目前，全球最大的网络是因特网(Internet)，它所采用的网络协议是 TCP/IP 协议。它是因特网的核心技术。TCP/IP 协议，具体地说就是传输控制协议(Transmission Control Protocol，TCP)和网际协议(Internet Protocol，即 IP)。其中，TCP 协议用于负责网上信息的正确传输，而 IP 协议则是负责将信息从一处传输到另一处。

TCP/IP 协议本质上是一种采用分组交换技术的协议。其基本思想是把信息分割成一个个不超过一定大小的信息包来传送。目的是：一方面可以避免单个用户长时间地占用网络线路；另一方面，可以在传输出错时不必重新传送全部信息，只须重传出错的信息包就行了。

TCP/IP 协议组织信息传输的方式是一种 4 层的协议方式。下图是一种简化了的层次模型：

应用层	Telnet 、FTP 和 e-mail 等
传输层	TCP 和 UDP
网络层	IP 、ICMP 和 IGMP
网络接口层	设备驱动程序及接口卡

TCP/IP 协议层次简化模型

一、因特网概述

因特网(Internet)是一个建立在网络互联基础上的最大的、开放的全球性网络。因特网拥有数千万台计算机和上亿个用户，是全球信息资源的超大型集合体。

因特网起源于 20 世纪 60 年代中期，由美国国防部高级研究计划局(ARPA)资助的 ARPANET，此后提出的 TCP/IP 协议为因特网的发展奠定了基础。

我国正式接入因特网是在 1994 年 4 月，当时为了发展国际科研合作的需要，中国科学院高能物理研究所和北京化工大学开通了到美国的因特网专线，并有千余科技界人士使用了因特网。

我国 Internet 的发展情况：

20 世纪 80 年代末、90 年代初才起步。

1989 年我国第一个公用分组交换网 CNPAC 建成运行。

我国已陆续建成与 Internet 互联的四个全国范围的公用网络：中国公用计算机互联网(CHINANET)、中国金桥信息网(CHINAGBN)、中国教育和科研计算机网(CERNET)、中国科学技术网(CSTNET)。

二、域名和网址

1. 网址

网址在因特网中，如果要从一台计算机访问网上另一台计算机，就必须知道对方的网址。这里所说的网址实际上指两个内涵，即 IP 地址、域名地址和 URL。

IP 地址(英语：Internet Protocol Address)是一种在 Internet 上的给主机编址的方式，也称为网络协议地址。常见的 IP 地址分为 IPv4 与 IPv6 两大类。

IPV4 就是有 4 段数字，每一段最大不超过 255。由于互联网的蓬勃发展，IP 地址的需求量愈来愈大，使得 IP 地址的发放愈趋严格，各项资料显示全球 IPv4 地址可能在 2005 至 2010 年间全部发完(实际情况是在 2011 年 2 月 3 日 IPv4 地址分配完毕)

IPv6 采用 128 位地址长度。在 IPv6 的设计过程中除了一劳永逸地解决了地址短缺问题以外，还考虑了在 IPv4 中解决不好的其他问题。

2. 域名

32 位二进制数 IP 地址对计算机来说是十分有效的，但记忆一组并无意义的且无任何特征的 IP 地址是困难的，为此，因特网引进了字符形式的 IP 地址，即域名。域名采用层次结构的基于“域”的命名方案，每一层由一个子域名组成，子域名间用“.”分隔。

其格式为：开头. 主机名. 主机类别. 国家名(可以不要)。

如 www. ycwb. com. cn：域名地址采用层次结构，一个域名一般有 3～5 个子段，中间用“.”隔开。

因特网上的域名由域名系统(Domain Name System，DNS)统一管理。DNS 是一个分布式数据库系统，由域名空间、域名服务器和地址转换请求程序三部分组成。有了 DNS，凡域名空间中有定义的域名都可以有效地转换为对应的 IP 地址，同样，IP 地址也可通过 DNS 转换成域名。在因特网上，域名和 IP 地址一样都是唯一的。

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、…

IP 地址是纯数字化的，不便于人们记忆和识别。因此，用一串具有某种意义又便于记忆的文字来标识网络中的计算机是必要的，这便引入了域名的概念。

因特网中的计算机必须首先具有 IP 地址才能使用，域名则直观地标识出计算机的地理位置、所属机构等信息，对于 TCP/IP 来说，域名必须被映射为 IP 地址才能使用。在因特网中，专门用来管理域名与 IP 地址之间映射关系的计算机叫域名服务器(Domain Name Server，DNS)。

域名采用层次结构的命名方法，因特网的顶层域名(第一级域名)分为两大类，通用的(网络性质划分)和国家(地区)的，常用的见下表：

通用的		国家(地区)的	
edu	教育机构	cn	中国
gov	政府部门	hk	香港
net	网络组织	mo	澳门
com	商业组织	tw	台湾
org	非赢利性组织	jp	日本
mil	军事部门	sg	新加坡
…		…	

与一级域名类似，二级域名也有两类：一类表示网络性质(同一级)；一类表示国内的各省、直辖市、自治区等(如 fj，bj 分别表示福建和北京)。

根据需要还可以有第三级、第四级……每一级的域名都由英文字母和数字组成(随着技术的发展有可能可以用其他文字)，最长不超过计划 63 个字符，不分大小写；层次低的域名写在左边，高的写在右边，之间用英文的点号分开，完整的域名不超过 255 个字符。

说明：域名的最左边(最底层)往往是一台主机的名字，通常用主机提供的服务表示，如 WWW、FTP、MAIL、BBS 等。因此，一个具体的域名也可以分成两个部分，即机器名. 域名。如 WWW. TSINGHUA. EDU. CN 这个域名，TSINGHUA. EDU. CN 是域名，WWW 是主机名。如果主机名被省略，系统默认为 WWW。

三、因特网提供的服务

Internet 的服务有电子邮件、远程登录、文件传输、信息服务等。

1. 万维网(WWW)

全球信息网，又称万维网(World Wide Web，WWW)，是一个全球规模的信息服务系统，由遍布于全世界的数以万计的 Web 站点组成。

万维网是瑞士日内瓦欧洲粒子实验室最先开发的一个分布式超媒体信息查询系统，目前它是因特网上最为先进、交互性能最好、应用最为广泛的信息检索工具，万维网包括各种各样的信息，如文本、声音、图像、视频等。万维网采用了“超文本”的技术，使得用户以通用而简单的办法就可获得因特网上的各种信息。

2. 电子邮件(E-mail)

电子邮件地址格式为：收信人邮箱名@邮箱所在主机的域名。

例：winner01@ 21cn. com。

电子邮件(Electronic mail，E-mail)是因特网上使用最广泛的一种服务。用户只要能与因特网连接，具有能收发电子邮件程序及个人的电子邮件地址，就可以与因特网上具有电子邮件地址的所有用户方便、快捷、经济地交换电子邮件。电子邮件可以在两个用户间交换，也可以向多个用户发送同一封邮件，或将收到的邮件转发给其他用户。电子邮件中除文本外，还可包含声音、图像、应用程序等各类计算机文件。此外，用户还可以以邮件方式在网上订阅电子杂志、获取所需文件、参与有关的公告和讨论组等。

3. 文件传输协议(FTP)

文件传输协议(File Transfer Protocol，FTP)：用于在计算机间传输文件，如下载软件等。

FTP 是因特网上文件传输的基础，通常所说的 FTP 是基于该协议的一种服务。FTP 文件传输协议允许因特网上的用户将一台计算机上的文件传输到另一台上，几乎所有类型的文件，包括文本文件、二进制可执行文件、声音文件、图像文件、数据压缩文件等，都可以用 FTP 传送。

4. 远程登录(Telnet)

远程登录(Telnet)：指通过 Internet 与其他主机连接。

Telnet 是远程登录服务的一个协议，该协议定义了远程登录用户与服务器交互的方式。允许用户在一台联网的计算机登录到一个远程分时系统时，像使用自己的计算机一样使用该远程系统。

四、浏览网页的相关概念

1. WWW 与 HTML

WWW 是因特网上集文本、声音、图像、视频等多媒体信息于一身的全球信息资源网络，是因特网上的重要组成部分。浏览器(Browser)是用户通向 WWW 的桥梁和获取 WWW 信息的窗口。通过浏览器，用户可以在浩瀚的因特网海洋中漫游，搜索和浏览自己感兴趣的所有信息。

WWW 的网页文件是用超文本标记语言 HTML(Hyper Text Markup Language)编写的，并在超文本传输协议 HTTP(Hyper Text Transmission Protocol)支持下运行的。超文本中不仅含有文本信息，还包括图形、声音、图像、视频等多媒体信息(故超文本又称超媒体)，更重要的是超文本中隐含着指向其他超文本的链接，这种链接称为超链(Hyper Links)。

2. URL

简单地讲，URL(Uniform Resource Locator，统一资源定位器)就是因特网上的资源地址。每个 Web 页面，包括主页，都有一个唯一的地址，其格式为：协议名：//IP 地址或域名。

协议名表示所提供的服务，如 http://www.online.sh.cn(上海热线)就是我们常用的万维网服务的 URL 地址。ftp://pchome.net 就是因特网上文件传输服务的 URL 地址。

3. 浏览器

WWW 浏览器是一个客户端的程序，其主要功能是使用户获取因特网上的各种资源。常用的浏览器有 Microsoft 的 Internet Explorer(IE)。

五、电子邮件的相关概念

1. 电子邮件概述

电子邮件(Electronic Mail，E-mail)是因特网上使用最广泛的一种服务。

2. 电子邮件使用的协议

邮件服务器使用的协议有简单邮件传输协议(Simple Message Transfer Protocol，SMTP)、电子邮件扩展协议(Multipurpose Internet Mail Extensions，MIME)和 POP 协议(Post Office Protocol)。POP 服务需要一个邮件服务器来提供，用户必须在该邮件服务器上取得帐号才可使用这种服务。目前使用较普遍的 POP 协议为第三版，故又称为 POP3 协议。

3. 信箱地址及其格式

使用电子邮件系统的用户首先要有一个电子邮件信箱，该信箱在因特网上有唯一的地址，以便识别。

像传统信件的信封有格式要求一样，电子邮件也有规范的地址格式。电子邮件的信箱地址由字符串组成，该字符串被字符“@”分成两部分(字符“@”在英语中可以读作“at”)，前一部分为用户标识，可以使用该用户在该计算机上的登录名或其他标识，只要能够区分该计算机上的不同用户即可，如“zhangshan”；后一部分用户信箱所在的计算机的域名，如 ecust.edu.cn(华东理工大学网络中心邮件服务器主机域名)。像 zhangshan@ecust.edu.cn 就是一个电子邮件的地址。

电子邮件地址的一般格式为：〈用户标识〉@〈主机域名〉。

【课堂练习】

1.【NOIP2000】 Internet 的规范译名应为(　　)。

A. 英特尔网　　B. 因特网　　C. 万维网　　D. 以太网

【答案】 B

【分析】 因特网又称国际互联网。我国于 1994 年正式联入因特网。全国科学技术名词审定委员会于 1997 年 7 月 18 日为 Internet 做出了命名，中文名词为“因特网”，注译是“全球最大的、开放的、由众多网络相互连接而成的计算机网络”。万维网是 WWW 的中文命名，是 World Wide Web 的缩写，意思是“基于超文本的，方便用户信息浏览和信息搜索的信息服务系统”。人们以 WWW 的方式浏览网上信息。以太网是 Xerox 公司发明的基带局域网标准，它采用带冲突检测的载波监听多路访问协议(CSMA/CD)，速率为 10 Mbps，传输介质为同轴电缆。现在，以太网一词泛指所有采用 CSMA/CD 协议的局域网，包括使用双绞线和光纤作为传输介质的快速以太网，当前学校中的教学网大多是快速以太网。

2.【NOIP2001】 E-mail 邮件本质上是一个(　　)。

A. 文件　　B. 电报　　C. 电话　　D. 传真

【答案】 A

【分析】 计算机是以文件的方式管理数据的，计算机中绝大多数数据都以文件的形式存在。

3.【NOIP2002 普及组】 E-mail 地址中用户名和邮件所在服务器名之间的分隔符号是(　　)。

A. #　　B. @　　C. &　　D. $

【答案】 B

【分析】电子邮件地址中必须有@符号。

4.【NOIP2003 普及组】下列电子邮件地址，正确的是(　　)。

A. wang@hotmail. com　　B. cai@jcc. pc. tool@rf. edu. jp

C. 162. 105. 111. 22　　D. ccf. edu. cn　　E. http://www. sina. com

【答案】A

【分析】电子邮箱的格式通常为 username@domain. com。其中 username 为用户名，“@”后是域名。如 hotmail 的邮箱格式一般为 123456789@hotmail. com。

5.【NOIP2004 普及组】下列网络上常用的名字缩写对应的中文解释错误的是(　　)。

A. WWW(World Wide Web)：万维网

B. URL(Uniform Resource Locator)：统一资源定位器

C. HTTP(Hyper Text 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

【答案】D

【分析】正译是：文件传输协议。

6.【NOIP2004】一台计算机如果要利用电话线上网，就必须配置能够对数字信号和模拟信号进行相互转换的设备，这种设备是(　　)。

A. 调制解调器　B. 路由器　　C. 网卡　　D. 网关　　E. 网桥

【答案】A

【分析】A. 调制解调器 modem(其实是 Modulator(调制器)与 Demodulator(解调器)的简称)，是一种计算机硬件，它能把计算机的数字信号翻译成可沿普通电话线传送的脉冲信号，而这些脉冲信号又可被线路另一端的另一个调制解调器接收，并译成计算机可懂的语言。

B. 路由器：连接因特网中各局域网、广域网的设备，它会根据信道的情况自动选择和设定路由，以最佳路径，按前后顺序发送信号的设备。路由器英文名 Router，所谓路由，就是指通过相互连接的网络把信息从源地点移动到目标地点的活动。路由和交换之间的主要区别就是交换发生在 OSI 参考模型的第二层(数据链路层)，而路由发生在第三层，即网络层。这一区别决定了路由和交换在移动信息的过程中需要使用不同的控制信息，所以两者实现各自功能的方式是不同的。家庭中可以使用无线路由器提供无线上网。

C. 计算机与外界局域网的连接是通过主机箱内插入有线或无线网卡。

D. 网关(Gateway)又称网间连接器、协议转换器。网关在网络层以上实现网络互连，是最复杂的网络互连设备，仅用于两个高层协议不同的网络互连。

E. 网桥(Bridge)是早期的两端口二层网络设备，用来连接不同网段。网桥的两个端口分别有一条独立的交换信道，不是共享一条背板总线，可隔离冲突域。

7.【NOIP2004 提高组】下列哪个网络上常用的名字缩写是错误的(　　)。

A. WWW(World Wide Web)　　B. URL(Uniform Resource Locator)

C. HTTP(Hyper Text Transfer Protocol)　D. FTP(Fast Transfer Protocol)

E. TCP(Transfer Control Protocol)。

【答案】D

【分析】应该是“File Transfer Protocol”。

8.【NOIP2005】常见的邮件传输服务器使用(　　)协议接收邮件。

A. HTTP　　B. SMTP　　C. TCP　　D. FTP　　E. POP3

【答案】E

【分析】SMTP 是发邮件协议，POP3 是收邮件协议，IMAP 是邮件访问协议。

第二章　程序设计基本知识

计算机的应用已不再局限于科学计算，而更多地用于控制、管理和数据处理等非数值计算的处理工作。为了编写一个“好”的程序，必须分析待处理的对象特性以及各处理对象之间存在的关系。

第 1 节　程序基本常识

1. 算法

算法是对特定问题求解步骤的一种描述，它是指令（规则）的有限序列，其中每一条指令表示一个或多个操作。简单地说，算法就是解决问题的操作步骤。

一个算法必须满足以下五个重要的特征。

（1）有穷性

对于任意一组合法的输入值，算法的每个操作步骤都能在有限的时间内完成，这包括合理的执行时间的含义。如果一个算法执行耗费的时间太长，即使最终得出了结果，也是没有意义的。

（2）确定性

算法中的每一步都必须有明确的定义，不允许有歧义性和多义性。确定性使算法的执行者或者阅读者能够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）输入

一个算法应该有 0 个或多个输入，以刻画运算对象的初始情况。所谓 0 个输入，是指有的算法表面上可以没有输入，实际上已被嵌入算法之中。

（4）输出

一个算法应该有一个或多个输出，以反映对输入数据加工后的结果。没有输出的算法是毫无意义的。

（5）可行性

一个算法必须遵循特定条件下的解题规则，算法描述的每一个操作都应该是特定的解题规则中允许使用的、可执行的，并可以通过执行有限次来实现。

2. 算法的复杂度

同一个问题可用不同的算法来解决，而一个算法质量的优劣将影响算法乃至程序的效率。算法分析的目的在于选择合适的算法和改进算法。一个算法的评价主要从时间复杂度和空间复杂度来考虑。

（1）时间复杂度

算法的时间复杂度（Time Complexity）是指算法所需要的计算工作量，用算法所执行的基本运算次数来度量。

常见的时间复杂度有：常数阶 $O(1)$，对数阶 $O(\log_2 n)$，线性阶 $O(n)$，线性对数阶 $O(n\log_2 n)$，平方阶 $O(n^2)$，立方阶 $O(n^3)$，……，指数阶 $O(2^n)$。随着问题规模 n 的不断增大，上述时间复杂度不断增大，算法的执行效率将越低。

（2）空间复杂度

算法的空间复杂度（Space Complexity）是指执行这个算法所需要的内存空间。算法执行期间所需的存储空间主要包括三部分：输入数据所占的存储空间、程序本身所占的空间和

算法执行过程中时所需的存储空间。

3. 算法的基本结构

算法的结构不仅决定了算法中各操作的执行顺序，而且也直接反映了算法的设计是否符合结构化原则。一般一个算法可以用顺序、选择和循环三种基本结构组合而成。

(1)顺序结构

顺序结构是最简单、最常用的算法结构，语句与语句之间、框与框之间按从上到下的顺序进行。

(2)选择结构

选择结构是先根据条件做出判断，再决定执行哪一种操作的算法结构，它必须包含判断框。当条件 P 成立(或称为真)时执行 A，否则执行 B，不可能两者同时执行，但 A 或 B 两个框中可以有一个是空的，即不执行任何操作。

(3)循环结构

在一些算法中，经常会出现从某处开始，按照一定条件，反复执行某一处理步骤的情况，这就是循环结构。反复执行的处理步骤为循环体，它可以细分为两类：当型循环结构和直到型循环结构。

4. 基础算法

(1)高精度计算

利用计算机进行数值计算，有时会遇到这样的问题：有些计算要求精度高，希望计算的数的位数可达几十位甚至几百位，虽然计算机的计算精度不断提高了，但因受到硬件的限制，往往达不到实际问题所要求的精度。我们可以利用程序设计的方法去实现这样的高精度计算。主要涉及高精度的加、减、乘、除运算，实现的时候采用逐位加、减、乘、除即可。

常见问题有“汉诺塔的步数”“国王的米粒”等。

(2)穷举算法

所谓穷举法，即枚举法，指的是从可能的解的集合中一一枚举各元素，用题目给定的检验条件判定哪些是无用的、哪些是有用的。能使命题成立，即为其解。

有些问题可以用循环语句和条件语句直接求解，有些问题用循环求解时循环次数太多，无法编写程序，则需要用到回溯、递归、分治等方法。

常见问题有“百钱买百鸡”“素数判断”等。

(3)数据排序

排序就是将杂乱无章的数据元素，通过一定的方法按关键字顺序排列的过程。

排序的方法很多，下面根据初赛的要求，简单介绍各种常见排序算法在一些方面的比较，尤其是时间复杂度和稳定性两个方面。稳定性，指在原序列中相同元素的相对位置与排好序的新序列中相同元素的相对位置是否相同。若相同，则该算法是稳定的，否则不稳定。

时间复杂性比较：

插入排序、冒泡排序、选择排序的时间复杂性为 $O(n^2)$；

其他非线性排序的时间复杂性为 $O(n\log n)$；

线性排序的时间复杂性为 $O(n)$。

稳定性比较：

插入排序、冒泡排序、二叉树排序、归并排序及其他线性排序是稳定的；

选择排序、希尔排序、快速排序、堆排序是不稳定的；

辅助空间的比较：

线性排序、归并排序的辅助空间为 $O(n)$，其他排序的辅助空间为 $O(1)$。

其他比较：

①插入、冒泡排序的速度较慢，但参加排序的序列局部或整体有序时，这种排序能达到较快的速度。反而在这种情况下，快速排序慢了，时间复杂度会达到其上限。当数据为随机数据时，快速排序远远快于插入、冒泡、选择排序，时间复杂度接近其下限。

②当 n 较小时，对稳定性不作要求时宜用选择排序，对稳定性有要求时宜用插入或冒泡排序。

③若待排序的记录的关键字在一个明显有限范围内且空间允许时用桶排序。

④当 n 较大时，关键字元素比较随机且对稳定性没要求，宜用快速排序。

⑤当 n 较大时，关键字元素可能出现本身是有序的且对稳定性有要求时，在空间允许的情况下，宜用归并排序。

常见问题有“合并果子”“逆序对问题”等。

(4)递推算法

递推法是一种重要的数学方法，在数学的各个领域中都有广泛的运用，也是计算机用于数值计算的一个重要算法。这种算法特点是：一个问题的求解需一系列的计算，在已知条件和所求问题之间总存在着某种相互联系的关系，在计算时，如果可以找到前后过程之间的数量关系(即递推式)，那么从问题出发逐步推到已知条件，此种方法叫逆推。无论顺推还是逆推，其关键是要找到递推式。这种处理问题的方法能使复杂运算化为若干步重复的简单运算，充分发挥出计算机擅长于重复处理的特点。递推算法的首要问题是得到相邻的数据项间的关系(即递推关系)。递推算法避开了求通项公式的麻烦，把一个复杂问题的求解，分解成了连续的若干步简单运算。一般说来，可以将递推算法看成是一种特殊的迭代算法。

常见问题有“斐波那契数列”“过河卒”等。

(5)递归算法

递归程序设计是 C++语言程序设计中的一种重要方法，它使许多复杂的问题变得简单，容易解决了。递归特点是：函数或过程调用它自己本身。其中直接调用自己称为直接递归，而将 A 调用 B，B 再调用 A 的递归叫作间接递归。

常见问题有“汉诺塔问题”“Ackerman 函数”等。

(6)搜索与回溯算法

搜索与回溯是计算机解题中常用的算法，很多问题无法根据某种确定的计算法则来求解，可以利用搜索与回溯的技术求解。回溯是搜索算法中的一种控制策略。它的基本思想是：为了求得问题的解，先选择某一种可能情况向前探索，在探索过程中，一旦发现原来的选择是错误的，就退回一步重新选择，继续向前探索，如此反复进行，直至得到解或证明无解。

如迷宫问题：进入迷宫后，先随意选择一个前进方向，一步步向前试探前进，如果碰到死胡同，说明前进方向已无路可走，这时，首先看其他方向是否还有路可走，如果有路可走，则沿该方向再向前试探；如果已无路可走，则返回一步，再看其他方向是否还有路可走；如果有路可走，则沿该方向再向前试探。按此原则不断搜索回溯再搜索，直到找到新的出路或从原路返回入口处无解为止。

常见问题有“八皇后问题”“骑士游历问题”等。

(7)贪心算法

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，他所做出的仅是在某种意义上的局部最优解。

贪心算法没有固定的算法框架，算法设计的关键是贪心策略的选择。必须注意的是，贪心算法不是对所有问题都能得到整体最优解，选择的贪心策略必须具备无后效性，即某个状态以后的过程不会影响以前的状态，只与当前状态有关。所以，对所采用的贪心策略，一定要仔细分析其是否满足无后效性。

常见问题有“删数问题”“排队打水问题”等。

(8)分治算法

分治就是指分而治之，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。当我们将问题分解成两个较小问题求解时的分治方法称之为二分法。

常见问题有“归并排序”“比赛日程安排”等。

(9)广度优先搜索

广度优先算法的核心思想是:从初始节点开始,应用算符生成第一层节点,检查目标节点是否在这些后继节点中,若没有,再用产生式规则将所有第一层的节点逐一扩展,得到第二层节点,并逐一检查第二层节点中是否包含目标节点。若没有,再用算符逐一扩展第二层的所有节点……如此依次扩展,检查下去,直到发现目标节点为止。这种搜索的次序体现沿层次向横向扩展的趋势,所以称之为广度优先搜索。

常见问题有"分油问题""八数码问题"等。

(10)动态规划

动态规划程序设计是解决多阶段决策过程最优化问题的一种途径、一种方法,而不是一种特殊算法。由于各种问题的性质不同,确定最优解的条件也互不相同,因而动态规划的设计方法对不同的问题,有各具特色的解题方法,而不存在一种万能的动态规划算法,可以解决各类最优化问题。需要满足"最优子结构"和"无后效性"的两项基本条件。实现时主要分成几个步骤:划分阶段、确定状态和状态变量、确定决策并写出状态转移方程、寻找边界条件、程序设计实现。大致可以分为以下几类:线性、区间、背包形、树形等。

常见问题有"最长不下降序列""最长公共子序列"等。

【课堂练习】

1.【NOIP1998 提高组】表达式(4 %(-3))与(-4 % 3)的值为(　　)。

A. -1,-1　　B. 1,-1　　C. -1,1　　D. 1,1

【答案】B

【分析】任何一个整数 n 都可以表示成 n=k * q+r,其中 0<=|r|<|q|,r 就是 n 除以 q 的余数,即 r==n%q,例如:-9=(-2) * 4+(-1),则-9 除以 4 的余数为-1。就是说被除数是负数,余数是负数,被除数是正数,余数是正数。

2.【NOIP2000】已知数组 A 中,每个元素 A[I][J]在存储时要占 3 个字节。设 I 从 1 变化到 8,J 从 1 变化到 10,分配内存时是从地址 SA 开始连续按行存储分配的。

试问:A[5][8]的起始地址为(　　)。

A. SA+141　　B. SA+180　　C. SA+222　　D. SA+225

【答案】A

【分析】设数组有 m 行 n 列,可以总结出公式:A[I][J]的起始地址=SA+((I-1) * n+(J-1)) * 3。将 m=8,n=10 代入以上公式,得 SA+((5-1) * 10+(8-1)) * 3=SA+141。

3.【NOIP2002】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(　　)。

A. 110　　B. 108　　C. 100　　D. 109

【答案】B

【分析】数据元素的存储位置均取决于第 1 个数据元素的存储位置,即 $LOC(a_i)=LOC(a_1)+(i-1)*C$,所以第 5 个元素的地址为 100+2 * (5-1)=108。

4.【NOIP2006】在编程时(使用任一种高级语言,不一定是 C++),如果需要从磁盘文件中输入一个很大的二维数组(例如 1000 * 1000 的 double 型数组),按行读(即外层循环是关于行的)与按列读(即外层循环是关于列的)相比,在输入效率上(　　)。

A. 没有区别　　B. 按行读的方式要高一些

C. 按列读的方式要高一些　　D. 取决于数组的存储方式

【答案】D

【分析】按行、按列,决定元素是否连续,可能会有差异。

5.【NOIP2000 普及组】算法是指(　　)。

A. 为解决问题而编制的计算机程序　　B. 为解决问题而采取的方法与步骤

C. 为解决问题而需要采用的计算机语言　D. 为解决问题而采用的计算方法

【答案】B

【分析】算法是对特定问题求解步骤的一种描述，它是指令(规则)的有限序列，其中每一条指令表示一个或多个操作。简单地说，算法就是解决问题的操作步骤。

6.【NOIP2001】下面关于算法的错误说法是(　　)。

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

【答案】B

【分析】算法除了以计算机语言实现外，还可以框图、文字等方式存在。

7.【NOIP2006 普及组】在下列关于计算机算法的说法中，不正确的是(　　)。

A. 一个正确的算法至少要有一个输入

B. 算法的改进，在很大程度上推动了计算机科学与技术的进步

C. 判断一个算法的好坏的主要标准是算法的时间复杂性与空间复杂性

D. 目前仍然存在许多涉及国计民生的重大课题，还没有找到能够在计算机上实施的有效算法

【答案】A

【分析】算法(algorithm)是在有限步骤内求解某一问题所使用的一组定义明确的规则。

一个算法应该具有以下五个重要的特征：

有穷性：一个算法必须保证执行有限步之后结束；

确切性：算法的每一步必须有确切的定义；

输入：一个算法有 0 个或多个输入，以刻画运算对象的初始情况，所谓 0 个输入是指算法本身定出了初始条件；

输出：一个算法有一个或多个输出，以反映对输入数据加工后的结果，没有输出的算法是毫无意义的；

可行性：算法原则上能够精确地运行，而且人们用笔和纸做有限次运算后即可完成。

8.【NOIP2000 普及组】在待排序的数据表已经为有序时，下列排序算法中花费时间反而多的是(　　)。

A. 堆排序　　B. 希尔排序　　C. 冒泡排序　　D. 快速排序

【答案】D

【分析】最坏情况下，是整个序列都已经有序，此时，快速排序退化为冒泡排序，要比较 n*(n−1)/2 次才能完成。

9.【NOIP2002】在所有排序方法中，关键字比较的次数与记录的初始排列次序无关的是(　　)。

A. 希尔排序　　B. 起泡排序　　C. 插入排序　　D. 选择排序

【答案】D

【分析】选择排序特点是排序总是从第一位开始，与起始位无关。所有趟数下来关键字总是比较 n(n−1)/2 次，每一趟关键字比较次数也是固定的。

10.【NOIP2006 普及组】在下列各种排序算法中，不是以“比较”作为主要操作的算法是(　　)。

A. 选择排序　　B. 冒泡排序　　C. 插入排序　　D. 基数排序

【答案】D

【分析】基数排序按个、十、百……轮流排序，非比较。

11.【NOIP2006】将 5 个数的序列排序，不论原先的顺序如何，最少都可以通过(　　)次比较，完成从小到大的排序。

A. 6　　B. 7　　C. 8　　D. 9

【答案】B

【分析】简单地说，5个数的全排列，共有5＊4＊3＊2＊1＝120(种)，而每次比较能得到或大或小两种情况，n次比较可得到2^n种情况，所以要想区分出这120种情况，至少要n＝7，即有2^7＝128＞120。也就是说至少要7次比较。

或者：

5个数7次排好，具体如下：

假设5个数分别编号1,2,3,4,5。

1v2,3v4……2次，假设1＞2,3＞4；

1v3,……1次，假设1＞3，则1＞3＞4；

把5插入1,3,4中，需要2次，结果有4种：

5＞1＞3＞4，则把2插入3,4中，需2次；

1＞5＞3＞4，则把2插入5,3,4中，需2次；

1＞3＞5＞4，同上；

1＞3＞4＞5，同上。

这样一共需2＋1＋2＋2＝7(次)。

12.【NOIP2000】某数列有1000个各不相同的单元，由低至高按序排列。现要对该数列进行二分法检索(binary search)，在最坏的情况下，须检视(　　)个单元。

A. 1000　　B. 10　　C. 100　　D. 500

【答案】B

【分析】由二分法定义得：2^9＜1000＜2^{10}，须检视10个单元。

13.【NOIP2001】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(　　)。

A. 2　　B. 3　　C. 4　　D. 5

【答案】C

【分析】按二分法，该数需依次和15,7,10,14比较，共4次。

14.【NOIP2004 提高组】由3个a,5个b和2个c构成的所有字符串中，包含子串“abc”的共有(　　)个。

A. 40320　　B. 39600　　C. 840　　D. 780　　E. 60

【答案】D

【分析】排列组合。

方法一：将abc当做d，原问题可以分为{2a,4b,1c,1d}和{1a,3b,2d}。

{1a,3b,2d}——6!/(3!＊2!)＝60，其中3!为b的重复，2!为d的重复；

{2a,4b,1c,1d}——8!/(2!＊4!)＝840。

840－60＝780。

方法二：2个a、4个b、1个c、1个d(abc)的重排列：(2＋4＋1＋1)!/(2!＊4!)＝840，再减去1个a、3个b、2个d(abc)的多重排列，6!/(2!＊3!)＝60(60为有出现两个abc子串的情况)，即840－60＝780。

方法三：(8＊7!)/(2!＊4!)－(5＋4＋3＋2＋1)＊4＝780。

说明：考虑“abc”的摆放位置，共有8种，余下的7个字符的全排列有7!种。但是，在这些7!种全排列中，a的重复摆放共有2!种，b的重复摆放有4!种。此外，在余下的7个字符中，仍有可能出现“abc”的排列，这与前面考虑的8种“abc”的摆放是重复的，也要去掉。这时，根据头一个“abc”的摆放起点位置，后一个“abc”分别有5、4、3、2、1种可能的摆放位置，而一旦第二个“abc”摆放好后，余下的一个a和三个b的摆放位置有C(4,1)种可能，因而得上式。

15.【NOIP2005 提高组】字符串“ababacbab”和字符串“abcba”的最长公共子串是(　　)。

A. abcba　　B. cba　　C. abc　　D. ab　　E. bcba

【答案】B

【分析】暴力模拟一下，或者动态规划一下。

16.【NOIP2000】电线上停着两种鸟(A,B)，可以看出两只相邻的鸟就将电线分为了一个线段。这些线段可分为两类：一类是两端的小鸟相同；另一类则是两端的小鸟不相同。已知：电线两个顶点上正好停着相同的小鸟，试问两端为不同小鸟的线段数目一定是(　　)。

A. 奇数　　B. 偶数　　C. 可奇可偶　　D. 数目固定

【答案】B

【分析】本题可考虑为每增加一只小鸟，两端为不同小鸟的线段数目的奇偶性是否发生变化。设想第一种情况，新增加的小鸟(如A)落在了同类鸟(如AA)中间，则两端为不同小鸟的线段数目不变，奇偶性不发生变化；第二种情况，新增加的小鸟(如A)落在了两只同类鸟(如BB)中间，两端为不同小鸟的线段数目增加2，奇偶性不发生变化；第三种情况，新增加的小鸟(如A)落在了一只同类鸟和一只不同类鸟(AB或BA)，则两端为不同小鸟的线段数目减少一个，增加一个，总数不变，奇偶性也不发生变化。可设定开始只有两只相同的小鸟停在两端，两端为不同小鸟的线段数目为0，是偶数，小鸟一只只飞进来，都不改变其奇偶性，因此一定是偶数。

17.【NOIP2007】在关系数据库中，存放在数据库中的数据的逻辑结构以(　　)为主。

A. 二叉树　　B. 多叉树　　C. 哈希表　　D. 二维表

【答案】D

【分析】在关系数据库中，存在数据库中的数据的逻辑结构以二维表为主，数据库中的表横行为记录，列标题为字段，类似excel表格，但又不是电子表格。

18.【NOIP2000】线性表若采用链表存储结构，要求内存中可用存储单元地址(　　)。

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

【答案】D

【分析】如果用链表，每个元素都有一个指针域，指向下一个节点，所以可以不连续，当然连续也可以。如果换成数组，那么必须连续，因为数组本身就是连续的。

19.【NOIP2000】下列叙述中，正确的是(　　)。

A. 线性表的线性存储结构优于链表存储结构

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

【答案】D

【分析】链表存储结构在某些方面优于线性存储结构，队列的操作方式是先进先出，堆栈的操作方式是后进先出，因此A、B、C都是错误的。

第 2 节　逻辑运算

逻辑运算先掌握各种运算，注意运算符的级别比较，做题时要细心。

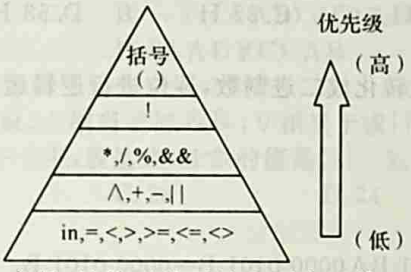
1. 概念介绍

非：!　　与：&　　或：|　　异或：∧

2. 运算级比较

括号＞非＞与＞或、异或(|和∧是同级的)。

如果加入加减乘除，就是以下这样：



注意：同级的运算符不分高低，计算时按照从左到右运算。

例题 1　若 A＝True，B＝False，C＝True，D＝False，以下逻辑运算表达式真的有(　　)。

A. (A∧B)∨(C∧D ∨¬A)　　B. ((¬A ∧ B)∨ C)∧ ¬B

C. (B∨C∨D)∨D∧A　　D. A ∧(D ∨¬C)∧B

【解答】根据运算级的比较，我们可以定下运算的顺序，然后按运算顺序计算结果。注意，这类题有个小技巧。比如 A 选项，可以先看中间的∨，为什么呢？因为∨的左右有一边是真就行，可以不去看另外一边。

A 选项的结果是：(A∧B)＝假，(C∧D ∨¬A)中 C∧D＝假，¬A＝假，所以(C∧D ∨¬A)＝假。于是 A 选项可以简写为：假∨(假 ∨假)＝假。

B 选项的结果是：如果¬B 是假，那么就可以不去看前面的((¬A ∧ B)∨ C)，可惜的是¬B 是真，那么就要看((¬A ∧ B)∨ C)，发现 C 是真，所以不看(¬A ∧ B)，于是 B 选项可以简写为：(？ ∨ 真)∧ 真＝真。

C 选项的结果是：D∧A＝假，所以不得不看前面部分(B∨C∨D)，只要 B、C、D 有一个是真，那么(B∨C∨D)＝真，而容易发现 C＝true。所以，C 选项可以简写为：真∨ 假＝真。

D 选项的结果是：我们很容易发现 D 选项的特殊结构为 ？ ∧？ ∧？，三个“?”有一个是假，那么 D 为假。A 和 B 不用计算便可看出，所以先发现 B＝假，故 D 选项的结果为假。

例题 2　计算 23+2 | 2 & 5 * 3−6 ^ 5＝(　　)。

【解答】数字也有逻辑运算，当然也可以混合加减乘除。

这里举例说明运算的操作：

&：22 & 5			|:22 | 5		
22：10110	⟶	10110	22：10110	⟶	10110
5：101 (缺位补零)		00101	5：101 (缺位补零)		00101
(垂直对应两位 & 运算)		00100＝4	(垂直对应两位 | 运算)		10111＝23

2004 年和 2005 年的比赛中都出现了集合运算问题，虽然后来没有再出现，但集合的运算还是需要掌握的。例如：全集{a,b,c,d,e,f,g}，集合 A{a,b,c}，集合 B{b,d,e}。

并运算(∪)：比如 A∪B，就是 A 集合和 B 集合里所有元素组成一个新集合，重复的元素只保留一份。A∪B —＞{a, b, c, b, d, e}—＞{a, b, c, d, e}。

交运算(∩):比如 A∩B,就是同时在 A 集合和 B 集合的元素组成一个新集合。A∩B —>{b}。

差运算(—):比如 A—B,就是 A 集合删去 A∩B 里的元素后组成一个新集合。A—B —>{a, c}。

非运算(~)(区别于逻辑非运算:¬):若是单目运算符,比如~A,非运算有个特殊的要求:一定要说明全集,那么~A 就为全集删去 A 集合中的元素,剩下的全集中的元素组成一个新集合。~A={d, e, f, g}。

【课堂练习】

1.【NOIP2002 提高组】已知 A=35 H,A ∧ 05 H ∨ A ∧ 30 H 的结果是(　　)。

A. 30 H　　B. 05 H　　C. 35 H　　D. 53 H

【答案】C

【分析】将上述十六进制数转化成二进制数,逐位进行逻辑运算,注意:与运算比或运算的优先级高。

35 H=0011 0101 B,

05 H=0000 0101 B,

30 H=0011 0000 B。

所以 A∧05 H=0011 0101 B∧0000 0101 B=0000 0101 B,

A∧30 H=0011 0101 B∧0011 0000 B=0011 0000 B,

A∧05 H∨A∧30 H=0000 0101 B∨0011 0000 B=0011 0101 B=35 H。

2.【NOIP2003】假设 A=true,B=false,C=true,D=true,逻辑运算表达式 A∧B∨C∧D 的值是(　　)。

A. true　　B. false　　C. 0　　D. 1　　E. NULL

【答案】A

【分析】注意:"与"运算比"或"运算的优先级高,所以 A∧B 和 C∧D 先运算,分别得到 false 和 true,然后 false∨true 得到 true。

3.【NOIP2003 提高组】设全集 E={1,2,3,4,5},集合 A={1,4},B={1,2,5},C={2,4},则集合(A∩B)∪~C 为(　　)。

A. 空集　　B. {1}　　C. {3,5}　　D. {1,5}　　E. {1,3,5}

【答案】E

【分析】集合运算的一元操作符优先级高于二元,求"补"的优先级比"交"和"并"高,交运算和并运算优先级相同。A∩B 是交运算,结果是{1},~C 的结果是{1,3,5},最后进行的运算是{1}∪{1,3,5},最终结果为{1,3,5}。

4.【NOIP2004 提高组】设全集 I={a, b, c, d, e, f, g},集合 A={a, b, c},B={b, d, e},C={e, f, g},那么集合(A—B)∪(~C∩B)为(　　)。

A. {a, b, c, d}　B. {a, b, d, e}　C. {b, d, e}　D. {b, c, d, e}　E. {d, f, g}

【答案】A

【分析】考查集合知识。

5.【NOIP2005 普及组】设 A=true,B=false,C=false,D=true,以下逻辑运算表达式值为真的是(　　)。

A. (A∧B)∨C∧D　　B. ((A∧B)∨C)∧D　　C. A∧(B∨C∧D)

D. (A∧B∨C)∨D　　E. (A∨B)∧C∧D

【答案】D

【分析】考查逻辑运算的知识,∧相当于与 &&,∨相当于或||。

6.【NOIP2005 提高组】设全集 I={a, b, c, d, e, f, g, h},集合 B∪A={a, b, c, d, e, f},C∩A={c, d, e},~B∩A={a, d},那么集合 C∩B∩A 为(　　)。

A. {c, e}　　B. {d, e}　　C. {e}　　D. {c, d, e}　　E. {d, f}

【答案】A

【分析】考查集合知识。

7.【NOIP2006 普及组】设 A=B=D=true,C=false,以下逻辑运算表达式值为真的有(　　)。

A. (¬A∧B)∨C∧D　　B. ¬ ((A∨B∨D)∧C)

C. ¬A∧B∨C∨D　　D. (A∧B∧C)∨¬ D

【答案】B

【分析】∧相当于与 &&,∨相当于或||,¬相当于非。

8.【NOIP2007 普及组】设 A=B=true,C=D=false,以下逻辑运算表达式值为假的有(　　)。

A. (¬A∧B)∨C∧D∨A　　B. ¬ (((A∧B)∨C)∧D)

C. A∧B∨C∨D∨D　　D. (A∧D∨C) ∧B

【答案】D

【分析】考查逻辑运算的知识,∧相当于与 &&,∨相当于或||,¬相当于非。

9.【NOIP2006 普及组】在 C++中,表达式 21^2 的值是(　　)。

A. 441　　B. 42　　C. 23　　D. 24

【答案】C

【分析】"^"异或操作。

21 转换为二进制 10101,2 转换为二进制 00010,两数异或得到结果 10111,再将其转换为十进制数,得到结果 23。

	16	8	4	2	1
.	1	0	1	0	1
xor^	0	0	0	1	0
	1	0	1	1	1

10.【NOIP2006】在 C++中,判断 a 不等于 0 且 b 不等于 0 的正确的条件表达式是(　　)。

A. ! a==0 || ! b==0　　B. ! ((a==0) && (b)==0)

C. ! (a==0 && b==0)　　D. a && b

【答案】D

【分析】0 为假,非 0 为真。

11.【NOIP2007】在 C++程序中,表达式 23|2^5 的值是(　　)。

A. 23　　B. 1　　C. 32　　D. 18

【答案】A

【分析】"^"为位运算异或,不同为 1,相同为 0,$23=(10111)_2$,$2=(10)_2$,$5=(101)_2$,位运算优先级 &>^>|,2^5=7,23|7=23。

12.【NOIP2007】在 C++程序中,判断 a 等于 0 或 b 等于 0 或 c 等于 0 的正确的条件表达式是(　　)。

A. ! ((a! =0) || (b! =0) || (c)! =0)

B. ! ((a! =0) && (b! =0) && (c)! =0)

C. ! (a==0 && b==0) || (c)! =0

D. (a=0) && (b=0) && (c)=0

【答案】B

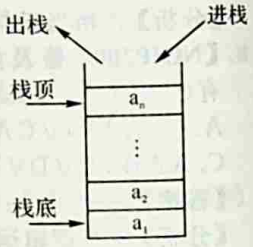
【分析】D 语法错误先排除,题目要求相当于 a==0 || b==0 || c==0,abc 只要有一个为真就成立,列个 2^3 的 01 真假值表,可以穷举验证一下。

第3节　栈

栈是只能在某一端插入和删除的特殊线性表。

用桶堆积物品，先堆进来的压在底下，随后一件一件往上堆。取走时，只能从上面一件一件取。堆和取都在顶部进行，底部一般是不动的。

栈就是一种类似桶堆积物品的数据结构，进行删除和插入的一端称栈顶，另一堆称栈底。插入一般称为进栈(PUSH)，删除则称为退栈(POP)。栈也称为后进先出表(LIFO表)。



一个栈可以用定长为n的数组s来表示，用一个栈指针top指向栈顶。若top=0，表示栈空，top=n时栈满。进栈时top加1，退栈时top减1。当top<0时为下溢。栈指针在运算中永远指向栈顶。

1. 进栈(PUSH)算法

①若top≥n时，则给出溢出信息，做出错处理(进栈前首先检查栈是否已满，满则溢出；不满则做②)。

②top++(栈指针加1，指向进栈地址)。

③s[top]=x，结束(x为新进栈的元素)；

2. 退栈(POP)算法

①若top≤0，则给出下溢信息，做出错处理(退栈前先检查是否已为空栈，空则下溢；不空则作②)。

②x=s[top](退栈后的元素赋给x)。

③top--，结束(栈指针减1，指向栈顶)。

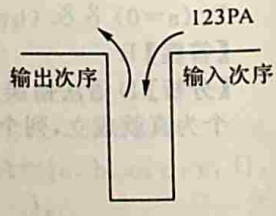
进栈、出栈的C++实现过程程序：

```
#define n 100
void push(int s[],int *top,int *x)                    //入栈
{
   if (*top==n)printf("overflow");
  else {*top++;s[*top]=*x;}
}
void pop(int s[],int *y,int *top)                     //出栈
{
 if (*top==0) printf("underflow");
else {*y=s[*top];*top--;}
}
```

对于出栈运算中的“下溢”，程序中仅给出了一个标志信息，而在实际应用中，下溢可用来作为控制程序转移的判断标志，是十分有用的。对于入栈运算中的“上溢”，则是一种致命的错误，将使程序无法继续运行，所以要设法避免。

从历届初赛可以看出，栈也是属于比较容易出题的知识点，选择题、解答题等题型都有可能出现。

例题1　设输入元素为1、2、3、P和A，输入次序为123PA，如图所示。元素经过栈后到达输出序列，当所有元素均到达输出序列后，有哪些序列可以作为高级语言的变量名？

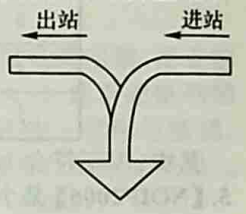


【解答】高级语言变量名的定义规则：以字母开头，字母与数字的组合。

由于必须以字母开头，所以，第一个可能出现的字母是P，那么

必然要求 123 已经先后入栈，这样便可得到以 P 开头的、并且 123 按逆序排列的(即 321)、同时 A 位于 P 后任一位置的变量名序列。此外，还需要考虑以 A 开头的可能情况，这只有一种情形：AP321。所以 AP321、PA321、P3A21、P32A1、P321A 可以作为高级语言的变量名。

例题 2　火车站列车调度示意图如右，假设调度站两侧的轨道为单向行驶轨道。



①如果进站的车厢序列为 123，则可能的出站车厢序列是什么？

②如果进站的车厢序列为 123456，问：能否得到 135426 和 435612 的出站序列？

【课堂练习】

1.【NOIP2001】若已知一个栈的入栈顺序是 1，2，3，…，n，其输出序列为 P_1，P_2，P_3，…，P_n，若 P_1 是 n，则 P_i 是　　(　　)

A. i　　B. n－1　　C. n－i＋1　　D. 不确定

【答案】C

【分析】归纳总结

P_1　　n

P_2　　n－1

P_3　　n－2

…

P_n　　1

总结：P_i 对应 n－i＋1

2.【NOIP2001 提高组】以下哪一个不是栈的基本运算(　　)。

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

【答案】B

【分析】堆栈删除和插入操作只能在栈顶进行。

3.【NOIP2002 提高组】设栈 S 和队列 Q 的初始状态为空，元素 e_1，e_2，e_3，e_4，e_5，e_6 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 e_2，e_4，e_3，e_6，e_5，e_1，则栈 S 的容量至少应该为(　　)。

A. 2　　B. 3　　C. 4　　D. 5

【答案】B

【分析】因为依次入栈，所以一个元素入队时，比它标号小的元素要么已入队，要么还在栈中，对每个元素计算比它标号小的元素总数减去队列中比它标号小的元素个数，得到尚在栈中的元素个数，找出最大值加 1(其本身刚出栈)，就是所要结果。例中 e_6 入队时，队中有 e_2，e_4，e_3 共 3 个元素，比 e6 标号小的有 5 个元素，故栈 S 的容量至少为 5－3＋1＝3。

4.【NOIP2003 提高组】已知元素(8，25，14，87，51，90，6，19，20)，则这些元素以(　　)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。

A. 20，6，8，51，90，25，14，19，87　　B. 51，6，19，20，14，8，87，90，25

C. 19，20，90，8，6，25，51，14，87　　D. 6，25，51，8，20，19，90，87，14

E. 25，6，8，51，87，90，19，14，20

【答案】D

【分析】根据堆栈先进后出的操作原则，出栈顺序和入栈顺序相反，将题中所给两数间的前后关系颠倒过来：

1	8 在 51 的前面		8 在 51 的后面
2	90 在 87 的后面		90 在 87 的前面
3	20 在 14 的后面		20 在 14 的前面
4	25 在 6 的前面		25 在 6 的后面
5	19 在 90 的后面		19 在 90 的前面

其中：A 不符合 1.5；B 不符合 2；C 不符合 1；E 不符合 1,2,3,4,5。符合条件的只有 D。

5.【NOIP2006】某个车站呈狭长形，宽度只能容下一台车，并且只有一个出入口。已知某时刻该车站状态为空，从这一时刻开始的出入记录为："进，出，进，进，进，出，出，进，进，进，出，出"。假设车辆入站的顺序为 1,2,3,…，则车辆出站的顺序为(　　)。

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

【答案】C

【分析】栈按先进后出顺序，元素上面有元素就不能出栈。

6.【NOIP2006】设栈 S 的初始状态为空，元素 a,b,c,d,e 依次入栈，以下出栈序列不可能出现的有(　　)。

A. a,b,c,e,d　　B. b,c,a,e,d　　C. a,e,c,b,d　　D. d,c,e,b,a

【答案】C

【分析】C 中 a 和 e 出栈后，从栈顶到栈底有 d,c,b,d 没有出栈 c 和 b 出不来。

7.【NOIP2007】地面上有标号为 A、B、C 的 3 根细柱，在 A 柱上放有 10 个直径相同中间有孔的圆盘，从上到下依次编号为 1,2,3,……，将 A 柱上的部分盘子经过 B 柱移入 C 柱，也可以在 B 柱上暂存。如果 B 柱上的操作记录为："进，进，出，进，进，出，出，进，进，出，进，出，出"，那么，在 C 柱上，从下到上的盘子的编号为(　　)。

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

【答案】D

【分析】本题就是一个栈，按照"进、出……"的描述，进栈、出栈的顺序是固定的，不得调整。这样一来，进、出栈的顺序就是：1 进，2 进，2 出，3 进，4 进，4 出，3 出，5 进，6 进，6 出，7 进，7 出，5 出。

第 4 节　队列

队列是限定在一端进行插入，另一端进行删除的特殊线性表。就像排队买东西，排在前面的人买完东西后离开队伍（删除），而后来的人总是排在队伍末尾（插入）。通常把队列的删除和插入分别称为出队和入队。允许出队的一端称为队头，允许入队的一端称为队尾。所有需要进队的数据项，只能从队尾进入，队列中的数据项只能从队头离去。由于总是先入队的元素先出队（先排队的人先买完东西），这种表也称为先进先出（FIFO）表。

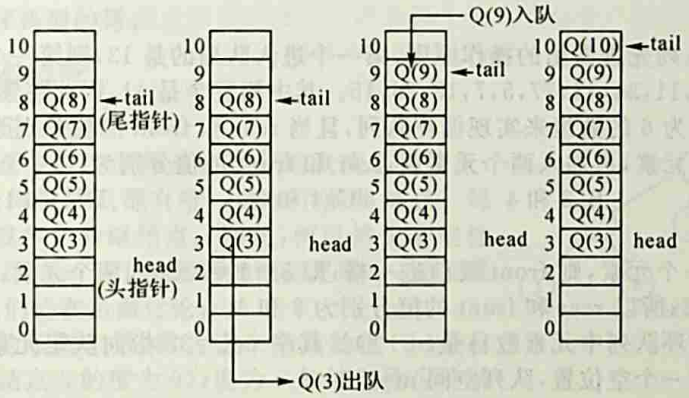
队列可以用数组 Q[m+1]来存储，数组的上界 m 即是队列所容许的最大容量。在队列的运算中需设两个指针：

head：队头指针，指向实际队头元素的前一个位置

tail：队尾指针，指向实际队尾元素所在的位置

一般情况下，两个指针的初值设为 0，这时队列为空，没有元素。下图（a）画出了一个由 6 个元素构成的队列，数组定义 Q[11]。

Q(i) i=3,4,5,6,7,8，头指针 head=2，尾指针 tail=8。



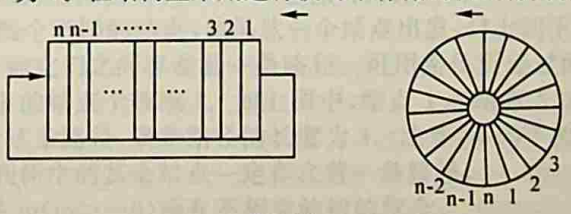
队

队列中拥有的元素个数为：L=tail－head 现要让排头的元素出队，则需将头指针加 1，即 head＋＋。这时头指针向上移动一个位置，指向 Q(3)，表示 Q(3)已出队。见上图（b）。

如果想让一个新元素入队，则需尾指针向上移动一个位置。即 tail＋＋这时 Q(9)入队，见上图（c）。

当队尾已经处理在最上面时，即 tail=10，见图上（d），如果还要执行入队操作，则要发生“上溢”，但实际上队列中还有三个空位置，所以这种溢出称为“假溢出”。

克服假溢出的方法有两种。一种是将队列中的所有元素均向低地址区移动，显然这种方法是很浪费时间的；另一种方法是将数组存储区看成是一个首尾相接的环形区域。当存放到 n 地址后，下一个地址就“翻转”为 1。在结构上采用这种技巧来存储的队列称为循环队列，见下图：



循环队

循环队的入队算法如下：

1. tail＋＋；
2. 若 tail＝n＋1，则 tail＝1；
3. 若 head＝tail 尾指针与头指针重合了，表示元素已装满队列，则作上溢出错处理；
4. 否则，Q(tail)＝x，结束(x 为新入队元素)。

【课堂练习】

1.【NOIP2000 普及组】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(　　)。

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

【答案】D

【分析】循环队列中 f 和 r 不一定满足 f＜r，如果满足 f＜r，则元素个数为 r－f；但有时也会 r＜f，所以需要加 n 然后对 n 求余才能统一表示。

2.【NOIP2003】已知队列(13，2，11，34，41，77，5，7，18，26，15)，第一个进入队列的元素是 13，则第五个出队列的元素是(　　)。

A. 5　　B. 41　　C. 77　　D. 13　　E. 18

【答案】B

【分析】根据队列先进先出的操作原则，第一个进入队列的是 13，则第一个出队的也是 13，向后依次为 2，11，34，41，77，5，7，18，26，15。其中第五个是 41，因此答案为 B。

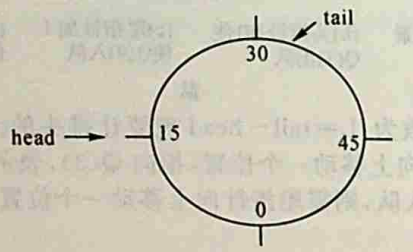
3. 若用一个大小为 6 的数组来实现循环队列，且当 rear 和 front 的值分别为 0 和 3。当从队列中删除一个元素，再加入两个元素后，rear 和 front 的值分别为(　　)。

A. 1 和 5　　B. 2 和 4　　C. 4 和 2　　D. 5 和 1

【答案】B

【分析】删除一个元素，则 front 要前进一格，即 3＋1＝4；加入两个元素，则 rear 要前进两格，即 0＋2＝2；所以 rear 和 front 的值分别为 2 和 4。

4. 如图所示的循环队列中元素数目是(　　)。其中 tail＝32 指向队尾元素，head＝15 指向队头元素的前一个空位置，队列空间 m＝60.



A. 42　　B. 16　　C. 17　　D. 41

【答案】C

【分析】元素数目可以根据公式(tail－head＋60)%60 得到，即(32－15＋60)%60＝17。

第5节　树

前两章学习的栈和队列属于线性结构。在这种结构中，数据元素的逻辑位置之间呈线性关系，每一个数据元素通常只有一个前件（除第一个元素外）和一个后件（除最后一个元素外）。在实际生活中，可以用线性结构描述数据元素之间逻辑关系的问题是很广泛的，但也有很多问题不能依靠线性结构来解决，例如家谱、行政组织机构等都是非线性的数据结构。其中树就是一种非线性的数据结构。

一、树的定义

一棵树是由 n(n>0)个元素组成的有限集合，其中：

(1)每个元素称为结点(node)；

(2)有一个特定的结点，称为根结点或树根(root)；

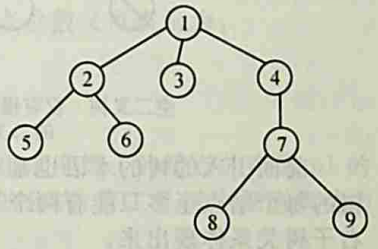
(3)除根结点外，其余结点能分成 m(m>=0)个互不相交的有限集合 T_0，T_1，T_2，……T_{m-1}。其中的每个子集又都是一棵树，这些集合称为这棵树的子树。

如下图是一棵典型的树：

二、树的基本概念

A. 树是递归定义的；

B. 一棵树中至少有 1 个结点。这个结点就是根结点，它没有前驱，其余每个结点都有唯一的一个前驱结点。每个结点可以有 0 或多个后继结点。因此，树虽然是非线性结构，但也是有序结构。至于前驱后继结点是哪个，还要看树的遍历方法，我们将在后面讨论；



C. 一个结点的子树个数，称为这个结点的度(degree，结点 1 的度为 3，结点 3 的度为 0)；度为 0 的结点称为叶结点(树叶 leaf，如结点 3、5、6、8、9)；度不为 0 的结点称为分支结点(如结点 1、2、4、7)；根以外的分支结点又称为内部结点(结点 2、4、7)；树中各结点的度的最大值称为这棵树的度(这棵树的度为 3。

D. 在用图形表示的树型结构中，对两个用线段(称为树枝)连接的相关联的结点，称上端结点为下端结点的父结点，称下端结点为上端结点的子结点。称同一个父结点的多个子结点为兄弟结点。如结点 1 是结点 2、3、4 的父结点，结点 2、3、4 是结点 1 的子结点，它们又是兄弟结点，同时结点 2 又是结点 5、6 的父结点。称从根结点到某个子结点所经过的所有结点为这个子结点的祖先。如结点 1、4、7 是结点 8 的祖先。称以某个结点为根的子树中的任一结点都是该结点的子孙。如结点 7、8、9 都是结点 4 的子孙。

E. 定义一棵树的根结点的层次(level)为 1，其他结点的层次等于它的父结点层次加 1。如结点 2、3、4 的层次为 2，结点 5、6、7 的层次为 3，结点 8、9 的层次为 4。一棵树中所有的结点的层次的最大值称为树的深度(depth)。如这棵树的深度为 4。

F. 对于树中任意两个不同的结点，如果从一个结点出发，自上而下沿着树中连着结点的线段能到达另一结点，称它们之间存在着一条路径。可用路径所经过的结点序列表示路径，路径的长度等于路径上的结点个数减 1。如上图中，结点 1 和结点 8 之间存在着一条路径，并可用(1、4、7、8)表示这条路径，该条路径的长度为 3。注意，不同子树上的结点之间不存在路径，从根结点出发，到树中的其余结点一定存在着一条路径。

G. 森林(forest)是 m(m>=0)棵互不相交的树的集合。

三、树的遍历

在应用树结构解决问题时，往往要求按照某种次序获得树中全部结点的信息，这种操作叫作树的遍历。遍历的方法有多种，常用的有：

A. 先序（根）遍历：先访问根结点，再从左到右按照先序思想遍历各棵子树。

如上图先序遍历的结果为：125634789。

B. 后序（根）遍历：先从左到右遍历各棵子树，再访问根结点。

如上图后序遍历的结果为：562389741。

C. 层次遍历：按层次从小到大逐个访问，同一层次按照从左到右的次序。

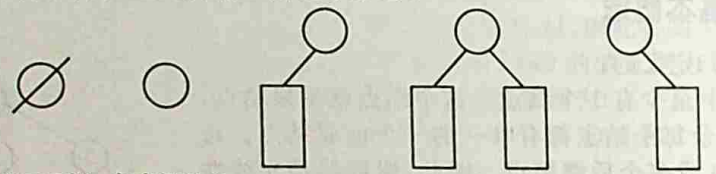
如上图层次遍历的结果为：123456789。

D. 叶结点遍历：有时把所有的数据信息都存放在叶结点中，而其余结点都是用来表示数据之间的某种分支或层次关系，这种情况就用这种方法。

如上图按照这个思想访问的结果为：56389。

四、二叉树基本概念

二叉树（binary tree，简写成 BT）是一种特殊的树型结构，它的度数为 2 的树。即二叉树的每个结点最多有两个子结点。每个结点的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：



前面引人的树的术语也基本适用于二叉树，但二叉树与树也有很多不同，如：首先二叉树的每个结点至多只能有两个子结点，二叉树可以为空，二叉树一定是有序的，通过它的左、右子树关系体现出来。

五、二叉树的性质

【性质 1】在二叉树的第 i 层上最多有 2^{i-1} 个结点（i>=1）。

证明：很简单，用归纳法：当 i=1 时，$2^{i-1}=1$ 显然成立；现在假设第 i−1 层时命题成立，即第 i−1 层上最多有 2^{i-2} 个结点。由于二叉树的每个结点的度最多为 2，故在第 i 层上的最大结点数为第 i−1 层的 2 倍，即 $2*2^{i-2}=2^{i-1}$。

【性质 2】深度为 k 的二叉树至多有 2^k-1 个结点（k>=1）。

证明：在具有相同深度的二叉树中，仅当每一层都含有最大结点数时，其树中结点数最多。因此利用性质 1 可得，深度为 k 的二叉树的结点数至多为：

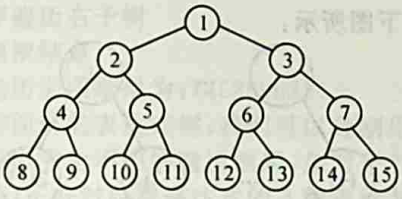
$$2^0+2^1+\cdots+2^{k-1}=2^k-1$$

故命题正确。

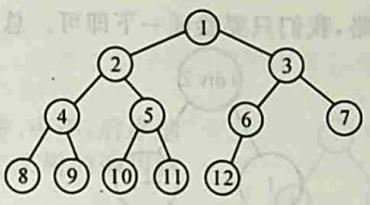
特别：一棵深度为 k 且有 2^k-1 个结点的二叉树称为满二叉树。如下图 A 为深度为 4 的满二叉树，这种树的特点是每层上的结点数都是最大结点数。

可以对满二叉树的结点进行连续编号，约定编号从根结点起，自上而下，从左到右，由此引出完全二叉树的定义，深度为 k，有 n 个结点的二叉树当且仅当其每一个结点都与深度为 k 的满二叉树中编号从 1 到 n 的结点一一对应时，称为完全二叉树。

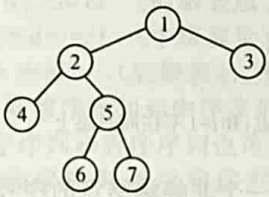
下图 B 就是一个深度为 4，结点数为 12 的完全二叉树。它有如下特征：叶结点只可能在层次最大的两层上出现；对任一结点，若其右分支下的子孙的最大层次为 m，则在其左分支下的子孙的最大层次必为 m 或 m+1。下图 C、D 不是完全二叉树，请大家思考为什么？



图A



图B



图C

图D

【性质 3】对任意一棵二叉树，如果其叶结点数为 n_0，度为 2 的结点数为 n_2，则一定满足：$n_0=n_2+1$。

证明：因为二叉树中所有结点的度数均不大于 2，所以结点总数（记为 n）应等于 0 度结点数 n_0、1 度结点 n_1 和 2 度结点数 n_2 之和：

$$n=n_0+n_1+n_2\cdots\cdots(式子\ 1)$$

另一方面，1 度结点有一个孩子，2 度结点有两个孩子，故二叉树中孩子结点总数是：

$$n_1+2n_2$$

树中只有根结点不是任何结点的孩子，故二叉树中的结点总数又可表示为：

$$n=n_1+2n_2+1\cdots\cdots(式子\ 2)$$

由式子 1 和式子 2 得到：

$$n_0=n_2+1$$

例题 1. 有 n 个结点的二叉树，已知叶结点个数为 n0，写出求度为 1 的结点的个数 n1 的计算公式；若此树是深度为 k 的完全二叉树，写出 n 为最小的公式；若二叉树中仅有度为 0 和度为 2 的结点，写出求该二叉树结点个数 n 的公式。

【解答】(1)记度为 2 的结点个数为 n2，

则 n=n0+n1+n2 ①

又因为除了根结点以外，其余结点均由父结点射出，所以

n=1+n1+2*n2 ②

由①②两式得 n1=n+1-2n0

(2)当树是深度为 k 的完全二叉树时，n 的最小值 $n_{min}=2^{k-1}$

(3)当二叉树中仅有度为 0 和度为 2 的结点时，

n=n0+n2，n=2n2+1

所以：n0=n2+1

n=2n0-1

【性质 4】具有 n 个结点的完全二叉树的深度为 floor($\log_2^n$)+1

证明：假设深度为 k，根据完全二叉树的定义，前面 k-1 层一定是满的，所以 $n>2^{k-1}-1$。但 n 又要满足 $n<=2^k-1$。所以，$2^{k-1}-1<n<=2^k-1$。变换一下为 $2^{k-1}<=n<2^k$。

以 2 为底取对数得到：$k-1<=\log_2^n<k$。而 k 是整数，所以 k=floor($\log_2^n$)+1。

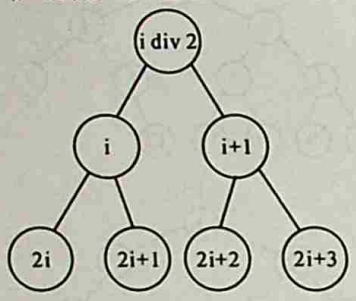
【性质 5】对于一棵 n 个结点的完全二叉树，对任一个结点（编号为 i），有：

①如果 i=1，则结点 i 为根，无父结点；如果 i>1，则其父结点编号为 i/2。

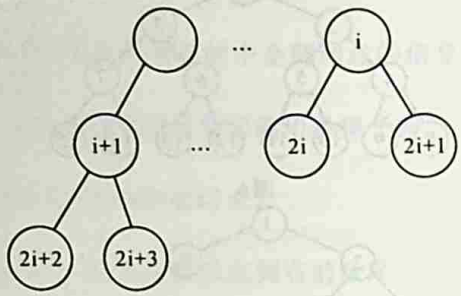
如果 2*i>n，则结点 i 无左孩子（当然也无右孩子，为什么？即结点 i 为叶结点）；否则左孩子编号为 2*i。

②如果 2*i+1>n，则结点 i 无右孩子；否则右孩子编号为 2*i+1。

证明:略,我们只要验证一下即可。总结如下图所示:



结点i和i+1在同一层上



结点i和i+1不在同一层上

例题 2. 已知一棵完全二叉树共有 892 个结点,试求:

(1)树的高度(2)叶子结点数(3)单支结点数(4)最后一个非终端结点的序号

【解答】 1)已知深度为 k 的二叉树至多有 2^k-1 个结点($k\geqslant 1$),由于 $2^9-1<892<2^{10}-1$,所以树的高度为 10。

2)对完全二叉树来说,度为 1 的结点只能是 0 或 1。由 $n=n0+n1+n2$ 和 $n0=n2+1$ 得:

①设 $n1=0$,则有:$892=n0+0+n2=n2+1+n2=2n2+1$,因得到的 n2 不为整数而出错;

②设 $n1=1$,则有:$892=n0+1+n2=n2+1+1+n2=2n2+2$,得到 $n2=445$,代入 $n0=n2+1$,得到 $n0=446$,故叶子结点数为 446。

3)由 2)可知单支结点数为 1。

4)对有 n 个结点的完全二叉树,最后一个树叶结点,即序号为 n 的叶结点其双亲结点 n/2,即为最后一个非终端结点,也即序号为向下取整(892/2)=446。此外,由 2)可知:$n2=445$,$n1=1$;则最后一个非终端结点的序号为 $445+1=446$。

六、遍历二叉树

在二叉树的应用中,常常要求在树中查找具有某种特征的结点,或者对全部结点逐一进行某种处理。这就是二叉树的遍历问题。所谓二叉树的遍历是指按一定的规律和次序访问树中的各个结点,而且每个结点仅被访问一次。"访问"的含义很广,可以是对结点作各种处理,如输出结点的信息等。遍历一般按照从左到右的顺序,共有 3 种遍历方法:先(根)序遍历,中(根)序遍历,后(根)序遍历。

(一)先序遍历的操作定义如下:

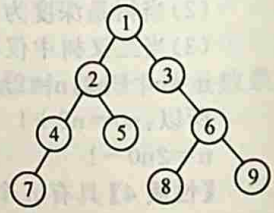
若二叉树为空,则空操作,否则:

①访问根结点

②先序遍历左子树

③先序遍历右子树

先序遍历右图二叉树的结果为:124753689



(二)中序遍历的操作定义如下:

若二叉树为空,则空操作,否则:

①中序遍历左子树

②访问根结点

③中序遍历右子树

中序遍历上图二叉树的结果为:742513869

(三)后序遍历的操作定义如下:

若二叉树为空,则空操作,否则:

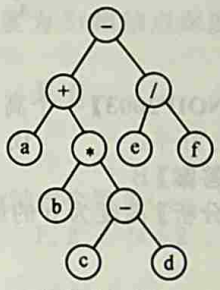
①后序遍历左子树

②后序遍历右子树

③访问根结点

后序遍历上图结果为：745289631

关于前面讲的表达式树，我们可以分别用先序、中序、后序的遍历方法得出完全不同的遍历结果，如对于右图二叉树的遍历结果如下，它们正好对应着表达式的3种表示方法。



表达式(a+b*(c−d)−e/f)的二叉树

−+a*b−cd/ef　（前缀表示、波兰式）

a+b*c−d−e/f　（中缀表示）

abcd−*+ef/−（后缀表示、逆波兰式）

结论：已知前序序列和中序序列可以确定出二叉树；

已知中序序列和后序序列也可以确定出二叉树；

但，已知前序序列和后序序列却不可以确定出二叉树；为什么？

例题3. 有二叉树中序序列为ABCEFGHD，后序序列为：ABFHGEDC。请画出此二叉树，并求前序序列。

【解答】 根据后序序列知根节点为C

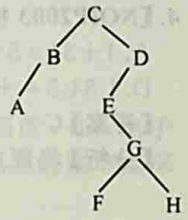
因此左子树：中序序列为AB

后序序列为AB

右子树：中序序列为EFGHD

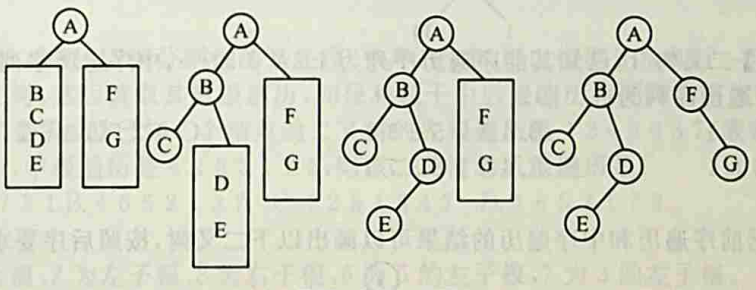
后序序列为FHGED

依次推得该二叉树的结构图。（如右图）



前序序列为：CBADEGFH。

例题4. 已知结点的前序序列为ABCDEFG，中序序列为CBEDAFG。构造出二叉树。过程见下图：



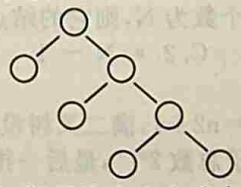
【课堂练习】

1.【NOIP2001 提高组】 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（　　）个结点。

A. 2^h-1　　B. 2h−1　　C. 2h+1　　D. h+1

【答案】 B

【分析】 符合题意最少节点的二叉树是如下形状：



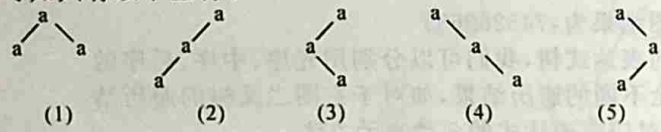
除根节点所在层节点数为1外，每层均为2个节点，因此节点数为2h−1。

2.【NOIP2002 提高组】 按照二叉数的定义，具有3个结点的二叉树有（　　）种。

A. 3　　B. 4　　C. 5　　D. 6

【答案】C

【分析】通过画图可知只有以下五种：

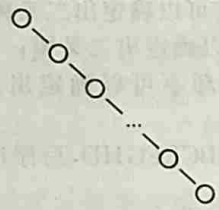


3.【NOIP2003】一个高度为 h 的二叉树最小元素数目是(　　)。

A. 2h+1　　B. h　　C. 2h−1　　D. 2h　　E. 2^h-1

【答案】B

【分析】高度为 h 的最小元素数目的二叉树如下图所示，它有 h 个节点。



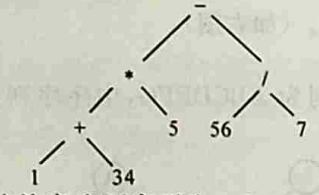
4.【NOIP2003 提高组】表达式(1+34)*5−56/7 的后缀表达式为(　　)。

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−*/

【答案】C

【分析】将原式按二叉树展开如图所示，按后根序遍历可得后缀表达式。



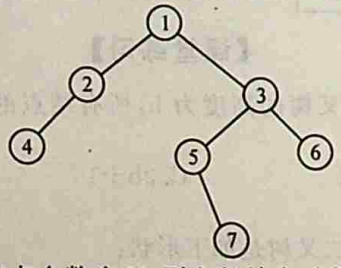
5.【NOIP2004】二叉树 T，已知其前序遍历序列为 1 2 4 3 5 7 6，中序遍历序列为 4 2 1 5 7 3 6，则其后序遍历序列为(　　)。

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　　E. 4 5 2 6 3 7 1

【答案】B

【分析】根据前序遍历和中序遍历的结果可以画出以下二叉树，按照后序要求读取即可。



6.【NOIP2004】满二叉树的叶结点个数为 N，则它的结点总数为(　　)。

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

【答案】C

【分析 1】根据二叉树的性质，n0＝n2＋1，满二叉树没有度为 1 的结点，经计算选 C。

【分析 2】总共是 2^{n-1}，第 n 排叶子总数 2^{n-1}，最后一排是 n，前面所有的和是 n−1，答案 2n−1。

7.【NOIP2005 普及组】完全二叉树的结点个数为 11，则它的叶结点个数为(　　)。

A. 4　　B. 3　　C. 5　　D. 2　　E. 6

【答案】E

【分析】左下角不许缺，右下可以。

8.【NOIP2005 普及组】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（　　）。

A. 无法确定　　B. B　　C. C　　D. D　　E. E

【答案】C

【分析】A|BC|DEF|GHI。

9.【NOIP2005 提高组】完全二叉树的结点个数为 4 * N+3，则它的叶结点个数为（　　）。

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

【答案】E

【分析】最后一个（叶）结点的编号为 4 * N+3，其父结点的编号为 2 * N+1，因此，4 * N+3−(2 * N+1)=2 * N+2

10.【NOIP2006】高度为 n 的均衡的二叉树是指：如果去掉叶结点及相应的树枝，它应该是高度为 n−1 的满二叉树。在这里，树高等于叶结点的最大深度，根结点的深度为 0，如果某个均衡的二叉树共有 2381 个结点，则该树的树高为（　　）。

A. 10　　B. 11　　C. 12　　D. 13

【答案】B

【分析】$2^{11}=2048$。如果根节点的深度为 1，则满二叉树节点总数为 $2^{}$（深度−1），由于 2048−1=2047<2381，故整个树高为 11（满二叉树）+1=12。由于本题设定的根节点深度为 0，故该题的树高为 11。

11.【NOIP2006 普及组】已知 6 个结点的二叉树的先根遍历是 1 2 3 4 5 6（数字为结点的编号，以下同），后根遍历是 3 2 5 6 4 1，则该二叉树的可能的中根遍历是（　　）。

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

【答案】B

【分析】先根、中根、后根分别指的是先序、中序、后序，可以根据先根和任一可能的中根形成一棵二叉树，然后读取其后根遍历，如果和题干中后根遍历相同即为答案。

12.【NOIP2007 普及组】已知 7 个结点的二叉树的先根遍历是 1 2 4 5 6 3 7（数字为结点的编号，以下同），中根遍历是 4 2 6 5 1 7 3，则该二叉树的后根遍历是（　　）。

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

【答案】A

【分析】1 为根，2 为左子根，3 为右子根，6 为 5 的左子根，7 为 3 的左子根。

第6节 图

图(Graph)是一种复杂的非线性结构。在人工智能、工程、数学、物理、化学、生物和计算机科学等领域中,图结构有着广泛的应用。奥林匹克信息学竞赛的许多试题,亦需要用图来描述数据元素间的联系。

图G由两个集合V和E组成,记为:G=(V,E),其中:V是顶点的有穷非空集合,E是V中顶点偶对(称为边)的有穷集。通常,也将图G的顶点集和边集分别记为V(G)和E(G)。E(G)可以是空集。若E(G)为空,则图G只有顶点而没有边。

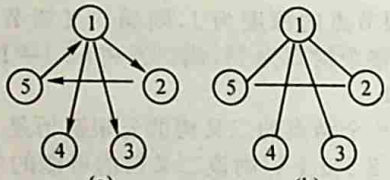
一、什么是图?

很简单,点用边连起来就叫做图,严格意义上讲,图是一种数据结构,定义为:graph=(V,E)。V是一个非空有限集合,代表顶点(结点),E代表边的集合。

二、图的一些定义和概念

(1)有向图:图的边有方向,只能按箭头方向从一点到另一点。(a)就是一个有向图。

(2)无向图:图的边没有方向,可以双向。(b)就是一个无向图。



(3)结点的度:无向图中与结点相连的边的数目,称为结点的度。

(4)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。

(5)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。

(6)权值:边的"费用",可以形象地理解为边的长度。

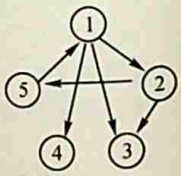
(7)连通:如果图中结点U,V之间存在一条从U通过若干条边、点到达V的通路,则称U、V是连通的。

(8)回路:起点和终点相同的路径,称为回路,或"环"。

(9)完全图:一个n阶的完全无向图含有n*(n-1)/2条边;一个n阶的完全有向图含有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。

(10)强连通分量:有向图中任意两点都连通的最大子图。右图中,1-2-5构成一个强连通分量。特殊地,单个点也算一个强连通分量,所以右图有三个强连通分量:1-2-5,4,3。



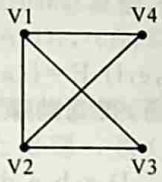
三、图的存储结构

二维数组邻接矩阵存储

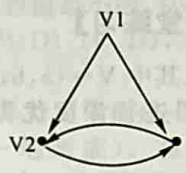
定义 int G[101][101];

G[i][j]的值,表示从点i到点j的边的权值,定义如下:

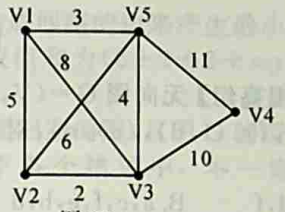
$$G[i][j]\begin{cases}1\text{或权值} & \text{当 } v_i \text{ 与 } v_j \text{ 之间有边或弧时,取值为1或权值}\\ 0\text{或}\infty & \text{当 } v_i \text{ 与 } v_j \text{ 之间无边或弧时,取值为0或}\infty\ \text{(无穷大)}\end{cases}$$



图(A)



图(B)



图(C)

上图中的3个图对应的邻接矩阵分别如下：

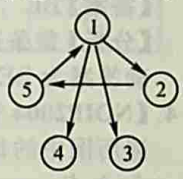
$$G(A)=\begin{bmatrix}0&1&1&1\\1&0&1&1\\1&1&0&0\\1&1&0&0\end{bmatrix}\quad G(B)=\begin{bmatrix}0&1&1\\0&0&1\\0&1&0\end{bmatrix}\quad G(C)=\begin{bmatrix}\infty&5&8&\infty&3\\5&\infty&2&\infty&6\\8&2&\infty&10&4\\\infty&\infty&10&\infty&11\\3&6&4&11&\infty\end{bmatrix}$$

四、深度优先与广度优先遍历

从图中某一顶点出发系统地访问图中所有顶点，使每个顶点恰好被访问一次，这种运算操作被称为图的遍历。为了避免重复访问某个顶点，可以设一个标志数组 visited[i]，未访问时值为 false，访问一次后就改为 true。

图的遍历分为深度优先遍历和广度优先遍历两种方法，两者的时间效率都是 O(n * n)。

1. 深度优先遍历



深度优先遍历与深搜 dfs 相似，从一个点 A 出发，将这个点标为已访问 visited[i]=true，然后再访问所有与之相连，且未被访问过的点。当 A 的所有邻接点都被访问过后，再退回到 A 的上一个点(假设是 B)，再从 B 的另一个未被访问的邻接点出发，继续遍历。

例如对右边的这个有向图深度优先遍历，假定先从 1 出发，程序以如下顺序遍历：

1→2→5，然后退回到 2，退回到 1。

从 1 开始再访问未被访问过的点 3，3 没有未访问的邻接点，退回 1。

再从 1 开始访问未被访问过的点 4，再退回 1。

起点 1 的所有邻接点都已访问，遍历结束。

2. 广度优先遍历

广度优先遍历并不常用，从编程复杂度的角度考虑，通常采用的是深度优先遍历。

广度优先遍历和广搜 bfs 相似，因此使用广度优先遍历一张图并不需要掌握什么新的知识，在原有的广度优先搜索的基础上，做一点小小的修改，就成了广度优先遍历算法。

五、一笔画问题

如果一个图存在一笔画，则一笔画的路径叫做欧拉路，如果最后又回到起点，那这个路径叫做欧拉回路。

我们定义奇点是指跟这个点相连的边数目有奇数个的点。对于能够一笔画的图，我们有以下两个定理。

定理 1：存在欧拉路的条件：图是连通的，有且只有 2 个奇点。

定理 2：存在欧拉回路的条件：图是连通的，有 0 个奇点。

两个定理的正确性是显而易见的，既然每条边都要经过一次，那么对于欧拉路，除了起点和终点外，每个点如果进入了一次，显然一定要出去一次，显然是偶点。对于欧拉回路，每个点进入和出去次数一定都是相等的，显然没有奇点。

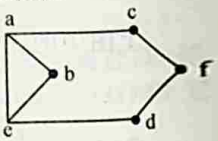
求欧拉路的算法很简单，使用深度优先遍历即可。

根据一笔画的两个定理，如果寻找欧拉回路，对任意一个点执行深度优先遍历；找欧拉路，则对一个奇点执行 dfs，时间复杂度为 O(m+n)，m 为边数，n 是点数。

【课堂练习】

1.【NOIP2001 提高组】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(　　)。

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



【答案】D

【分析】依题中描述将无向图画出如图所示:

按照深度优先搜索的规则进行搜索,得到序列{ a,b,e,d,f,c }。

2.【NOIP2002 提高组】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(　　)倍。

A. 1/2　　B. 1　　C. 2　　D. 4

【答案】B

【分析】在有向图中,所有顶点的入度之和等于所有顶点的出度之和。

3.【NOIP2003 提高组】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(　　)。

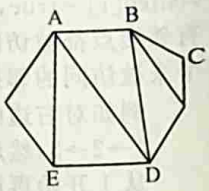
A. {5,4,4,3,1}　　B. {4,2,2,1,1}　　C. {3,3,3,2,2}

D. {5,4,3,2,1}　　E. {2,2,2,2,2}

【答案】BE

【分析】每条边被两个顶点计算度数,因此度数和必为偶数,ACD 的度数和为奇数,因此正确答案为 BE。

4.【NOIP2004 普及组】在右图中,从顶点(　　)出发存在一条路径可以遍历图中的每条边一次,而且仅遍历一次。



A. A 点　　B. B 点　　C. C 点

D. D 点　　E. E 点

【答案】E

【分析】欧拉图问题,E 是奇点。

5.【NOIP2005 普及组】平面上有五个点 A(5, 3), B(3, 5), C(2, 1), D(3, 3), E(5, 1)。以这五点作为完全图 G 的顶点,每两点之间的直线距离是图 G 中对应边的权值。以下哪条边不是图 G 的最小生成树中的边(　　)。

A. AD　　B. BD　　C. CD　　D. DE　　E. EA

【答案】D

【分析】笛卡尔坐标系画出来,生成树是 n−1 条总长最短的边连所有点。

6.【NOIP2005 提高组】平面上有五个点 A(5, 3),B(3, 5),C(2, 1),D(3, 3),E(5, 1)。以这五点作为完全图 G 的顶点,每两点之间的直线距离是图 G 中对应边的权值。图 G 的最小生成树中的所有边的权值综合为(　　)。

	A	B	C	D	E
A		$2\sqrt{2}$	$\sqrt{13}$	2	2
B	$2\sqrt{2}$		$\sqrt{17}$	2	$2\sqrt{5}$
C	$\sqrt{13}$	$\sqrt{17}$		$\sqrt{5}$	3
D	2	2	$\sqrt{5}$		$2\sqrt{2}$
E	2	$2\sqrt{5}$	3	$2\sqrt{2}$	

A. 8　　B. $7+\sqrt{5}$　　C. 9　　D. $6+\sqrt{5}$　　E. $4+2\sqrt{2}+\sqrt{5}$

【答案】D

【分析】根据 kruskal 算法，选取权值最小的、且不构成回路的边来产生最小生成树，因此，选出的边为(A,D)，(A,E)，(B,D)，(C,D)，其权值和为(2+2+2+sqrt(5))，即答案为 D。

7.【NOIP2007 提高组】欧拉图 G 是指可以构成一个闭回路的图，且图 G 的每一条边恰好在这个闭回路上出现一次(即一笔画成)。在以下各个描述中，不一定是欧拉图的是(　　)。

A. 图 G 中没有度为奇数的顶点

B. 包括欧拉环游的图(欧拉环游是指通过图中每边恰好一次的闭路径)

C. 包括欧拉闭迹的图(欧拉迹是指通过途中每边恰好一次的路径)

D. 存在一条回路，通过每个顶点恰好一次

E. 本身为闭迹的图

【答案】D

【分析】通过每个顶点一次的是哈密尔顿图，欧拉图是每条边一次，一笔画问题，闭迹回到起点封口。

8.【NOIP2004 普及组】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(　　)。

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

E. C_0，C_1，C_2，C_3，C_6，C_7，C_5，C_4

【答案】D

【分析】拓扑排序：在学习 C_5 之前要先学习 C_3 和 C_7，D 答案的 C_5 排在了 C_3 前面。

第三章　问题求解

第 1 节　组合数学初步

1. 排列与组合

历史

1772 年，旺德蒙德以[n]p 表示由 n 个不同的元素中每次取 p 个的排列数。而欧拉则于 1771 年以及于 1778 年以表示由 n 个不同元素中每次取出 p 个元素的组合数。至 1872 年，埃汀肖森引入了以表相同之意，这组合符号(Signs of Combinations)一直沿用至今。

1830 年，皮科克引入符号 nCr 以表示由 n 个元素中每次取出 r 个元素的组合数；1869 年或稍早些，剑桥的古德文以符号 nPr 表示由 n 个元素中每次取 r 个元素的排列数，这用法延用至今。按此法，nPn 便相当于现在的 n!。

1880 年，鲍茨以 nCr 及 nPr 分别表示由 n 个元素取出 r 个的组合数与排列数；至 1899 年，克里斯托尔以 nPr 及 nCr 分别表示由 n 个不同元素中每次取出 r 个不重复元素的排列数与组合数，并以 nHr 表示相同意义下之可重复的排列数，这三种符号也通用至今。

两个基本原理是排列和组合的基础

(1)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 m1 种不同的方法，在第二类办法中有 m2 种不同的方法，……，在第 n 类办法中有 mn 种不同的方法，那么完成这件事共有 N＝m1＋m2＋m3＋…＋mn 种不同方法。

(2)乘法原理：做一件事，完成它需要分成 n 个步骤，做第一步有 m1 种不同的方法，做第二步有 m2 种不同的方法，……，做第 n 步有 mn 种不同的方法，那么完成这件事共有 N＝m1 * m2 * m3 * … * mn 种不同的方法。

这里要注意区分两个原理，要做一件事，完成它若是有 n 类办法，是分类问题，第一类中的方法都是独立的，因此用加法原理；做一件事，需要分 n 个步骤，步与步之间是连续的，只有将分成的若干个互相联系的步骤，依次相继完成，这件事才算完成，因此用乘法原理。

这样完成一件事的分“类”和“步”是有本质区别的，因此也将两个原理区分开来。

排列和排列数

(1)排列：从 n 个不同元素中，任取 m(m≤n)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。

从排列的意义可知，如果两个排列相同，不仅这两个排列的元素必须完全相同，而且排列的顺序必须完全相同，这就告诉了我们如何判断两个排列是否相同的方法。

(2)排列数公式：从 n 个不同元素中取出 m(m≤n)个元素的所有排列，

排列的记号和公式为 $P_n^m=\dfrac{n!}{(n-m)!}$

当 m＝n 时，为全排列 nPn＝n * (n－1) * (n－2)…3 * 2 * 1＝n!

组合和组合数

(1)组合：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。

从组合的定义知，如果两个组合中的元素完全相同，不管元素的顺序如何，都是相同的组合；只有当两个组合中的元素不完全相同时，才是不同的组合。

(2) 组合数：从 n 个不同元素中取出 m(m≤n)个元素的所有组合的个数。

(3) 这里要注意排列和组合的区别和联系，从 n 个不同元素中，任取 m(m≤n)个元素，“按照一定的顺序排成一列”与“不管怎样的顺序并成一组”这是有本质区别的。

组合的记号为 C(n,k)或$\binom{n}{k}$：

$$\binom{n}{k}=\frac{P(n,k)}{P(k,k)}$$

根据 P(n,k)的定义：

$$P(n,k)=\frac{n!}{(n-k)!}$$

$$\binom{n}{k}=\frac{n!}{k!\cdot(n-k)!}$$

常见计数方法

(1)特殊优先

特殊元素，优先处理；特殊位置，优先考虑。

例. 六人站成一排，求：

①甲不在排头，乙不在排尾的排列数；

②甲不在排头，乙不在排尾，且甲乙不相邻的排法数。

【分析一】

先考虑排头，排尾，但这两个要求相互有影响，因而考虑分类。

第一类：乙在排头，有 P(5,5)种站法；

第二类：乙不在排头，当然他也不能在排尾，有 4 * 4 * P(4,4)种站法；

共 P(5,5)+4 * 4 * P(4,4)种站法。

【分析二】

第一类：甲在排尾，乙在排头，有 P(4,4)种方法。

第二类：甲在排尾，乙不在排头，有 3 * P(4,4)种方法。

第三类：乙在排头，甲不在排头，有 4 * P(4,4)种方法。

第四类：甲不在排尾，乙不在排头，有 P(3,3) * P(4,4)种方法。

共 P(4,4)+3 * P(4,4)+4 * P(4,4)+P(3,3) * P(4,4)=312 种。

(2)捆绑与插空

例. 8 人排成一队。

①甲乙必须相邻；

②甲乙不相邻；

③甲乙必须相邻且与丙不相邻；

④甲乙必须相邻，丙丁必须相邻；

⑤甲乙不相邻，丙丁不相邻。

【分析】①甲乙必须相邻，就是把甲乙捆绑(甲乙可交换)和 7 人排列 P(7,7) * 2。

②甲乙不相邻，P(8,8)−P(7,7) * 2。

③甲乙必须相邻且与丙不相邻，先求甲乙必须相邻且与丙相邻 P(6,6) * 2 * 2；甲乙必须相邻且与丙不相邻 P(7,7) * 2−P(6,6) * 2 * 2。

④甲乙必须相邻，丙丁必须相邻 P(6,6) * 2 * 2。

⑤甲乙不相邻，丙丁不相邻，P(8,8)−P(7,7) * 2 * 2+P(6,6) * 2 * 2。

例. 某人射击 8 枪，命中 4 枪，恰好有三枪连续命中，有多少种不同的情况？

【分析】因为连续命中的三枪与单独命中的一枪不能相邻，因而这是一个插空问题。另外没有命中的之间没有区别，不必计数。即在四发空枪之间形成的 5 个空中选出 2 个的排列，即 P(5,2)。

例. 马路上有编号为 1,2,3,……,10 十个路灯，为节约用电又看清路面，可以把其中的

三只灯关掉，但不能同时关掉相邻的两只或三只，在两端的灯也不能关掉的情况下，求满足条件的关灯方法共有多少种？

【分析】即关掉的灯不能相邻，也不能在两端。又因为灯与灯之间没有区别，因而问题为在 7 盏亮着的灯形成的不包含两端的 6 个空中选出 3 个空放置熄灭的灯。

所以共 C(6,3)＝20 种方法。

不定方程的正整数解个数

例：求方程 $x_1+x_2+x_3+\cdots+x_N=M$ 的正整数解的个数。

【分析】想象现在有 M 个球一字排列，要把它们分成 N 组，我们用 N－1 块木板将它们分割成 N 段，N－1 块木板放在中间的 M－1 个空格中，由于每个空格中只能放一块木板，所以这是一个组合问题，答案是：

$$\binom{M-1}{N-1}$$

又因为每一种放置木板的方法对应了一组不定方程的解，所以原问题的答案也是：

$$\binom{M-1}{N-1}$$

例：求 $x_1+x_2+x_3+\cdots+x_N=M$ 方程的非负整数解的个数。

【分析】设 $y_i=x_i+1$，则 $y_1+y_2+y_3+\cdots y_N=M+N$，问题转化为上一种情况，于是答案为：

$$\binom{M+N-1}{N-1}$$

2. 几个特殊的数列

Catalan 数

在一个有 n＋2 条边的凸多边形中，我们可以画出 n－1 条不相交的对角线将多边形分为 n 个三角形，设所有满足条件的方案数是 h_n，定义 $h_0=1$，求 h_2、h_4。

【分析】

我们试图找出 h_n 的递推关系和通项公式。

考虑凸 n＋2 边形的任意一条边，我们设为基边，考虑这条边所在的三角形，这个三角形会把凸 n＋2 边形剖成两块，一块有 k＋2 条边(可以剖分成 k 个三角形)，那么另一块有 n＋1－k 条边(可以剖分成 n－1－k 个三角形)，根据乘法原理，这样的剖分方案数是 $h_k * h_{n-1-k}$，又因为 k 显然可以从 0 一直变化到 n－1(考虑边界情况，并注意到对角线不相交，所以这里不会出现重复计数)，故：

$$h_n=\sum_{k=0}^{n-1} h_k * h_{n-1-k} \tag{1}$$

数列$\{h_n\}$就是著名的 Catalan 数列，在组合数学的许多问题上都有很重要的应用。Catalan 数列的递推关系正如上式，再加上一点生成函数的知识就可以推导出它的通项公式，但是生成函数已远远超出本书所介绍的知识范畴，有兴趣的读者可以参考《组合数学》(《Introductory Combinatorics》，Richard A. Brualdi 著)中第 7 章和第 8 章的相关内容。笔者在此仅仅给出 Catalan 数列的通项公式，并不要求读者明白它是怎么来的：

$$h_n=\frac{1}{n+1}\binom{2n}{n}$$

我们会在以后看到这个数列非常有用，希望读者认真记住这个公式。

从另一种模型推导 Catalan 数

定理：n 个＋1 和 n 个－1 构成的 2n 项 $a_1,a_2,a_3,\cdots\cdots,a_{2n}$，并且其部分和满足：

$a_1+a_2+a_2+\cdots+a_k\geqslant 0(k=1,2,3,\cdots\cdots,2n)$

对所有 k 都成立的数列个数等于第 n 个 Catalan 数：

$$h_n=\frac{1}{n+1}\binom{2n}{n} \tag{2}$$

证明：如果 n 个+1 和 n 个−1 的序列满足部分和都不小于 0，则称其为可接受的，否则为不可接受的，令 A_n 为 n 个+1 和 n 个−1 形成的可接受序列的个数，令 U_n 为不可接受的序列个数。我们尝试计算出 A_n+U_n 的值和 U_n 的值以得到 A_n 的值。

任意一个有 n 个+1 和−1 的构成的 2n 项必然属于且仅属于可接受序列和不可接受序列之一，于是有：

$$A_n+U_n=\binom{2n}{n}$$

下面我们试图计算出 U_n 的值。

对于任意一个不可接受的数列中一定存在一个最小的 k，使得 $a_1+a_2+a_3+\cdots+a_k<0$，于是我们可以得到一些显然的结论：k 是一个奇数，前 k−1 个数中+1 和−1 恰好各占一半且 $a_k=-1$。我们将这 k 个数取相反数，后面的数不变，由此得到了一个有 n+1 个+1 和 n−1 个−1 的数列。注意这个操作是可逆的：对于一个有 n+1 个+1 和 n−1 个−1 的数列，我们只需要找到最小的 k，满足 $a_1+a_2+a_3+\cdots+a_k>0$，然后将前 k 个数取反即可得到原数列。所以不可接受的序列和有 n+1 个+1 和 n−1 个−1 的数列是一一对应的，故：

$$U_n=\binom{2n}{n+1}=\frac{n}{n+1}\binom{2n}{n}$$

再结合：

$$A_n+U_n=\binom{2n}{n}$$

可知定理成立。

Catalan 数列的应用

因为 Catalan 数列满足递推关系(1)和定理(2)，所以它在许多看似没有联系的模型中都有重要的应用。

(1)括号序列：给出一个有 n 个运算符 n+1 个运算数的算式，要求在算式中任意添加括号，那么本质不同的运算顺序有多少种？

【分析】考虑最后一个运算的符号，它的左边有 k 个运算符，右边则有 n−1−k 个运算符，由乘法原理可知这个问题满足递推关系(1)，于是答案是 Catalan 数列。

(2)有 2n 个人排队进入剧场，入场费 50 元，每人都带着一张 50 元或 100 元的钞票，剧院售票处没有任何零钱，有多少种情况满足无论什么时候售票处都能找得开零钱。

(3)一个 n * n 的网格图，从左下走到右上，每次只能向上或向右走，不能走到左下到右上的对角线的上方，一共有多少种走法？

【分析】2、3 两个问题本质是由 n 个+1 和 n 个−1 组成的序列中任意部分和都非负的序列总数(模型的转换希望读者认真思考)，于是这两个问题的答案也是 Catalan 数列。

(4)有 n 个结点的形态不同的二叉树一共有多少棵？

分析请读者自行完成。

这个问题的答案也满足递推关系(1)。

(5)出栈序列统计，有 n 个数和一个栈，本质不同的合法序列一共有多少种？

提示：将一次入栈操作视为+1，一次出栈操作视为−1，剩余的分析请读者自行完成。

第 2 类 Stirling 数

请读者注意，第 1 类 Stirling 数和第 2 类 Stirling 数的出现都是基于对差分数列的研究，但是笔者无意给读者介绍这些在信息学初赛中毫无用处的数列，所以差分数列和第一类 Stirling 数的相关知识完全略去，请有兴趣的读者参考《组合数学》(《Introductory Combinatorics》，Richard A. Brualdi 著)中第 8 章第 2 节的相关内容。下面仅仅从第 2 类 Stirling 数的一个小应用引入第 2 类 Stirling 数。

定理：第二类 Stirling 数 S(n,k)是将 n 个元素的集合划分成 k 个不可辨认的非空盒子的划分的个数。注意，这里的不可辨认是指不能把一个盒子与另一个盒子分辨开，它们看起来都一样。

下面给出第二类 Stirling 数的递推关系和证明：

S(n,k)=k*S(n−1,k)+S(n−1,k−1)；S(n,1)=1(n≥1)，S(n,n)=1。

上面的递推式可以用组合证明：一方面，如果将元素 n 单独拿出来划分成 1 个集合，那么方法数是 S(n−1,k−1)；另一方面，如果元素 n 所在的集合不止一个元素，那么可以先将剩下的 n−1 个元素划分好了以后再选一个集合把 n 放进去，方法数是 k*S(n−1,k)。

例：盒子与球

有 3 个一模一样的盒子和 6 个不同颜色的球，将这些球放到 3 个盒子里并且保证每个盒子里至少有 1 个球，求放置的方案总数。

答案：第二类 Stirling 数 S(6,3)。

3. 容斥原理与错位排列问题

容斥原理

$$\left|\bigcup_{i=1}^{n}A_i\right|=\sum_{i=1}^{n}|A_i|-\sum_{i,j|i\neq j}|A_i\cap A_j|+\sum_{i,j,k|i\neq j\neq k}|A_j\cap A_j\cap A_k|-\cdots\cdots+|A_1\cap\cdots\cap A_n|$$

在计数时，必须注意无一重复，无一遗漏。为了使重叠部分不被重复计算，人们研究出一种新的计数方法，这种方法的基本思想是：先不考虑重叠的情况，把包含于某内容中的所有对象的数目先计算出来，然后再把计数时重复计算的数目排斥出去，使得计算的结果既无遗漏又无重复，这种计数的方法称为容斥原理。

例题：有多少能被 3 或 5 或 7 整除的小于 1000 的正整数？

【分析】设 A_1、A_2、A_3 分别表示被 3 整除、被 5 整除、被 7 整除的数构成的集合，根据公式，

$$\left|\bigcup_{i=1}^{3}A_i\right|=\sum_{i=1}^{3}|A_i|-\sum_{i\neq j}|A_i\cap A_j|+\sum|A_1\cap A_2\cap A_3|$$

于是答案是：

$$\left[\frac{1000}{3}\right]+\left[\frac{1000}{5}\right]+\left[\frac{1000}{7}\right]-\left[\frac{1000}{15}\right]-\left[\frac{1000}{21}\right]-\left[\frac{1000}{35}\right]+\left[\frac{1000}{105}\right]=543$$

错位排列问题

对于一个 1……n 的任意排列，有一种情况是第 i 个位置上的数不是 i 对任意 i 都成立，例如 n=3 时，有且仅有排列 231 和 312 是满足要求的，这样的排列被称为错位排列。现在的问题是，能否对于任意 n，给出错位排列的个数？

【分析】利用容斥原理。

设 A_i 表示 i 这个数在原来位置上的排列的集合，则错位排列的个数：

$$D_n=n!-\left|\bigcup_{i=1}^{n}A_i\right|$$

现在我们来算 $\left|\bigcup_{i=1}^{n}A_i\right|$，

先算 $\sum|A_i|$，因为有 1 个数在固定的位置上，有 C(n,1)种情况，对于每一种情况，剩下的 n−1 个数任意排列，于是有 C(n,1)*(n−1)! =n! 种情况。

再算 $\sum|A_i\cap A_j|$，因为有 2 个数在固定的位置上，有 C(n,2)种情况，对于每一种情况，剩下 n−2 个数任意排列，于是有 C(n,2)*(n−2)! =n! /2 种情况。

……

……

一般地，因为有 k 个数在固定的位置上，另外(n−k)个数任意排列，所以：

$$\sum|A_{i_1}\cap A_{i_2}\cap A_{i_3}\cap\cdots\cap A_{i_k}|=\binom{n}{k}*(n-k)!=\frac{n!}{k!*(n-k)!}*(n-k)!=\frac{n!}{k!}$$

综上，由容斥原理，我们可以得到错位排列的公式：

$$D_n = n!\left(1-\frac{1}{1!}+\frac{1}{2!}-\frac{1}{3!}+\cdots+(-1)^n*\frac{1}{n!}\right)$$

错位排列的递推关系：

考虑 1 号位置的那个数，不妨设为 k，k>1；

以下有两种情况：

(1)k 号位置的那个数是 1，于是剩下 n−2 个数也是一个错位排列，有 D_{n-2} 种情况；

(2)k 号位置的那个数不是 1，那么不妨设 1 原来所在的位置应该在 k，这样就是除 k 以外的 n−1 个数的一个错位排列，方案数为 D_{n-1}。

最后，因为 k 的取值可以是 2 到 n 中的任意一个，于是我们得到了错位排列的递推关系：

$$D_n=(n-1)*(D_{n-1}+D_{n-2})$$

初始值 $D_1=0$，$D_2=1$。

事实上，在 n 较小的时候，计算 D_n 更多的是通过递推关系而不是通项公式，但是通项公式和递推关系以及它们的推导过程希望读者能牢牢记住。

两个计数技巧——算两次和补集转化

算两次：

在对一个集合进行计数的时候，我们往往可以通过不止一条途径得到答案，那么用两种不同的方法计算得到答案应该是相同的，这样的技巧称为算两次，事实上算两次的应用并不仅仅局限于验证答案的正确性，请看下面的例题：

一棵二叉树中有 7 个结点有 2 个儿子，求二叉树中有多少叶子结点(没有儿子的结点)？

【分析】设 S 表示二叉树的结点总数，表示有 i 个儿子的结点的个数。

一方面，S 可以这样计算：所有结点的个数总和，即 $S=S_1+S_2+S_0$；

另一方面，S 可以这样计算，根结点＋所有有 1 个儿子的结点的儿子个数＋所有有 2 个儿子的结点的儿子个数，即 $S=1+S_1+2*S_2$；

两次计算结果相互对照可以得出 $S_0=S_2+1$，于是例题的答案是 8。

这样的技巧我们可以推广到 k 叉树中得到更一般的结论：

$$S_0=S_2+S_2*2+S_4*3+\cdots+S_k*(k-1)+1$$

补集转化：

在计算一个集合 A 中元素的个数的时候，往往会发现直接求解很困难，但是如果我们换一种思路，利用全集 C 和 A 在 C 中的补集中元素的个数迂回求解一般可以达到更好的效果，这样求解的技巧称为补集转化，多用于集合本身元素个数不好分析但是全集和补集都很容易求解的情况，由于这样的情况大量存在，所以补集转化思想是一个很重要的计数技巧。

计算中用到补集转化的例子比比皆是，我们甚至不需要刻意去找这样的例题，稍微留心的读者大概就已经注意到了，我们在上文求解错位排列的通项公式的时候就用了补集转化的思想。另一个稍远的例子在前面第 2 部分“从另一种模型推导 Catalan 数”部分，可接受序 A_n 列的个数不好求，但是 A_n+U_n 的值和 U_n 的值都可以很方便地求到，于是我们利用补集转化的技巧得到了 A_n 的值。

4. 鸽巢原理

鸽巢原理

鸽巢原理，又名狄利克雷抽屉原理、鸽笼原理。

其中一种简单的表述法为：若有 n 个笼子和 n+1 只鸽子，所有的鸽子都被关在鸽笼里，那么至少有一个笼子有至少 2 只鸽子。

另一种为：若有 n 个笼子和 kn+1 只鸽子，所有的鸽子都被关在鸽笼里，那么至少有一个笼子有至少 k+1 只鸽子。

鸽巢原理的加强形式

令 $q_1,q_2\cdots\cdots,q_n$ 为正整数，如果将：

$$q_1+q_2+q_3+\cdots+q_n-n+1$$

个鸽子放入 n 个笼子里，那么，或者第 1 个笼子至少有 q_1 个鸽子，或者第 2 个笼子至少有 q_2 个鸽子，……，或者第 n 个笼子至少有 q_n 个鸽子。

若有 n 个笼子和 kn−1 只鸽子，所有的鸽子都被关在鸽笼里，那么至少有一个笼子至多只有 k−1 只鸽子。

鸽巢原理的简单应用

一位国际象棋大师有 11 周的时间备战一场锦标赛，他决定每天至少下一盘棋，但为了不使自己过于疲劳他还决定每周不能下棋超过 12 盘，证明存在连续的若干天，期间这位大师恰好下了 21 盘棋。

证明：a_1 令表示第 1 天所下的盘数，a_2 表示前 2 天所下的盘数，a_3 表示前 3 天所下的盘数……由于每天都要下至少一盘棋，故数列 $a_1,a_2,a_3,\cdots\cdots,a_{77}$ 是一个严格递增的序列。此外，$a_1\geqslant 1$，而且由于每周下棋最多是 12 盘，即 $a_{77}\leqslant 12*11=132$。因此，我们有：

$$1\leqslant a_1<a_2<a_3<\cdots<a_{77}\leqslant 132$$

此外，序列 $a_1+21,a_2+21,a_3+21,\cdots\cdots,a_{77}+21$ 也是一个严格递增的序列：

$$22\leqslant a_1+21\leqslant a_2+21<a_3+21<\cdots<a_{77}+21\leqslant 153$$

于是，这 154 个数：

$$a_1,a_2,a_3,\cdots\cdots,a_{77},a_1+21,a_2+21,a_3+21,\cdots\cdots,a_{77}+21$$

中的每一个都是 1 到 153 之间的一个整数。根据鸽巢原理可知，它们中间有两个是相等的。既然 $a_1,a_2,a_3,\cdots\cdots,a_{77}$ 中没有相等的数，并且 $a_1+21,a_2+21,a_3+21,\cdots\cdots,a_{77}+21$ 中也没有相等的数，因此必然存在一个 i 和一个 j 使得 $a_i=a_j+21$。从而，这位国际象棋大师在第 j+1，j+2，j+3，……，i 天这段时间内一共下了 21 盘棋。

例题【NOIP2010 提高组】记 T 为一队列，初始时为空，现有 n 个总和不超过 32 的正整数依次入列。如果无论这些数具体为何值，都能找到一种出队的方式，使得存在某个时刻队列 T 中的数之和恰好为 9，那么 n 的最小值是_____。

【分析】组合数学题目的一种常见思路是先猜测答案再证明。我们试图通过鸽巢原理猜出答案。

设 S_i 表示前 i 个数的和，并规定 $S_0=0$，事实上原题要求出最小的 n，使得存在 $0\leqslant i<j\leqslant n$ 满足 $S_j=S_i+9$。于是我们可以把 S 数组看做鸽子，用不能同时取的一组（即差为 9）的集合构造笼子，构造方法如下：一共有 18 个集合按如下方式选取，{0,9}、{1,10}、{2,11}、{3,12}、{4,13}、{5,14}、{6,15}、{7,16}、{8,17}、{18,27}、{19,28}、{20,29}、{21,30}、{22,31}、{23,32}、{24}、{25}、{26}，根据题意，我们一旦在某个集合中取了两个元素，那么一定存在某个时刻队列 T 中数之和恰好为 9，于是由鸽巢原理我们可以知道 n=18（S 数组有 19 个元素）一定满足条件。

下面我们来证明 n=17 不可行，事实上只要构造出一组不合法的序列即可。可以选择如下不合法的序列：1 1 1 1 1 1 1 1 10 1 1 1 1 1 1 1 1。

综上：原问题所求的 n 的最小值为 18。

5. Nim 取石子游戏

最简单的取石子游戏

考虑这样一个简单的问题：

有一堆石子，共 n 个，两个人轮流取石子，每次最少取 1 个，最多取 m 个，取完最后一个石子的人获胜，问先手有没有必胜方案。

这是一个非常经典的题目，相信大部分读者都做过这样的题目。这里直接给出答案：当 n mod (m+1)=0 时，先手必败；否则先手必胜，必胜策略是每一次都取走当前石子数 mod (m+1) 的值。

SG 函数

现在我们来考察上面所说的策略为什么是对的。

首先说明必败态和必胜态的概念。必败态是说当前这个状态，先手无论如何操作都必败；必胜态则是说当前这个状态，先手无论如何操作都必胜。注意，在一般确定操作状态（例如不会有每次给一个随机数根据随机数操作的情况）的组合游戏中，只会存在这两种状态，如果先手和后手都足够聪明，不会出现介于必胜态和必败态之间的状态。

一个重要的性质：从必败态走到的每一个状态都是必胜态，从必胜态操作一步走到的所有状态中至少有一个是必败态。规定无法操作的状态为必败态。

在最简单的取石子游戏中，设 f[N]表示还剩 N 个石子的状态，那么显然 f[N]可以转移到 f[N−1]到 f[N−M]，这里面只要有一个必败态 f[N]就必胜否则必败。

那么我们是否可以不通过递推的方式判断必胜态还是必败态呢？

下面引入 SG 函数。

SG 函数的规定如下：最终状态（不可操作状态）的 SG 函数为 0，其余状态的 SG 函数规定为所有不等于它的某一个后继状态的 SG 函数的最小非负整数，例如一个状态有 2 个后继状态，它们的 SG 函数分别为 2 和 3，则当前状态的 SG 函数为 0；2 个后继状态的 SG 函数分别为 0 和 2，则当前状态的 SG 函数为 1。

SG 函数判断状态是否必胜的规则是如果当前状态的 SG 函数为 0，则当前状态必败，否则当前状态必胜。容易发现引入 SG 函数后我们之前递推判断必胜必败的规则也可用于 SG 函数的递推上。往往我们可以找到一些状态和 SG 函数之间的规律，从而避免递推。

例题【NOIP2010 提高组】

```
#include <iostream>
using namespace std;
const int NUM=5;
int r(int n){
    int i;
    if (n <=NUM)   return n;
    for (i=1; i <=NUM; i++)
    if (r(n-i) < 0) return i;
    return-1;
}
int main(){
    int n;      cin>>n;
    cout<<r(n)<<endl;
    return 0;
}
```

输入：16

输出：_____

【分析】这个程序实际上就是通过递推的方式来求最少取 1 个石子，最多取 5 个石子的博弈游戏的 SG 函数，但是我们可以分析出规律：状态 f[N]的 SG 函数＝n mod 6，于是应该输出 4。

Nim 取石子游戏

现在有若干堆石子，两人轮流操作，每人每次可以从任意一堆中取任意多个，但是不能不取，取完最后一个石子的人获胜（即无法再取的人就输了）。

这个问题就是最经典的 Nim 取石子问题。

我们在此不加证明地直接给出这个游戏状态的 SG 函数。

对于一个状态，设有 N 堆石子，每堆个数分别为 $a_1, a_2, a_3, \cdots, a_N$，那么这个状态的 SG 函数为 a_1 xora_2 xora_3 xor……xor a_N。根据 SG 函数的定义，a_1 xor a_2 xor a_3 xor……xor a_N＞0 的状态为必胜态，否则为必败态。

在这里笔者仅仅给出一个每个状态的SG函数不会等于其任意一个后继状态的SG函数的证明，更详细的关于SG函数为什么等于所有数xor值的证明读者可以参照下面一部分《Nim取石子游戏的必胜态策略分析》中的分析自行完成。

xor函数有一个相当优美的性质，它是一个可逆函数(所谓可逆函数，就是存在反函数的函数)，并且它的反函数就是本身。

举一个简单的例子，a xor b=c，那么一定有 b xor c=a 和 a xor c=b。

下面我们用反证法进行证明，假设存在一个状态的SG函数和它的一个后继状态的SG函数相等，设 $k=a_1$ xor a_2 xor a_3 xor…xor a_N，当前的操作是在第M堆中取若干石子，设 $s=k$ xor a_M，即除 a_M 以外其他所有数的xor值，设在 a_M 这一堆中取走了x个，因为当前状态和这个后继状态SG函数相等，则有 s xor a_M=s xor(a_M-x)=k，由于xor函数的反函数就是它本身，于是可知 $a_M=a_M-x$，从而 x=0，与游戏者不能不取的条件矛盾，故原命题成立。

Nim取石子游戏的必胜态策略分析

最后我们来分析一下Nim取石子游戏中的必胜态策略。

由上一部分的证明可以看出，如果有一个状态的SG函数为0，那么它的任何后继状态的SG函数一定不为0，也就是说它的任何后继状态都是必胜态，现在我们只需要找出一个方案，使得任何一个必胜态依此操作后都会形成一个必败态。

考虑任意一个必胜态的SG函数值s，将它写成二进制的表示，找出最高位的一个1。因为这一位是1，又由xor函数的运算规则可知，至少有一堆的石子数写成二进制之后这一位也是1，不妨设是第k堆，有x个石子。设 $x'=x$ xor s，由xor函数的性质可知，用 x' 代替x作为第k堆的石子数之后，所有石子堆的石子数xor的值为0，于是我们只需证明 $x'<x$。事实上因为s的最高位的1，不妨设为第c位，由xor函数的运算规则和x这一位也是1的约定可知：x和 x' 在比c更高的数位上数字完全相同，且x第c位为1，x' 的第c位为0，所以我们甚至无需继续比较就可以得出结论：$x'<x$。

综上，按照如上的构造方法，在第k堆中取走 $x-x'$ 个石子就可以使一个必胜状态变成必败状态，于是必胜策略的可行性得证。

例题【NOIP2006普及组】现有5堆石子，石子数依次为3,5,7,19,50，甲乙两人轮流从任一堆中任取(每次只能取自一堆，不能不取)，取最后一颗石子的一方获胜。甲先取，问甲有没有获胜策略(即无论乙怎样取，甲只要不失误，都能获胜)？如果有，甲第一步应该在哪一堆里取多少？请写出你的结果：_____。

【分析】首先我们来看第一个问题，判断当前这个状态是否是必胜态。

因为 3^5^7^19^50=32>0，所以当前的状态是必胜态，先手有必胜策略。

下面我们来回答第二个问题，根据上一部分的策略，先看32的最高位，最高位为6，而石子的个数中，50的第6位为1，于是我们应该让那一堆剩下 50 xor 32=18 个石子，即从50那一堆中取走32个。

6. 图论浅谈——二分图理论与Ramsey理论

二分图

二分图又称作二部图，是图论中的一种特殊模型。设G=(V,E)是一个无向图，如果顶点V可分割为两个互不相交的子集(A,B)，并且图中的每条边(i,j)所关联的两个顶点i和j分别属于这两个不同的顶点集(i in A,j in B)，则称图G为一个二分图。

二分图最著名也是最重要的一个应用就是最大匹配，但是在这里，最大匹配与初赛无关，我们来看另一个也很重要但是往往容易被忽视的结论：

一个图是二分图的充分必要条件是这个图中不含奇环。

充分性是说如果一个图不含奇环，那么这个图是二分图；必要性是说如果一个图是二分图那么这个图一定不含奇环。证明显然。

另一个显然的事实是，如果二分图的某一类点有x个，另一类点有y个，那么这个二分图中最多有xy条边。

例题【NOIP2010 提高组】无向图 G 有 7 个顶点，若不存在由奇数条边构成的简单回路，则它至多有_____条边。

【分析】由条件可知这个图是一个二分图，设某一类点有 x 个，那么另一类点有(7－x)个，原题即求 x(7－x)的最大值，其中 x 是正整数。可以用一元二次方程的性质或者一些经典的不等式，很快就能得到答案为 12。

我们将这个题目推广开去，如果一个图有 N 个点，不存在由奇数条边构成的简单回路，则它至多有$\left[\frac{N^2}{4}\right]$条边。

注意，如果我们用一个看似更松一点的约束条件：这个图中不存在三角形，那么这个问题的答案并不会增加，取等号的条件也不变，仍为一个一类点有$\left[\frac{N}{2}\right]$个、另一类点有$\left[\frac{N}{2}\right]$个的完全二分图。

7. 从两个形似的问题看数学模型的构建

在这一节里，我们将讨论最小任意交换排序和最小相邻交换排序。笔者将在最后给出一些分析数学问题和建立数学模型的建议，希望能对读者有一点启发，起到一点抛砖引玉的作用。

最少任意交换排序

【NOIP2008 普及组】将数组{8,23,4,16,77,－5,53,100}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(　　)次。

A. 4　　B. 5　　C. 6　　D. 7

【分析】初看此题，毫无头绪，我们不妨从最简单的贪心和模拟开始，看看能不能找到一点思路。

首先第一个数是 8，排好序后第一个数应该是－5，于是做一次交换；第二个数是 23，排好序后应该是 4，于是做一次交换。此时序列变成{－5,4,23,16,77,8,53,100}。然后再把 23 和 8 交换，77 和 23 交换，77 和 53 交换即可排好序，一共用了 5 次交换。我们用了贪心的思想做到了正确答案 5，但是很可惜目前我们还不知道这个答案为什么是对的。

为此需要引入置换的概念。

设 X 是一个有限集。不失一般性，取 X＝{1,2,3,……,N}。

X 的一个置换：

$$i_1, i_2, i_3, \cdots\cdots, i_N$$

是一个 1 到 N 的排列，这个置换可以视为 X 到其自身定义的一个一对一的函数：

$$f: X \to X$$

其中 $f(1)=i_1, f(2)=i_2, \cdots\cdots, f(N)=i_N$，用行列式可以表示为：

$$\begin{pmatrix} 1 & 2 & \cdots & N \\ i_1 & i_2 & \cdots & i_N \end{pmatrix}$$

事实上，如果我们把 k 和 i_k 之间连一条有向边，那么一个置换可以用一个有向图唯一地表示，特别地，这个有向图是由一些有向环组成的(包括只有一个点的自环)，注意到我们每次交换两个元素只有当这两个元素在一个环上时才有意义，并且一个有 k 个点的环至少需要交换 k－1 次才能完成排序，于是最少任意交换排序的答案就是 N－置换环的个数。

最少相邻交换排序

现在我们把上面一个问题的要求变一下，每次只能交换两个相邻元素，那么这时的最少交换次数又会是多少呢？

【分析】这个问题和上面那个问题看上去很相似，然而很可惜，它们只是形似，本质的数学模型没有一点相近的地方。

为解决这个问题，我们先考察相邻交换的性质。注意到相邻交换只会改变相邻两个数的相对大小而不会影响到别的数，因此我们考虑能不能从改变相对大小这个方面入手。在

数列中，有一个值和相对大小密切相关，那就是逆序对数。所谓逆序对，就是指一个数列中的两个数 i 和 j，i 在数列中的位置在 j 的前面并且 i>j。显而易见的是，一个排好序的数列的逆序对数为 0。现在我们看看相邻交换对逆序对数有什么影响，如果前一个数小于后一个数，交换后逆序对数加 1；否则逆序对数减 1。因为排好序的数组不含逆序对，所以我们希望逆序对数减得尽量快，这样可以尽可能减少交换次数。而另一方面，对于一个没有排好序的数组，我们一定可以找到两个相邻的数是逆序对，这样当逆序对数大于 0 的时候，我们总可以通过一次交换使逆序对数减 1。于是最小相邻交换排序的交换次数就等于原数列的逆序对数。

浅谈组合数学问题分析中数学模型的构建

(1)尝试，或者说探索，这是一个很重要的步骤，在这一阶段我们可以用模拟、贪心等各种手段来探索问题的解，很可能在这一步我们就得到的答案甚至不需要进行后续的分析。

(2)从小数据入手，分而治之。很多题目把数据规模改小之后和原题的处理方法是差不多的，这一点可以在第 1 步探索时试出来。

【NOIP2006 普及组】

现有 80 枚硬币，其中有一枚是假币，其重量稍轻，所有真币的重量都相同，如果使用不带砝码的天平称重，最少需要称几次，就可以找出假币？你还要指出第 1 次的称重方法。请写出你的结果：_____。

80 这个数据规模很大，我们不妨从小数据入手。如果有 2 枚硬币，只有 1 枚假币，那么很显然一次称重就可以；3 枚硬币也是 1 次就可以(天平的两边各放一枚)。3 枚以上的可以分成 3 堆，其中两堆数目相同，然后把两堆相同数目的放到天平上称重，这样一次我们就可以知道假币在哪一堆中，然后这个问题就被用一次称重转化到数据规模为原来 1/3 的问题了，于是原问题的答案是$[\log_2 N]=4$。

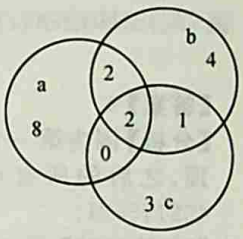
(3)分析题目中的有效信息，准确地说，是找到题目中的“特点”，例如计数时的一些限制，最少交换排序的交换方式，这些题目特有的性质是我们分析的要点。

(4)构建有效的数学模型。这是最重要的一步，但事实上在一般的考试题目是没有必要进行严格的分析的，第一步和第二步往往已经可以解决问题。模型的构建是和题目密切相关的，也就是根据题目的特点构建的。比如最小相邻排序问题中，我们需要寻找一个量，使得交换相邻两个数时，这个量的变化是确定的，这样逆序对数就给我提供了一个恰当的帮助。

最后，数学问题是千变万化层出不穷的，我们也无法给读者提供一个能解决所有问题的方法，希望这一部分《组合数学初步》能给读者一些思维方法和分析问题上的启迪，起到一点抛砖引玉的作用，但是要想真正做好数学题，仅靠这一点知识基础是不够的，希望读者能寻找到一些合适的题目多加练习。

第 2 节　入门篇

1.【NOIP1998】某班有 50 名学生，每位学生发一张调查卡，上写 a，b，c 三本书的书名，将读过的书打✓，结果统计数字如下：只读 a 者 8 人；只读 b 者 4 人；只读 c 者 3 人；全部读过的有 2 人；读过 a，b 两本书的有 4 人；读过 a，c 两本书的有 2 人；读过 b，c 两本书的有 3 人；(1)读过 a 的人数是 ；(2)一本书也没有读过的人数是。



【答案】(1)读过 a 的人数是12 人。(2)一本书也没读过的人数是30 人。

【分析】如图，可清晰的看到结果，比抽象推理更容易避免重复计算或丢失数据。读过 a 的人数是 8＋2＋2＋0＋12，一本书也没有读过的人数是 50－8－2－2－4－1－3＝30。

2.【NOIP1999】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：

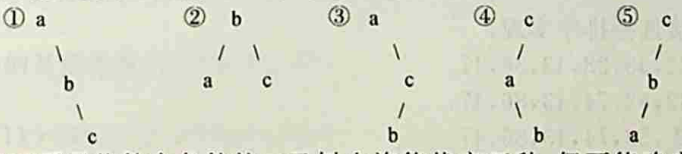
【答案】$n^2 - n + 1$

【分析】可以通过观察，n 的平方正好是右侧加法式子的中位数(如果加数个数是奇数，则就是最中间数；如果加数个数是偶数，则就是中间两数的平均值)，这个值正好和最小奇数差了 n－1，就得到了结果。

3.【NOIP2000】已知，按中序遍历二叉树的结果为：abc。

问：有多少种不同形态的二叉树可以得到这一遍历结果，并画出这些二叉树。

【答案】有 5 种不同形态的二叉树可以得到这一遍历结果。



【分析】本题直接画图寻找符合条件的二叉树也许能找齐五种，但不能确定是否找全。要想确定找全还是用分类寻找法比较稳妥，如果节点数增多，优点更为明显。

首先将要寻找的二叉树分三类：

第一类：以 a 为根的二叉树。

第二类：以 b 为根的二叉树。

第三类：以 c 为根的二叉树。

对于第一类以 a 为根的情况：因 abc 是中序遍历结果，可知 bc 为其右子树。

对于右子树 bc，再考虑以 b 为根的情况和以 c 为根的情况：

以 b 为根的子树，因中序遍历为 bc，c 必为其右儿子，得到①；

以 c 为根的子树，因中序遍历为 bc，b 必为其左儿子，得到②。

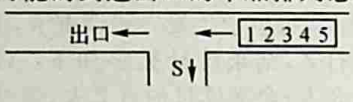
对于第二类以 b 为根的情况：因 abc 是中序遍历结果，可知 a 为其左儿子，c 为其右儿子，得到③。

对于第三类以 c 为根的情况：因 abc 是中序遍历结果，可知 ab 为其左子树。

以 a 为根的子树，因中序遍历为 ab，b 必为其右儿子，得到④；

以 b 为根的子树，因中序遍历为 ab，a 必为其左儿子，得到⑤。

4.【NOIP2002】如下图，有一个无穷大的的栈 S，在栈的右边排列着 1，2，3，4，5 共五个车厢。其中每个车厢可以向左行走，也可以进入栈 S 让后面的车厢通过。现已知第一个到达出口的是 3 号车厢，请写出所有可能的到达出口的车厢排列总数(不必给出每种排列)。



【答案】9

【分析】因为第一个到达出口的是 3 号车厢，所以可以肯定，1 号车厢在栈底，2 号车厢在栈顶，之后的所有可能序列有 9 种，分别为 2145，2154，2415，2451，2541，4215，4251，4521，5421。

5.【NOIP2002】将 N 个红球和 M 个黄球排成一行。例如：N＝2，M＝3 可得到以下 6 种排法：

红红黄黄黄 红黄红黄黄 红黄黄红黄 黄红红黄黄 黄红黄红黄 黄黄黄红红

问题：当 N＝4，M＝3 时有多少种不同排法？(不用列出每种排法)

【答案】35

【分析】这是一个可重复排列的问题，求解方法为(4＋3)！/4！/3！＝35。

6.【NOIP2004】75 名儿童到游乐场去玩。他们可以骑旋转木马，坐滑行铁道，乘宇宙飞船。已知其中 20 人这三种东西都玩过，55 人至少玩过其中的两种。若每样乘坐一次的费用是 5 元，游乐场总共收入 700，可知有名儿童没有玩过其中任何一种。

【答案】10

【分析】集合的交并补。

3 种东西都玩过的共用去：3＊5＊20＝300(元)，

只玩过两种东西的共用去：2＊5＊(55－20)＝350(元)，

那么：只玩过一种东西的人数为：(700－300－350)÷5＝10(人)，

所以：什么也没有玩的人数为：75－55－10＝10(人)。

7.【NOIP2005 普及组】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，最少需要交换次。

【答案】5

【分析】用直接选择排序实现：

25，74，32，53，28，43，86，47

25，**28**，32，53，74，43，86，47

25，28，**32**，53，74，43，86，47

25，28，32，**43**，74，53，86，47

25，28，32，43，**47**，53，86，74

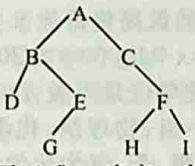
25，28，32，43，47，**53**，86，74

25，28，32，43，47，53，**74**，86

第3节　普及篇

1.【NOIP1998】给出一棵二叉树的中序遍历：DBGEACHFI 与后序遍历：DGEBHIFCA，画出此二叉树。

【答案】此二叉树为：



【分析】根据中序和后序遍历的结果可以唯一确定一棵树。

2.【NOIP2001】在 a,b,c,d,e,f 六件物品中，按下面的条件能选出的物品是：

(1)a,b 两样至少有一样；

(2)a,d 不能同时取；

(3)a,e,f 中必须有 2 样；

(4)b,c 要么都选，要么都不选；

(5)c,d 两样中选一样；

(6)若 d 不选，则 e 也不选。

【答案】a，b，c，f

【分析】这是一个逻辑推理问题，硬性手工推导很容易出错，建议采用穷举法，这里一共有六件物品，每件物品都有“取”和“不取”两种情况，所以所有的可能性是 $2^6=64$ 种，每种试验一下并不复杂，答案是唯一的。

3.【NOIP2002】在书架上放有编号为 1,2,…,n 的 n 本书。现将 n 本书全部取下然后再放回去，当放回去时要求每本书都不能放在原来的位置上。例如：n=3 时：

原来位置为：1　2　3；

放回去时只能为：3　1　2　或　2　3　1　这两种。

问题：求当 n=5 时满足以上条件的放法共有多少种？（不用列出每种放法）

【答案】44

【分析】此题要求的是错排数，详见上文介绍。

方法一：

F(n)=(n−1)*(F(n−1)+F(n−2))

F(1)=0

F(2)=1

F(3)=2*(0+1)=2

F(4)=3*(2+1)=9

F(5)=4*(F(4)+F(3))=44

方法二：

总数 S=n!(1!−1/1!+1/2!−1/3!……(−1)^n*1/n!)

方法三：

f(0)=1

f(x)=x*f(x−1)+1(x mod 2=0)

f(x)=x*f(x−1)−1(x mod 2=1)

所以 f(5)=5*f(4)−1=5*(4*f(3)+1)−1=5*(4*2+1)−1=44

方法四：

5!−5*4!+10*3!−10*2!+5*1!−1=44

4.【NOIP2003】现在市场上有一款汽车 A 很热销，售价是 2 万美元。汽车 A 每加仑汽油可以行驶 20 英里。普通汽车每年大约行驶 12000 英里。油价是每加仑 1 美元。不久我公司就要推出新款节油汽车 B，汽车 B 每加仑汽油可以行驶 30 英里。现在我们要为 B 制定价格（它的价格略高于 A）：我们预计如果用户能够在两年内通过节省油钱把 B 高出 A 的价钱弥补回来，则他们就会购买 B，否则就不会购买 B。那么 B 的最高价格应为_____万美元。

【答案】2.04

【分析】可以列不等式求解。设 B 的最高售价为 x 美元，加上两年消耗的油钱就是两年的总支出，则 20000+12000*2/20≥x+12000*2/30，解不等式得 x≤20400 元，即最高售价不得高于 2.04 万美元，注意填空的单位是万美元。

5.【NOIP2005 普及组】有 3 个课外小组：物理组，化学组和生物组。今有张、王、李、赵、陈 5 名同学，已知张、王为物理组成员，张、李、赵为化学组成员，李、赵、陈为生物组成员。如果要在 3 个小组中分别选出 3 位组长，一位同学最多只能担任一个小组的组长，共有种选择方案。

【答案】11

【分析】考虑物理和生物组，这两组成员没有重复。所以，这两组组长的选法有 6 种。其中，组长为王陈两人的情况有 1 种，这时化学组组长人选有 3 种；组长没有王陈的情况有 2 种，这时化学组组长人选有 1 种；组长有且仅有王陈其中 1 人的情况有 3 种，这时化学组组长人选有 2 种，所以，总共有 1*3+2*1+3*2=11 种方案。情况不是很多，穷举就可以出来。

6.【NOIP2005 提高组】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（回答应为一个由 0 和/或 1 组成的字符串）。

【答案】11011

【分析】经典的取石子问题，有结论当 n%3=0 时先手必败，否则必胜。当 n=100，200，400，500 时，A 都有必胜的把握，但是，当 n=300 时，A 无必胜的把握。

7.【NOIP2006 普及组】（寻找假币）现有 80 枚硬币，其中有一枚是假币，其重量稍轻，所有真币的重量都相同，如果使 用不带砝码的天平称重，最少需要称几次，就可以找出假币？你还要指出第 1 次的称重方法。请写出你的 结果：_____。

【答案】4 次，第一步：分成 3 组：27，27，26，将前 2 组放到天平上。

【分析】三分法：第一次分成（27，27，26），第一次称 27 和 27，如果一样重说明假币在 26 中，如果有一边轻，则假币在轻的一边，然后第二次称有假币的，方法和第一次一样，依次类推。

第 4 节　提高篇

1.【NOIP2001】平面上有三条平行直线，每条直线上分别有 7，5，6 个点，且不同直线上三个点都不在同一条直线上。问用这些点为顶点，能组成多少个不同四边形？

【答案】2250

【分析】本题用到排列组合的知识，也考查学生的空间想象能力。

将四边形进行分类，各类的计算用到组合公式

因一条直线上的点不可能组成四边形，所以可将四边形先分成使用两条直线上点的四边形和使用三条直线上的点的四边形。

设三条直线为 A、B、C，分别有 7，5，6 个点。

首先考虑第一种使用两条直线的四边形，可以再细分为三类：

(1) 使用直线 A、B 的四边形；

(2) 使用直线 A、C 的四边形；

(3) 使用直线 B、C 的四边形。

对于(1)，因不能在一条直线上取 1、3 个点(只能组成三角形)，故四边形的四个点中来自 A、B 各 2 个点。由题目可知，不同直线上三个点都不在同一直线上，问题转换成在 A 上任取 2 个点，在 B 上任取 2 个点有多少种取法。有排列组合公式得：

$(1)=C_7^2 * C_5^2=21 * 10+210$

同理得$(2)=C_7^2 * C_6^2=21 * 15=315$　　$(3)=C_6^2 * C_5^2=15 * 10=150$

(1)+(2)+(3)=675；

再来考虑第二种使用三条直线的四边形：

因使用三条直线，必有一条直线上有 2 个点。

可以再细分为三类：

(1) 直线 A 上有 2 个点的四边形；

(2) 直线 B 上有 2 个点的四边形；

(3) 直线 C 上有 2 个点的四边形。

对于(1)，问题转换成在 A 上取 2 个点，在 B 上取 1 个点，在 C 上取 1 个点，共有多少种取法。有排列组合公式得：

$(1)=C_7^2 * 6 * * 5=21 * 6 * 5=630$

同理得$(2)=C_5^2 * 7 * 6=10 * 42=420$　　$(3)=C_6^2 * 7 * 5=15 * 35=525$

(1)+(2)+(3)=1575；

最后结果为 1575+675=2250。

2.【NOIP2002】设有一棵 k 叉树，其中只有度为 0 和 k 两种结点，设 n_0，n_k，分别表示度为 0 和度为 k 的结点个数，试求出 n_0 和 n_k 之间的关系(n_0=数学表达式，数学表达式仅含 n_k、k 和数字)。

【答案】$n_0=(k-1)\ n_k+1$

【分析】

二叉树：

层数	n_0	n_k
1	1	0
2	2	1
3	3	2
3	4	3
$n_0=n_k+1$		

三叉树：

层数	n_0	n_k
1	1	0
2	3	1
3	5	2
3	7	3
3	9	4
$n_0=2*n_k+1$		

四叉树：

层数	n_0	n_k
1	1	0
2	4	1
3	7	2
3	10	3
3	13	4
$n_0=3*n_k+1$		

总结以上得：

$n_0=(k-1)*n_k+1$。

3.【NOIP2003】无向图 G 有 16 条边，有 3 个 4 度顶点、4 个 3 度顶点，其余顶点的度均小于 3，则 G 至少有_____个顶点。

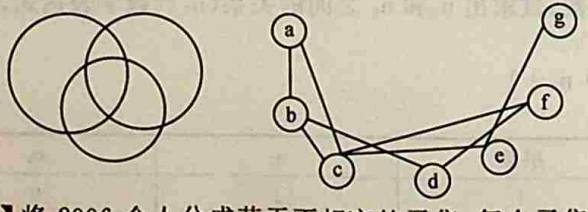
【答案】11

【分析】无向图有 16 条边，每条边被 2 个顶点计算度数，共计度数为 32。3 个 4 度顶点、4 个 3 度顶点，共 7 个顶点，度数为 24，尚余 32－24＝8 个度，其余顶点的度均小于 3，最大度数为 3，至少有 4 个顶点，加上前 7 个顶点，至少 11 个顶点。

4.【NOIP2004 提高组】已知 a，b，c，d，e，f，g 七个人中，a 会讲英语；b 会讲英语和汉语；c 会讲英语、意大利语和俄语；d 会讲汉语和日语；e 会讲意大利语和德语；f 会讲俄语、日语和法语；g 会讲德语和法语。能否将他们的座位安排在圆桌旁，使得每个人都能与他身边的人交谈？如果可以，请以“a b”开头写出你的安排方案。

【答案】abdfgec

【分析】可以画一张图：



5.【NOIP2006 提高组】将 2006 个人分成若干不相交的子集，每个子集至少有 3 个人，并且：

(1)在每个子集中，没有人认识该子集的所有人。

(2)同一子集的任何 3 个人中，至少有 2 个人互不认识。

(3)对同一子集中任何 2 个不相识的人，在该子集中恰好只有 1 个人认识这两个人。

则满足上述条件的子集最多能有_____个？

【答案】401

【分析】运用图论的思想，用一个结点代表一个人，如果两个人互相认识就用线连上，不认识就不连；

原题的要求就变成了这样：

①没有一个节点与其他所有点相连；

②每个子集中，任何三个结点中，至少两个不相连（没有三角形）；

③同一子集中的任意不直接相连的两点，彼此之间有只通过一个结点的路径。

然后拿一张纸，一支笔画，

三个节点，不行；

四个节点，不行；

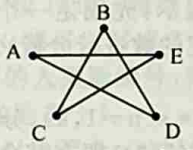
五个节点，连成五边形或者五角星就可以了；

六、七个节点好像无法凑成一个满足要求的集合；

八点子集是存在的（正八边形，每点再向正对的点连线）符合要求。

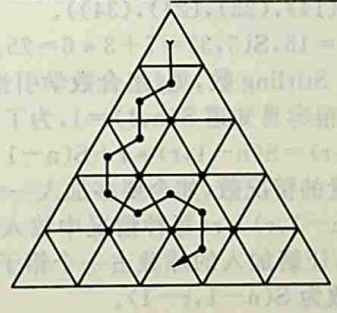
可以是：5＊x＋8＊y＝2006 使 x＋y 最大，y＝2，x＝398，答案是 x＋y＝400；

但是题目要的是最多，2006＝5＊400＋6＊1，最多 401 个子集。



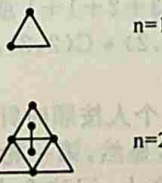
3个条件均满足

6.【NOIP2006 提高组】将边长为 n 的正三角形每边 n 等分，过每个分点分别做另外两边的平行线，得到若干个正三角形，我们称为小三角形。正三角形的一条通路是一条连续的折线，起点是最上面的一个小三角形，终点是最 下面一行位于中间的小三角形。在通路中，只允许由一个小三角形走到另一个与其有公共边的且位于同 一行或下一行的小三角形，并且每个小三角形不能经过两次或两次以上（图中是 n＝5 时一条通路的例子）。设 n＝10，则该正三角形的不同的通路的总数为_____。



【答案】9！　（或 362880）

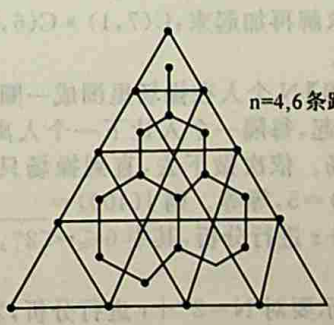
【分析】方法一：上一层三角形的底的数量就是通路的数量，1＊2＊3＊……＊(n－1)＊n。



n=1,0条路径

n=2,1条路径

n=3,2条路径



n=4,6条路径

方法一：

n=1　0条路径(1-1)!

n=2　1条路径(2-1)!

n=3　2条路径(3-1)!

n=4　6条路径(4-1)!

递推

n=10　(10-1)!条路径。

方法二：上层下到下层的第一个点，然后你就会发现，这些点通向中间那个终点的路只有一条。而上层到下层的第一个到达点的个数是(n-1)所以答案就是(n-1)!

解释：n=4第1层到第2层有1个路径，第2层到第3层有2个路径，第3层到第4层有3个路径，故总路径为1*2*3=6。推广到n层，(n-1)!个路径。

7.【NOIP2007普及组】(子集划分)将n个数{1,2,…,n}划分成r个子集。每个数都恰好属于一个子集，任何两个不同的子集没有共同的数，也没有空集。将不同划分方法的总数记为S(n,r)。例如，S(4,2)=7，这7种不同的划分方法依次为{(1),(234)}，{(2),(134)}，{(3),(124)}，{(4),(123)}，{(12),(34)}，{(13),(24)}，{(14),(23)}。当n=6,r=3时，S(6,3)=_____。(提示：先固定一个数，对于其余的5个数考虑S(5,3)与S(5,2)，再分这两种情况对原固定的数进行分析)。

【答案】90

【分析】方法一：s(n,k)=s(n-1,k-1)+k*s(n-1,k)，分别是an单独一堆，an与其他一起。将6单独放到1个子集的方法数是S(5,2)，即在其余的2个子集中放其余5个数。将6不单独放到1个子集的方法数是3*S(5,3)，即在3个子集里先放好其余5个数。然后将6依次放入3个子集中的某一个。即S(6,3)=S(5,2)+3*S(5,3)，类似可得：

S(5,2)=S(4,1)+2*S(4,2)，

S(5,3)=S(4,2)+3*S(4,3)，

已知S(4,2)=7，容易看出，S(4,1)=1，S(4,3)=6(这时，只有一个子集中含有两个数，共6种可能：(12)，(13)，(14)，(23)，(24)，(34))。

于是，S(5,2)=1+2*7=15，S(5,3)=7+3*6=25，S(6,3)=15+3*25=90。

S(n,k)通常称为第2类Stirling数，见《组合数学引论》，孙淑玲等著。

方法一：用递归思想做，很容易知道S(n,1)=1，为了利用这点，先分析S(n-1,r)与S(n-1,r-1)的情况，而S(n,r)=S(n-1,r)*r+S(n-1,r-1)其具体情况分析如下：

假设已知n-1个球放置的情况数，如今要多加入一个球，有两种新情况：一是放入原有球的箱子，即情况数为S(n-1,r)*r(每种情况中放入任意箱子都是新情况，所以乘以箱子数r)；二是取出所有球，让新加入的球独占一个箱子，则余下r-1个箱子可放球，有n-1个球可用来放，即情况数为S(n-1,r-1)。

综上，得出S(n,r)=S(n-1,r)*r+S(n-1,r-1)

【分析3】7个球放入4个箱子无非是2+2+2+1或者3+2+1+1或者4+1+1+1三种情况。所以分别求解再加起来：C(7,1)*C(6,2)*C(4,2)*C(2,2)/P(3,3)+C(7,3)*C(4,2)+C(7,4)。

8.【NOIP2007提高组】N个人在操场里围成一圈，将这N个人按顺时针方向从1到N编号，然后，从第一个人起，每隔一个人让下一个人离开操场，显然，第一轮过后，具有偶数编号的人都离开了操场。依次做下去，直到操场只剩下一个人，记这个人的编号为J(N)，例如，J(5)=3，J(10)=5，等等。则J(400)=____。

(提示：对$N=2^m+r$进行分析，其中$0\leqslant r<2^m$。)

【答案】289

【分析】题目的提示要对$N=2^m+r$进行分析，发现结果与m无关，只与r有关：

J	$N=2^m+r$
J(1)=1	2^0+0
J(2)=1	2^1+0
J(3)=3	2^1+1
J(4)=1	2^2+0
J(5)=3	2^2+1
J(6)=5	2^2+2
J(7)=7	2^2+3
J(8)=1	2^3+0
J(9)=3	2^3+1
J(10)=5	2^3+2

当有 2^m 个人围成一圈，按 1212…报数，凡报到 2 的人离开，最后剩下的 1 个人，一定是 1 号。那么对于任意的 N，都可以写成 $2^m+r(0<=r<2^m)$，这样第 r 个人出列后，剩下的人数就是 2^m，这时对剩下的人重新编号，排在第 1 号的就是最后剩下的那个人，将这个人还原到原队伍中的编号就是 2r+1。

对于本题，因为 $400=2^8+144=2^m+r$，r=144，所以结果就是 2r+1=144 * 2+1=289。

第四章　阅读程序

第 1 节　入门篇

1.【NOIP1998】
```
#include<iostream>
#include<cstdio>
#include<cmath>
using namespace std;
int s,maxx;
int a[11];
int main()
{
    for(int i=1; i<=10;++i) cin>>a[i];
    maxx=a[1]; s=a[1];
    for(int i=2; i<=10;++i)
    {
        if(s<0) s=0;
        s=s+a[i];
        if(s>maxx) maxx=s;
    }
    printf("%d", maxx);
}
```
输入:8　9　−1　24　6　5　11　15　−28　9

输出:max=

【答案】77

【分析】本题是一个简单的利用累加算法数列求和问题,通过循环输入 10 个数,再利用一个循环计算数列的和并输出计算结果。其中语句 if (s < 0) s=0;的作用是去掉 s < 0 的数列和的值,并令其为 0。语句 if (s > maxx) maxx=s; 的作用是保留最大的序列和。

2.【NOIP1998】
```
#include<iostream>
#include<cstdio>
using namespace std;
const int n=10;
int s;
int co(int i1)
{
    int j1,s1;
    s1=n;
    for(int j1=n-1; j1>=n-i1+1;--j1)
        s1=(s1*j1)/(n-j1+1);
    return s1;
}
```

```
int main()
{
    s=n+1;
    for(int i=2; i<=n;++i) s=s+co(i);
    printf("S=%d",s);
}
```

【答案】1024

【分析】首先观察函数 co 的作用。程序是累乘算法，找一个具体值作为参数代入可以发现计算公式。例如执行 co(6)，

```
co(6)=10 * (9 * 8 * 7 * 6 * 5)/2/3/4/5/6
     =(10 * 9 * 8 * 7 * 6 * 5)/(2 * 3 * 4 * 5)
     =10! / (4! * 6!)
     =10! /(6! * (10-6)!)
```

=C(n,m)(当 n=10，m=6 时)。

3.【NOIP2001】

```
#include<cstdio>
using namespace std;
inline int fun(int x){
    if (x==0 || x==1) return 3;
    else return x-fun(x-2);
}
int main(){
    printf("%d", fun(9));
    putchar('\n');
    return 0;
}
```

输出：

【答案】7

【分析】递归求解，可以转换为递推求解。

f(0)=f(1)=3；　　f(2)=2-f(0)=-1；　　f(3)=3-f(1)=0；

f(4)=4-f(2)=5；　　f(5)=5-f(3)=5；　　f(6)=6-f(4)=1；

f(7)=7-f(5)=2；　　f(8)=8-f(6)=7；　　f(9)=9-f(7)=7。

4.【NOIP2001】

```
#include<cstdio>
using namespace std;
int ack(int m, int n){
    if (m==0) return n+1;
    else if (n==0) return ack(m-1, 1);
    else return ack(m-1, ack(m, n-1));
}
int main(){
    printf("%d\n", ack(3, 4));
    putchar('\n');
    return 0;
}
```

输出：

【答案】125

【分析】经典的阿克曼函数的递归求解，可以手工完成，因为涉及到两个参数，建议画出一

张二维表。

5.【NOIP2001】
```
#include<cstdio>
using namespace std;
int i, j, h, m, n, k, b[11];
int main()
{
    scanf("%d", &n);
    for (i=1; i<=10; i++){
        m=n; j=11;
        while (m>0){
            j=j-1; b[j]=m % 10; m=m / 10;}
        for (h=1; h<=10; h++) n=n+b[h];
    }
    printf("%d", n);
}
```
输入 1234　　输出：

【答案】1348

【分析】本题程序的运算为：将 n 的各位数字分离，存入数组 b，将各位数字的和与 n 相加，重复对 n 作同样的运算 10 次。n 的变化依次为：

1234→1244→1255→1268→1285→1301→1306→1316→1327→1340→1348

6.【NOIP2003】
```
#include<iostream>
#include<cstdio>
#include<cstring>
using namespace std;
int a, x, y, ok1, ok2;
int main()
{
 a=100;
 x=20;
 y=20;
 ok1=5;
 ok2=0;
 if((x>y) || ((y!=20) && (ok1==0)) && (ok2!=0))
   a=1;
 else if ((ok1!=0) && (ok2==0))
   a=-1;
 else
   a=0;
 cout << a << endl;
 return 0;
}
```
输出：_____

【答案】-1

【分析】简单的顺序和分支结构模拟，注意：可以将变量的值逐一记录下来。

7.【NOIP2003】
```
#include <iostream>
#include <cstdio>
```

```
#include <cstring>
#include <string>
using namespace std;
string a, t;
int i, j;
int main()
{
    a=" morning";
    j=1;
    for (i=2; i<=7; i++)
    if (a[j] < a[i])
        j=i;
    j=j-1;
    for (i=1; i<=j; i++)
      printf("%c", a[i]);
    return 0;
}
```

输出：_____

【答案】mo

【分析】简单的模拟题。

8.【NOIP2003】

```
#include <iostream>
#include <cstdio>
#include <cstring>
using namespace std;
int a, b, c, d, sum;
int main()
{
 cin >> a >> b >> c >> d;
 a=a % 23;
 b=b % 28;
 c=c % 33;
 sum=a * 5544+b * 14421+c * 1288-d;
 sum+=21252;
 sum %=21252;
 if (sum==0)
   sum=21252;
 cout << sum << endl;
 return 0;
}
```

输入：283 102 23 320 输出：_____

【答案】8910

【分析】本题好像只是考一考选手的计算能力，只要细心一些，计算不会出错，就能解决问题。运算过程如下表所示：

行号	a	b	c	d	sum
8	283	102	23	320	无值
11	7	18	23	320	无值
12	7	18	23	320	327690
13	7	18	23	320	348942
14	7	18	23	320	8910
15	7	18	23	320	8910

9.【NOIP2004 普及组】

```
#include <stdio.h>
int main(){
    int i,j;
    char str1[]="pig-is-stupid";
    char str2[]="clever";
    str1[0]='d';str1[1]='o';
    for(i=7,j=0;j<6;i++,j++)
    str1[i]=str2[j];
    printf("%s\n",str1);
    return 0;
}
```

输出：

【答案】dog-is-clever

【分析】字符串替换

10.【NOIP2005 普及组】

```
#include <stdio.h>
int main() {
    int a, b;
    scanf("%d", &a);
    b=(a * (a * a))+1;
    if (b%3==0) b=b / 3;
    if (b%5==0) b=b / 5;
    if (b%7==0) b=b / 7;
    if (b%9==0) b=b / 9;
    if (b%11==0) b=b / 11;
    if (b%13==0) b=b / 13;
    if (b%15==0) b=b / 15;
    printf("%d \n", (100 * a - b) / 2);
    return 0;
}
```

输入：10

输出：

【答案】499

【分析】注意“/”是整数运算求商。

11.【NOIP2005 普及组】

```
#include <stdio.h>
int main() {
    char str[20]="Today-is-terrible!";
```

```
    int i;
    for (i=6; i <=10; i++)
        if (str[i]=='-') str[i - 1]='x';
    for (i=12; i >=0; i--)
        if (str[i]=='t') str[i+1]='e';
    printf("%s\n", str);
    return 0;
}
```

输出：

【答案】Today-ix-terrible!

【分析】字符串处理。

12.【NOIP2006】

```
#include <iostream>
#define N  7
int fun(char s[],char a,int n)
{
 int j;
 j=n;
 while(a<s[j] && j>0) j--;
 return j;
}
void main()
{
 char s[N+1];
 int k;
 for(k=1;k<=N;k++)
     s[k]='A'+2*k+1;
 cout <<fun(s,'M',N)<<endl;
}
```

输出：_____

【答案】5

【分析】s 从第 1 个到第 7 个的子串＝“DFHJLNP”，就是询问 s 中最后一个不小于 M 的位置，模拟即可。

第2节　普及篇

1.【NOIP1998】
```
#include<iostream>
#include<cstdio>
using namespace std;
const int n=4;
int i,j,i1,j1,k,s,t,s1,l,swapp;
char temp,a[n*2+1];
int main()
{
    for(int i=1; i<=2*n;++i) a[i]=getchar();
    s=0; t=0;
    for(int i=1; i<=n*2;++i)
    if(a[i]=='1') s++;
    else if(a[i]=='0') t++;
    if(s!=n||t!=n) printf("error");
    else
    {
        s1=0;
        for(int i=1; i<=2*n-1;++i)
          if(a[i]!=a[i+1]) s1++;
        printf("jamp=%d\n",s1); swapp=0;
        for(int i=1; i<=2*n-1;++i)
        for(int j=i; j<=2*n;++j)
          if(a[i]!=a[j])
          {
              temp=a[i]; a[i]=a[j]; a[j]=temp;
              s=0;
              for(int l=1; l<=2*n-1;++l)
              {
                  if(a[l]!=a[l+1]) s++;
              }
              if(s>swapp)
              {
                  swapp=s; i1=i; j1=j;
              }
              temp=a[i]; a[i]=a[j]; a[j]=temp;
          }
          if(swapp>0) printf("maxswap=%d i=%d j=%d",swapp-s1,i1,j1);
    }
}
```
输入:10101100　　输出:

【答案】jamp=5
　　maxswap=2 i=6 j=7

【分析】本题执行步骤较多，必须明白程序的作用才能得到正确答案。下面逐段说明程序含义。

程序段一：

```
for(int i=1; i<=2*n;++i) a[i]=getchar(); //从键盘读取数据
s=0; t=0;
for(int i=1; i<=n*2;++i)
  if(a[i]=='1') s++;
  else if(a[i]=='0') t++;  //统计0和1的个数
if(s!=n||t!=n) printf("error");  //排错处理
```

本程序是初始化程序段，给数a[i]赋值0和1，并用累加算法分别统计0和1的个数，进行排错处理。

程序段二：

```
s1=0;
for(int i=1; i<=2*n-1;++i)
if(a[i]!=a[i+1]) s1++;
printf("jamp=%d\n",s1);
```

本程序统计相邻不同的数对对数，最大值为7，输出结果。可以看出10101100的相邻不同的数对是5。因此，输出jamp=5。

程序段三：

```
swapp=0;
for(int i=1; i<=2*n-1;++i)
for(int j=i; j<=2*n;++j)
if(a[i]!=a[j])
{
   temp=a[i]; a[i]=a[j]; a[j]=temp;  //交换数对
   s=0;
   for(int l=1; l<=2*n-1;++l)
   {
      if(a[l]!=a[l+1]) s++;  //统计相邻不同的数对对数
   }
   if(s>swapp)
   {
      swapp=s; i1=i; j1=j;  //保存最大值并记录位置
   }
   temp=a[i]; a[i]=a[j]; a[j]=temp;  //还原数对
}
```

双重循环类似于选择排序算法，不过不是比较大小，而比较是否相等，可以知道双重循环将进行所有的比较。在所有的比较中进行交换，找最大的相邻不同数对对数，可以看出1010100中交换第6、7个数得到相邻不同数对对数为7为最大值。可知swapp=7，因此输出swapp−s1=7 − 5=2。其中5是程序段二计算的结果。i，j的值是其位置6，7。

2.【NOIP1999】

```
#include<iostream>
using namespace std;
int a[21];
int main()
{
   int x, y, y1, jk, j1, g, e;
```

```
x=3465; y=264; jk=20;
for (j1=1; j1 <=20;++j1) a[j1]=0;
while (y ! =0)
{
   y1=y % 10;
   y=y / 10;
   while (y1 ! =0)
   {
      g=x;
      for (e=jk; e >=1;--e)
      {
         g=g+a[e];
         a[e]=g % 10;
         g=g / 10;
      }
      y1=y1 -1;
   }
   jk=jk-1;
}
j1=1;
while (a[j1]==0) j1=j1+1;
for (jk=j1; jk <=20;++jk) cout<<""<<a[jk];
}
```

输出：

【答案】9 1 4 7 6 0

【分析】答案就是 x 和 y 的乘积逐位输出，题中利用循环拆解 y 的每一位，依次将 x 按序加入到结果数组中，相当于是利用加法完成了乘法的功能。

3.**【NOIP1999】**

```
#include<iostream>
using namespace std;
int i, j, k, a[101];
int main()
{
   for (i=0; i <=100;++i) a[i]=i;
   for (k=5; k >=2;--k)
   {
      for (i=1; i <=100;++i) if (i % k==0) a[i]=0;
      for (i=1; i <=99;++i)
        for (j=1; j <=100-i;++j)
          if (a[j] > a[j+1])
          {
             a[j]=a[j]+a[j+1];
             a[j+1]=a[j]-a[j+1];
             a[j]=a[j]-a[j+1];
      }
   }
   j=1;
```

```
    while (a[j]==0 && j < 100) j=j+1;
    for (i=j; i <=100;++i) a[0]=a[0]+a[i];
    cout<<a[0];
}
```

输出：

【答案】970

【分析】题意是将位置是 5 的倍数的数值改为 0，然后调整为升序；继续将位置是 4 的倍数的数值改为 0，然后调整为升序；将位置是 3 的倍数的数值改为 0，然后调整为升序；将位置是 2 的倍数的数值改为 0，然后调整为升序。非常巧妙的是每次都是将 20 个非零数字改为 0，最后留下的正好是 1，6，11，16，…，96，利用等差数列求和公式求解即可。

4.【NOIP2000】

```
#include<cstdio>
#include<iostream>
#include<cmath>
using namespace std;
int b[105],c[105];
bool p1;
int main()
{
    int i,j,j1,j2,p,q;
    cin>>q>>p; j=1; p1=true; b[j]=q; j1=0;
    while (q>0&&p1)
    {
        j1++; c[j1]=q*10/p; q=q*10-c[j1]*p;
        if (q>0)
        {
            j2=1;
            while(b[j2]! =q&&j2<=j) j2++;
            if(b[j2]==q)
            {
                p1=false; cout<<"0.";
                for(i=1;i<=j2-1;i++) cout<<c[i];
                cout<<'{';
                for(i=j2;i<=j1;i++) cout<<c[i];
                cout<<'}';
            }
            else { j++; b[j]=q; }
        }
        if (q==0)
        {
            cout<<"0.";
            for(i=1;i<=j1;i++) cout<<c[i];
            cout<<endl;
        }
    }
    return 0;
}
```

输入 ① 1 8 输出

输入 ② 2 7 输出

【答案】① 0.125 ② 0.{285714}

【分析】题意是输出两个数相除的商，如果是有限小数，就输出结果；如果是无限循环小数，则将循环节部分用大括号括出来。题中是根据每次除法的余数来判断的，如果余数为 0，则为有限小数；如果余数曾经出现过，则说明出现了循环节。

5.【NOIP2001】

```
#include<cstdio>
using namespace std;
int f, a[9];
int main(){
    for (int i=1; i<=8; i++){
        f=i % 2;
        if (f==0) a[i]=0;
          else a[i]=1;
        for (int j=1; j<=i; j++)
          if (f==0) a[i]=a[i]+j;
            else a[i]=a[i] * j;
    }
    for (int i=1; i<=8; i++)
    printf("%5d",a[i]);
    return 0;
}
```

输出：

【答案】 1 3 6 10 120 21 5040 36

【分析】只要简单模拟就能找到规律了，如果 i 是奇数，则 a[i]为 1～i 的乘积，如果 i 是偶数，则 a[i]为 1～i 的累加和。

6.【NOIP2001】

```
#include<cstdio>
using namespace std;
int n, k, i, a[41];
void find(int x){
     int s, i1, j1;
     bool p;
     i1=0;
     p=1;
     while (p){
         i1=i1+1;
         s=0;
         for (j1=1; j1<=n; j1++)
           if (a[j1] > a[i1]) s=s+1;
         if (s==x-1){
           printf("%d\n",a[i1]);
           p=0;
         }
     }
}
int main(){
```

```
    scanf("%d%d", &n, &k);
    for (int i=1; i<=n; i++)
        scanf("%d",&a[i]);
    find(k);
    find(n-k);
    return 0;
}
```

输入:10 4
　　12 34 5 65 67 87 7 90 120 13
输出:

【答案】67　34

【分析】程序是寻找 n 个数中第 k 大数和第 n-k 大数,简单模拟一遍就可以发现规律。

7.【NOIP2002】

```
#include <iostream>
#include <cstdio>
using namespace std;
int a[21], i, j, k, n, l0, l1, lk;
int main()
{
    scanf("%d %d", &n, &k);
    for (i=0; i<=n-1;++i) a[i]=i+1;
    a[n]=a[n-1];
    l0=n-1;
    lk=n-1;
    for (i=1; i<=n-1;++i)
    {
        l1=l0-k;
        if (l1 < 0) l1=l1+n;
        if (l1==lk)
        {
            a[l0]=a[n];
            lk=lk-1;
            a[n]=a[lk];
            l0=lk;
        }
        else
        {
            a[l0]=a[l1];
            l0=l1;
        }
    }
    a[l0]=a[n];
    for (i=0; i<=n-1;++i) printf("%4d", a[i]);
    printf("\n");
    return 0;
}
```

输入:10　4

输出：

【答案】7 8 9 10 1 2 3 4 5 6

【分析】程序涉及到10个数据，可以模拟完成。最终应该可以发现，程序完成的功能是将1～n循环右移k位。

8.【NOIP2002】

```
#include <iostream>
#include <cstdio>
using namespace std;
int a[11], i, j, s, sp1;
bool p;
int main()
{
  sp1=1;
  a[1]=2;
  j=2;
  while (sp1 < 10)
  {
    j=j+1;
    p=true;
    for (i=2; i <=j-1;++i)
      if (j % i==0) p=false;
    if (p)
    {
      sp1=sp1+1;
      a[sp1]=j;
    }
  }
  j=2;
  p=true;
  while (p)
  {
    s=1;
    for (i=1; i <=j;++i) s=s * a[i];
    s=s+1;
    for (i=2; i <=s-1;++i)
      if (s % i==0) p=false;
    j=j+1;
  }
  printf("%d\n\n", s);
  return 0;
}
```

输出：

【答案】30031

【分析】本题程序分为两段：

第一段如下，其作用是将数组a设成长度为10素数数组：

```
sp1=1;
a[1]=2;
```

```
    j=2;
    while (spl < 10)
    {
       j=j+1;
       p=true;
       for (i=2; i <=j-1;++i)
         if (j % i==0) p=false;  //此循环判断 j 是否为素数,如果是则 p 为真值
       if (p)  //当 j 为素数时,将其放入数组 a
       {
          spl=spl+1;
          a[spl]=j;
       }
    }
```

第二段如下,其作用是寻找满足条件的最小合数:

```
j=2;
p=true;
while (p)
{
   s=1;
   for (i=1; i <=j;++i) s=s * a[i];
   s=s+1;
   for (i=2; i <=s-1;++i) //判断 s 是否为素数,如是和数,则循环结束
     if (s % i==0) p=false;
   j=j+1;  //s 中累乘素数的个数
}
```

s 在循环中的变化如下:

2 * 3+1=7 (素数)
2 * 3 * 5+1=31 (素数)
2 * 3 * 5 * 7+1=211 (素数)
2 * 3 * 5 * 7 * 11+1=2311 (素数)
2 * 3 * 5 * 7 * 11 * 13+1=30031 (素数)

9.【NOIP2002】

```
#include <iostream>
#include <cstdio>
using namespace std;
int a[21], i, j, p, n, q, s;
int main()
{
   scanf("%d %d %d", &p, &n, &q);
   j=21;
   while (n > 0)
   {
      j=j-1;
      a[j]=n % 10;
      n=n / 10;
   }
   s=0;
```

```
    for (i=j; i<=20;++i) s=s * p+a[i];
    printf("%d\n", s);
    j=21;
    while (s > 0)
    {
       j=j-1;
       a[j]=s % q;
       s=s / q;
    }
    for (i=j; i<=20;++i) printf("%d", a[i]);
    printf("\n");
    return 0;
}
```

输入:7 3051 8

输出:

【答案】

1065

2051

【分析】程序的含义是读入一个 p 进制数 n,先转换位十进制输出,然后转换为 q 进制数输出。题中第一部分是将 n 逐位拆解放入 a 数组;然后当成 p 进制利用秦九韶算法转换成十进制并输出;然后按照 q 进制逐步拆解放入 a 数组并输出。

10.【NOIP2004】

```
#include <stdio.h>
int main(){
  int u[4], a, b, c, x, y, z;
  scanf("%d %d %d %d",&(u[0]), &(u[1]), &(u[2]), &(u[3]));
  a=u[0]+u[1]+u[2]+u[3]-5;
  b=u[0]*(u[1]-u[2]/u[3]+8);
  c=u[0]*u[1]/u[2]*u[3];
  x=(a+b+2)*3-u[(c+3)%4];
  y=(c*100-13)/a/(u[b%3]*5);
  if((x+y)%2==0)z=(a+b+c+x+y)/2;
  z=(a+b+c-x-y)*2;
  printf("%d\n",x+y-z);
  return 0;
}
```

输入:2 5 7 4

输出:

【答案】263

【分析】考查运算优先级,直接计算。

优先级	运算符	结合性
1	()[]	从左到右
2	! +(正)-(负)~++--	从右到左
3	*/%	从左到右
4	+(加)-(减)	从左到右
5	<<> >>	从左到右

（续表）

优先级	运算符	结合性
6	< <= > >=	从左到右
7	== ！=	从左到右
8	&（按位与）	从左到右
9	^	从左到右
10	|	从左到右
11	&&	从左到右
12	||	从左到右
13	?：	从右到左
14	= += -= *= /= %= &= |= ^= <<= >>= >>>=	从右到左

11.【NOIP2004 提高组】

```
#include <stdio.h>
int number, ndata, data[100], sum;
void solve(int s,int sign,int n){
    int i;
    for(i=s;i<ndata;i++){
        sum+=sign*(number/n/data[i]);
        solve(i+1,-sign,n*data[i]);
    }
}
int main(){
  int i;
  scanf("%d %d",&number,&ndata);
  sum=0;
  for(i=0;i<ndata;i++)scanf("%d", &(data[i]);
  solve(0,1,1);
  printf("%d\n",sum);
  return 0;
}
```

输入：1000 3 5 13 11

输出：

【答案】328

【分析】考查递归，层数不深，可以暴力模拟。

第一次循环调用三次递归函数 solve(i+1,-sign,n*data[i])：分别为：

Solve(1,-1, 1*5)；solve(2,-1,1*13)；solve(3,-1, 1*11)；

第一层 200

第二层 -15

第三层 1

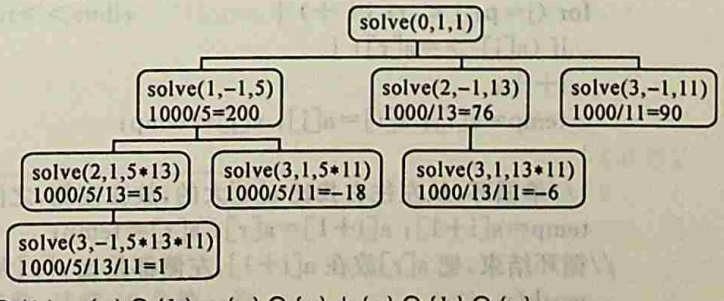
第二层 -18

第一层 76

第二层 -6

第一层 90

总和为 328



另外，{a}+{b}+{c}-{a}∩{b}-{c}∩{b}-{a}∩{c}+{a}∩{b}∩{c}。

12.【NOIP2005 提高组】#include<stdio.h>

```
#include <math.h>
int a[50];
void work(int p, int r) {
  if (p < r) {
    int i=p-1, j, temp;
    for (j=p; j < r; j++) {
      if (a[j] >=a[r]) {
        i++;
        temp=a[i]; a[i]=a[j]; a[j]=temp;
      }
    }
    temp=a[i+1]; a[i+1]=a[r]; a[r]=temp;
    work(p, i);
    work(i+2, r);
  }
}
int main() {
    int n, i, sum=0;
    scanf("%d", &n);
    for (i=0; i < n; i++) scanf("%d", &(a[i]));
    work(0, n-1);
    for (i=0; i < n-1; i++) sum+=abs(a[i+1]-a[i]);
    printf("%d\n", sum);
    return 0;
}
```

输入:10 23 435 12 345 3123 43 456 12 32-100

输出:

【答案】3223

【分析】work(p,r)是二分快排,关键是搞清楚这个函数。排好序后,算邻近 2 个数的差的绝对值。由于是 abs,所以排序不需要知道是从小到大还是从大到小,最后的结果是头尾两数的绝对值和。

```
int a[50];
void work(int p, int r) {
    if (p < r) {
      int i=p-1, j, temp;
      for (j=p; j < r; j++) {
        if (a[j] >=a[r]) {
        i++;
        temp=a[i]; a[i]=a[j]; a[j]=temp;
      }
      }//单循环,从左往右找比 a[r]大的,放在左侧,之间不排序
      temp=a[i+1]; a[i+1]=a[r]; a[r]=temp;
     //循环结束,把 a[r]放在 a[i+1],左侧都是比 a[r]大的,右侧都是比 a[r]小的
      work(p, i);
      work(i+2, r);
```

```
    }
}
```

13.【NOIP2007】

```
#include <iostream.h>
void fun(int *a,int *b)
{
    int *k;
    k=a; a=b; b=k;
}
void main( )
{
    int a=3, b=6, *x=&a, *y=&b;
    fun(x,y);
    cout<<a<<","<<b<<endl;
}
```

输出：_____

【答案】3,6

【分析】*x,*y,x,y 为指针，&a 为取 a 变量的地址，fun()交换了 x,y 的内容，但输出的还是原来的变量。

14.【NOIP2007】

```
#include <iostream>
#include <iomanip>
#include <math>
void main()
{
    int a1[51]={0};
    int i,j,t,t2,n=50;
    for (i=2;i<=sqrt(n);i++)
    if(a1[i]==0)
    {
        t2=n/i;
        for(j=2;j<=t2;j++) a1[i*j]=1;
    }
    t=0;
    for (i=2;i<=n;i++)
      if(a1[i]==0)
      {
          cout<<setw(4)<<i; t++;
          if(t%10==0) cout<<endl;
      }
    cout<<endl;
}
```

输出：_____

【答案】

2 3 5 7 11 13 17 19 23 29

31 37 41 43 47

【分析】此题即利用筛法求素数，注意 10 个换一行。

第3节　提高篇

1.【NOIP1998】

```
#include<iostream>
#include<cstdio>
#include<cmath>
using namespace std;
const int n=5;
int k;
int a[2*n+1][2*n+1];
int main()
{
    k=1;
    for(int i=1; i<=2*n-1;++i)
    if(i<=n)
    if (i%2==1)
      for(int j=i; j>=1;--j)
      {
        a[i-j+1][j]=k; k++;
      }
      else for(int j=1; j<=i;++j)
      {
        a[i-j+1][j]=k; k++;
      }
    else if(i%2==1)
      for(int j=n; j>=i-n+1;--j)
      {
        a[i-j+1][j]=k; k++;
      }
      else for(int j=i-n+1; j<=n;++j)
      {
        a[i-j+1][j]=k; k++;
      }
    for(int i=1; i<=n;++i)
    {
      for(int j=1; j<=n;++j) printf("%d    ",a[i][j]);
      cout<<endl;
    }
}
```

【答案】

1	3	4	10	11
2	5	9	12	19
6	8	13	18	20
7	14	17	21	24
15	16	22	23	25

【分析】题意是要输出一个蛇形方阵，可通过模拟二维数组填数的方法逐步发现规律。

2.【NOIP1998】

```
#include<iostream>
#include<cstdio>
using namespace std;
int i,j,s;
int b[6];
int main()
{
    s=1;
    for(int i=1; i<=5;++i) b[i]=i;
    j=1;
    while(j>0)
    {
    j=5;
    while(j>0&&b[j]==10+j-5) j--;
    if(j>0)
    {
    s++; b[j]++;
    for(int i=j+1; i<=5;++i) b[i]=b[j]+i-j;
    }
    }
    printf("S=%d",s);
}
```

【答案】S=252

【分析】本题的主要目的是计算变量 s 的值。通过列表观察各变量的变化可以总结出规律。

J 的值变化一次，b[5]的值被重新赋值，变化规律是：

第一次 b[5]变化 5 次后，最后 b[5]=10；

第二次 b[4]变化引起 b[5]的变化，其变化次数为 4 次；

第三次 b[3]变化引起 b[4]的变化，b[4]变化引起 b[5]的变化，循环变化；

同理 b[1]，b[2]，b[3]也有同样的变化，s 是计数器，记录变化的次数。

变化次数 s=6+(5+4+3+2+1)+(15+10+5+4+1)+(35+20+10+4+1)+(70+35+15+6)=252

3.【NOIP2000】

```
#include<cstdio>
#include<iostream>
#include<cmath>
using namespace std;
const int n=7,m=6;
int g[n+5][m+5];
double d; bool p;
double disp(int x1,int y1,int x2,int y2)
{
    return sqrt((x1-x2)*(x1-x2)+(y1-y2)*(y1-y2));
}
int main()
{
```

```
int i,j,x0,y0,x1,y1,x2,y2;
for(i=0;i<=n;i++) for(j=0;j<=m;j++) g[i][j]=0;
cin>>x1>>y1>>x2>>y2;
g[x1][y1]=g[x2][y2]=1; p=true;
while (p)
{
      p=false; d=disp(x1,y1,x2,y2); x0=x1; y0=y1;
      for(i=4;i<=n;i++) for(j=0;j<=m;j++)
      if(d)>disp(i,j,x2,y2)&&g[i][j]==0
      {
        d=disp(i,j,x2,y2); x0=i; y0=j;
      }
      if (x0! =x1||y0! =y1)
      {
        x1=x0; y1=y0; p=true; g[x1][y1]=1;
      }
      d=disp(x1,y1,x2,y2); x0=x2; y0=y2;
      for(i=0;i<=3;i++) for(j=0;j<=m;j++)
      if(d)<disp(x1,y1,i,j)&&g[i][j]==0
      {
        d=disp(x1,y1,i,j); x0=i; y0=j;
      }
      if (x0! =x2||y0! =y2)
      {
        x2=x0; y2=y0; p=true; g[x2][y2]=1;
      }
   }
   cout<<x1<<' '<<y1<<' '<<x2<<' '<<y2<<endl;
  return 0;
}
```

输入：　7　6　0　0

输出：

【答案】4 3 0 2

【分析】本题程序较为复杂，需要分化为块，逐块理解。

```
double disp(int x1,int y1,int x2,int y2)
{
    return sqrt((x1-x2)*(x1-x2)+(y1-y2)*(y1-y2));
}
```

由表达式可以看出这是两点距离计算公式。此函数的作用是计算点(x1,y1)到点(x2,y2)的距离。

```
for(i=0;i<=n;i++) for(j=0;j<=m;j++) g[i][j]=0;
```

矩阵初始化；

```
cin>>x1>>y1>>x2>>y2;
g[x1][y1]=g[x2][y2]=1; p=true;
```

读入初始结点和终止结点，赋值为1标志着该点被访问过。

```
while (p)
```

```
{
    p=false; d=disp(x1,y1,x2,y2); x0=x1; y0=y1;
    for(i=4;i<=n;i++) for(j=0;j<=m;j++)
    if(d)>disp(i,j,x2,y2)&&g[i][j]==0
    {
      d=disp(i,j,x2,y2); x0=i; y0=j;
    }
    if (x0! =x1||y0! =y1)
    {
      x1=x0; y1=y0; p=true; g[x1][y1]=1;
    }
```

在下半个矩阵中找寻一个点，使得它与点(x2,y2)距离尽可能的大且该点没有被访问过。

```
  d=disp(x1,y1,x2,y2); x0=x2; y0=y2;
    for(i=0;i<=3;i++) for(j=0;j<=m;j++)
    if(d)<disp(x1,y1,i,j)&&g[i][j]==0
    {
      d=disp(x1,y1,i,j); x0=i; y0=j;
    }
    if (x0! =x2||y0! =y2)
    {
      x2=x0; y2=y0; p=true; g[x2][y2]=1;
    }
  }
```

在上半个矩阵中找寻一个点，使得它与点(x1,y1)距离尽可能的小且该点没有被访问过。

根据以上分析，可知当输入为(7,6)和(0,0)时，两个移动的点分别移动序列为：

(7,6)(4,0)(4,6)(4,1)…(4,3)

(0,0)(0,6)(1,0)(1,6)…(0,2)

所以程序的最后运行结果为：4302。

4.【NOIP2001】

```
#include<cstdio>
using namespace std;
int x, y1, y2, y3;
int main(){
  scanf("%d", &x); y1=0; y2=1; y3=1;
  while (y2 <=x) {
  y1=y1+1; y3=y3+2; y2=y2+y3;
  }
  printf("%d",y1);
}
```

输入：23420　　输出：

【答案】153

【分析】本题计算的是和的超过 X 的奇数序列最少有多少项。因为输入的 X 比较大，计算量大，不能计算直接得到结果。可采用试取 N，根据计算结果选择增大或减小，直至找到解为止。具体过程如下：

因奇数数列的通式公式为 a[n]=2*n-1；

取 n=100,第 N 项为:2*N-1=199;
根据高斯算法 S[n]=(a[1]+a[100])*50=200*50=10000;
与 X 相比小,N 需增大;
取 N=150,第 N 项为:2*N-1=299;
S[n]=(a[1]+a[150])*75=300*75=22500;
与 X 相比小,N 仍需增大;
取 N=152,第 N 项为:2*N-1=303;
S[n]=(a[1]+a[152])*76=303*76=23028;
与 X 相比小,N 需增大;
取 N=153,第 N 项为:2*N-1=305;
S[n]=(a[1]+a[153])*77=305*77=23485;
与 X 相比大,N 不需增大。

5.【NOIP2002】

```
#include <iostream>
#include <cstdio>
#include <cmath>
using namespace std;
double d1, d2, x, Min;
int main()
{
    Min=10000;
    x=3;
    while (x < 15)
    {
        d1=sqrt(9+(x-3) * (x-3));
        d2=sqrt(36+(15-x) * (15-x));
        if (d1+d2 < Min) Min=d1+d2;
        x=x+0.001;
    }
    printf("%10.2lf\n", Min);
    return 0;
}
```

输出:

【答案】15.00

【分析】本题需要平面几何的基本知识。

语句 d1=sqrt(9+(x-3) * (x-3));可以看成 d1=sqrt((0-3) * (0-3)+(x-3) * (x-3),根据平面几何两点间距离公式,可以理解成平面直角坐标系中点(0 ,x)到点(3 ,3)的距离。语句 d2=sqrt(36+(15-x) * (15-x));可以看成 d2=sqrt((0-6) * (0-6)+(x-15) * (x-15)),可以理解成平面直角坐标系中点(0,x)到点(6,15)的距离。程序是求 x 在 0 到 15 之间时 d1+d2 的最小值。可理解为平面直角坐标系中点(0,x)到点(3,3)与点(6,15)距离和的最小值。因(0,x)到点(3,3)的距离与到点(-3,3)的距离相同。所以点(0,x)到点(3,3)与(6,15)距离与点(0,x)到点(-3,3)与(6,15)距离相同。而点(-3,3)与(6,15)所决定的直线过 x 轴,交点为(0,7)。所以当 x=7 时,有最小距离 sqrt((-3-6) * (-3 - 6)+(3 - 15) * (3 - 15))=15。

6.【NOIP2002】

```
#include <iostream>
#include <cstdio>
using namespace std;
```

```
int i, n, jr, jw, jb;
char ch[21], ch1;
int main()
{
    scanf("%d", &n);
    for (i=1; i<=n;++i) cin>>ch[i];
    jr=1;
    jw=n;
    jb=n;
    while (jr<=jw)
    {
        if (ch[jw]=='R')
        {
            ch1=ch[jr];
            ch[jr]=ch[jw];
            ch[jw]=ch1;
            jr=jr+1;
        }
        else if (ch[jw]=='W') jw=jw-1;
        else
        {
            ch1=ch[jw];
            ch[jw]=ch[jb];
            ch[jb]=ch1;
            jw=jw-1;
            jb=jb-1;
        }
    }
    for (i=1; i<=n;++i) printf("%c", ch[i]);
    printf("\n");
    return 0;
}
```

输入:10
RBRBWWRBBR
输出:

【答案】RRRRWWBBBB

【分析】本题对数组 ch 中的字母进行调整。从最右边开始分三种情况进行处理。

第一种情况 ch[jw]=='R',进行如下处理:

```
ch1=ch[jr];
ch[jr]=ch[jw];
ch[jw]=ch1;
jr=jr+1;
```

jr 是数组左边连续的 R 中最靠右的 R 的下标加 1,如果左边没有 R 则加 1。ch[jw]与 ch[jr]进行交换,作用将 R 移至左边,和连续的 R 放在一起,同时修改 jr 的值。

第二种情况 ch[jw]=='W',进行如下处理:

```
jw=jw - 1;
```

数组未做任何移动，只修改 jw 的值。

第三种情况经分析条件为(ch[jw]！='W'&& ch[jw]！='R')进行如下处理：

```
ch1=ch[jw];
ch[jw]=ch[jb];
ch[jb]=ch1;
jw=jw-1;
jb=jb-1;
```

jb 是数组右边经过处理的非 R 和 W 元素中最靠左的元素下表加 1，初始值为 n。ch[jw]与 ch[jb]进行交换，作用是将 ch[jw]移至右边，和连续的非 R 和 W 元素放在一起，同时修改 jb 的值。jw=jw - 1;表示接着处理左边的元素。用一句话来说就是将除 R、W 以外的字母移至右边。

最后的输出结果是所有的 R 移到左边，所有的 W 在中间，其余的字母在后边。

7.【NOIP2003】

```
#include <iostream>
#include <cstdio>
#include <cstring>
using namespace std;
int a[6], sum, n, mx, i, j, k;
bool cover[22001];
int main()
{
 cin >> a[5] >> a[4] >> a[3] >> a[2] >> a[1] >> a[0];
 if(a[5]==0 && a[3]==0 && a[1]==0)
 {
    a[5]=a[4];
    a[4]=a[2];
    a[3]=a[0];
    a[2]=0;
    a[0]=0;
 }
 for (i=0; i<=5; i++)
    if (a[i] > 10) a[i]=10+(a[i] % 2);
 sum=0;
 for (i=0; i<=5; i++) sum=sum+a[i] * (6-i);
 if (sum % 2 ! =0)
 {
    puts("Can't be divided.");
    return 0;
 }
 sum=sum / 2;
 mx=0;
 cover[0]=true;
 for (i=1; i<=sum * 2; i++) cover[i]=false;
 for (i=0; i<=5; i++)
 {
    j=0;
    while (j < a[i])
```

```
    {
        for (k=mx; k >=0; k--)
        {
            if(cover[k]) cover[k+6-i]=true;
        }
        mx=mx+6-i;
        j=j+1;
    }
  }
  if(cover[sum]) puts("Can be divided. ");
  else puts("can't be divided. ");
  return 0;
}
```

输入:4 7 9 20 56 48 输入:1000 7 101 20 55 1 输入:2000 5 1 1 0 0

输出:_____ 输出:_____ 输出:_____

【答案】Can't be divided　Can be divided　Can't be divided

【分析】根据程序的执行过程直接模拟即可,数据只有涉及到 6 个。

8.【NOIP2004】

```
#include <stdio.h>
char c[3][200];
int s[10],m, n;
void numara(){
  int i, j, cod, nr;
for(j =0;j<n;j++){
    nr=0;cod=1;
    for (i=0;i<m;i++){
      if(c[i][j]=='1'){
        if (! cod){cod=1;s[nr]++;nr=0;}
      }
      else{
        if(cod){nr=1;cod=0;}
        else nr++;
      }
    }
    if (! cod)s[nr]++;
  }
}
int main(){
  int i, j;
  scanf("%d %d\n", &m, &n);
  for(i=0;i<m;i++)gets(c[i]);
  numara();
  for(i=1;i<=m;i++)
    if(s[i]! =0)printf("%d %d ", i, s[i]);
  return 0;
}
```

输入:

3 10
1110000111
1100001111
1000000011
输出：
【答案】1 4 2 1 3 3
【分析】首先注意输出只循环3次，每次2个数有空格隔开。Numara中j循环10，负责列，而i循环3次。Cod是标记开关。S[10]是计数器数组。一列中由1变0或由0变1的个数s[1]，一列中有连续2个0的个数为s[2]，一列中有3个0的个数为s[3]。

9.【NOIP2004 提高组】
```
#include<stdio.h>
const int u[3]={1,-3, 2};
const int v[2]={-2, 3};
int g(int n){
  int i, sum=0;
  for(i=1;i<=n;i++)sum+=u[i%3]*i;
  return sum;
}
int main(){
  int n, i, sum=0;
  scanf("%d", &n);
  for(i=1;i<=n;i++)sum+=v[i%2]*g(i);
  printf("%d\n", sum);
  return 0;
}
```
输入：103
输出：
【答案】-400
【分析】u[i%3]和v[i%2]，虽然103项，2和3公倍数为6，6个一循环的等差数列，硬算出前面部分项，有6组等差数列，然后按公式计算。

1	3	-3	7	3	-13	13	3	-23
2	-2	1	8	-2	3	14	-2	5
3	3	4	9	3	12	15	3	20
4	-2	-8	10	-2	-18	16	-2	-28
5	3	2	11	3	4	17	3	6
6	-2	8	12	-2	16	18	-2	24

10.【NOIP2005 提高组】
```
#include<stdio.h>
#include<string.h>
int main(){
    char str[60];
    int len, i, j, chr[26];
  char mmin='z';
    scanf("%s", str);
    len=strlen(str);
    for (i=len-1; i>=1; i--)
    if (str[i-1] < str[i]) break;
    if (i==0) {
      printf("No result! \n");
```

```
return 0;
    }
      for (j=0; j < i-1; j++) putchar(str[j]);
memset(chr, 0, sizeof(c)hr);
   for (j=i; j < len; j++) {
     if (str[j] > str[i-1] && str[j] < mmin)
     mmin=str[j];
     chr[str[j]-'a']++;
   }
   chr[mmin-'a']--;
   chr[str[i-1]-'a']++;
   putchar(mmin);
   for(i=0; i < 26; i++)
     for(j=0; j < chr[i]; j++)
       putchar(i+'a');
   putchar('\n');
   return 0;
}
```

输入：zzyzcccbbbaaa

输出：

【答案】zzzaaabbbcccy

【分析】算法类似桶排，字符串处理，注意数组起始位置为0，循环控制变量<i-1，此题chr[26]是字母计数器，统计字母出现的次数，最后根据计数器表输出。

11.【NOIP2005 提高组】

```
#include <stdio.h>
long g(long k) {
    if (k <=1) return k;
    return (2002 * g(k-1)+2003 * g(k-2)) % 2005;
}

int main() {
   long n;
   scanf("%ld", &n);
printf("%ld\n", g(n));
return 0;
}
```

输入：2005

输出：

【答案】31

【分析】递归转递推再加同余问题，方法比较多。$g(x)=(-3g(x-1)-2g(x-2))\%2005$，然后我们要找到$f(x)=-3g(x-1)-2g(x-2)$的通项公式，在没有其他办法的情况下，我们可以通过列举来找规律：$f(1)=1$，$f(2)=-3f(1)-2f(0)=-3$，$f(3)=-3f(2)-2f(1)=9-2=7$，$f(4)=-15$……

通过上面的几个结果我们基本可以得出它的规律：2^1-1，$-(2^2-1)$，2^3-1，$-(2^4-1)$……

通项公式：$f(x)=(-1)^{x-1}(2^x-1)$

答案应该是：

g(2005)

$=(-1)^{2004}*(2^{2005}-1)\%2005$

$=(2^{2005}-1)\%2005$

$=2^{401*5}\%2005-1$

$=(2^{401}*2^{401}*2^{401}*2^{401}*2^{401})\%2005-1$

(2 与 401 互质才能用费马小定理,2005 不是质数)

因为 $2^{401-1}\equiv 1(\%401)$,2 和 401 互质,因为 5 和 2 互质

$=(2*2*2*2*2)\%2005-1$

$=31$

$2^4 \bmod 5=1$,所以 $2^{400} \bmod 5=1$,$2^{401} \bmod 5=2$

由费马小定理知 $2^{401} \bmod 401=2$

又$(5,401)=2005$,$2^{401} \bmod 2005=2$,所以 $2^{2005} \bmod 2005-1=2^5-1=31$

12.【NOIP2007 普及组】

```
#include <iostream.h>
#include "ctype.h"
void expand(char s1[],char s2[])
{
 int i,j,a,b,c;
 j=0;
 for(i=0;(c=s1[i])! ='\0';i++)
 if(c=='-')
 {
  a=s1[i-1]; b=s1[i+1];
  if( isalpha(a)&&isalpha(b) || isdigit(a)&&isdigit(b) )
      //函数 isalpha(a)用于判断字符 a 是否为字母,isdigit(b) 用于判断字符 b 是否为
  数字,如果是,返回 1,否则返回 0
  {
   j--;
   do s2[j++]=a++;
   while(tolower(a)<tolower(s1[i+1]));
  }
  else s2[j++]=c;}
  else s2[j++]=c;
  s2[j]='\0';
 }
 void main()
{
 char s1[100],s2[300];
    cin>>s1;
    expand(s1,s2);
    cout<<s2<<endl;
}
```

输入:wer2345d-h454-82qqq　　输出:_____

【答案】wer2345defgh45456782qqq

【分析】函数的名字为 expand,可以猜猜它的功能,if(c=='-'),里面有个 do while,这个是本题的关键语句。此题为字符串的扩展操作,对于形如"d-h"、"4-8"的三字符串,去掉中间的减号,将字母或数字连续写出,即分别输出为"defgh"、"45678"。

13.【NOIP2007 提高组】

```
#include"stdio.h"
    charch[]={'q','A','S','O','R','T','E','X','A','M','P','L','E'};
```

```
    intn=12;
void shift(int k, int n)
{
    char v;
    int j;
    v=ch[k]; j=k+k;
    while (j<=n)
    {
      if((j<n) &&(ch[j]<ch[j+1])) j++;
      if (v<ch[j])
      {
        ch[j/2]=ch[j]; j*=2;
      }
      else
        return;
      ch[j/2]=v;
    }
}
void hpsrt(void)
{
   int k;
   char tmp;
   for (k=n/2; k>0; k--) shift(k,n);
   printf("No.1: ");
   for(k=1; k<=n; k++) putchar(c)h[k];
   putchar('\n');
   for (k=n; k>0; k--)
   {
      tmp=ch[1]; ch[1]=ch[k]; ch[k]=tmp; shift(1,k-1);
   }
}
main()
{
   int k;
   hpsrt();
   printf("No.2: ");
   for(k=1; k<=n; k++) putchar(c)h[k];
   putchar('\n');
}
```

输出：_____

_____

【答案】

No.1: XTORSEAAMPLE

No.2: AAEELMOPRSTX

【分析】考查堆排序，第一次建堆，把原数组维护成大根堆，X 被移到堆顶，但数组并不是有序的，后面用循环进行堆排序，把最大的交换到数组最后，规模变为 N－1，直到最后一个，数组从小到大排序。

第五章　完善程序

第1节　入门篇

1.【NOIP1999】

【问题描述】

下面程序的功能是从键盘读取A,B数组的元素,A,B数组均已从小到大排好序(无相同元素),现将A,B合并为数组C,同样要求数组C也是从小到大排好序(有相同元素时只保留一个)。

程序中N表示数组A,B的长度,i,j,k分别表示数组A,B,C的取数或存数的指针。

【程序清单】

```
#include <iostream>
#include <cstdio>
using namespace std;
const int n=8;
const int m=2 * n;
int a[n+1], b[n+1], c[m+1], i, j, k;
void copy(int &x, int &y, int &i, int &j)
{
    i=i+1; y=x; j=j+1;
}
int main()
{
   for (i=1; i <=n;++i) cin>>a[i];
   for (i=1; i <=n;++i) cin>>b[i];
   i=1; j=1;  ①
   while (  ②  )
   {
      if (a[i] < b[j]) copy(a[i], c[k+1], k, i);
      else if (b[j] < a[i]) copy(b[j], c[k+1], k, j);
      else
      {
         copy(a[i], c[k+1], k, i);
         ③
      }
   }
   while (  ④  ) copy(a[i], c[k+1], k, i);
   while (  ⑤  ) copy(b[j], c[k+1], k, j);
   for (i=1; i <=k;++i) cout<<"     "<<c[i];
   return 0;
}
```

【答案】①k=0；　②i <=n && j <=n　③j=j+1；　④i <=n　⑤j <=n

【分析】两个数组合并，将 a 数组和 b 数组合并放入 c 数组，指向当前元素位置的变量分别是 i、j、k(第 1 空，c 数组还没放数，所以 k=0)，如果 a 数组和 b 数组都没有比较到结束(第 2 空)，在比较过程中将小的一个复制到 c 数组的末尾；如果相同，则将 a 数组当前元素复制到 c 数组的末尾，同时 b 数组的位置变量 j 也要加 1(第 3 空)。循环结束，将剩余的未复制到 c 数组的部分全部复制到 c 数组的末尾(第 4 空、第 5 空)。

2.【NOIP2001】输入 n 个 0 到 100 之间的整数，由小到大排序输出，每个数占 4 格，每行输出 8 个。

【程序清单】

```
#include <iostream>
#include <cstdio>
using namespace std;
int i, j, k, n, x, b[101];
int main(){
    scanf("%d\n",&n);
    for (i=0; i <=100; i++) b[i]=0;
    for (i=1; i <=n; i++){
        scanf("%d", &x);
        b[x]=    ①    ;
    }
    ②    ;
    for (i=0; i <=100; i++)
      while (    ③    ){
        printf(    ④    );
        k=k+1;
        b[i]=b[i]-1;
        if (    ⑤    ) putchar('\n');
      }
    putchar('\n');
    return 0;
}
```

【答案】① b[x]+1　② k=0　③ b[i]>0　④ "%4d", i　⑤ k % 8==0

【分析】简单的桶排序。每读入一个数就在该数作为下标的位置增加 1(第 1 空)；为了每行输出 8 个，程序由 k 来控制，但 k 需要赋初值 0(第 2 空)；循环每个下标，当该位置有值(第 3 空)，输出该位置(第 4 空)；如果输出个数正好是 8 的倍数(第 5 空)，则输出回车。

3.【NOIP2003】一元二次方程

【问题描述】

方程 $ax^2+bx+c=0$，要求给出它的实数解。

【输入格式】

三个实数：a，b，c，是方程的三个系数($a\neq 0$)。

【输出格式】

如果无实数解，则输出“No solution”；

如果有两个相等的实数解，则输出其中一个，四舍五入到小数点后面 3 位；

如果有两个不等的实数解，则解与解之间用逗号隔开，同样要四舍五入到小数点后 3 位。

【输入样例】

1 2 1

【输出样例】

-1.000

【程序清单】

```
#include <iostream>
#include <cstdio>
#include <cstring>
#include <cmath>
using namespace std;
double a, b, c, m;
int main()
{
    cin >> a >> b >> c;
    m=b * b-4 * a * c;
    if (    ①    )
    {
        printf("%.3lf",        ②        );
        printf(" , ");
        printf("%.    ③    lf", (-1 * b-sqrt(m))/ (2 * a));
    }
    else if (    ④    )
        printf(        ⑤        );
    else
        puts("No solution");
    return 0;
}
```

【答案】① m>0　② (-1*b+sqrt(m))/(2*a)　③ 3

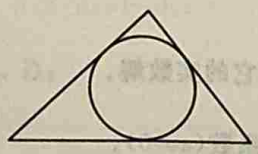
④ abs(m)<0.0001　⑤ "%.3lf",-1*b/(2*a)

【分析】程序根据一元二次方程的求根公式进行判断求解；如果 m 大于 0(第 1 空)，说明有两个不同的根，输出第一个根(第 2 空)、逗号、第二个根，保留 3 位小数(第 3 空)；如果 m 等于 0，注意浮点数不能简单判等，而是绝对值约等于 0(第 4 空)，这时只要输出一个根即可(第 5 空)；如果小于 0，就没有实数根了。

4.【NOIP2004 普及组】三角形内切圆的面积

【问题描述】

给出三角形三边的边长，求此三角形内切圆(如下图所示，三角形的内切圆是和三角形三边都相切的圆)的面积。



【输入格式】

三个正实数 a,b,c(满足 a+b>c,b+c>a,c+a>b)，表示三角形三边的边长。

【输出格式】

三角形内切圆的面积，结果四舍五入到小数点后面 2 位。

【输入样例】

3 4 5

【输出样例】

3.14

【程序清单】

```
#include <iostream>
#include <cstdio>
#include <cmath>
using namespace std;
int main(){
    float a, b, c, r, s, t;
    scanf("%f %f %f", &a, &b, &c);
    s=(    ①    ) / 2;
    t=    ②    (s * (s-a) * (s-b) * (s-c));
    r=t / s;
    printf("    ③    \n", 3.1415927 * r *    ④    );
    return 0;
}
```

【答案】① a+b+c　② sqrt　③ %.2f　④ r

【分析】海伦公式，数论知识。

5.【NOIP2005 普及组】判断质数

【问题描述】

给出一个正整数，判断这个数是否是质数。

【输入格式】

一个正整数 n(1 ≤ n ≤ 10000)。

【输出格式】

如果 n 是质数，输出"YES"；否则，输出"NO"。

【输入样例】

10

【输出样例】

NO

【程序清单】

```
#include <iostream>
#include <cstdio>
using namespace std;
int main() {
    int    ①    ;
    scanf("%d", &n);
    if (n==2) puts(    ②    );
    else if (    ③    || n % 2==0) puts("NO");
    else {
        i=3;
        while (i * i <=n) {
            if (    ④    ) {
                puts("NO"); return 0;
            }
            i=i+2;
        }
```

```
        puts("YES");
    }
    return 0;
}
```

【答案】① n，i（或者 i，n）　② "YES"
③ n==1（或者 n − 1==0）　④ n % i==0 （或者 ! n % i）
【分析】题目给的范围中 1<=n<=10000，1 要当心，特殊处理。
①送分 n，i，变量定义；
②2 是质数单独处理，"YES"；
③空 n==1 比较容易，题目数据范围里有；
④不难，整除就是 NO，n%i==0。

6.【NOIP2005】木材加工

【问题描述】

木材厂有一些原木，现在想把这些木头切割成一些长度相同的小段木头，需要得到的小段的数目是给定的。当然，我们希望得到的小段越长越好，你的任务是计算能够得到的小段木头的最大长度。

木头长度的单位是 cm。原木的长度都是正整数，我们要求切割得到的小段木头的长度也是正整数。

【输入格式】

第一行是两个正整数 N 和 K(1 ≤ N ≤ 10000，1 ≤ K ≤ 10000)，N 是原木的数目，K 是需要得到的小段的数目。

接下来的 N 行，每行有一个 1 到 10000 之间的正整数，表示一根原木的长度。

【输出格式】

输出能够切割得到的小段的最大长度。如果连 1cm 长的小段都切不出来，输出"0"。

【输入样例】

```
3 7
232
124
456
```

【输出样例】

```
114
```

【程序清单】

```
#include <iostream>
#include <cstdio>
using namespace std;
int n, k, len[10000];
int isok(int t) {
    int num=0, i;
    for (i=0; i < n; i++) {
        if (num >=k) break;
        num= ____①____;
    }
    if (____②____) return 1;
    else return 0;
}
int main() {
```

```
    int i, left, right, mid;
    scanf("%d%d", &n, &k);
    right=0;
    for (i=0; i < n; i++) {
        scanf("%d", &(len[i]));
        if (right < len[i]) right=len[i];
    }
    right++;
    ③____;
    while (____④____ < right) {
        mid=(left+right) / 2;
        if (____⑤____) right=mid;
        else left=mid;
    }
    printf ("%d\n", left);
    return 0;
}
```

【答案】① num+len[i] / t　② num >=k　③ left=0
④ left+1　⑤ ! isok(mid)　(或者 isok(mid)==0)

【分析 1】

①num+len[i] / t,按 mid 的长度来切,能切出的段数;

②num >=k,按 mid 来切,能否达到要求,或者超出,都可以;

③left=0,题目中有要求,1cm 都切不出来的情况,left 负数也可以的,但不大正常;

④left+1,如果填的是 left 的话,会导致死循环,left=4,right=5,mid 一直是 4;

⑤! isok(mid) (或者 isok(mid)==0),结合 isok()的 return 值判断,以及二分后的左右分区来分析。

【分析 2】

二分法,首先打擂台让 right 是最长值,

isok(t)就是测试一个长度中间值是否能切出 k 根,

num 是个累加器,每次先清 0, if(num>=k) break;

说明切到 k 根了,不需要再切了。

①不难得到 num+=len[i]/t;

②是 return 1;说明是 isok,切成功, num>=k,返回主程序,然后尝试让长度更长点。

③left=0 左边界,上面一句 right++是右侧;

④答案是 left+1<right, left<right 可以吗?

注意不能是 left<right,如 left=114,right=115,mid=115,就死循环了。

⑤成立, right=mid,下次 mid 会更小点,多切几根才能到 k;否则 mid 会更大,木棍可以更长点。isok(mid) 够切才会返回 1,而 right=mid,是没切成功,所以⑤isok(mid)==0。

7.【NOIP2006 普及组】(全排列)下面程序的功能是利用递归方法生成从 1 到 n(n<10)的 n 个数的全部可能的排列(不一定按升序输出)。例如,输入 3,则应该输出(每行输出 5 个排列):

123　132　213　231　321
312

【程序清单】

```
#include <iostream>
```

```
#include <cstdio>
#include <iomanip>
using namespace std;
int n,a[10];          //a[1],a[2],…,a[n]构成 n 个数的一个排列
long count=0;         //变量 count 记录不同排列的个数,这里用于控制换行
void perm(int k)
{
    int j,p,t;
    if(      ①      )
    {
        count++;
        for(p=1;p<=n;p++)
        cout <<setw(1)<<a[p];
        cout <<" ";
        if(      ②      )  cout <<endl;
        return;
    }
    for(j=k;j<=n;j++)
    {
        t=a[k];a[k]=a[j];a[j]=t;
        ③      ;
        t=a[k];      ④      ;
    }
}
int main()
{
    int i;
    cout <<"Entry n:"<<endl;
    cin >>n;
    for(i=1;i<=n;i++)  a[i]=i;
    ⑤      ;
    return 0;
}
```

【答案】① k==n（或 n==k） ② count%5==0 ③ perm(k+1)
④ a[k]=a[j];a[j]=t（分号可以用逗号代替） ⑤ perm(1)

【分析】DFS 实现全排列

①k==n,这段是 dfs 结束输出,底下有循环输出语句;

②count%5==0 送分,题目中的要求,5 个换一行;

③深搜的下一步 perm(k+1);

④a[k]=a[j];a[j]=t 在③的上面有交换的代码,深搜回溯是对称出现的,恢复操作,而且给了 t=a[k]的提示;

⑤perm (1),从第一个位置开始深搜;

应该还有从后往前的写法。

8.【NOIP2007 普及组】(求字符串的逆序)下面的程序的功能是输入若干行字符串,每输入一行,就按逆序输出该行,最后键入-1 终止程序。

【程序清单】

```
#include <iostream>
#include <cstdio>
#include <cstring>
using namespace std;
int maxline=200,kz;
int reverse(char s[])
{
    int i,j,t;
    for(i=0,j=strlen(s)-1; i<j;  ①  ,  ②  )
    {
        t=s[i]; s[i]=s[j]; s[j]=t;
    }
    return 0;
}
int  main()
{
    char line[100];
    cout<<"continue? -1 for end."<<endl;
    cin>>kz;
    while(  ③  )
    {
        cin>>line;
        ④  ;
        cout<<line<<endl;
        cout<<"continue? -1 for end."<<endl;
        cin>>kz;
    }
    return 0;
}
```

【答案】①i++　②j--　③kz!=-1　④reverse(line)

【分析】reverse 函数是头尾交换，循环从两头向内收拢，i,j 游标向中间，①②空还是比较简单的，③空通过最后一句 cin>>kz 以及题干中的要求可以推出，④函数调用，参数为数组名。

第2节　普及篇

1.【NOIP1998】

【问题描述】

输入一长度不超过80个字符的字符串(称为源串),该字符串由小写英文字母、空格组成,并以'.'结束。单词是由连续字母组成,两个单词之间至少有一个空格。

本程序的功能为:首先找出字符串中所有单词并保留一个空格作为单词分隔,存入数组ch中。然后用键盘输入一个待查找的单词,以字符'$'结束。采用顺序查找的方法在ch中进行查找,若找到,则输出该单词在ch中出现的序号(若有多个位置出现该单词,则只输出第一个序号位置)。若不存在,则输出'NOT FOUND'。

【程序清单】

```
#include<iostream>
#include<cstdio>
using namespace std;
char a[85],b[85],ch[85];
int i,j,k,n,m;
int main()
{
    n=0;
    do
    {
        ①      ; a[n]=getchar();
    }while(a[n]! ='.');
    k=0;
    for(i=1; i<=n;++i)
        if(a[i]>='a'&&a[i]<='z')
        {
            k++;
            ②      ;
        }else if(k! =0)
            if(ch[k]! ='')
            {
                k++; ch[k]='';
            }
        m=0;      ③      ;
    do
    {
        m++; b[m]=getchar();
    }while(      ④      );
    i=1; j=1; k=1; b[m]='';
    while(i<=n&&j<=m)
    {
        if(      ⑤      )
        {
```

```
            i++; j++;
        }
        else
        {
            while(ch[i]! ='')  ⑥  ;
            i=i+1; j=1; k++;
        }
    }
    if(  ⑦  )
    {
        printf("%d    ",k);
    }
    else printf("NOT FOUND");
    return 0;
}
```

【答案】①n=n+1　②ch[k]=a[i]　③n=k　④b[m]! ='$'
⑤ch[i]==b[j]　⑥i=i+1　⑦j>m-1

【分析】程序分成五大块，第一块是输入字符数组 a，每次下标自加 1(第 1 空)；第二块是将 a 数组压缩放入 ch 数组，要保证每个单词的右边只有一个空格，碰到是字母就放入 ch 中(第 2 空)，else 来到 k! =0 表示碰到的不是字母(可能是空格，也可能是“.”)，但已经开始数过字母了，且 ch[k]! ='' 表示之前放入 ch 的不是字母，说明现在可以放入一个空格，如果 ch[k]==''，说明已经放了空格就不要再放入多余的空格了，就能保证单词的右侧只有一个空格；n 重新设为 ch 的右端 k(第 3 空)，k 还要派其他用处；第三块是输入字符数组 b，第 4 空要判断输入有没有到'$'表示输入结束；第四块是逐位判断 ch 中有没有单词和 b 是完全匹配的，开始都置为首位 1，开始逐位比较，如果相等(第 5 空)，都前进一格(i++，j++)，如果当前不相等，说明 ch 当前的单词不能和 b 进行匹配，所以必须拿 ch 的下一个单词和 b 进行比对，第 6 空是排除当前还没比完的这个单词的剩余部分(第 6 空)，i=i+1是为了跳过空格；一旦比对成功，j 就大于 m，或者全部比完也没匹配，则 i 大于 n；第五块是判断输出，如果因为 j 达到了 m 或者超过了 m 才结束循环的(第 7 空)，那就是匹配的，k 就是在 ch 中匹配单词的首位。

2.【NOIP1998】FBZ 串问题

【问题描述】

已知一个由 0，1 字符组成的长度为 2^n 的字符串。请按以下规则将已给出的字符串分解为 FBZ 串：

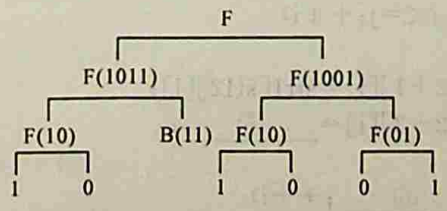
(1)若其中字符全为“1”，则称其为“B”串；

(2)若其中字符全为“0”，则称其为“Z”串；

(3)若不全为“0”，同时也不全为“1”，则称“F”串。若此串为 F 串，则应将此串分解为 2 个长为 2^{n-1} 的子串。

对分解后的子串，仍按以上规则继续分解，直到全部为 B 串或为 Z 串为止。

例如 n=3 时，给出 0-1 串为：“10111001”



【输出样例】

FFFBZBFFBZFZB

【问题求解】

给出01串，分解成FBZ串。

【程序清单】

```
#include<iostream>
#include<cstdio>
using namespace std;
const int n=8;
int i,j,st11,st12,st2,s,t;
char str1[n*2+1][n+1],str2[44];
int main()
{
    for(i=1; i<=n*2;++i)
    for(j=1; j<=n;++j) str1[i][j]='';
    st11=1; st12=1; st2=0;
    for(i=1; i<=n;++i) str1[1][i]=getchar();
    while(    ①    )
    {
        s=0; t=0;
        for(i=1; i<=n;++i)
        {
            if(str1[st12][i]=='1') s++;
            if(str1[st12][i]=='0') t++;
        }
        if(    ②    )
        {
            st2++; str2[st2]='B';
        }
        else if(    ③    )
        {
            st2++; str2[st2]='Z';
        }
        else
        {
            st2++; str2[st2]='F'; j=(s+t)/2;
            for(s=n*2-2; s>=    ④    ;--s)
              for(t=1; t<=n;++t)
              str1[s+2][t]=str1[s][t];
            st11+=2;
            for(int i=1; i<=j;++i)
            {
                str1[st12+1][i]=str1[st12][i];
                str1[st12+2][i]=    ⑤    ;
            }
            for(int i=    ⑥    ;++i)
```

```
        {
            strl[st12+1][i]=''; strl[st12+2][i]='';
        }
      }
      st12++;
    }
    for(int i=1;i<=st2;++i) printf("%c",str2[i]);
    return 0;
}
```

【答案】①st12<=st11　②t=0　③s=0　④st12+1　⑤strl[st12][i+j]　⑥j+1;n

【分析】本题如上图所示是一棵二叉树，最后的输出可理解为树的先序根遍历。st12 为被扩展结点指针，st11 为扩展结点指针。二叉树的遍历没有采取常见的递归算法，而采用循环来处理，循环结束的条件是没有结点可以扩展，即 st12>st11，因此①应填 st12<=st11。根据题意②③容易确定，应填 t=0 和 s=0。为了储存新产生的结点，程序采取了每扩展一个结点，产生它的两个儿子，就将当前结点一下行下推两行，将两个新结点存入空出的两行的方法。因此④应填 st12+1。

```
for(int i=1; i<=j;++i)
{
    strl[st12+1][i]=strl[st12][i];
    strl[st12+2][i]=    ⑤    ;
}
```

这段程序在填写新产生的两个结点数据，因此⑤应填 strl[st12][i+j];

```
for(int i=    ⑥    ;++i)
{
    strl[st12+1][i]=''; strl[st12+2][i]='';
}
```

这段程序的目的是让每行用不上的位置均添上空格，因此⑥应填 j+1;n。

3.【NOIP1999】

【问题描述】

用生成法求出 1,2,…,r 的全排列(r<=8)。

【程序说明】

用数组:a[r+1]表示排列；

初始化时，a[i]=1(i=1,2,….f)；

设中间的某一个排列为 a[1],a[2],…a[r]；

则求出下一个排列的算法为：

(1)从后面向前找，直到找到一个顺序为止(设下标为 j-1，则 a[j-1]<a[j])；

(2) 从 a[j]-a[r]中，找出一个 a[k]比 a[j-1]大的最小元素；

(3) 将 a[j-1]与 a[k]交换；

(4) 将 a[j],a[j+1]……a[r]由小到大排序。

【程序清单】

```
#include <iostream>
#include <cstdio>
using namespace std;
const int r=7;
int n, i, s, k, j, il, t, a[r+1];
void print1()
```

```
{
    int ik;
    for (ik=1; ik <=r;++ik) cout<<"          "<<a[ik];
    cout<<endl;
}
int main()
{
    for (i=1; i <=r;++i)    ①    ;
    print1();
    s=1;
    for (i=2; i <=r;++i) s=s * i;
    s=s-1;
    for (i=    ②    ;++i)
    {
        j=r;
        while (    ③    ) j=j-1;
        k=j;
        for (i1=j+1; i1 <=r;++i1)
          if (    ④    ) k=i1;
        t=a[j-1]; a[j-1]=a[k]; a[k]=t;
        for (i1=j; i1 <=r-1;++i1)
          for (k=i1+1; k <=r;++k)
            if (    ⑤    )
            {
            t=a[i1]; a[i1]=a[k]; a[k]=t;
            }
        print1();
    }
    return 0;
}
```

【答案】①a[i]=i　②1; i <=s　③a[j-1] > a[j]　④a[i1] > a[j-1] && a[i1] < a[k]　⑤a[i1] > a[k]

【分析】首先举例说明全排列算法。

以长度为 7 的数列为例：

a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]
5	2	3	7	4	2	1

(1) 从右向左找第一个左比右小的位置，因 a[3]<a[4]，记位置 3；

(2) 从位置 4 到位置 7 找比 a[3]大的元素位置，记位置 5；

(3) 将(1)找到的位置 3 和(2)找到的位置 5 互换；

a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]
5	2	4	7	3	2	1

(4) 将位置 4 到位置 7 的元素由小到大排序；

a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]
5	2	4	1	2	3	7

这就是生成的新排列。

下面分段说明程序的功能。

程序段一：

```
void print1()
{
        int ik;
        for (ik=1; ik <=r;++ik) cout<<"        "<<a[ik];
    cout<<endl;
}
```

这是一个输出过程，其作用是输出数组 a。

程序段二：

```
for (i=1; i <=r;++i)  ①  ; //此处应填 a[i]=i,输出第一种排列 1,2,3,...,r
print1();
```

程序段三：

```
s=1;
for (i=2; i <=r;++i) s=s * i;
s=s-1;
```

由排列组合知识可知，共有排列 r! 组，本段程序采用累乘算法计算 r!，程序段三输出了一组，还有 r! -1 组。

程序段四：

```
for (i=  ②  ;++i) //因为总共生成 r! -1 组排列,此处应填 1; i <=s
{
    j=r;
    while (  ③  ) j=j-1; //从右至左找到第一个右大于左的情况,其位置记为 j,此处应填 a[j-1]>a[j]
    k=j;
    for (i1=j+1; i1 <=r;++i1)
      if (  ④  ) k=i1; //在 a[j]右边寻找最小的大于 a[j-1]的元素,其位置记为 k,此处应填 a[i1] > a[j-1] && a[i1] < a[k],左边保持大于 a[j-1],右边保证最小。
    t=a[j-1]; a[j-1]=a[k]; a[k]=t;
    for (i1=j; i1 <=r-1;++i1) //用选择排序算法将第 j 个元素至第 r 个元素从小到大排序
  for (k=i1+1; k <=r;++k)
      if (  ⑤  )     //此处应填 a[i1]>a[k]
      {
          t=a[i1]; a[i1]=a[k]; a[k]=t;
      }
    print1();
}
```

本段程序用全排列算法生成并输出 r! -1 组排列。

4.【NOIP2000】多项式的乘法

【问题描述】

例如有如下多项式：

$P(X)=2X^2-X+1$，　　$Q(X)=X+1$

则：

$$P(X)\cdot Q(X)=(2X^2-X+1)(X+1)=2X^3+X^2+1$$

【程序说明】

多项式的表示：系数、指数。

如上例中：

P(X)：	系数	指数	Q(X)	系数	指数
	2	2		1	1
	−1	1		1	0
	1	0		0	0
	0	0			

P＊Q 的结果存入 C 中。其输出格式是：依次用一对括号内的(系数,指数)分别来表示。如上例的输出结果表示为：(2,3)(1,2)(1,0)。

【程序清单】

```
#include<iostream>
#include<cstdio>
#include<cmath>
using namespace std;
int p[15][5],q[15][5],c[25][5];
int main()
{
   int i,j,k,l,jp,jq,jc,x,y,x1,y1;
   jp=0;
   cin>>x>>y;
   while (x! =0)
   {
      jp++; p[jp][1]=x;  p[jp][2]=y; cin>>x>>y;
   }
   jq=0;
   cin>>x>>y;
   while (x! =0)
   {
      jq++; q[jq][1]=x;  q[jq][2]=y; cin>>x>>y;
   }
   jc=1;  c[jc][1]=0;  c[jc][2]=-1000;
   for(i=1;i<=jp;i++)
   {
      ____①____;
      y=p[i][2];
      for(j=1;j<=jq;j++)
      {
         ____②____;
         y1=y+q[j][2];
         k=1;
         while (y1<c[k][2]) k++;
         if (y1==c[k][2]) ____③____;
         else {
              for(l=jc;l>=k;l--)
```

```
                {
                    c[l+1][1]=c[l][1];
                    c[l+1][2]=c[l][2];
                }
                c[k][1]=x1; c[k][2]=y1;
                ④ ____;
            }
        }
    }
    for(int i=1;i<=jc;i++)
        if ____⑤____ cout<< '(' << c[i][1] << ',' << c[i][2] << ')' ;
    return 0;
}
```

【答案】①x=p[i][1]　②x1=x * q[j][1]　③c[k][1]=c[k][1]+x1
④jc=jc+1 或 jc++　⑤(c)[i][1]! =0

【分析】两个多项式 p 和 q 进行乘法操作，把结果存入 c。通过程序说明部分可以看出，输入多项式的每一项系数和指数，至两数都为 0 结束，所以程序开始的部分就是输入两个多项式 p 和 q。接下来是做乘法操作放入 c，先假定 c 的最高次项的系数和指数分别为 0 和 -1000，方便之后将结果存入指定的位置。穷举 p 的每一个有效位，取出第 i 位的系数和指数(x 和 y，其中第 1 空是取系数，则 x=p[i][1])，外循环内部穷举 q 的每一个有效位，与 p 的第 i 位做乘法，其中系数做乘法、指数做加法(第 2 空是系数做乘法，则 x1=x * q[j][1])；将乘法的结果，包括系数和指数放入 c 中合适的位置(从低位到高位寻找)，如果指数正好相等，则将系数加入其中(第 3 空为 c[k][1]=c[k][1]+x1)；如果需放入的那一项的指数在 c 中不存在，则进行一个移位操作，即将 c 的后续位置统一右移一位，则将需放入的一项放入，同时 c 的实际长度要增加 1 位(第 4 空为 jc++)；在运算过程中，系数经过不断的加法可能为 0，则该项不需要输出，所以，第 5 空应该是(c)[i][1]! =0。

5.【NOIP2004 普及组】Joseph 问题

【问题描述】

原始的 Joseph 问题的描述如下：有 n 个人围坐在一个圆桌周围，把这 n 个人依次编号为 1，…，n。从编号是 1 的人开始报数，数到第 m 个人出列，然后从出列的下一个人重新开始报数，数到第 m 个人又出列，…，如此反复直到所有的人全部出列为止。比如当 n=6，m=5 的时候，出列的顺序依次是 5，4，6，2，3，1。

现在的问题是：假设有 k 个好人和 k 个坏人。好人的编号的 1 到 k，坏人的编号是 k+1到 2k。我们希望求出 m 的最小值，使得最先出列的 k 个人都是坏人。

【输入格式】

仅有的一个数字是 k(0 < k <14)。

【输出格式】

使得最先出列的 k 个人都是坏人的 m 的最小值。

【输入样例】

4

【输出样例】

30

【程序清单】

```
#include <iostream>
#include <cstdio>
using namespace std;
```

```
long k, m, begin;
int check(long remain){
    long result=(    ①    ) % remain;
    if (    ②    ){
        begin=result; return 1;
    }
else return 0;
}
int main(){
    long i, find=0;
    scanf("%ld", &k);
    m=k;
    while(    ③    ) {
        find=1; begin=0;
        for (i=0; i < k; i++)
            if (! check(    ④    )){
                find=0; break;
            }
        m++;
    }
    printf("%ld\n",    ⑤    );
    return 0;
}
```

【答案】① begin+m-1　② result>=k（或者 k<=result）　③ ! find（或者 find==0）　④ 2*k-i　　⑤ m-1

【分析】m=k，m++；这个 m 就是题目所求的值，从 k 开始，圈是从 0 开始。m 从 k 开始穷举，while 中的 for 循环是测试所有的坏人是否能通过数 m 全部踢出去，如果不行，出现好人被踢出去，直接 break，说明 find==0 就是没找到，find==1 就是找到。

③find==0，而④应该是依次测试数 m 会不会把好人踢出去，函数中形参名称为 remain 是个提示，剩下的人数；

④应该是依次测试中每次剩下的人数的 2*k-i。check 函数中的 begin 是全局变量，用来记录数 m 后光标的位置，result 中的%就说明是绕圈后的结果位置，if()成立后 begin=result; return 1; 说明坏人出局，对应④中的! check()，所以②应该是 result>=k。

①的位置是 begin+m，由于是从 0 开始，①填入 begin+m-1。

6.【NOIP2006 普及组】由键盘输入一个奇数 P（P<100,000,000），其个位数字不是 5，求一个整数 S，使 P*S=1111...1（在给定的条件下，解 S 必存在）。要求在屏幕上依次输出以下结果：

(1)S 的全部数字。除最后一行外，每行输出 50 位数字。　(2)乘积的数字位数。

例 1：输入 p=13，由于 13*8547=111111，则应输出(1)8547　(2)6

例 2：输入 p=147，则输出结果应为(1)7558578987150415721844293272864701436 13
(2)42，即等式的右端有 42 个 1。

【程序说明】

```
#include <iostream>
#include <cstdio>
#include <iomanip>
```

```
using namespace std;
int main()
{
    long p,a,b,c,t,n;
    while (1)
    {
        cout <<"输入 p，最后一位为 1 或 3 或 7 或 9:"<<endl;
        cin >>p;
        if ((p%2! =0)&&(p%5! =0))  // 如果输入的数符合要求,结束循环
        ①    ;
    }
    a=0; n=0;
    while (a<p)
    {
      a=a*10+1; n++; // 变量 a 存放部分右端项,n 为右端项的位数
    }
    t=0;
      do
    {
          b=a/p;
          cout <<setw(1)<<b;
          t++;
          if (    ②    )
            cout <<endl;
          c=    ③    ;  a=    ④    ; n++;
    } while(c)>0;
    cout<<endl<<"n="<<    ⑤    <<endl;
    return 0;
}
```

【答案】①break　②t%50==0　③a-p*b（或 a-b*p）
④c*10+1（或 10*c+1）　⑤--n

【分析】高精度除法

①break 上一行和题目要求有提示，保证输入正确；

②下面有个 count<<endl 根据题目要求，50 个一行，上面有个计数器 t++，得知 t%50==0；

③需要自己列个除法竖式模拟一下，a 是被除数 111……111，b 是商，p 是除数，c=a-b*p；

④余数继续放大 10 倍，尾巴上再添 1，a=c*10+1；

⑤n++在循环内，会多增一次，--n。

7.【NOIP2006 提高组】（选排列）下面程序的功能是利用递归方法生成从 1 到 n(n<10)的 n 个数中取 k(1<=k<=n)个数的全部可能的排列（不一定按升序输出）。例如，当 n=3，k=2 时，应该输出（每行输出 5 个排列）：

12　13　21　23　32

31

【程序说明】

```
#include <iostream>
```

```
#include <cstdio>
using namespace std;
int n,k,a[10];
long count=0;
void perm2(int j)
{
    int i,p,t;
    if(    ①    )
    {
        for(i=k;i<=n;i++)
        {
            count++;
            t=a[k]; a[k]=a[i]; a[i]=t;
            for(    ②    )
            printf("%1d",a[p]);  /* "%1d"中是数字 1,不是字母 l */
            printf("  ");
            t=a[k];a[k]=a[i];a[i]=t;
            if(count%5==0) printf("\n");
        }
        return;
    }
    for(i=j;i<=n;i++)
    {
        t=a[j];a[j]=a[i];a[i]=t;
        ③    ;
        t=a[j];    ④    ;
    }
}
main()
{
    int i;
    printf("\nEntry n,k (k<=n):\n");
    scanf("%d%d",&n,&k);
    for(i=1;i<=n;i++)  a[i]=i;
    ⑤    ;
    return 0;
}
```

【答案】①j==k（或 k==j）　②p=1;p<=k;p++　③perm2(j+1)
④a[j]=a[i];a[i]=t　⑤perm2(1)

【分析】①j==k（或 k==j），试到最后，成功的标识；
②p=1;p<=k;p++，输出一种排列；
③perm2(j+1)，尝试下一个排列；
④a[j]=a[i];a[i]=t，回溯的标识；
⑤perm2(1)，从第一个位置开始深搜。

8.【NOIP2007 提高组】格雷码 Gray Code

格雷码是对十进制数的一种二进制编码。编码顺序与相应的十进制数的大小不一

致。其特点是：对于两个相邻的十进制数，对应的两个格雷码只有一个二进制位不同。另外，最大数与最小数之间也仅有一个二进制位不同，以 4 位二进制数为例，编码如下：

十进制数	格雷码	十进制数	格雷码
0	0000	8	1100
1	0001	9	1101
2	0011	10	1111
3	0010	11	1110
4	0110	12	1010
5	0111	13	1011
6	0101	14	1001
7	0100	15	1000

如果把每个二进制的位看作一个开关，则将一个数变为相邻的另一个数，只须改动一个开关。因此，格雷码广泛用于信号处理、数一模转换等领域。

下面程序的任务是：由键盘输入二进制数的位数 n（n<16），再输入一个十进制数 m（0≤m<2n），然后输出对应于 m 的格雷码（共 n 位，用数组 gr[]存放）。为了将程序补充完整，你必须认真分析上表的规律，特别是对格雷码固定的某一位，从哪个十进制数起，由 0 变为 1，或由 1 变为 0。

【程序说明】

```
#include <iostream>
#include <cstdio>
using namespace std;
int main()
{
    int bound=1,m,n,i,j,b,p,gr[15];
    printf("input n,m\n");
    scanf("%d%d",&n,&m);
    for(i=1;i<=n;i++) bound=    ①    ;
    if(m<0||m>=bound)
    {
        printf("Data error! \n");
        ②    ;
    }
    b=1;
    for(i=1;i<=n;i++)
    {
        p=0; b=b*2;
        for(    ③    ;j<=m;j++)
          if(    ④    )
            p=1-p;
        gr[i]=p;
    }
    for(i=n;    ⑤    )
    printf("%1d",gr[i]);  //在"%1d"中出现的是数字 1，不是字母 l
    printf("\n");
```

```
    return 0;
}
```

【答案】①bound * 2　②return 或 exit(0)　③j=0
④(j%b-(b/2))==0　⑤i>=1;i--或　i>0;i--

【分析】按照题目后面的提示，针对示例中各个为的 0/1 取反，可以找到如下规律：

第一位上从十进制数 $1(2^{1-1})$ 开始进行 0/1 取反，以后每过 $2(2^1)$ 个数进行 0/1 取反；

第二位上从十进制数 $2(2^{2-1})$ 开始进行 0/1 取反，以后每过 $4(2^2)$ 个数进行 0/1 取反；

第三位上从十进制数 $4(2^{3-1})$ 开始进行 0/1 取反，以后每过 $8(2^3)$ 个数进行 0/1 取反；

依此类推，第 i 位上从十进制 2^{i-1} 开始进行 0/1 取反，以后每过 2^i 个数进行 0/1 取反。

① bound * 2，比较容易，底下一行就有对 m 的检验，bound 是为上界，题干中有明确的要求 2^n；

② return 0，错误输入直接退出；

③ j=0，格雷码是从 0 开始的，到 m 结束；

④ j%b==b/2，前面的 $b=2^i$，为该位变化的间距，因为 i 位上第一个 0/1 转换是从值 2^i-1 的十进制开始的，所以 j=b/2；

⑤ i>=1;i--，输出 gr[]，上面生成格雷码时循环 i 从 1-n。n+1 位格雷码的集合=n 位格雷码集合(顺序)加前缀 0+n 位格雷码集合(逆序)加前缀 1。

第3节　提高篇

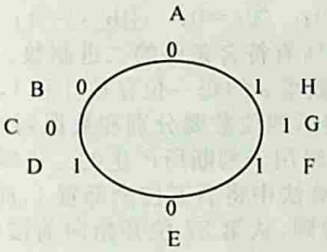
1.【NOIP2000】

【问题描述】

将 2^n 个 0 和 2^n 个 1，排成一圈。从任一个位置开始，每次按逆时针的方向以长度为 n+1 的单位进行数二进制数。

要求给出一种排法，用上面的方法产生出来的 2^{n+1} 个二进制数都不相同。

例如，当 n=2 时，即 2^2 个 0 和 2^2 个 1 排成如下一圈：



比如，从 A 位置开始，逆时针方向取三个数 000，然后再从 B 位置上开始取三个数 001，接着从 C 开始取三个数 010……可以得到 000,001,010,101,011,111,110,100 共 8 个二进制数且都不相同。

【程序说明】

以 n=4 为例，即有 16 个 0,16 个 1，

数组 a 用以记录 32 个 0,1 的排法，

数组 b 统计二进制数是否已出现过。

【程序清单】

```
#include<iostream>
#include<cstdio>
#include<cmath>
using namespace std;
int a[37],b[32],i,j,k,s,p;
int main()
{
    for(i=1;i<=36;i++) a[i]=0;
    for(i=28;i<=32;i++) a[i]=1;
    p=1; a[6]=1;
    while (p==1)
    {
       j=27;
       while(a[j]==1) j--;
       ①    ;
       for(i=j+1;i<=27;i++)    ②
       for(i=0;i<=31;i++) b[i]=0;
       for(i=1;i<=32;i++)
       {
          ③
          for(k=i;k<=i+4;k++) s=s*2+a[k];
          ④
```

```
    }
    s=0;
    for(i=0;i<=31;i++) s+=b[i];
    if ____⑤____ p=0;
  }
  for(i=1;i<=32;i++)
  for(j=i;j<=i+4;j++) cout<<a[j];
  cout<<endl;
  return 0;
}
```

【答案】①a[j]=1　②a[i]=0;　③s=0;　④b[s]=1;　⑤(s==32)

【分析】本题采用穷举法输出所有符合条件的二进制数。具体算法如下：

(1)数据结构的处理：用数组 a 的每一位存放 0 和 1，当 n=4 时，有 16 个 0 和 16 个 1，由于数据是环行表示，所以最后四位数要分别和数据列的前四位形成新的二进制数，因此数组 a 要定义为：a[37]，b 数组用于判断所产生的二进制数是否重复；

(2)为了减少循环次数，算法中将后五位全部置 1，前五位全部置 0；

(3)用二进制加法运算规则，从第 27 位开始向前按位搜索，如果为 1 则继续向前，否则将当前位置置为 1，从当前位置的后一位到第 27 位全置为 0(二进制加法运算)。则有：①为 a[j]=1；②为 a[i]=0；

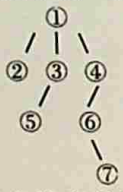
(4)扫描 32 位二进制数，顺序将每五位二进制数转换为十进制数，并在数组 b 中做出标识。则有④为 b[s]=1；

(5)判断是否产生 2n+1 个不同的二进制数，即 32 个二进制数，如果没有达到 32 个则重复以上(3)(4)(5)过程，否则置 p 为 0，退出循环。

2.【NOIP2000】

【问题描述】

求出一棵树的深度和宽度。例如有如下的一棵树：



其树的深度为从根结点开始到叶结点结束的最大深度，树的宽度为同一层上结点数的最大值。在上图中树的深度为 4，宽度为 3。

用邻接表来表示树，上图中的树的邻接表见表 1。

【程序说明】

数组 tree 表示树，用邻接表来表示(假设树的度为 4)；

数组 q 表示队列，其中 SP1——取出指针，SP2——存入指针，q[i,0]表示层数；

数组 d，统计同一层上的结点数(假设≤20 层)。

表 1

1	2	3	4	0	0
2	0	0	0	0	0
3	5	0	0	0	0
4	6	0	0	0	0
5	0	0	0	0	0
6	7	0	0	0	0
7	0	0	0	0	0

【程序清单】

```
#include <iostream>
#include <cstdio>
using namespace std;
int i,j,sp1,sp2,l;
int tree[25][10],q[105][10],d[25];
int main()
{
   for(i=1;i<=14;i++) for(j=1;j<=6;j++) tree[i][j]=0;
   for(i=1;i<=14;i++) tree[i][1]=i;
   tree[1][2]=2;  tree[1][3]=3;  tree[1][4]=4; tree[2][2]=5;  tree[2][3]=6;
   tree[3][2]=7;  tree[3][3]=8;  tree[4][2]=9; tree[4][3]=10; tree[4][4]
   =11;
   tree[7][2]=12; tree[7][3]=13; tree[13][2]=14;
   sp1=1; sp2=1;
   for(i=1;i<=6;i++) q[1][i]=tree[1][i];
   q[1][0]=1;
   while ____①____
   {
      l=____②____;  j=2;
      while ____③____
      {
         sp2++; q[sp2][0]=l; q[sp2][1]=q[sp1][j];
         for(i=2;i<=6;i++) q[sp2][i]=tree[q[sp1][j]][i];
         j++;
      }
      sp1++;
   }
   cout<<____④____<<endl;
   for(i=0;i<=20;i++) d[i]=0;
   for(i=1;i<=sp2;i++) d[q[i][0]]=____⑤____;
   int max=d[1];
   for(i=2;i<=20;i++)
   if(d)[i]>max max=d[i];
   cout<<max<<endl;
   return 0;
}
```

【答案】①(sp1<=sp2)　②q[sp1][0]+1　③(q[sp1][j]! =0)
④q[sp2][0]　⑤d[q[i][0]]+1

【分析】该题的基本算法为：

(1)设置数据的存储结构，用数组 tree 表示树的存储结构－>邻接表；程序中的树设定为 14 个结点，每个结点最多 4 个孩子，第 5 个孩子为 0，用于判断某层的结束。

(2)设 q 数组表示队列，其中 sp1 表示出队的指针，sp2 表示入队的指针，q[i][0]表示层数。

(3)d 数组统计同一层上的结点数，以便求得宽度。

(4)初始化 tree[i][j]=0；将树中的 14 个结点赋初值：

for(i=1;i<=14;i++) tree[i][1]=i;

(5)将树的结构用二维数组邻接形式存储,其树的结构如上图:

tree[1][2]=2; tree[1][3]=3; tree[1][4]=4; tree[2][2]=5;

tree[2][3]=6; tree[3][2]=7; tree[3][3]=8; tree[4][2]=9;

tree[4][3]=10; tree[4][4]=11; tree[7][2]=12; tree[7][3]=13;

tree[13][2]=14;

(6)将根结点及其他的后继结点——孩子结点入队,队列指针指向队首:sp1=1; sp2=1; 并且记录指针指向树的第一层:q[1][0]=1,(用 q[i][0]表示层数)。

(7)用双重循环实现"树的深度搜索":

(a)取出一个结点,层数加 1 即有②q[sp1][0]+1,如果后继结点不为 0,则将后继结点入队,即有③(q[sp1][j]! =0);

(b)进队操作:

sp2++; q[sp2][0]=l; q[sp2][1]=q[sp1][j];

for(i=2;i<=6;i++) q[sp2][i]=tree[q[sp1][j]][i];

j++;

(c)再取当前队列的首位(出队),sp1++,重复 a,b,c 过程;

(d)输出深度,④为 cout<<1<<endl 或 cout<<q[sp2][0]<<endl;

(e)重复直到队列为空,①为 sp1<=sp2;

(8)用循环语句完成树的宽度的搜索,⑤为 d[q[i][0]]+1;

(9)用循环语句寻找层次中,最多孩子的结点数。

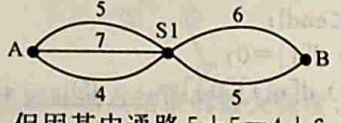
3.【NOIP2001】

【问题描述】

在 A,B 两个城市之间设有 N 个路站(如下图中的 S1,且 N<100),城市与路站之间、路站和路站之间各有若干条路段(各路段数≤20,且每条路段上的距离均为一个整数)。

A,B 的一条通路是指:从 A 出发,可经过任一路段到达 S1,再从 S1 出发经过任一路段,…,最后到达 B。通路上路段距离之和称为通路距离(最大距离≤1000)。当所有的路段距离给出之后,求出所有不同距离的通路个数(相同距离仅记一次)。

例如:下图所示是当 N=1 时的情况:



从 A 到 B 的通路条数为 6,但因其中通路 5+5=4+6,所以满足条件的不同距离的通路条数为 5。

【算法说明】

本题采用穷举算法。

【程序说明】

N:记录 A,B 间路站的个数

数组 D[I][0]记录第 I-1 到第 I 路站间路段的个数

D[I][1],D[I][2],…记录每个路段距离

数组 G 记录可取到的距离

【程序清单】

```
#include <iostream>
#include <cstdio>
using namespace std;
int i, j, n, s, b[101], d[101][21];
bool g[1001];
```

```
int main()
{
    scanf("%d", &n);
    for (i=1; i<=n+1; i++){
        scanf("%d", &d[i][0]);
        for (j=1; j<=d[i][0]; j++) scanf("%d", &d[i][j]);
    }
    d[0][0]=1;
    for (i=1; i<=n+1; i++) b[i]=1;
    b[0]=0;
    for (i=0; i<=1000; i++) g[i]=0;
    while (    ①    ){
        s=0;
        for (i=1; i<=n+1; i++)
            s=    ②    ;
        g[s]=1; j=n+1;
        while (    ③    ) j--;
        b[j]++;
        for (i=j+1; i<=n+1; i++) b[i]=1;
    }
    s=0;
    for (i=1; i<=1000; i++)    ④    ;
    printf("%d\n", s);
    return 0;
}
```

【答案】① b[0]==0　② s+d[i][b[i]];　③ b[j]==d[j][0]　④ s=s+g[i]

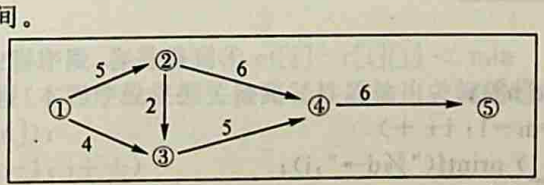
【分析】本题采用回溯穷举法得到所有的可能情况。相邻两个路站或者城市之间有多个路段，从 1 ～ D[I][0]不等，开始都设 b[i]的初值为 1，即所有都采用第一个路段作为初始状态，统计路径长度总和(第 2 空就是求和 s+d[i][b[i]])并做标记，然后从右往左看哪几个取到了最后一个路段(路段数量的最大值，即第 3 空 b[j]==d[j][0])，到某个未达最大值的地方停下来，该位置的路段取编号大 1 的路段，后续的全都设为第 1 个路段，继续大循环，循环条件就是第 1 空 b[0]==0，即如果不是所有位置都达到最后一个路段(即路段数量的最大值)，继续做循环；最后统计 g 数组中的出现了多少个被打过标记的点就是答案(第 4 空 s=s+g[i])。

4.【NOIP2001】求关键路径

【问题描述】

设有一个工程网络如下图表示(无环路的有向图)：

其中，顶点表示活动，①表示工程开始，⑤表示工程结束(可变，用 N 表示)，边上的数字表示活动延续的时间。



如上图中，活动①开始 5 天后活动②才能开始工作，而活动③则要等①、②完成之后才能开始，即最早也要 7 天后才能工作。

在工程网络中，延续时间最长的路径称为关键路径。上图中的关键路径为：①—②—

③—④—⑤共 18 天完成。

【程序说明】

关键路径的算法如下：

r[1..n][1..n]；　　表示活动的延续时间，若无连线，则用－1 表示；

eet[1..n]　　表示活动最早可以开始的时间；

et[1..n]　　表示活动最迟应该开始的时间；

关键路径通过点 j，具有如下的性质：eet[j]＝et[j]。

【约定】

结点的排列已经过拓扑排序，即序号前面的结点会影响序号后面结点的活动。

【程序清单】

```
#include <iostream>
#include <cstdio>
using namespace std;
int i, j, n, max, min, w, x, y, r[21][21], eet[21], et[21];
int main(){
    scanf("%d", &n);
    for (i=1; i<=n; i++)
      for (j=1; j<=n; j++)
        r[i][j]=-1;
    scanf("%d%d%d", &x, &y ,&w);
    /*输入从活动 x 到活动 y 的延续时间，以 0 为结束*/
    while (x !=0){
        r[x][y]=w; ____①____
    }
    eet[1]=0;
    for (i=2; i<=n; i++){
        max =0;
        for (i=1; i<=n; i++)
          if (r[j][i] !=-1)
            if (____②____) max=r[j][i]+eet[j];
        eet[i]=max;
    }
    ③____;
    for (i=n-1; i>=1; i--){
        min=10000;
        for (j=1; j<=n; j++)
          if (r[i][j] !=-1)
            if (____④____) min=et[j]-r[i][j];
        et[i]=min;
    }
    printf("%d",eet[n]);
    for (i=1; i<=n-1; i++)
      if (____⑤____) printf("%d→",i);
    printf("%d",n);
    return 0;
}
```

【答案】①scanf("%d%d%d", &x, &y, &w);　②r[j][i]+eet[j] > max
③et[n]=eet[n];　④et[j] - r[i][j] < min　⑤eet[i]==et[i]

【分析】由题目可知，关键路径通过j，具有eet[j]=et[j]，那么只需计算eet[j]，et[i](1<=i<=n)，就能很快得到关键路径所通过的节点，也就能得到关键路径。在所在程序中很容易找到计算eet[i]，et[i]的程序块，问题也就很容易解决，下面我们分段分析所给程序。

第一段：初始化程序段(本程序初始化数组r并输入数据)

```
scanf("%d", &n); //输入节点个数 n
for (i=1; i<=n; i++)
  for (j=1; j<=n; j++)
    r[i][j]=-1; //初始化数组 r，值-1 表示节点 i 到节点 j 没有先后活动关系
scanf("%d%d%d", &x, &y ,&w); //输入从活动 x 到活动 y 的延续时间，以 0 为结束
while (x ! =0){
    r[x][y]=w; ____①____
    //根据上面题目所给提示："以 0 为结束"可知循环结束取决于录入数据，也就是说循环体中要有输入语句。容易判断①scanf("%d%d%d", &x, &y ,&w);
}
```

第二段：处理程序段1(本程序段每个工程的最晚开始时间)

```
eet[1]=0; //认为工程从第 0 天开始
for (i=2; i<=n; i++){
    max =0;
      for (i=1; i<=n; i++)
        if (r[j][i] ! =-1)
          if (  ____②____  ) max=r[j][i]+eet[j];
      eet[i]=max;
    }
    //因为是求最值的程序段，容易得到② r[j][i]+eet[j] > max
```

第三段：处理程序2(本程序计算每个工程最早开始时间)

```
  ____③____
//因为下面的程序要计算 et，循环变量是从 n-1 到 1，易知 et[n]要被初始化，③应填 et[n]=eet[n];
for (i=n-1; i>=1; i--){
   min=10000;
   for (j=1; j<=n; j++)
     if (r[i][j] ! =-1)
       if (  ____④____  ) min=et[j]-r[i][j];
   et[i]=min;
}
//因为是求最值的程序段，容易得到④ et[j]-r[i][j] < min
```

第四段：输出程序段(本程序段根据关键路径性质输出关键路径)

```
printf("%d",eet[n]);
for (i=1; i<=n-1; i++)
  if (  ____⑤____  ) printf("%d→",i);
printf("%d",n);
//满足关键路径性质的结点才被输出，易知⑤为关键路径性质，应填 eet[i]=et[i]
```

5.【NOIP2002】

【问题描述】

将 n 个整数分成 k 组(k≤n,要求每组不能为空),显然这 k 个部分均可得到一个各自的和 $S_1,S_2,\cdots,S_k$,定义整数 P 为:

$$P=(S_1-S_2)^2+(S_1-S_3)^2+\cdots+(S_1-S_k)^2+(S_2-S_3)^2+\cdots+(S_{k-1}-S_k)^2$$

【问题求解】

求出一种分法,使 P 为最小(若有多种方案仅记一种)。

【程序说明】

数组:a[1],a[2],...a[n]存放原数;
s[1],s[2],...,s[k]存放每个部分的和;
b[1],b[2],...,b[n]穷举用临时空间;
d[1],d[2],...,d[n]存放最佳方案。

【程序清单】

```
#include <iostream>
#include <cstdio>
using namespace std;
int a[101], b[101], d[101], s[31], i, j, k, n, sum, cmin;
int main()
{
    scanf("%d %d", &n, &k);
    for (i=1; i<=n;++i) scanf("%d", &a[i]);
    for (i=1; i<=n;++i) b[i]=1;
    cmin=1000000;
    while (b[0]==1)
    {
        for (i=1; i<=k;++i) ____①____
        for (i=1; i<=n;++i) ____②____
        sum=0;
        for (i=1; i<=k-1;++i)
            for ____③____ sum=sum+(s[i]-s[j]) * (s[i]-s[j]);
        if ____④____
        {
            cmin=sum;
            for (i=1; i<=n;++i) d[i]=b[i];
        }
        j=n;
        while ____⑤____ j=j-1;
        b[j]=b[j]+1;
        for (i=j+1; i<=n;++i) ____⑥____
    }
    printf("%d\n", cmin);
    for (i=1; i<=n;++i) printf("%40d", d[i]);
    printf("\n");
    return 0;
}
```

【答案】①s[i]=0; ②s[b[i]]=s[b[i]]+a[i]; ③(j=i+1; j<=k;++j)

④(cmin ＞ sum)　⑤(b[j]＝＝k)　⑥ b[i]＝1;

【分析】本题采用回溯穷举法得到所有的可能情况，从中选择最小值保留，最后输出该方案。a 数组存储原始数据，b 数组记录每个数所在小组的编号，比如 b[3]＝5 表示 a[3]属于第 5 小组；在穷举过程中假设每个数开始都是属于第 1 小组的，即 b[i]＝1，利用逐位递增法穷举每个数所在的小组从 1 ～ k，如果所有的数所在的小组都属于第 k 小组之后，说明所有情况已经穷举完毕。

对于每种情况需要把每组的和 s 数组初始化为 0(第 1 空 s[i]＝0;)；将每个数分别加到各自所在 s[i]小组内(第 2 空 s[b[i]]＝s[b[i]]＋a[i];)；利用双重循环求得 s 数组两两之间差平方的总和，为了不重复，第 3 空应为(j＝i＋1; j ＜＝k;＋＋j)；需要拿此时的总和 sum 和全局最小值 cmin 比较(第 4 空(cmin ＞ sum))并刷新方案；获得下一组穷举方案的方案是：按照原始数据序号从大到小寻找第一个数不属于第 k 组(第 5 空(b[j]＝＝k))，则将该位置 j 所在的小组编号加 1，比该位置 j 大的位置全部重新改为第 1 小组(第 6 空 b[i]＝1;)，继续大循环；最后输出最小方案。题中所说“要求每组不能为空”，其实数据肯定是拆解的份数越多，所求的差平方总和越小，所以，最后方案肯定是所有的第 1 ～ k 组都是有数据的。

6.【NOIP2002】

【问题描述】

有 n 种基本物质(n≤10)，分别记为 P_1，P_2，…，P_n，用 n 种基本物质构造物质，这些物品使用在 k 个不同地区(k≤20)，每个地区对物品提出自己的要求，这些要求用一个 n 位的数表示：$a_1a_2\cdots a_n$，其中：

ai ＝1	表示所需物质中必须有第 i 种基本物质
＝－1	表示所需物质中必须不能有第 i 种基本物质
＝0	无所谓

【问题求解】

当 k 个不同要求给出之后，给出一种方案，指出哪些物质被使用，哪些物质不被使用。

【程序说明】

数组	b[1],b[2]……b[n]	表示某种物质
	a[1..k][1..n]	记录 k 个地区对物品的要求，其中：
	a[i][j]＝1	表示第 i 个地区对第 j 种物品是需要的
	a[i][j]＝0	表示第 i 个地区对第 j 种物品是无所谓的
	a[i][j]＝－1	表示第 i 个地区对第 j 种物品是不需要的

【程序清单】

```
#include <iostream>
#include <cstdio>
using namespace std;
int a[21][11], b[21], i, j, k, n;
bool p;
int main()
{
   scanf("%d %d", &n, &k);
   for (i=1; i<=k;++i) for (j=1; j<=n;++j) scanf("%d\n", a[i][j]);
   for (i=0; i<=n;++i) b[i]=0;
   p=true;
   while (    ①    )
   {
      j=n;
```

```
    while (b[j]==1) j=j-1;
    ②
    for (i=j+1; i<=n;++i) b[i]=0;
    ③
    for (i=1; i<=k;++i)
        for (j=1; j<=n;++j)
            if (a[i][j]==1 && b[j]==0 || ____④____) p=true;
}
if (____⑤____) printf("找不到! \n");
else for (i=1; i<=n;++i)
    if (b[i]==1) printf("物质%d需要\n", &i);
    else  printf("物质%d不需要\n", &i);
return 0;
}
```

【答案】①p && b[0]==0　②b[j]=1;　③p=false;
④a[i][j]==-1 && b[j]==1　⑤p或b[0]==1

【分析】本题采用回溯穷举法得到所有的可能情况，从中寻找一种可行方案并输出，如果没有可行方案则输出“找不到!”。

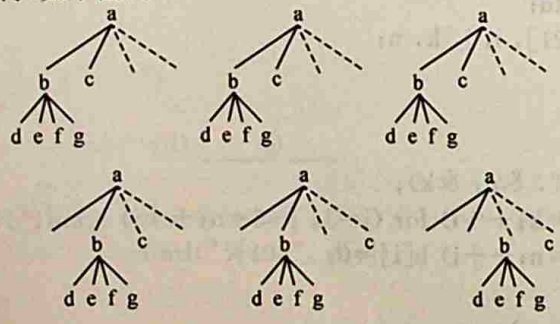
p值初始为true，第1空要求两个方面共同成立才继续循环，即p还是true，b数组还可以继续穷举，答案：p && b[0]==0；第2空所在位置就是寻找下一种方案，位置从大到小寻找第一个未取的数位j(b[j]=1;)；第3空是假设p为false，即假设当前这种方案可行，下面的双重循环是判断是否存在不符合要求的位置，如果存在，则p改为true，所谓不符合要求，即本该是1的位置在当前方案中为0，或者本该是0的位置在当前方案中为1，第4空即第二种情况a[i][j]==-1 && b[j]==1；第5空是最后输出，如果p值为true，或者b[0]==1，说明所有方案都已经尝试过了也没有找到可行方案。

7.【NOIP2005提高组】N叉树

【问题描述】

我们都了解二叉树的先根遍历，中根遍历和后根遍历。当知道先根遍历的结果和中根遍历结果的时候，我们可以唯一地确定二叉树；同样的，如果知道了后根遍历的结果和中根遍历结果，二叉树也是唯一确定的。但是如果只知道先根遍历和后根遍历的结果，二叉树就不是唯一的了。但是我们可以计算满足条件的不同二叉树的一共有多少个。这不是一个很困难的问题，稍微复杂一点，我们把这个问题推广到N叉树。

我们用小写英文字母来表示N叉树的结点，不同的结点用不同的字母表示。比如，对于4叉树，如果先根遍历的结果是abdefgc，后根遍历的结果是defgbca，那么我们可以得到6个不同的4叉树(如下图)。



【输入格式】

输入数据包括3行。

第一行是一个正整数N(1 ≤ N ≤ 20)，表示我们要考虑N叉树。

第二行和第三行分别是两个字符串序列，分别表示先根遍历和后根遍历的结果。

【输出格式】

输出不同的 N 叉树的数目。题目中给的数据保证得到的结果小于 2^{31}。

【输入样例】

```
4
abdefgc
defgbca
```

【输出样例】

```
6
```

【程序清单】

```
#include <iostream>
#include <cstdio>
#include <cstring>
using namespace std;
char str1[100], str2[100];
int N;
long com[100][100];
long getcom(int x, int y) {
    if (y==0 || x==y) ____①____;
    else if (com[x][y] ! =0) return com[x][y];
    else {
        com[x][y]=getcom(x-1, y)+ ____②____;
        return com[x][y];
    }
}
long count(int a, int b, int c){
    long sum=1;
    int k=0;
    int s=a+1, t=c, p;
    if (a==b) return 1;
    while(s <=b){
      p=t;
      while(str1[s] ! =str2[t]) t++;
      sum=sum * count(s, s+t-p, p);
      s= ____③____;
      ____④____;
      k++;
    }
    return ____⑤____ * getcom(N, k);
}
int main(){
    int len;
    scanf("%d", &N);
    scanf("%s%s", str1, str2);
    len=strlen(str1);
    printf("%ld\n", count(____⑥____));
    return 0;
```

}

【答案】①return 1（或者 return 1L，或者 return 1l）

②getcom(x-1, y-1)　③s+t-p+1

④t++（或者++t，或者 t=t+1，或者 t+=1）　⑤sum　⑥0, len-1, 0

【分析】总体思路是这样的：

确定某一棵子树中各层分支结点的结构和顺序，再利用求组合数的公式，求出可能的 N 叉子树的数目；

对当前子树的每一颗子树，再递归地进行计数处理；

对各子树中可能的排列数目，再利用乘法原理进行相乘计算；

Count()中：s=a+1，子串的起点位置，if(a==b)就是叶子，s<=b，子串未扫完。

① return 1（或者 return 1L，或者 return 1l），递归边界处理；

② getcom(x-1, y-1)，组合公式；

③ s+t-p+1，新子串的根在 str1 的位置；

④ t++（或者++t，或者 t=t+1，或者 t+=1），下个子串在 str2 的起点位置；

⑤ sum，组合公式累乘；

⑥ 0, len-1, 0，str1 中的起点，终点，str2 的起点。

8.【NOIP2006 提高组】TSP 问题

【问题描述】

给定 n 个城市，构成一个完全图，任何两城市之间都有一个代价（例如路程、旅费等），现要构造遍历所有城市的环 路，每个城市恰好经过一次，求使总代价达到最小的一条环路。

遗传算法是求解该问题的一个很有效的近似算法。在该算法中，一个个体为一条环路，其编码方法 之一是 1 到 n 这 n 个数字的一个排列，每个数字为一个城市的编号。例如当 n=5 时，“3 4 2 1 5” 表示该方案实施的路线为 3->4->2->1->5->3。遗传算法的核心是通过两个个体的交叉操作，产生两 个新的个体。下面的程序给出了最简单的一种交叉算法。具体过程如下：

(1)选定中间一段作为互换段，该段的起止下标为 t1, t2，随机生成 t1, t2 后，互换两段。

(2)互换后，在每个新的排列中可能有重复数字，因而不能作为新个体的编码，一般再做两步处理：

(2.1) 将两个互换段中，共同的数字标记为 0，表示已处理完。

(2.2) 将两个互换段中其余数字标记为 1，按顺序将互换段外重复的数字进行替换。例如：n=12，两个个体分别是：

a1: 1 3 5 4　* 2　6 7　9　* 10 12 8 11

a2: 3 2 1 12 * 6 7 10 11 * 8 5 4 9 t1=5, t2=8。上述每一行中，两个星号间的部分为互换段。假定数组的下标从 1 开始，互换后有：

a1: 1 3 5 4　　* 6 7 10 11　* 10 12 8 11

a2: 3 2 1 12　* 2 6　7　9　*　8 5　4　9

然后，将数字 6,7 对应的项标记为 0，星号内数字 2,9,10,11 对应的项标记为 1，并且按顺序对应关系为：10<->2, 11<->9。于是，将 a1[9]=10 替换为 a1[9]=2，将 a2[2]=2 替换为 a2[2]=10，类似再做第 2 组替换。这样处理后，就得到了两个新个体：

a1: 1　3 5　4　6 7 10 11　　2 12　8　9

a2: 3 10 1 12　2 6 7 9　　8　5　4　11

(3)输出两个新个体的编码。

【程序说明】

```
#include <iostream>
#include <cstdio>
```

```
#include <cstdlib>
#define N 20
using namespace std;
int a1[N],a2[N],kz1[N],kz2[N],n;
int rand1(int k)
{
   int t=0;
   while(t<2|| t>k)
   t=(int)((double)rand()/RAND_MAX * k);
   return t;
}
void read1(int a[],int m)
{读入数组元素 a[1]至 a[m],a[0]=0,略。}
void wrt1(int a[],int m)
{输出数组元素 a[1]至 a[m],略。}
void cross(int a1[], int a2[],int t1, int t2, int n)
{
   int i,j,k,t,kj;
   for(i=t1; i<=t2; i++)
   {
       t=a1[i];  ①  ;
   }
   for(i=1;i<=n;i++)
     if(i<t1 || i>t2)
       kz1[i]=kz2[i]=-1;
     else
       ②  ;
   for(i=t1;i<=t2;i++)
     for(j=t1;j<=t2;j++)
       if(a1[i]==a2[j])
       {
         ③  ; break;
       }
   for(i=t1;i<=t2;i++)
   if(kz1[i]==1)
   {
      for(j=t1;j<=t2;j++)
      if(kz2[j]==1)
      {
        kj=j; break;
      }
      for(j=1;j<=n;j++)
      if(  ④  )
      {
        a1[j]=a2[kj];break;
      }
      for(j=1;j<=n;j++)
```

```
        if(    ⑤    )
        {
          a2[j]=a1[i]; break;
        }
    }
    kz1[i]=kz2[kj]=0;
}
int main()
{
    int k,t1,t2;
    printf("input (n>5):\n");  scanf("%d",&n);
    printf("input array 1 (%d'numbers):\n",n);  read1(a1,n);
    printf("input array 2 (%d'numbers):\n",n);  read1(a2,n);
    t1=rand1(n-1);
    do
    {
        t2=rand1(n-1);
    }while(t1==t2);
    if(t1>t2)
    {
        k=t1; t1=t2; t2=k;
    }
    ⑥    ;
    wrt1(a1,n);  wrt1(a2,n);
    return 0;
}
```

【答案】①a1[i]=a2[i];a2[i]=t　②kz1[i]=kz2[i]=1
③kz1[i]=kz2[j]=0　④a1[j]==a1[i]　&&　kz1[j]==-1
⑤a2[j]==a2[kj]　&&　kz2[j]==-1　⑥cross(a1,a2,t1,t2,n)

【分析】根据题目模拟，分布，把题目的步骤要标清楚，题目和代码一一对应，不算太绕。

①a1[i]=a2[i];a2[i]=t，根据题意，第一步，将 a1,a2 对应位置交换；

②kz1[i]=1;kz2[i]=1，此空上面几行根据题意，非有效部分被标记为-1，kz1[]和kz2[]是用来标记的，有效部分中相同数字被标记为 0，不相同标记为 1；

③kz1[i]=0;kz2[j]=0，相同标记为 0；

④上面有部分代码是根据标记是 1，查找的代码，i 和 j 用来定位需要交换的数字；然后根据题意，在外部替换，a1[j]==a1[i]　&&　kz1[j]==-1；

⑤a2[j]==a2[kj]　&&　kz2[j]==-1，理由同上；

⑥cross(a1,a2,t1,t2,n)，2 段的起点、终点、总长。

9.【NOIP 提高组 2007】连续邮资问题

某国发行了 n 种不同面值的邮票，并规定每封信最多允许贴 m 张邮票，在这些约束下，为了能贴出{1,2,3,…,maxvalue}连续整数集合的所有邮资，并使 maxvalue 的值最大，应该如何设计各邮票的面值？例如，当 n=5、m=4 时，面值设计为{1,3,11,15,32}，可使 maxvalue 达到最大值 70(或者说，用这些面值的 1 至 4 张邮票可以表示不超过 70 的所有邮资，但无法表示邮资 71。而用其他面值的 1 至 4 张邮票如果可以表示不超过 k 的所有邮资，必有 k≤70)。

下面是用递归回溯求解连续邮资问题的程序。数组 x[1：n]表示 n 种不同的邮票面值，并约定各元素按下标是严格递增的。数组 bestx [1：n]存放使 maxvalue 达到最大值

的邮票面值(最优解)，数组 y[maxl]用于记录当前已选定的邮票面值 x[1：i]能贴出的各种邮资所需的最少邮票张数。请将程序补充完整。

【程序说明】

```
#include <iostream>
#include <cstdio>
#define NN 20
#define maxint 30000
#define maxl 500                /* 邮资的最大值 */
using namespace std;
int n,m,bestx[NN],x[NN],y[maxl],maxvalue=0;
void result()
{
    输出结果:最大值:maxvalue 及最优解:bestx[1：n](略)
}
void backtrace(int i,int r)
{
   int j,k,z[maxl];
   for(j=0;j<=    ①    ;j++)
   if(y[j]<m)
     for(k=1;k<=m-y[j];k++)
     if(y[j]+k<=y[    ②    ])
       y[    ③    ]=y[j]+k;
   while(y[r]<maxint) r++;
   if(i>n)
   {
     if(r-1>maxvalue)
     {
        maxvalue=    ④    ;
        for(j=1;j<=n;j++) bestx[j]=x[j];
     }
     return;
   }
   for(k=0;k<maxl;k++)
     z[k]=y[k];
   for(j=    ⑤    ;j<=r;j++)
   {
     x[i]=j;
     ⑥    ;
     for(k=0;k<maxl;k++) y[k]=z[k];
   }
}
int main()
{
   int j;
   printf("input n,m:\n");
   scanf("%d%d",&n,&m);
   for(j=1;j<maxl;j++)
```

```
        y[j]=maxint;
    y[0]=0; x[0]=0; x[1]=1;
    backtrace(2,1);
    result();
    return 0;
}
```

【答案】① x[i-2]*(m-1)　② j+x[i-1]*k　③ j+x[i-1]*k　④ r-1,本段 if 是打擂台程序,如果超过 maxvalue,可以做更新操作,下面代码将 bestx[j]更新就是提示;　⑤ x[i-1]+1　⑥ backtrace(i+1,r)

【分析】基本思路:搜索所有可行解,找出最大连续邮资区间的方案。

解向量:用 n 元组 x[1:n]表示 n 种不同的邮票面值,并约定它们从小到大排列。x[1]=1 是唯一的选择。

可行性约束函数:已选定 x[1:i-1],最大连续邮资区间是 1-r,接下来 x[i]的可取值范围是 x[i-1]+1->r+1。

如何确定 r 的取值:计算 x[1:i]的最大连续邮资区间在本算法中被频繁使用到,因此势必要找到一个高效的方法。考虑到直接递归的求解复杂度太高,我们不妨尝试计算用不超过 m 张面值为 x[1:i]的邮票贴出邮资 k 所需的最少邮票数 y[k]。通过 y[k]可以很快推出 r 的值。事实上,y[k]可以通过递推在 O(n)时间内解决。

```
    for (int j=0; j<=x[i-2]*(m-1); j++)
        if (y[j]<m)
        for (int k=1; k<=m-y[j]; k++)
            if (y[j]+k<y[j+x[i-1]*k])
              y[j+x[i-1]*k]=y[j]+k;
    while (y[r]<maxint) r++;
```

尝试计算用不超过 m 张面值为 x[1:i]的邮票贴出邮资 k 所需的最少邮票数 y[k]。通过 y[k]可以很快推出 r 的值。事实上,y[k]可以通过递推在 O(n)时间内解决。附已填空的程序及逐模块备注。

```
#include <iostream>
#include <cstdio>
#define NN 20
#define maxint 30000
#define maxl 500                          //邮资的最大值
using namespace std;
intn,m,bestx[NN],x[NN],y[maxl],maxvalue=0;
voidresult()
{
    输出结果:最大值:maxvalue 及 最优解:bestx[1:n](略)
}
voidbacktrace(int i,int r)          //i 是指第 i 种面值不超过 n     r 最高总面额
{     int j,k,z[maxl];
    //用不超过 m 张面值为 x[1:i]的邮票贴出邮资 k 所需的最少邮票数 y[k]。通过 y[k]可以很快推出 r 的值
    for(j=0;j<=①x[i-2]*(m-1);j++)    //j 穷举总面额      递推
      if(y[j]<m)                          //m 是张数上限 4
for(k=1;k<=m-y[j];k++)                //k 穷举还有几张钞票可以用
    if(y[j]+k<=y[②j+x[i-1]*k])
        y[ ③j+x[i-1]*k ]=y[j]+k;
```

```
while(y[r]<maxint) r++; //maxint 是标记,没被改动过,r 找连续邮资值最大值上限
if(i>n)//递归结束调节,i 超过 n 张,backtrace(i,r)i 是指第 i 种邮票面值,是线索
{
    if(r-1>maxvalue)                    //r-1 是因为前面 while 会多走一步
    {
        maxvalue=④r-1;                  //打擂台更新,这空比较简单,上一行就是提示
    for(j=1;j<=n;j++)                   //线索,n 是 n 种面值,
        bestx[j]=x[j];          //题干中提到 :数组 x[1 : n]表示 n 种不同的邮票面值,
    }
    return;                             //找到结果返回
}
for(k=0;k<maxl;k++)
    z[k]=y[k];                          //保存 y[]到 z[] ,呼应下面
    //数组 x 记录当前已经确定的邮票面值,整数 r 表示当前使用不超过 m 张邮票能贴出的最大连续邮资区间
    //x[i]的取值要和前面 i 个数各不相同,最小应该是 x[i-1]+1,最大就是 r
    for(j=⑤x[i-1]+1;j<=r;j++) //j 穷举 上一种面值+1, 上限是目前能到的 r 值
    {
        x[i]=j;                         //先填进去试试
            ⑥backtrace(i+1,r);          //尝试 i+1 的面值,
          for(k=0;k<maxl;k++)           //回溯的时候恢复现场
        y[k]=z[k];
    }
}
int main()
{
    int j;
    printf("inputn,m:\n");
    scanf("%d%d",&n,&m);
    for(j=1;j<maxl;j++)
        y[j]=maxint;                    //标记,等会有值的,可以检查连续邮资上限用
    y[0]=0;x[0]=0;x[1]=1;               //x[1]  第一种钱是 1¥
    backtrace(2,1);
    result();
    return 0;
}
```