第一章　C 语言概述

简单描述 C 语言中主函数与被调同的函数的关系

主函数又称 main 函数，一个程序有且只有一个，程序从主函数开始运行且在主函数中结束，被调用函数可以有多个，可被主函数和其他函数调用。

请简述 C 程序的特点

C 程序主要是由函数构成的；一个函数由函数首部和函数体构成；C 程序总是从 main 中开始执行且从 main 中结束；每个语句和数据声明的最后必须有分号（;）；C 语言本身没有输入和输出语句；可以用“//对 C 程序中任何一行或数行做注释。C 语言书写格式自由，一行内可以写多个语句，一个语句可以分多行写；

上机运行 C 程序的方法步骤

上机器输入和编辑源程序，对源程序进行编译，得到目标函数，将目标函数与数据库连接，得到可执行的目标程序，运行可执行的目标程序

编辑好源代码后为什么还要进行编译才能执行

由于 C 语言是高级编程语言，而计算机的底层是二进制的机器语言，且不同的设备之间的二进制语言不互通，因此我们需要通过编译将程序转化为机器语言，也可以通过不同的编译系统，适应不同的机器。

计算机由什么控制？为什么需要高级编程语言?

计算机由程序控制，高级语言接近人类自然语言和数学语言，容易学习和推广，不依赖具体计算机，通用性强。

机器语言和汇编语言与高级编程语言的区别

机器语言和汇编语言依赖具体计算机，属于低级语言，难学难用，无通用性。高级语言接近人类自然语言和数学语言，容易学习和推广，不易赖有具体计算机，通用性强。

C 语言数据类型有哪几类？基本数据类型包括哪几类?

C 中的数据类型包括：基本类型、构造类型、指针类型、空类型等。
其中基本类型包括：整型、字符型、实型（浮点型）等。
构造类型包括：枚举型、结构体、共用体、数组

第二章数据的存储与运算

35、数据在计算机中以什么进制形式存储？二进制数 1010 表示十进制的数是多少?
数据以二进制存储，1010 表示数字 8*1+4*0+2*1+1*0 等于 10

36、简述整数、实数与字符的存储方式

整数的存储方式是直接存储，如 0001010；实数采用指数形式存储，如 124.5 则储存为|+|，1245|+|3|；字符则是以二进制直接存储，读写时通过计算机按照 ASCII 码转换，如 65 的二进制是 1000001，在 ASCI 码中对应的是字符 A

37、简述常量和变量的关系

常量是指程序运行过程中值不改变的量；变量是指在程序运行过程中可以改变的量；

38、简述变量名、变量地址、存储单元与变量的值四者的关系

变量名是一个容易记的名字，代表一个存储单元的地址，也就是变量地址，在存储单元中可以储存变量的值；存储的关键是存储单元，每个存储单元对应一个变量地址，访问时通过变量地址访问，存储单元里面可以存储变量值，为了方便访问存储单元，可以用变量名代表变量地址

39、简述整型有哪些类型以及实型变量有哪些类型

整型分为基本整型 int、长整型 long int 和短整型 shortint；实型变量分为单精度 float、双精度 double 和长双精度 long double

40、简述编译时“错误”(error)和“警告”(warning)的区别

在程序编译时，可能会出现致命“错误”（error）和“警告”（warning），系统出现“错误”时，无法运行，系统出现系统发出“警告”，提醒用户注意。在编译时出现“警告”，不属于致命性错误，可以继续运行并得到结果，但不保证结果完全精确，由用户自己分析决定。

41、简述字符常量、转义字符和字符变量

字符常量是用单引号括起来的字符，如‘&’；转义字符是用来作为输出信息时的控制符号如‘\n’代表换行；字符变量是用来存放字符常量的，它只能放一个字符；

字符常量与字符串常量有什么区别?

字符常量就是一个字符，用单引号括起来，占一个字节；而字符串常量是由若干个字符组合而成，用双引号括起来，存储时自动在后面加“\0”，即使同样是一个字符，字符串常量后面还要加一个“\0”。

42、符号常量的作用和使用方法

符号常量是在 C 语言中，可以用一个标识符来表示一个常量，这个标识符称之为符号常量。符号常量用来简化编码时多次重复出现的特定数值，如 PI=3.1415926，我们可以用 PI 替代。使用步骤：进行一次“顶编译”如#define PI 3.1415926；然后进行正式的编译工作

43、简述基本运算符及其运算规则

基本运算符有+–*/%，运算规则是先乘除后加减，同级情况下自左向右，进行%运算要双目为整形。

44、简述++a 与 a++的区别

++a 是先进行自加运算，后进行其他运算，而 a++这是先进行其他运算后进行自加，如 a=1，b=a++，则 b=1；b=++a,则 b=2

第三章顺序程序设计

1.书中原题：C 语言为什么把输入输出的功能作为函数，而不作为语句的基本部分

不把输入输出作为 C 语句的目的是使 C 语言编译系统简单，因为将语句翻译成二进制的指令是在编译阶段完成的，把输入输出操作放在函数中处理，就可以使 C 语言本身的规模较小，编译程序简单，很容易在各种机器上实现，程序具有可移植性。

37、程序有哪些基本结构

顺序结构；选择结构；循环结构

37、**书中原题：怎样区分表达式和表达语句?C 语言为什么要设表达式语句？什么时候用表达式?什么时候用表达语句?**

表达式是操作符、操作数和标点符号组成的序列，其目的是用来说明计算过程。表达式语句则是在表达式后加分号，如 i=1 是表达式，i=1；则为表达语句。设表达语句的目的是规范代码语法格式，符合编译器的设置。我们在进行计算时会用到表达式进行运算，运算完成后需要表达式语句进行规范表达，程序中的计算功能主要由表达式语句来完成。

39、简述表示算法的几种方式

有自然语言表示算法、用流程图表示算法、用 N–S 流程图表示算法、用伪代码表示算法

C 语言中的语句有几类？控制语句有哪些?

C 语句可以分为以下五类：

(1) 控制语句。完成一定的控制功能。

(2) 函数调用语句；

(3) 表达式语句；

(4) 空语句；

(5) 可以用{ }把一些语句括起来成为复合语句。

C 只有 9 种控制语句，它们是：

(1) if（）else（条件语句）
(2) for（）~（转向语句）
(3) while（）~（循环语句）
(4) do~while（）（循环语句）
(5) continue（结束本次循环语句）
(6) break（中止执行 switch 或循环语句）
(7) switch（多分支选择语句）
(8) goto（转向语句）
(9) return（从函数返回语句）

第四章 选择结构程序设计

书中原题：什么是算术运算？什么是关系运算？什么是逻辑运算？

1.算术运算就是指加减乘除和整数的模运算（即取余数运算）。
2.关系运算就是比较运算，将两个数值进行比较，判断其比较结果是否符合给定的条件。
3.逻辑运算指两个条件进行运算，有逻辑与、逻辑或、逻辑非三种。

36、C 程序中常用的输入输出数据的函数

在 C 程序中，数据的输入输出主要用 printf 和 scanf 函数来实现的。这两个函数是格式输入输出函数，在进行输入输出时，程序设计人员必须指定输入输出数据的格式，即根据数据的不同类型指定不同的格式。

书中原题：C 语言中如何表示“真”和“假”？系统如何判断一个量的“真”和“假”？

解：如果有一个逻辑表达式，若其值为“真”，系统会以 1 表示，若其值为“假”，会以 0 表示。但是在判断一个逻辑量的值时，系统会以 0 作为“假”，以非 0 作为“真”，例如 3&.&5 的值为“真”，系统给出 3&.&.5 的值为 1。

52、C 语言中的常用条件判断的函数和关系运算符

常用 if（条件）..else.…函数进行条件条件判断，常用关系运算符有<><=>==！=

53、常用逻辑运算符及其含义

C 语言提供 3 种逻辑运算符
(1) &&逻辑与
(2) ‖逻辑或
(3)!逻辑非

54、C 语言中常用的多支选择函数

常用 switch 实现多支选择，如
switch（表达式）
{case 1：语句 1
case 2：语句 2
default：语句 n+1}

55、C 语言中常用的条件运算符条件运算符（?:）是 C 语言中唯一的三目（元）运算符。条件表达式是一种特殊的选择结构

56、使用 if 实现多支选择结构需要注意什么

内嵌 if 也应包括 else 部分；把内嵌的 if 放在外层的 else 子句中；加大括号，限定范围；程序写成锯齿形，同一层次的 if 和 else 在同一列上。

57、使用 switch 为什么需要用 break

在用 switch 语句实现多分支选择结构时，“case 常量表达式”只起语句标号作用，如果“switch”后面的表达式的值与“case”后面的常量表达式的值相等，就执行 case 后面的语句。但特别注意：执行完这些语句后不会自动结束，会继续执行下一个 case 子句中的语句。因此，应在每个 case 子句最后加一个 break 语句，才能正确实现

多分支选择结构。

第五章 循环结构程序设计

31、为什么需要循环结构

循环结构可以帮助我们完成需要重复常量的问题

32、循环的分类

有两种循环：一种是无休止的循环，如地球围绕太阳旋转，永不终止；每一天 24 小时，周而复始。另一种是有终止的循环，达到一定条件循环就结束了，如统计完第 50 名学生成绩后就不再继续了。计算机程序只处理有条件的循环，算法的特性是有效性、确定性和有穷性，如果程序永远不结束，是不正常的。

33、简述 C 语言中常用的三种循环语句有

while、do...while 和 for

34、简述 while 循环和 do..while 循环的区别

while 是先进行判断，符合条件再进入循环体；do..while 是先做一次循环体后再进行判断

35、提前结束循环的两种方法

用 break 提前结束整个循环，用 continue 提前结束本次循环

36、构建有效循环体的条件

需要重复执行的操作，记循环体；循环结束的条件

第六章 利用数组处理批量数据

46、什么是数组，为什么要用数组

数组是有序数据的集合。数组中的每一个元素都属于同一个数据类型。用一个统一的数组名和下标来唯一地确定数组中的元素。这样就把具有同一属性的若干个数据组织成一个整体，它们再也不是互相孤立无关的单个数据，而是互相关联的，便于统一处理。

47、如何定义和引用数组

在定义数组时需要指定这批变量的类型、数组名称和数组中包含多少个元素（即变量），通过数组下表引用数组，如 int a 定义了一个一维数组长度为 5，a 则代表数组的第一个数值

48、数组如何进行排序

对一组数据进行排序的方法很多，如“起泡法”排序。“起泡法”的思路是：先将第 1 个数和第 2 个数比较，如果第 2 个数比第 1 个数小，就将两个数互换，这样，小的数就排到前面了。然后再将第 2 个数和第 3 个数比较，如果第 3 个数比第 3 个数小，就将两个数互换，这样，第 3 个数就是 3 个数中最大的了。依此规律，将相邻两个数比较，将小的调到前头，

50、字符串的常用处理函数

gets（字符数组）：从终端输入一个字符串到字符数组；
puts（字符数组）：将一个字符串（以 0 结束的字符序列）输出到终端；
strcat(字符数组 1，字符数组 2)：连接两个字符数组中的字符串，把字符串 2 接到字符串 1 的后面；strcpy（字符数组 1，字符串 2)：将字符串 2 复制到字符数组 1 中去；
strcmp(字符串 1，字符串 2)：比较字符串 1 和字符串 20 如果字符串 1=字符串 2,则函数值为 0；如果字符串 1>字符串 2，则函数值为一个正整数；如果字符串 1<字符串 2，则函数值为一个负整数；strlen（字符数组)：测试字符串长度；

51、C 语言中字符串是如何存放的?什么时候代表结束?

字符串是以字符数组形式存放的，为了确定字符串的范围，C 编译系统在每一个字符串的后面加一个作为字符串结束标志。0 不是字符串的组成部分，输出字符串时不包括要区分字符数组和字符串，字符串可以放在字符数组中，如果宇符串的长度为 n，则能存放该字符串的字符数组的长度应≥n+1

第七章 用函数实现模块化程序设计

45、简述全局变量与局部变量

变量的作用域是指变量有效的范围。根据定义变量的位置不同，变量分为局部量和全局变量。凡是在函数内或复合语句中定义的变量都是局部变量，其作用域限制在函数内或复合语句内，函数或复合语句外不能引用该变量。在函数外定义的变量都是全局变量，其作用域为

从定义点到本文件末尾，可以用 extern 对变量作“外部声明”，将作用域扩展到本文件中作 extern 声明的位置处，或在其他文件中用 extern 声明将作用域扩展到其他文件。用 static 声明的静态全局变量禁止其他文件引用该变量，只限本文件内引用。

46、什么是变量的生存期

变量的生存期指的是变量存在的时间。全局变量的生存期是程序运行的整个时间。局部变量的生存期是不相同的。用 auto 或 register 声明的局部变量的生存期与所在的函数被调用的时间段相同，函数调用结束，变量就不存在了。用 static 声明的局部变量在函数调用结束后内存不释放，变量的生存期是程序运行的整个间。凡不声明为任何存储类别的都默认为 auto(自动变量)

7.6.1 什么是内部函数

如果一个函数只能被本文件中其他函数所调用、它称为内部函数，在定义内部函数时，
在函数名和函数类型的前面加 static，即 static 类型标识符 函数名（形参表）；

7.6.2 什么是外部函数

(1) 如果在定义函数时，在函数首部的最左端加关键字 extern、则此函数是外部函数，可
供其他文件调用。

47、简述内部函数与外部函数的区别

函数有内部函数与外部函数之分。函数本质上是外部的，可以供本文件或其他文件中的函数调用，但是在其他文件调用时要用 extern 对函数进行声明。如果在定义函数时 用 static 声明，表示其他文件不得调用此函数，即把它“屏蔽”起来。

48、什么是函数

在 C 语言中，函数是用来完成某一个特定功能的。C 程序是由一个或多个函数组成的。函数是 C 程序中的基本单位。执行程序就是执行主函数和由主函数调用其他函数。因此编写 C 程序，主要就是编写函数。

49、简述库函数和自己定义的函数的区别

有两种函数：系统提供的库函数和用户根据需要自己定义的函数。如果在程序中使用库函数，必须在本文件的开头用 include 指令把与该函数有关的头文件包含到本文件中来（如用数学函数时要加上 include<math.h>）。如果用自己定义的函数，必须先定义，后调用。需要注意：如果函数的调用出现在函数定义位置之前，应该在调用函数之前用函数的原型对该函数进行引用声明。

50、函数的定义和声明的关系

函数的“定义”和“声明”不是一回事。函数的定义是指对函数功能的确立，包括指定函数名、函数值类型、形参及其类型以及函数体等，它是一个完整的、独立的函数单位。而函数的声明的作用则是把函数的名字、函数类型以及形参的类型、个数和顺序通知编译系统，以便在调用该函数时系统按此进行对照检查。

51、如何将数值作为函数的参数

用数组元素作为函数实参，其用法与用普通变量作实参时相同，向形参传递的是数组元素的值用数组名作函数实参，向形参传递的是数组首元素的地址，而不是数组全部元素的值。如果形参也是数组名，则形参数组首元素与实参数组首元素具有同一地址，两个数组共占同一段内存空间。利用这一特性，可以在调用函数期间改变形参数组元素的值，从而改变实参数组元素的值。这是很有用的。要弄清其概念与用法。

第八章 指针

58、简述指针和指针变量

指针就是地址，凡是出现“指针”的地方，都可以用“地址”代替。例如，变量的指针就是变量的地址，指针变量就是地址变量。

59、为什么是“指针是带类型的地址”

指针代表的不是一个纯地址（即内存单元的编号），而且还是一个带类型的地址。每一个指针（地址）型数据都是有类型属性的 (如“int”, 关 foat”)。在 C 语言中, 指针就是地址”指的就是“带类型的地址”。如 P 的类型用“*int”表示，其中“*”表示它是指针型变量, “int”表示其基类型为整型。

60、什么是指针运算

将该指针变量的原值（是一个地址）和它指向的变量所占用的内存单元字节数相加减，将一个变量地址赋给一个指针变量，指针间的比较等

38、.什么是指针？什么是指针变量？它们有什么关系

(1) 在计算机科学中，指针（Pointer）是编程语言中的一个对象，利用地址，它的值直接指向（points to)存在电脑存储器中另一个地方的值。由于通过地址能找到所需的变量单元，可以说，地址指向该变量单元。因此, 将地址形象化的称为”指针”。

(2)存放地址的变量称为指针变量。指针变量是一种特殊的变量，它不同于一般的变量，一般变量存放的是数据本身，而指针变量存放的是数据的地址。

(3) 指针就是地址，地址就是指针。地址就是内存单元的编号。指针变量就是存放内存地址的变量。

(4)指针和指针变量是两个不同的概念，但要注意的是，通常我们叙述时会把指针变量简称为指针。

61、举例说明指针和变量定义时的区别

int i：定义整型变量 i

int*p：p 为指向整型数据的指针变量

int a[n]：定义整型数组 a，它有 n 个元素

int*p[n]：定义指针数组 p，它由 n 个指向整型数据的指针元素组成

62、在用数组名作为函数实参时，既然实际上相应的形参是指针变量，为什么还允许使用形参数组的形式呢?

这是因为在 C 语言中用下标法和指针法都可以访问一个数组 (如果有一个数组 a,则 a 和*(a+i)是无条件等价的)，用下标法表示比较直观，便于理解。因此许多人愿意用数组名作形参，以便与实参数组对应。从应用的角度看，用户可以认为有一个形参数组, 它从实参数组那里得到起始地址, 因此形参数组与实参数组共占同一段内存单元，在调用函数期间，如果改变了形参数组的值，也就是改变了实参数组的值。当然在主调函数中可以利用这些已改变的值。对 C 语言比较熟练的专业人员往往喜欢用指针变量作形参。

第九章 使用结构体类型处理组合数据

34、C 中的数据类型有哪些种类

C 语言中的数据类型包括两类：一类是系统已经定义好的标准数据类型（如 int，char，float.double 等），编程者不必自己定义，可以直接用它们去定义变量；另一类是用户根据需要在一定的框架范围内自己设计的类型, 先要向系统作出声明，然后才能用它们定义变量。其中最常用的有结构体类型，此外还有共用体类型和枚举类型。

35、什么是结构体

C 语言允许用户自己建立由不同类型数据组成的组合型的数据结构，它称为结构体。

36、结构体变量的指针是什么

结构体变量的指针就是结构体变量的起始地址，注意它的基类型是结构体类型的数据。可以定义指向结构体变量的指针变量，这个指针变量的值是结构体变量的起始地址。指向结构体变量的指针变量常用于作为函数参数和链表中（用来指向下一个结点）。

37、结构体和结构体指针能组建什么

把结构体变量和指向结构体变量的指针结合起来，可以建立动态数据结构 (如链表)。开辟动态内存空间用 malloc 函数，函数的返回值是所开辟的空间的起始地址。利用所开辟的空间作为链表的一个结点，这个结点是一个结构体变量，其成员由两部分组成：一部分是实际的有用数据，另一部分是一个指向结构体类数据的指针变量，利用

它指向下一个结点。

38、结构体和共用体的区别

共用体与结构体不同，其各成员不是分别占独立的存储单元，而是共享同一段存储空间，因此，各成员的值不会同时存在，在某一瞬时，只有最后一次被赋值的成员是有意义的。

39、什么是枚举类型

枚举类型是把可能的值全部一一列出，枚举变量的值只能是其中之一。实际生活中有些问题没有现成的数学公式来解决，只能把所有的可能性一一列出，测试其是否满足条件，这时用枚举变量比较方便。

第十章　利用文件保存数据

书中原题：10.1 对 C 文件操作有些什么特点?

C 语言把文件看作是一个字符(字节）的序列，即由一个一个字符（字节)的数据顺序组成。一个输入输出流就是一个字节流或二进制流。在 C 文件中，数据由一连串的字符(字节）组成，中间没有分隔符，对文件的存取是以字符(字节）为单位的，可以从文件读取一个字符或向文件输出一个字符。输入输出数据流的开始和结束仅受程序控制而不受物理符号(如回车换行符)控制，这就增加了处理的灵活性。这种文件称为流式文件。

书中原题：什么是缓冲文件系统和文件缓冲区?

C 语言采用“缓冲文件系统”处理文件，所谓缓冲文件系统是指系统自动地在内存区为程序中每一个正在使用的文件开辟一个文件缓冲区。

书中原题：1.什么是文件型指针？通过文件指针访问文件有什么好处?

答：缓冲文件系统中，关键的概念是“文件类型指针”，简称“文件指针”。每个被使用的文件都在内存中开辟一个相应的文件信息区，用来存放文件的有关信息(如文件的名字、文件状态及文件当前位置等)。这些信息是保存在一个结构体变量中的。该结构体类型是由系统声明的，取名为 FILE。通过文件指针访问文件的好处是：可以随机访问文件，有效表示数据结构，动态分配内存，方便使用字符串，有效使用数组。

书中原题：2.对文件的打开与关闭的含义是什么？为什么要打开和关闭文件?

答：文件使用前必须“打开”，用完后应当“关闭”。所谓打开，是建立相应的文件信息区，开辟文件缓冲区。由于建立的文件信息区没有名字，只能通过指针变量来引用，因此一般在打开文件时同时使指针变量指向该文件的信息区，以便程序对文件进行操作。所谓关闭，是撤销文件信息区和文件缓冲区，指针变量不再指向该文件。使用完如果不关闭文件可能会丢失数据。

31、什么是文件?

文件是在存储在外部介质上数据的集合，操作系统把所有输入输出设备都作为文件来管理。每一个文件需要有一个文件标识，包括文件路径、文件主干名和文件后缀。

C 语言程序中主要用到两种文件：

(1)　程序文件。包括源程序文件、目标文件、可执行文件等。这种文件是用来存放程序的，以便实现程序的功能。

(2)　数据文件。文件的内容不是程序，而是供程序运行时读写的数据，如在程序运行过程中输出到磁盘的数据，或者供程序运行时读入内存的数据。

10.1.2 文件名

一个文件要有一个唯一的文件标识，以便用户识别和引用。文件标识包括三部分；

(1)文件路径：(2) 文件名主干；(3)文件后缀。

文件名主干的命名规则遵循标识符的命名规则。后缀用来表示文件的性质，一般不超过 3 个字母。

32、数据文件有哪些类型

数据文件有两类：ASCII 文件和二进制文件。ASCII 文件又称文本文件，它的每一字节放一个字符的 ASCII 代码。二进制文件是把内存中的数据按其在内存中的存储形式原样输出到磁盘上存放。字符型数据只能以 ASCII 形式存储，数值型数据可以用 ASCII 形式存储到磁盘上，也可以用二进制形式存储。

33、C 语言采用什么系统读写文件

C 语言采用缓冲文件系统，为每一个使用的文件在内存开辟一个文件缓冲区，在计算机输入时，先从文件把数据读到文件缓冲区，然后从缓冲区分别送到各变量的存储单元。在输出时，先从内存数据区将数据送到文件缓冲区，待放满缓冲区后一次输出，这有利于提高效率。

34、简述文件指针、文件信息区

文件指针是缓冲文件系统中的一个重要的概念。在文件打开时，在内存建立一个文件信息区，用来存放文件有关信息。这个信息区的数据组织成结构体类型，系统把它命名为 FILE 类型。文件指针是指向 FILE 类型数据的，具体来说就是指向某一文件信息区的开头。通过这个指针可以得到文件的有关信息，从而对文件进行操作。这就是指针指向文件的含义。

35、文件操作的步骤

文件使用前必须“打开”，用完后应当“关闭”。所谓打开，是建立相应的文件信息区，开辟文件缓冲区。由于建立的文件信息区没有名字，只能通过指针变量来引用，因此一般在打开文件时同时使指针变量指向该文件的信息区，以便程序对文件进行操作。所谓关闭，是撤销文件信息区和文件缓冲区，指针变量不再指向该文件。

36、文件读写的方式有哪些

有两种对文件的读写方式，顺序读写和随机读写。对于顺序读写而言，对文件读写数据的顺序和数据在文件中的物理顺序是一致的。对于随机读写而言，对文件读写数据的顺序和数据在文件中的物理顺序一般是不一致的。可以在任何位置写入数据，任何位置读取数据。

第一章 数据结构引论

什么是数据结构

数据结构是指按一定的逻辑结构组成的一批数据，按某种存储结构将这批数据存储于计算机中，并在这批数据上定义了一个运算集合。

数据结构涉及哪几个方面?

涉及三个方面，即数据的逻辑结构，数据的存储结构，以及数据的运算集合

1.2.1 数据、 数据元素、 数据项和数据对象

数据（Data）是客观事物的符号表示，是所有能输入到计算机中并被计算机程序处理的符号 的总称。

数据元素(Data Element)是数据的基本单位，在计算机中通常作为一个整体进行考虑和处理。 在有些情况下，数据元素也称为元素、记录等。

数据项（Data Item）是组成数据元素的、有独立含义的、不可分割的最小单位。

数据对象（Data Object）是性质相同的数据元素的集合，是数据的一个子集。

数据类型和抽象数据类型

数据类型：一组性质相同的值的集合以及定义在这个集合上的一组操作的总称

抽象数据类型：包括数据对象、数据元

38、抽象数据类型是什么？它有什么特点?

抽象数据类型（Abstract Data Type, ADT）一般指由用户定义的、表示应用问题的数学模型，

以及定义在这个模型上的一组操作的总称，具体包括三部分：数据对象、数据对象上关系的集合以及对数据对象的基本操作的集合。一旦定义了一个抽象数据类型及具体实现，程序设计中就可以像使用基本数据类型那样，十分方便地使用抽象数据类型。抽象数据类型的设计者根据这些描述给出操作的具体实现，抽象数据类型的使用者依据这些描述使用抽象数据类型。

38、**数据的操作类型有哪些?可以分为哪两类?**

基本的操作主要有：插入、删除、更新、查找、排序从操作的特性来分，所有的操作可以归结为两类：加工型操作：改变了（操作之前的）结构的值：引用型操作：即不改变结构的值，只是查询或求得结构的值。上述 5 种操作中除“查找”为引用型操作外，其余都是加工型操作。

两个数据结构的逻辑结构和存储结构都相同，但是它们的运算集合中有一个运算的定义不一样，它们是否可以认作是同一个数据结构？为什么？

【答】：不能，运算集合是数据结构的重要组成部分，不同的运算集合所确定的数据结构是不一样的，例如，栈与队列它们的逻辑结构与存储结构可以相同，但由于它们的运算集合不一样，所以它们是两种不同的数据结构。

38、什么是数据？数据的最小单位是什么？

数据是对客观事物的符号表示，是计算机科学中所有能输入到计算机中并能被计算机程序处理的符号的总称。数据项是数据的最小单位。

38、数据的逻辑结构有哪两种？这两者的区别是什么？

【答案】数据的逻辑结构有以下两大类：线性结构与非线性结构。线性结构有且仅有一个开始结点和一个终端结点，且所有结点都最多只有一个直接前驱和一个直接后继。非线性结构可以有一个或多个开始节点或终结节点。

9、**线性结构的特点是什么？非线性结构的特点是什么？**

【答】：线性结构元素之间的关系是一对一的，在线性结构中只有一个开始结点和一个终端结点，其他的每一个结点有且仅有一个直接前驱和一个直接后继结点。而非线性结构则没有这个特点，元素之间的关系可以是一对多的或多对多的。（补充：线性结构是一个有序数据元素的集合。常用的线性结构有：线性表，栈，队列，双队列，数组，串。非线性结构，其逻辑特征是一个结点元素可能有多个直接前趋和多个直接后继。常见的非线性结构有：二维数组，多维数组，广义表，树（二叉树等）。）

10、数据结构的存储方式有哪几种？

【答】：数据结构的存储方式有顺序存储、链式存储、散列存储和索引存储等四种方式。

算法有那些特性

算法（Algorithm）是为了解决某类问题而规定的一个有限长的操作序列。

一个算法必须满足以下五个重要特性。

（1）有穷性：一个算法必须在执行有穷步后结束。

（2　确定性：算法中的每条指令必须由确切的含义，不能有二义性。

（3）可行性：算法中描述的操作能够通过有限次的基本运算来实现。

（4）输入：一个算法必须有 0 个或者有限个输入。

（5）输出：一个算法必须有 1 个或者有限个输出。

1.4.2 评价算法优劣的基本标准

一个算法的优劣应该从以下几方面来评价。

(1)正确性。在合理的数据输入下，能够在有限的运行时间内得到正确的结果。

(2) 可读性。一个好的算法，首先应便千人们理解和相互交流 ， 其次才是机器可执行性。可读性强的算法有助于人们对算法的理解，而难懂的算法易千隐藏错误，且难千调试和修改。

(3) 健壮性。当输入的数据非法时，好的算法能适当地做出正确反应或进行相应处理，而不会产生一些莫名其妙的输出结果。

(4) 高效性。高效性包括时间和空间两个方面。时间高效是指算法设计合理，执行效率高，可以用时间复杂度来度量；空间高效是指算法占用存储容量合理，可以用空间复杂度来度量。时间复杂度和空间复杂度是衡量算法的两个主要指标。

第五章 算法及复杂度

32、算法分析的两个主要指标是什么？

时间复杂度和空间复杂度是衡量算法的两个主要指标。

13、算法的时间复杂度指的是什么?如何表示?

时间复杂度是关于问题规模的函数，通常用 o（）表示，常见时间复杂度按照数量级递增排列为：o(1)<o(log2n)<o(n)<o(nlog2n)<o(n2)<o(n3)<0(n)<0(2)

14、算法的空间复杂度指的是什么?

【答】：空间复杂度是指算法在执行过程中临时占用的存储空间大小，包括：①算法本身所占用的存储空间；②算法的输入输出数据所占用的存储空间；③算法在运行过程中临时占用的存储空间。

9、试述 0（1)和 0(2)的区别。

根据符号 0 的定义易知 0(1)=0（2)。用 0(1)或 0(2)表示同一个函数时，差别仅在于其中的常数因子。

31、算法与程序有什么区别?

算法是描述一个问题求解的步骤序列，而程序是算法在特定计算机上的实现。算法不依赖于计算机，而程序依赖于特定计算机和特定编程语言。算法必须满足 5 个特性，即有输入、有输出、确定性、有穷性和可行性，而程序可能不满足有穷性

33、什么是算法的稳定性?

在一组待排序记录中，如果存在任意两个相等的记录 A 和 B，且在待排序记录中 A 在 B 前，如果在排序后 A 依然在 B 前，即它们的前后位置在排序前后不发生改变，则称为排序算法为稳定的。

第二章 线性表

顺序表：线性表的顺序存储，逻辑顺序和物理顺序相同。随机访问，存储密度高，插入删除需要移动大量元素。静态顺序表、动态顺序表。

链表：线性表的链式表示，不要求存储地址联系，失去了随机访问的特性。适合插入删除。

单链表：链表结点存放后继指针。访问后继结点时间复杂度为 O(1),前驱为 O(n)。

双链表：链表结点存放前驱和后继指针，这样就克服了单链表不能从后往前遍历的缺陷。prior、next

循环链表：循环单链表、循环双链表。循环单链表需要将尾结点指针指向头结点，这样就可以形成环。循环双链表还需要将头结点的 prior 指针指向尾结点。

61、什么是顺序表? 什么是栈? 什么是队列?

【答】：当线性表采用顺序存储结构时，即为顺序表。栈是一种特殊的线性表，它的特殊性表现在约定了在这种线性表中数据的插入与删除操作只能在这种线性表的同一端进行（即栈顶），因此，栈具有先进后出、后进先出的特点。队列也是一种特殊的线性表，它的特殊性表现在约定了在这种线性表中数据的插入在表的一端进行，数据的删除在表的另一端进行，因此队列具有先进先出，后进后出的特点。

头指针与头结点的区别

头指针：是指向第一个节点存储位置的指针，具有标识作用，头指针是链表的必要元素，无论链表是否为空，头指针都存在。

头结点：是放在第一个元素节点之前，便于在第一个元素节点之前进行插入和删除的操作，头结点不是链表的必须元素，可有可无，头结点的数据域也可以不存储任何信息。

6.顺序表与链表的比较

（1）存取方式：顺序表可以顺序存取，随机存取，链表是能顺序存取。

（2）逻辑结构和物理结构上：顺序存储是逻辑上相邻的元素物理上也相邻，链式存储是逻辑上相邻的元素物理上不一定相邻。

（3）插入删除操作:顺序表插入删除操作平均需要移动半个表长的元素，链表只需要修改相应的指针即可。

（4）空间分配：顺序存储在静态存储分配下需要预先分配足够大的空间，动态存储分配虽然可以扩充空间，但是需要移动大量的元素；链式存储空间在需要时申请即可。

4.简述各存储结构

顺序存储：采用一组连续的存储空间存储数据。

优点：简单，可以实现随机存取，元素占用最少的存储空间。

缺点：只能使用连续的存储单元，会产生较多的外部碎片。

链式存储：数据的存储空间是离散的，借助元素中的指针来表示元素之间的逻辑关系。

优点：不会产生外部碎片，能充分利用所有存储单元。

缺点：只能顺序存取，每个元素因为存储指针而占用额外的存储空间。

索引存储：在存储元素数据的同时还建立附加的索引表。

优点：检索速度快。
缺点：索引表会占用额外的存储空间。
散列存储：根据元素的关键字直接得出元素的存储地址。
优点：检索，增加，删除元素的速度都很快。
缺点：散列函数选择不好可能会出现元素单元的冲突，而解决冲突需要额外的开销。

第三章栈

38、什么是递归?

一个直接调用自身或通过一系列过程调用语句间接地调用自身的过程，称做递归过程。

39、什么是栈？栈的元素存取遵循什么原则?

栈是线性表，被限定仅在栈顶进行插入或删除操作。栈中数据元素同属一种数据类型，数据元素之间呈线性关系。栈的元素存取遵循先进后出原则

第四章 队

39、什么是队列？队列的元素存取遵循什么原则?

【答案】队列是一种先进先出（FIFO）的线性表。它只允许在表的一端插入元素，而在另一端删除元素。允许插入的一端叫做队尾（rear），允许删除的一端则称为队头（front)。队列中数据元素同属一种数据类型，数据元素之间呈线性关系。队列的元素存取遵循先进先
出原则。

8.栈和队列的区别

队列和栈都是操作受限的线性表，队列是只允许在一段插入，在另一端删除的线性表，进入队列的元素按先入先出的原则进行处理，在层次遍历和 BFS 算法、狄杰斯特拉算法中使用到
；栈是指能在表尾进行插入和删操作的线性表，对于插入到栈的元素按先进后出的规则进行处理，插入和删除操作都在栈顶进行

8、如何实现以任意长字符串为元素的队列？将一个字符串入队的运算耗时如何?

在队列中存储字符串指针可以使队列中元素大小相同。按此方法在输入长度为 m 的字符
串时需要 O(m)时间，而入队运算只要 0（1）时间

62、循环队列存储在一个数组中，数组大小为 n，队首指针和队尾指针分别为 front 和 rear，请写出求循环队列中当前结点个数的表达式。

【答】：循环队列中当前结点个数的计算公式是：(n+rear−front)%n

14.队列在层次遍历中的作用：

答：首先根结点入队，接着队根结点的子结点进行预处理，等预处理完后，根结点出队，接着刚刚处理的子结点入队，这部分的子结点又进行预处理，直到所有的结点都入队出队处理完毕。

串、数组、广义表（以防万一考到）

串(string)(或字符串）是由零个或多个字符组成的有限序列
数组是由类型相同的数据元素构成的有序集合，每个元素称为数组元素，每个元素受 n(n? 1）个线性关系的约束，每个元素在 n 个线性关系中的序号儿 i2, …，in 称为该元素的下标，可 以通过下标访问该数据元素。
广义表：顾名思义，广义表是线性表的推广，也称为列表。

第六章 树

39、什么是树?

树(Tree)是 n(大于等于 0)个结点组成的有限集，当 n= 0 称为空树；当 n>0 时为非空树，
对于非空树 T:
(1)有一个特定的称为根（root）的结点。它只有直接后继，没有直接前驱。
(2)除根结点以外的其它结点可分为 m（m>=O）个互不相交的有限集 Ti , T2 , …，Tm，其中每 一个集合本身又是一棵树，并称为根的子树，每棵子树的根结点有且只有一个直接前驱，但可以有 0 个或多个直接后继。

39、什么是二叉树？请简述二叉树的五种基本形态

二叉树是每个结点最多有两个子树的有序树。通常子树的根被称作“左子树”(left subtree)和“右子树”(right subtree)。二叉树常被用作二叉查找树和二叉堆或是二叉排序树。(2)二叉树也是递归定义的，其结点有左右子树之分，

逻辑上二叉树有五种基本形态：

(1)空二叉树

(2)只有一个根结点的二叉树

(3 只有左子树

(4)只有右子树

(5) 完全二叉树

39、什么是森林?

m（m 大于等于 0)棵互不相交的树的集合。

13.名词解释，满二叉树，完全二叉树，二叉排序树，平衡二叉树。

满二叉树：高度为 H，结点数为 2^H–1 的二叉树为满二叉树。

完全二叉树：除最后一层外，其余各层的节点数量达到最大值，并且最后一层只能在右侧缺少节点。

二叉排序树：左子树上所有的关键字均小于根结点，右子树上所有关键字均大于根结点。左子树和右子树又分别是一棵二叉排序树。

平衡二叉树：树中每一个结点的左子树，右子树高度之差的绝对值小于等于 1

14.二叉树的存储结构

顺序存储：用一组连续的地址单元自上到下，自左到右的存储完全二叉树的结点元素。

链式存储：采用二叉链表来存储树的每个节点。

15.线索二叉树

将二叉链表中的空指针改成指向前驱节点或后继的线索。线索链表解决了无法直接找到该结点在某种遍历序列中的前驱和后继结点的问题，解决了 二叉链表找左、右孩子困难的问题。

16.树的存储结构

双亲表示法：采用一组连续的存储空间存储每个结点，同时在每个结点后面增设一个伪指针指向其双亲节点。

孩子表示法：将每个结点的孩子结点用单链表链接起来形成一个线性结构。

孩子兄弟表示法：以二叉链表作为树的存储结构，其左指针指向第一个孩子结点，右指针指向其相邻的兄弟结点。可以方便地实现树转换为二叉树的操作，易于查 找结点的孩子等，但缺点是从当前结点查找其双亲结点比较麻烦。

17.哈夫曼树

权：树中的结点往往被赋予一个有意义的数值称为该结点的权。

结点的带权路径长度：从树的根到任意节点的路径长度与该结点的权值之积称为该结点的带权路径长度。

树的带权路径长度：树中所有叶结点的带权路径之和为该树的带权路径长度。

哈夫曼树：带权路径长度最小的树为哈夫曼树。

78、从概念上讲，树、森林和二叉树是三种不同的数据结构，将树、森林转化为二叉树的基本目的是什么？并指出树和二叉树的主要区别。

（1）基本目的

树的孩子兄弟链表表示法和二叉树的二叉链表表示法本质是一样的，只是解释不同，也就是说树（树是森林的特例，即森林中只有…棵树的特殊情况）可用二叉树唯·表示，并可使用二叉树的一些算法去解决树和森林中的问题。

（2）主要区别

一是二叉树的度至多为 2，树无此限制；二是二叉树有左右子树之分，即使在只有一个分支的情况下，也必须指出是左子树还是右子树，树无此限制；三是二叉树允许为空，树一般不允许为空（有些书上考虑到与二叉树的转

换，允许树为空)。

79、设有一棵算术表达式树，用什么方法可以对该树所表示的表达式求值?

有两种方法，具体如下：

对该算术表达式（二叉树）进行后序遍历，得到表达式的后序遍历序列，再按后缀表达式求值。递归求出左子树表达式的值，再递归求出右子树表达式的值，最后按根结点运算符（+、一、*、/等）进行求值。

第七章 散列表

31、什么是冲突?

散列函数可能会把两个或者两个以上的不同关键字映射到同一地址，这种情况叫“冲突”。

32、什么是散列函数?

一种将查找表中的关键字按函数关系映射成对应地址的函数称为哈希函数，又称散列函数。是根据关键码值 (Key value）而直接进行访问的数据结构。也就是说，它通过把关键码值映射到表中一个位置来访问记录，以加快查找的速度。

33、解决散列法中出现的冲突问题常采用的方法是?

线性探测法、双散列法、链地址法。

34、什么是直接定址法

取关键字或关键字的某个线性函数值为哈希地址。即：H(key)=key 或 H（key）=a*key+b。其中 a 和 b 为常数（这种哈希函数称作自身函数)，它适合关键字分布基本连续的情况

37.散列表

散列函数：一个把查找表中关键字映射成为该关键字对应地址的函数。

散列表：根据关键字直接进行访问的散列表，建立了关键字与存储地址的直接映射关系。

散列函数的构造方法：

（1）直接定址法：直接取关键字的某个线性函数值为其存储地址。

（2）除留余数法：假定散列表表长为 m，取一个不大于 m 且最接近 m 的质数 p，通过关键字取余 p 来得到关键字的存储地址。

（3）平方取中法：取关键字平方的中间几位作为该关键字的存储地址。

解决冲突的办法：

（1）开放地址法：

1.线性探测法：冲突发生时顺序查看表中下一个单元，直到找出一个空闲单元为止。(可能会造成大量元素在相邻的散列地址上堆积，降低查找效率)

2.二次探测法：令增量序列为 0,1 的平方，–1 的平方……（能够避免出现堆积问题，但是不能够探测到散列表中的所有单元)

3.伪随机探测法

2. 链地址法的基本思想是：把具有相同散列地址的记录放在同一个单链表中，称为同义词链表。有个散列地址就有 m 个单链表，同时用数组 HT[0…m–1]存放各个链表的头指针，凡是

第八章 优先队列

31、什么是优先队列?

在优先队列中，队列中的数据被赋予了优先级。当访问元素时，优先级最高的会先被删除。优先队列是最高级数据先出。

9、说明如何用优先队列来实现栈和队列。

对每个元素用其插入次序作为其优先级。按此优先级，栈是一个极大化堆，而队列是一个极小化堆。

33、为什么要用堆来实现优先队列?

优先队列所需要实现的两种操作，不同于队列和栈，它需要一个有序的元素序列，但不要求全部有序，只需要从这些元素中找到最大（或最小）的一个元素。而堆刚好满足这个条件，而插入元素这个操作，即是由下至上的堆

有序化。队列，栈都是用数组或者链表来实现的，针对优先队列，用数组和链表实现也是可以的，在队列较小，大量使用两种操作之一时，或者所操作的元素的顺序已知时，用数组和链表十分有用，但是，在最坏的情况下，优先队列用这两张方法实现所需的时间却是线性的。而用堆在最坏情况下的时间则是对数级别

34、什么是堆?
当一颗完全二叉树（用一维数组做树的存储结构）的每个节点都大于等于（小于等于)它的两个子节点时。它被称为堆。

第九章
38.排序
内部排序：排序期间元素全部存放在内存中。(插入，交换，选择，归并，基数)
外部排序：排序期间元素无法同时存放在内存中，必须在排序期间根据要求不停地在内，外存移动。(多路归并)

39.插入排序
直接插入排序：每次将一个待排序的记录按其关键字的大小插入到已排好序的子序列中。(1.从前面的有序子表中查出待插入元素应该插入的位置 2.将已排序的记录逐步向后移动，给待插入元素腾出位置，并将待插入元素复制到插入位置。 适用顺序存储/链式存储)

折半插入排序：如果是顺序存储的线性表，可以通过折半查找的方式来查找待插入元素在有序子序列的位置。确定待插入位置之后，可以统一的向后移动位置。

希尔排序：将待排序列按相隔某个增量分割成若干个子序列，对各个子序列进行直接插入排序，逐渐缩小增量，重复上述步骤，直到序列基本有序，再对全体记录进行一次直接插入排序。

40.交换排序
冒泡排序：从前到后（从后往前）依次两两比较相邻元素的值，若为逆序，就交换元素，每一趟排序完成之后，就有一个元素被放在最终位置上，重复上述步骤，当一趟排序不发生任何元素的交换为止。
快速排序：每次从待排序列中选择一个元素作为枢轴（通常是序列的首元素），把比枢轴小的元素放前面，比枢轴大的元素放后面，最后确定枢轴元素的最终位置并将枢轴放入，再对枢轴前后得到的子序列再重复上述步骤，直到每部分只有一个元素或为空为止，则所有元素放在最终位置上。

41.选择排序
简单选择排序：每趟排序选择关键字最小的元素与序列前面的元素进行交换，每次排序均可确定一个元素的最终位置。
堆排序：先将待排元素建成初始堆，以大根堆为例，堆顶元素为最大值，将其输出后，把堆底元素送入堆顶，此时堆不满足大根堆的性质，将堆顶元素向下调整（从堆的最后一个非叶子节点开始，从左到右，从下到上的顺序进行调整），成为大根堆之后再输出堆顶元素，重复上述过程，直到输出所有元素。

42.归并排序
“归并”：就是将两个或两个以上的有序表组成一个新的有序表。
假定待排序表有 n 个记录，将其视为 n 个有序的子表，然后两两归并得到 n/2 个有序子表，再进行两两归并，直到合并成一个长度为 n 的有序表为止。

43.基数排序
创建 0~9 的十个数组，将待排序表的所有元素先按个位进行分类，将分类后的元素按索引大小取出形成新的队列，再对队列按十位，百位的顺序进行分类，重复上述过程，最后形成一个有序序列。

44.选取排序算法需要考虑的因素
(1) 待排元素的数目；
(2) 稳定性的要求；
(3) 元素本身信息量的大小；

（4）关键字的结构及其分布情况；
若 n 较小，可直接采用直接插入排序或者简单选择排序。由于直接插入排序比简单选择排序移动的元素要多，所以当元素本身信息量比较大时可选用简单选择排序。
若待排序表已基本有序可以选用直接插入排序或者冒泡排序。
当 n 比较大时，可以选用快速排序、堆排序、归并排序。快速排序被认为是当前基于比较的内部排序中最好的排序方法，当待排序表记录随机分布时，使用快速排序速度最快。堆排序所用存储空间少于快速排序，且不会出现快速排序的最坏情况，这两种算法都是不稳定的，若要稳定排序则选用归并排序算法。

45.外部排序
采用多路归并法，包括两个相对独立的阶段：
（1）根据内部缓冲区的大小，将待排文件分成若干个大小合适的子文件，将子文件带入内存采用内部排序算法排序完成后再写回外存。
（2）对这些归并段进行逐趟归并，使归并段逐渐由小到大，直到得到整个有序文件为止。

46.提高外部排序算法的效率
由于待排文件无法全部放入内存，所以排序期间必须要频繁的进行内外存之间数据的交换，这会耗费大量的时间。所以可以通过增加归并路数来减少归并趟数，进而减少 I/O 次数。而增加归并路数又会增加内部排序的时间，所以引入了败者树。
增加初始归并段个数，并且不受内存空间的限制，引入了置换–选择算法。
文件经过置换–选择算法之后得到的是长度不同的初始归并段，如何组织长度不等的出使归并段的归并顺序，使得 I/O 次数最少，就引入了最佳归并树。

47.败者树（大的为失败者，小的为胜利者）
可视为一棵完全二叉树，每个叶结点存放各归并段在归并过程中参加比较的记录，非叶结点用来记录左右子树中的“失败者”，胜利者继续向上比较直到根节点。输出最后的胜利者。

48.置换选择算法
根据缓冲区的大小，由外存读入记录，当记录充满缓冲区时，选择最小的输出，其空缺位置由下一个记录来取代，输出记录称为当前初始归并段的一部分，如果新输出的记录比新建立归并段最大的记录小，就不能成为该归并段的一部分，只能成为下一个归并段的选择。重复上述步骤，直到缓冲区中所有记录都比当前归并段最大记录小时，就生成了一个初始归并段，用同样的方法继续生成下一个归并段，直到全部记录都处理完毕为止。

49.最佳归并树
对于 K 路归并算法，可用构造 K 叉哈夫曼树的方法来构造最佳归并树。

第十章 图

39、什么是图?

图(Graph)是由顶点的有穷非空集合和顶点之间边的集合组成，通常表示为：G(V，E)，其中，G 表示一个图，V 是图 G 中顶点的集合，E 是图 G 中边的集合。

19.图的名词解释

连通图：在无向图中如果两顶点之间有路径存在，就称这两个顶点是连通的。如果无向图中任意两个顶点是连通的，就称图为连通图。
极大连通子图（连通分量）：该连通子图包含所有的边。
极小连通子图：在保证连通的情况边最少的子图。
强连通图：在有向图中，如果顶点 m 到顶点 n 有路径存在且 n 到 m 也有路径存在，就称这两个顶点是强连通的。图中任何两个顶点都是强连通的，该图就是强连通图。
强连通分量：有向图中的极大强连通子图为强连通分量。
生成树：连通图中包含所有顶点的极小连通子图。
网：在图中每条边都可以标上具有某个意义的数值，称为该边的权值。边上带权值得图为网，

20.图的存储方式

邻接矩阵法：用一个一维数组存储图的顶点信息，用二维数组存储各顶点的邻接关系。存储顶点邻接关系的二维数组称为邻接矩阵。
邻接表法：图中每个顶点与其有邻接关系的顶点拉成一个单链表，每个顶点都有一个单链表。
十字链表法：十字链表法是有向图的一种链式存储结构。在十字链表中，有向图中的每一条弧都有一个对应的节点，每个顶点都有对应的一个节点。
邻接多重表法：

3. 邻接矩阵表示法的优缺点
3、.邻接超阵表示法的优缺点
⑴ 优点
①便于判断两个顶点之间是否有边，即根据 A[i][j]=0 或 1 来判断。
②便于计算各个顶点的度。对于无向图、邻接矩阵第/行元素之和就是顶点 i 的度；对于有
向图、第行元素之和就是顶点/的出度、第/列元素之和就是顶点/的入度。
（2）缺点
①不便于增加和删除顶点。
②不便于统计边的数目、需要扫描邻接矩阵所有元素才能统计完毕，时间复杂度为 0（n）。
③空间复杂度高。如果是有向图、n 个顶点需要 n 个单元存储边。如果是无向图，因其邻接矩阵是对称的、所以对规模较大的邻接矩阵可以采用压缩在储的方法，仅存储下三角(或上三角）的元素，这样需要减（m–1）/2 个单元即可。但无论以何种方式存储，邻接矩阵表示法的空间复杂度均为 O（n），这对于秘疏图而言尤其浪费空间。

21.广度优先搜索
类似于层次遍历，先访问起始顶点，在访问与其相邻的所有顶点，再顺序访问与这些顶点相邻的顶点，重复上述过程，知道图中所有顶点都被访问过为止。

22.深度优先搜索
类似于先序遍历，从起始顶点出发，先访问与其相邻的顶点，再访问与该顶点相邻且未被访问过的顶点，重复上述步骤，当与其相邻的所有顶点都被访问过，依次回退曾经访问过的顶点，若某个顶点还有与其相邻且未被访问过的顶点，则从该点开始重复上述过程，直到所有顶点均被访问过。
遍历
先序遍历：根左右
中序通历:左根右
后序遍历：左右根
层次遍历（一层一层，利用队列）

最小生成树
普里姆算法：从某个顶点出发，找与该顶点相邻的全值最小的边，并把这条边的顶点加入到顶点集 S 中，再分别从顶点集 S 内和 S 外找一个连接起来权值最小的表，也把该顶
点加入进来，直到所有顶点都加入进来了
克鲁斯卡尔算法：首先是一个具有 n 个顶点，无边的非连通图，再选出权值最小并且不能构成环的边，加入进来，直到选了 n–1 条边

26.最短路径
当图是带权图时，把从一个顶点到图中任意一个顶点的路径所经过的边的权值之和称为该路径的带权路径长度。把带权路径最短的那条路径称为最短路径。
最短路径算法:
弗洛伊德算法：求任意两个顶点之间的最短路径，权值可为负，利用动态规划的思想。先找到的最短路径会直接影响之后找到的最短路径，有三层循环，时间复杂度
是 O（n 的三次方 3），空间复杂度 O（n 的 2）适用于稠密图
狄杰斯特拉算法：求单源最短路径，权值不能为负，利用贪心策略，在已经求得最短路径的基础上，求更长距离的最短路径，适合于稀疏图

29.拓扑排序

AOV 网：用顶点表示活动，有向边表示活动之间的关系的有向图记为 AOV 网。在 AOV 网中选择一个没有前驱的顶点输出，并删除该顶点和以该顶点为起点的所有有向边，重复上述过程直到 AOV 网为空或者不存在无前驱的顶点为止。

30.关键路径

在带权有向图中，用顶点表示时间，以有向边表示活动，以边上的权值表示完成该活动所需要的时间，称为 AOE 网。

AOV 网与 AOE 网的区别：

AOV 网和 AOE 网都是有向无环图，区别在于它们的顶点和边所代表的含义是不同的，AOE 网中的边有权值，AOV 网的边无权值，仅代表活动之间的关系。

关键路径：从源点到汇点的所有路径中，具有最大路径长度的路径称为关键路径，在这条路径上活动称为关键活动，完成整个工程的最短时间就是关键路径的长度。