第一部分 C语言知识复习资料

第一章 C语言基本知识（90分）

【考点1】C程序

用C语言编写的程序称为C语言源程序，源程序文件的**后缀名为".c"。**源程序经编译后生成后缀名为".obj"的目标文件，再把目标文件与各种库函数连接起来，生成".exe"可执行文件。C语言有三种基本结构：顺序结构、选择结构、循环结构。

【考点2】main函数

又称主函数，是C程序的入口。main后面跟一对小括号和一对花括号，花括号括起来的部分称为main函数的函数体。一个C程序从main函数开始执行，到main函数体执行完结束，而不论main函数在整个程序中的位置如何。每一个程序有且仅有一个main函数，其他函数都是为main函数服务的。

【考点3】存储形式

计算机在电脑中保存数据是采用二进制形式，由0或1构成的二进制称为位（bit），八个位构成一个字节（Byte），1个Byte=8个bit。二进制、八进制、十六进制转化为十进制采用乘法，十进制转化为二进制、八进制、十六进制采用除法。数据的存放位置就是它的地址。

【考点4】注释

是对程序的说明，可出现在程序中任意合适的地方，注释从"/*"*开始到最近一个*"*/*"结束，其间任何内容都不会被计算机执行，注释不可以嵌套。

【考点5】书写格式

每条语句的后面必须有一个分号，分号是语句的一部分。一行内可写多条语句，一个语句可写在多行上。

【考点6】标识符

是标识名字的有效字符序列，可以理解为C程序中的单词。

标识符的命名规则是：

（1）标识符只能由字母、数字和下划线组成，字母区分大小写。

（2）标识符的第一个字符必须是字母或下划线，不能为数字。

C语言标识符分如下3类

（1）关键字。它们在程序中有固定的含义，不能另作他用。如int、for、switch等。

（2）预定义标识符。预先定义并具有特定含义的标识符。如define、include等。

（3）用户标识符。用户根据需要定义的标识符，符合命名规则且不与关键字相同。

【考点7】常量与变量

常量是指在程序运行过程中，其值不能改变的量。常量分为整型常量、实型常量、字符常量、字符串常量、符号常量5种。在程序运行过程中其值可以改变的量称为变量。C语言中没有字符串变量。存放字符串使用字符数组。

【考点8】整型数据

整型常量有十进制、八进制、十六进制三种表示形式，*没有二进制形式。八进制整型常量加前导数字0，十六进制常量加前导0X，八进制常量中不会出现8。

整型变量可分为基本整型（int）、短整型（short）、长整型（long）、和无符号整型（unsigned）。一个基本整型占4个字节。其它类型的整型占用字节数和取值范围详见教材第9页。

【考点9】实型数据

实型数据有两种表示形式：小数形式和指数形式。掌握判定指数形式合法性。

口诀：E前E后必有数，E后必须为整数。

实型变量分为单精度型（float）和双精度型（double），单精度型占四个字节。

【考点10】算术运算

算术运算符一共有+、—、*、/、%这五个。求余运算要求运算对象只能为整型，除法运算符两边运算对象都为整型时，运算结果也为整型即舍掉小数部分。

【考点11】强制类型转换

将一个运算对象转换成指定类型，格式为（类型名）表达式，注意小括号位置。

【考点12】赋值

赋值运算符为“=”，不同于关系等于“= =”。赋值表达式格式为：变量名=表达式，赋值运算符左边必须为变量，赋值运算是把赋值运算符右边表达式的值赋给左边变量。

复合赋值运算符是将算术运算符或位运算符与赋值运算符组合在一起组成的运算符，掌握复合赋值表达式转化为赋值表达式的方法。如n+=100可转化为n=n+100。

【考点13】自加自减运算

自加运算符“++”与自减运算符“--”是单目运算符，运算对象必须是变量。自增自减运算分前缀运算和后缀运算，它们所对应的表达式的值是有区别的，如j=i++;等价于j=i;i=i+1;而j=++i;等价于i=i+1;j=i;。

口诀：加加在前先加后用，加加在后先用后加。

【考点14】逗号运算

逗号运算符运算优先级最低，可将多个表达式构成一个新的表达式。

考试真题：

1、下列叙述中错误的是（ **D** ）——2006年4月选择第44题

A）C语言源程序经编译后生成后缀为.obj的目标程序

B）C语言经过编译、连接步骤之后才能形成一个真正可执行的二进制机器指令文件

C）用C语言编写的程序称为源程序，它以ASCII代码形式存放在一个文本文件中

D）C语言中的每条可执行语句和非执行语句最终都将被转换成二进制的机器指令

2、下列叙述中错误的是（ **B** ）——2006年4月选择第45题

A）算法正确的程序最终一定会结束

B）算法正确的程序可以有零个输出

C）算法正确的程序可以有零个输入

D）算法正确的程序对于相同的输入一定有相同的结果

3、下列叙述中错误的是（**A**）——2006年9月选择第11题

A）一个C语言程序只能实现一种算法

B）C程序可以由多个程序文件组成

C）C程序可以由一个或多个函数组成

D）一个C函数可以单独作为一个C程序文件存在

4、下列叙述中正确的是（**D**）——2006年9月选择第12题

A）每个C程序文件中都必须要有一个main（）函数 算法正确的程序对于相同的输入一定有相同的结果

B）在C程序中main（）函数的位置是固定的

C）C程序中所有函数之间都可以相互调用，与函数所处位置无关

D）在C程序的函数中不能定义另一个函数

5、在算法中，对需要执行的每一步操作，必须给出清楚、严格的规定，这属于算法的（**C**）——2007年4月选择第11题

A）正当性

B）可行性

C）确定性

D）有穷性

6、下列叙述中错误的是（**D**）——2007年4月选择第12题

A）计算机不能直接执行用C语言编写的源程序

B）C程序经C编译程序编译后，生成后缀为.obj的文件是一个二进制文件

C）后缀为.obj的文件，经连接程序生成后缀为.exe的文件是一个二进制文件

D）后缀为.obj和.exe的二进制文件都可以直接运行

7、下列叙述中错误的是（**C**）——2007年4月选择第14题

A）C语言是一种结构化程序设计语言

B）结构化程序由顺序、分支、循环3种基本结构组成

C）使用3种基本结构构成的程序只能解决简单问题

D）结构化程序设计提倡模块化的设计方法

8、对于一个正常运行的C程序，下列叙述中正确的是（**A**）—2007年4月选择第15题

A）程序的执行总是从main函数开始，在main函数结束

B）程序的执行总是从程序的第一个函数开始，在main函数结束

C）程序的执行总是从main函数开始，在程序的最后一个函数中结束

D）程序的执行总是从程序中的第一个函数开始，在程序的最后一个函数结束

9、C语言源程序名的后缀是（**B**）——2007年9月选择第11题

A）.exe **B）.c** C）.obj D）.cp

10、下列叙述中正确的是（**C**）——2007年9月选择第14题

A）C语言程序将从源程序中第一个函数开始执行

B）可以在程序中由用户指定任意一个函数作为主函数，程序将从此开始执行

C）C语言规定必须用main作为主函数名，程序将从此开始执行，在此结束

D）Main可作为用户标识符，用以命名任意一个函数作为主函数

\11. 以下选项中合法的标识符是（**C**）（2009年3月）

A）1_1 B）1-1 **C）_11** D）1_

\12. 以下选项中不合法的标识符是**(C)**（2008年4月）

A）print B）FOR **C）&a** D）_00

\13. 可在C程序中用做用户标识符的一组标识符是（**A**）（2007年9月）

A）and B）Date C）Hi D）case _2007 y-m-d Dr.Tom Bigl

\14. 按照C语言规定的用户标识符命名规则，不能出现在标识符中的是**(B)**（2007年4月）

A）大写字母 **B）连接符** C）数字字符 D）下划线

\15. 以下不合法的用户标识符是**(C)** （2006年4月）

A）j2_KEY B）Double **C) 4d** D) *8*

\16. 以下选项中不属于字符常量的是 **(B)**（2008年4月）

A）'C' **B）"C"** C）'\xCC0' D）'\072'

\17. 以下合法的字符常量的是 **(A)**

A）'\x13' B）"\081" C）'65' D）”\n”

\18. 已知大写字母A的ASCII码是65，小写字母a的ASCII码是97。下列不能将变量c中的大写字母转换为对应小写字母的语句是 **（A）**（2007年4月）

A）c=(c-„A")%26+"a" B）c=c+32 C）c=c-„A" +"a" D）c=(„A"+c)%26-„a"

\19. 以下选项中，值为1的表达式**B**。（2006年9月）

A）1-„0" **B）1-„\0"** C）"1"-0 D）"\0"-„0"

\20. 以下选项中，能用作数据常量的是 **（ D ）**（2009年3月）

A）o115 B）0118 C）1.5e1.5 **D） 115L**

\21. 以下选项中不能作为C 语言合法常量的是**（B）**。

A）'cd' **B）0.1e+6** C）"\a" D）'\011'

\22. 以下不合法的数值常量是 **(C)**

A）011 B）lel **C）8.0 E0.5** D）0xabcd

\23. C 源程序中不能表示的数制是 **（ C ）**。（2008年9月）

A）二进制

B）八进制

C）十进制

D）十六进制

\24. 以下关于long、int和short类型数据占用内存大小的叙述中正确的是 **(D)**（2007年9月）

A）均占4个字节

B）根据数据的大小来决定所占内存的字节数

C）由用户自己定义

D）由C语言编译系统决定

\25. 以下选项中，合法的一组C语言数值常量是 **(B)**（2007年9月）

A．028 **B．12.** C．177 D.0x8A

5 e-3 **0Xa23** 4e1.5 10,000

-0xf **4.5e0** 0abc 3.e5

第二章 顺序结构

【考点1】运算符、表达式、语句

运算对象加运算符构成表达式，表达式加分号构成表达式语句，运算对象可以是表达式、常量、变量。如赋值运算符加运算对象构成赋值表达式，赋值表达式加分号又可构成赋值语句。

【考点2】运算符的优先级和结合顺序

运算符按参加运算的对象数目可分为单目运算符、双目运算符和三目运算符。初等运算符的优先级别最高，然后依次是单目运算符、算术运算符、关系运算符、逻辑运算符（除逻辑非！）、条件运算符、赋值运算符、逗号运算符。位运算符优先级介于算术运算符与逻辑运算符之间。结合顺序大多为自左向右，而自右向左的有三个：单目运算符、条件运算符和赋值运算符。

【考点3】printf函数

格式为：printf(输出控制，输出列表)。输出控制是用一对双引号括起来的，包含格式说明和原样信息。输出列表包含若干输出项。

【考点4】printf函数中格式说明

%d对应整型，%f对应单精度实型，%c对应字符型，%o对应八进制无符号整型，%x对应无符号十六进制整型，%u对应无符号整型，%e对应指数型，%s对应字符串型。可在%和格式字符之间加一个数来控制数据所占的宽度和小数位数。

【考点5】scanf函数

输入项要求带取地址符&。当用键盘输入多个数据时，数据之间用分隔符。分隔符包括空格符、制表符和回车符，但不包括逗号。

【考点】6如何交换两个变量

要使用中间变量，语句为：t=x; x=y; x=t;。

第三章 选择结构

【考点1】关系运算

C语言用非0表示逻辑真，用0表示逻辑假。关系运算符有6个，分别是>，>=，<，<=，==，!=，前四种优先级高于后两种。关系表达式真时为1，假时为0。注意a<b<c是不可以的，可用(a<b)&&(b<c)来表示。

【考点2】逻辑运算

逻辑运算符共有3个：逻辑与（&&），逻辑或（||），逻辑非（!）。注意短路现象，例a++||b++，如果表达式a++的值非零，则表达式b++不再执行。

【考点3】if语句

可以单独出现，也可以与else匹配出现。if语句可以嵌套，这时else总是与离它最近的且没有与else匹配的if匹配。

【考点4】条件运算

是唯一的三目运算符，格式为：表达式1?表达式2:表达式3。表达式1值为非0时，整个表达式值为表达式2的值，表达式1值为0时，整个表达式值为表达式3的值。

口诀：真前假后

【考点5】switch语句

格式及执行过程详见教材P33，要注意每条case后有没有break语句的区别。还要注意switch后小括号**里面的表达式不能为实型，case后表达式不能有变量。**

口诀：switch表不为实，case表不为变。

第四章 循环结构

【考点1】三种循环结构

三种循环结构分别为：while，do-while，for，三种结构的格式及执行顺序详见教材第36、39、40页。注意for循环中的小括号中必须是两个分号；循环一定要有结束条件，否则成了死循环；do-while()循环最后的while()；后一定要有分号。

【考点2】break与continue

break是终止所在整个循环，而continue是提前结束本轮循环。break语句可出现在**循环结构与switch**语句中，continue只出现在循环结构中。

【考点3】循环的嵌套

就是循环里面还有循环，计算要一层一层分析，一般只考查两层嵌套，循环嵌套通常是处理二维数组。

【考点4】循环结构的复习

循环结构是重点，笔试所占分值一般在13分左右，在上机考试中也是必考点，应用性很强。要求学员重点理解并多加练习，领会掌握。

第五章 字符型数据 位运算

【考点1】字符常量

一个字符常量用一对单引号括起来，字符常量只能包括一个字符，'ab'是非法的。空格常用'□'来表示。字符常量可用对应的ASCII码表示，需记住：'0'的ASCII码为48，'A'的ASCII码为65，'a'的ASCII码为97。

【考点2】转义字符

一对单引号中以一个反斜线后跟一个特定字符或八进制、十六进制数来构成转义字符。比如'\n'表示换行，'\101'或'\x41'表示ASCII码为65的字符'A'。

【考点3】字符型数据可以和整型数据相互转换

如：'0'-0=48 'A'+32='a' char a=65;printf("%d%c",a,a);结果为65A

【考点4】位运算符

C语言提供6种位运算符：**按位求反~，按位左移<<，按位右移>>，按位与&，按位异或|，按位或^。**一般情况下需要先转化进制。异或运算的规则：0异或1得到1，0异或0得到0，1异或1得到0。可记为"**相同为0，不同为1"。**

【考点5】putchar与getchar函数

可用于输出或输入单个字符，这两个函数是stdio.h文件中的库函数，它们是printf与scanf函数的简化。

第六章 函数

【考点1】函数的定义

函数是具有一定功能的一个程序块。函数的首部为：**函数类型 函数名（类型1 形参1，类型2 形参2，……）。在函数定义中不可以再定义函数，即不能嵌套定义函数。函数类型默认为int型。**

【考点2】库函数

调用C语言标准库函数时要包含include命令，include命令行以#开头，后面是""或<>括起来的后缀为".h"的头文件。以#开头的一行称为编译预处理命令行，编译预处理不是C语言语句，不加分号，不占运行时间。

【考点3】函数的返回值

函数通过return语句返回一个值，返回的值类型与函数类型一样。return语句只执行一次，执行完或函数体结束后退出函数。

【考点4】函数的声明

函数要"先定义后调用"，或"先声明再调用后定义"。**函数的声明一定要有函数名、函数返回值类型、函数参数类型，但不一定要有形参的名称。**

【考点5】函数的调用

程序从上往下执行，当碰到函数名后，把值传给调用函数，当程序得到了返回值或调用函数结束，再顺序往下执行。

【考点6】函数的参数及值传递

形式参数简称形参，是定义函数时函数名后面括号中的参数。实在参数简称实参，是调用函数时函数名后面括号中的参数。实参和形参分别占据不同的存储单元。实参向形参单向传递数值。

"传值"与"传址"的区别：传数值的话，形参的变化不会改变实参的变化。传地址的话，形参的变化就有可能改变实参所对应的量。

【考点7】函数的递归调用

函数直接或间接地调用自己称为函数的递归调用。递归调用必须有一个明确的结束递归的条件。在做递归题时可把递归的步骤一步步写下来，不要弄颠倒了。

【考点8】要求掌握的库函数
sqrt()算术平方根函数，fabs()绝对值函数，pow()幂函数，sin()正弦函数

第七章 指针

【考点1】指针变量
【考点2】指针变量的定义指针变量是用来存储地址的，而一般变量是存储数值的。指针变量可指向任意一种数据类型，但不管它指向的数据占用多少字节，一个指针变量占用四个字节。

格式为：类型名 *指针变量名。二维指针int **p;可以理解为基类型为(int *)类型。**
【考点3】指针变量的初始化*
指针变量在使用前必须要初始化，把一个具体的地址赋给它，否则引用时会有副作用，如果不指向任何数据就赋"空值"NULL。
【考点4】指针变量的引用
*&是取地址符，*是间接访问运算符，它们是互逆的两个运算符。在指针变量名前加间接访问运算符就等价它所指向的量。
【考点5】指针的运算
***p++和(*p)++之间的差别：*p++是地址变化，(*p)++是指针变量所指的数据变化。一个指针变量加一个整数不是简单的数学相加，而是连续移动若干地址。当两个指针指向同一数组时，它们可以比较大小进行减法运算。**

第八章 数组

【考点1】数组的定义
数组是一组具有相同类型的数据的集合，这些数据称为数组元素。格式为：**类型名 数组名[常量表达式]。数组的所占字节数为元素个数与基类型所占字节数的乘积。**
【考点2】数组的初始化
第一维长度可以不写，其它维必须写。int a[]={1,2};合法，int a[][3]={2,3,4};合法，int a[2][]={2,3,4};非法。数组初始化元素值默认为0，没有初始化元素值为随机。如在int a[5]={0,1,2};中，元素a[4]值为0；而在int a[5];中，元素a[4]值为一个不确定的随机数。
【考点3】元素的引用
数组元素的下标从0开始，到数组长度减1结束。所以int a[5];中数组最后一个元素是a[4]。要把数组元素看作一个整体，可以把a[4]当作一个整型变量。
【考点4】二维数组
数组a[2][3]={1,2,3,4,5,6};中含6个元素，有2行3列。第一行为a[0]行，第2行为a[1]行，a[0]、a[1]叫行首地址，是地址常量。*(a[0]+1)是第一行第一个元素往后跳一列，即元素a[0][1]值为2*，(a[0]+3)是第一行第一个元素往后跳三个，即元素a[1][0]值为4。
【考点5】行指针
是一个指针变量，占四个字节，行指针指向一行连续数据，形式为：int (*p)[2];，*p只能存放含有两个整型元素的一维数组的首地址。注意*(p)两边的小括号不能省略，否则就成了指针数组，是若干指针元素的集合。
【考点6】数组名
数组名是数组的首地址。数组名不能单独引用，不能通过一个数组名代表全部元素。数组名是地址常量，不能对数组名赋值，所以a++是错误的。但数组名可以作为地址与一个整数相加得到一个新地址。
【考点7】元素形式的转换
助记："脱衣服法则"a[2]变成*(a+2)，a[2][3]变成*(a+2)[3]再可变成*((a+2)+3)。

第九章 字符串

【考点1】字符串常量及表示

字符串常量是由双引号括起来的一串字符，如"ABC"。在存储字符串时，系统会自动在其尾部加上一个空值'\0'，空值也要占用一个字节，也就是字符串"ABC"需要占四个字节。

【考点2】字符数组

C语言没有字符串变量，只能采用字符数组来存储字符串。数组的大小应该比它将要实际存放的最长字符串多一个元素，从而存放'\0'。

【考点3】字符串赋值

可以用下面的形式进行赋值：char str[]="Hello!";或char *p;p="Hello!";，但不能用下面的形式：char str[10];str="Hello";因为**str是一个地址常量，不能进行赋值操作。**

【考点4】字符串的输入与输出

可以用scanf和printf函数，如scanf("%s",str);，也可用专门处理字符串的两个函数gets和puts函数，还可以对字符数组逐个元素进行赋值，但一定要在最后赋一个'\0'。使用gets函数可以接收空格，使用puts函数在最后输出一个换行。

【考点5】字符串函数

要掌握的四个字符串函数：字符串拷贝函数strcpy（），求字符串长度函数strlen（），字符串链接函数strcat（），字符串比较函数strcmp（）。使用这些函数需在预处理部分包含头文件"string.h"。
字符串长度要小于字符数组的长度，例：char str[10]="Hello";sizeof(str)的值为10（数组长度），strlen(str)的值为5（字符串长度）。这些函数是考试常用到的函数，大家一定要熟练应用这几个函数。

第十章 结构体与共用体

【考点1】结构体类型的说明

结构体是若干个类型数据的集合，结构体类型说明格式如下：**struct 类型名 {类型1 成员名1;类型2 成员名2;……};，以上整个部分是一个数据类型，与整型的int是同样地位。可用typedef把结构体类型替换成一个只有几个字母的简短标识符。**

【考点2】结构体变量的定义

结构体变量是用说明的结构体类型所定义的一个变量，与结构体类型不是一回事。一个结构体变量所占字节数为其所有成员所占字节数之和。如struct stu{char name[10];int age;} a,b;则表明定义了两个结构体变量a,b,每个变量占4个字节。a,b与int i,j;中的变量i,j是同样地位。

【考点3】结构体成员的引用

引用成员可用以下3种方式：（1**结构体变量名.成员名；（2）指针变量名->成员名：（3）（*指针变量名）.成员名。点（.）称为成员运算符，箭头（->）称为结构指向运**）

【考点4】链表

链表是由一个个结点构成的，一个结点就是一个结构体变量。每个结点可以分为数据域与指针域两个部分，数据域用来存放要存储的数据，指针域用来指向下一个结点。链表是考试中的难点，在C语言和公共基础部分都会考到，要领悟掌握。

【考点5】共用体

共用体的使用格式与结构体相似，共用体定义的关键字为union，共用体所占字节数是所有成员中字节数最大的那个。

第十一章 文件

【考点1】文件类型指针

文件指针是一个指向结构体类型的指针，定义格式为：FILE *指针变量名。在使用文件时，都需要先定义文件指针。

【考点2】文本文件与二进制文件

文本形式存放的是字符的ASCII码，二进制形式存放的是数据的二进制。例如"100"如果是文本形式就是存储'1'、'0'、'0'三个字符的ASCII码（00110001 00110000 00110000），如果是二进制形式就把100转化成二进制（01100100）。

【考点3】打开文件
文件的打开形式如下：FILE *fp; fp=fopen(“c:\lab.c”,”rb”);。fopen函数的前面一部分为文件名，后面一部分为文件的使用方式。打开方式详见教材第127页，**其中r代表读，w代表写，a代表添加，b代表二进制位的。**
【考点4】文件函数
l 判断文件结束feof函数，调用形式为：feof(FILE *fp); //文件指针，如果文件结束，则函数返回1，否则返回0
l fseek用来设置文件的位置，接着的读或写操作将从此位置开始。函数的调用形式如下：
fseek（文件指针，位移量，移动起始点）；
eg。FILE*fp=fopen(“C:\xiaoyu.c”,”rb”);
fseek(fp,12,SEEK_SET); //表示将文件fp的位置移动到文件起始位置后面的第12个字节的位置上
或者 fseek(fp,12,0);

fseek(fp,-12,SEEK_END; //表示将文件fp的位置移动到文件末尾位置前面的第12个字节的位置上
或者 fseek(fp,12,2);

如果是
fseek(fp,12,SEEK_CUR); //表示将文件fp的位置从当前位置移动后面的第12个字节的位置上
或者 fseek(fp,12,1);

l ftell用来获得文件当前的位置，函数给出当前位置相对于文件开头的字节数。函数调用形式如下：
ftell(FILE *fp) //给出当前闻之相对于开头的字节数 ，出错时，返回-1L

l 文件位置移到开头rewind函数功能等价于 fseek(fp,0,SEEK_SET)
l 文件字符输入输出ch=fgetc(FILE *fp)函数用于从fp指定的文件中读入一个字符并把它作为函数值返回
l fputc(char ch,FILE *fp),将字符ch写到文件指针fp所指的文件中去，输出成功，则返回输出的字符，失败，则返回一个EOF值
l 文件输入输出fscanf函数和fprintf函数，一般形式fscanf(文件指针，格式控制字符串，输入项表或输入项表)
l 文件字符串输入输出fgets函数（fgets(str,n,fp),功能是从fp所指文件中读入n-1个字符放入str为起始地址的空间内）和fputs函数(fputs(str,fp))
l 读写二进制文件fread函数和fwrite函数。

以上函数要求知道格式会用，清楚是用于二进制文件还是文本文件，要把教材文件这章仔细复习下，不要在考试的时候把这些文件函数搞混了。
典型例题：
给定程序中,函数fun的功能是将形参给定的字符串、整数、浮点数写到文本 文件中，再用字符方式从此文本文件中逐个读入并显示在终端屏幕上。 请在程序的下划线处填入正确的内容并把下划线删除, 使程序得出正确的结果。
注意：源程序存放在考生文件夹下的BLANK1.C中。
不得增行或删行，也不得更改程序的结构!
给定源程序：

```
#include <stdio.h>
void fun(char *s, int a, double f)
{
__1__ fp;
char ch;
fp = fopen("file1.txt", "w");
fprintf(fp, "%s %d %f\n", s, a, f);
fclose(fp);
fp = fopen("file1.txt", "r");
printf("\nThe result :\n\n");
```

```
ch = fgetc(fp);
while (!feof(__2__)) {
putchar(__3__); ch = fgetc(fp); }
putchar('\n');
fclose(fp);
}
main()
{ char a[10]="Hello!"; int b=12345;
double c= 98.76;
fun(a,b,c);
}
```

第十二章 深入讨论

【考点1】编译预处理

凡以#开头的这一行，都是编译预处理命令行，编译预处理不加分号，不占运行时间。宏替换仅是简单的文本替换，如#define f(x) (x)*(x)和#define f(x) x*x替换f(2+2)时就有区别，前者展开为(2+2)*(2+2)，后者为2+2*2+2。

如果源文件f2.c中有#include"f1.c"可以理解为把源文件f1.c原样包含到f2.c中，使f1.c和f2.c融合到一起成为一个C程序编译。所以一个C程序必有主函数，但一个C源文件未必有主函数。

【考点2】标识符作用域

局部变量是在函数内或复合语句内定义的变量，**作用域为定义它的函数内。局部变量有三种类型：自动auto，寄存器register和静态sta**tic。

自动变量随着函数的使用与否创建消失；寄存器变量分配在cpu中，没有内存地址；静态变量占用固定存储单元，在程序执行过程不释放，直到程序运行结束。

全局变量是在函数外定义的变量，作用域从定义它的位置到整个源文件结束为止，生存期为整个程序运行期间。全局变量都是静态变量。

eg

.#include<stdio.h>

int fun()

{

static int x = 1;

x=*2;*

return x;

}

main()

{

int i,s=1;

for(i=1;i<=3;i++)

s=fun();//i=1 S=2 i=2 S=8

printf("%d\n",s);

} 64

【考点3】动态存储分配

malloc(size)用来创建连续size个字节存储区，返回值类型为void *型。malloc函数常用于动态创建链表结点，如int *p; p=(int *)malloc(sizeof(int));。

calloc（n,size）创建n个同一类型的存储空间，可以理解为n个malloc。

free(p)释放动态分配的存储单元。

第二部分 公共基础知识资料

第一章 数据结构与算法

【考点1】算法的基本概念

算法：是指一组有穷的指令集，是解题方案的准确而完整的描述。算法不等于程序，也不等于计算方法。

算法的基本特征：

确定性，算法中每一步骤都必须有明确定义，不允许有多义性；

有穷性，算法必须能在有限的时间内做完，即能在执行有限个步骤后终止；

可行性，算法原则上能够精确地执行；

拥有足够的情报。

算法的组成要素：一个算法由数据对象的运算和操作以及其控制结构这两部分组成。

算法的基本运算和操作：算术运算，逻辑运算，关系运算，数据传输。

算法的基本控制结构：顺序，选择，循环。

算法基本设计方法：列举法、归纳法、递推、递归、减半递推技术。

【考点2】算法的复杂度

算法效率的度量——算法的复杂度：时间复杂度和空间复杂度。

算法时间复杂度：指执行算法所需要的计算工作量。通常，一个算法所用的时间包括编译时间和运行时间。

算法空间复杂度：指执行这个算法所需要的内存空间。包括算法程序所占的空间，输入的初始数据所占的空间，算法执行过程中所需的额外空间。

空间复杂度和时间复杂度并不相关。

【考点3】数据结构的基本概念

数据：数据是客观事物的符号表示，是能输入到计算机中并被计算程序识别和处理的符号的总称，如文档，声音，视频等。

数据元素：数据元素是数据的基本单位。

数据对象：数据对象是性质相同的数据元素的集合。

数据结构：是指由某一数据对象中所有数据成员之间的关系组成的集合。

【考点4】逻辑结构和存储结构

数据结构可分为数据的逻辑结构和存储结构。

数据的逻辑结构是对数据元素之间的逻辑关系的描述，与数据的存储无关，是面向问题的，是独立于计算机的。它包括数据对象和数据对象之间的关系。

数据的存储结构也称为数据的物理结构，是数据在计算机中的存放的方式，是面向计算机的，它包括数据元素的存储方式和关系的存储方式。

数据结构和逻辑结构的关系：一种数据的逻辑结构可以表示成多种存储结构即数据的逻辑结构和存储结构不一定一一对应。

常见的存储结构有：**顺序，链接，索引**等。采用不同的存储结构其数据处理的效率是不同的。

【考点5】线性结构和非线性结构

线性结构的条件（一个非空数据结构)：（1）有且只有一个根结点；（2）每一个结点最多有一个前件，也最多有一个后件。

非线性结构：不满足线性结构条件的数据结构。

栈、队列、双向链表是线性结构，树、二叉树为非线性结构。

【考点6】线性表及其顺序存储结构

线性表是由一组数据元素构成，数据元素的位置只取决于自己的序号，元素之间的相对位置是线性的。

在复杂线性表中，由若干项数据元素组成的数据元素称为记录；由多个记录构成的线性表称为文件。

非空线性表的结构特征：

（1）有且只有一个根结点a1，它无前件；

（2）有且只有一个终端结点an，它无后件；

（3）除根结点与终端结点外，其他所有结点有且只有一个前件，也有且只有一个后件。

结点个数n称为线性表的长度，当n=0时，称为空表。

线性表的顺序存储结构具有以下两个基本特点：

（1）线性表中所有元素所占的存储空间是连续的；

（2）线性表中各数据元素在存储空间中是按逻辑顺序依次存放的。

元素ai的存储地址为：ADR(ai)=ADR(a1)+(i-1)*k，ADR(a1)为第一个元素的地址，k代表每个元素占的字节数。

顺序表的运算：查找、插入、删除。

【考点7】线性链表

线性链表是线性表的链式存储结构，数据结构中的每一个结点对应于一个存储单元，这种存储单元称为存储结点，简称结点。结点由两部分组成：(1) 用于存储数据元素值，称为数据域；(2) 用于存放指针，称为指针域，用于指向前一个或后一个结点。

在链式存储结构中，存储数据结构的存储空间可以不连续，各数据结点的存储顺序与数据元素之间的逻辑关系可以不一致，而数据元素之间的逻辑关系是由指针域来确定的。

链式存储方式既可用于表示线性结构，也可用于表示非线性结构。

线性单链表中，HEAD称为头指针，HEAD=NULL（或0）称为空表。

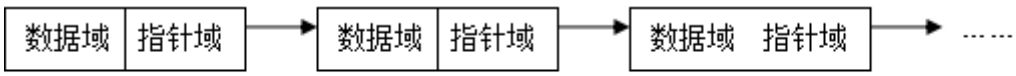


图 1 单链表的结构

双向链表有两个指针：左指针（Llink）指向前件结点，右指针（Rlink）指向后件结点。

图 2 双链表的结构

循环链表：循环链表与单链表的不同的是它的最后一个结点的指针域存放的事指向第一个结点的指针而单链表存放的是空指针。

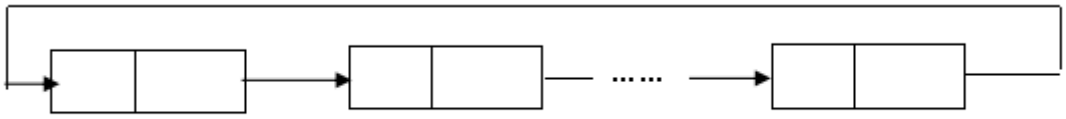


图 3 循环链表的结构

线性链表的基本运算：查找、插入、删除。

【考点8】栈

1、栈的基本概念

栈是一种特殊的线性表，只允许在表的一端进行插入和删除的线性表；插入，删除的一端为栈顶，另一端为栈底；当表中没有元素时为空栈。

栈是一种后进先出（或先进后出Last In First Out）的线性表。栈具有记忆功能。栈的实例：火车调度，子弹夹。

2、栈的存储结构

顺序存储结构：用一组地址连续的存储单元即一维数组来存储；

链式存储：用线性链表来存储；

3、栈的基本运算

(1) 入栈运算，在栈顶位置插入元素；

(2) 退栈运算，删除元素(取出栈顶元素并赋给一个指定的变量)；

(3) 读栈顶元素，将栈顶元素赋给一个指定的变量，此时指针无变化。

【考点9】队列

1.队列的基本概念

队列是一种特殊的线性表，只允许在表的一端插入，在另一端删除，允许插入的一端是队尾（rear），允许删除的一端为队头（front）；当表中没有元素是空队列；队列是一种先进先出的线性表。(FIFO)

2、队列的存储结构

顺序存储：一维数组。

链式存储：线性链表。队尾插；队头删

3、队列的运算:

(1) 入队运算：从队尾插入一个元素； (2) 退队运算：从队头删除一个元素。

队列的顺序存储结构一般采用循环队列的形式。循环队列s=0表示队列为空；s=1且front=rear表示队满。

计算循环队列的元素个数：“尾指针减头指针”，若为负数，再加其容量即可。

【考点10】树的基本概念

树是一种非线性结构，是n个结点的有限集。当n=0 时为空树，n>0时为非空树。结点的度：结点所拥有的子树的个数。

叶子结点：度为0的结点。

分支结点：除叶子结点以外的结点。

结点的层次：根结点在第一层，同一层上左右结点的子结点在下一层。

树的深度：所处层次最大的那个结点的层次。

树的度：树中所有结点的度的最大值。

【考点11】二叉树及其基本性质

1、二叉树的概念

二叉树是一种特殊的树形结构，每个结点最多只有两棵子树，且有左右之分不能互换，因此，二叉树有五种不同的形态，见教材12页。

2、二叉树的性质

性质1 在二叉树的第k层上，最多有2k-1(k≥1）个结点。

性质2 深度为m的二叉树最多有2m-1个结点。

性质3 在任意一棵二叉树中，度为0的结点（叶子结点）总是比度为2的结点多一个。

性质4 具有n个结点的二叉树，其深度不小于[log2n]+1,其中[log2n]表示为log2n的整数部分。

3、二叉树的存储结构：详见教材第13-14页。

【考点12】满二叉树与完全二叉树

满二叉树：除最后一层外，每一层上的所有结点都有两个子结点。在满二叉树中，每一层上的结点数都达到最大值，即在满二叉树的第k层上有2k-1个结点，且深度为m的满二叉树有2m - 1个结点。

完全二叉树是指这样的二叉树：除最后一层外，每一层上的结点数均达到最大值；在最后一层上只缺少右边的若干结点。

满二叉树是完全二叉树，而完全二叉树一般不是满二叉树。

【考点13】完全二叉树的性质

性质1 具有n个结点的完全二叉树的深度为[log2n]+1。

性质2 完全二叉树中度为1的结点数为0或1。

【考点14】二叉树的遍历

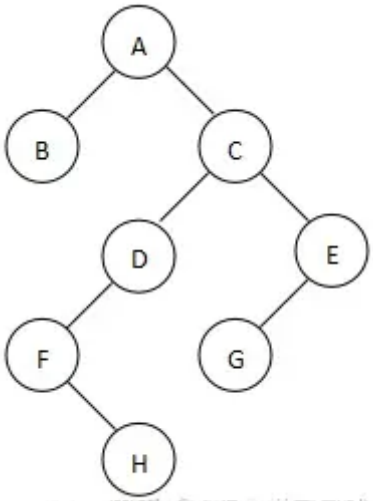


图 4 二叉树的遍历

前序遍历：先访问根结点、然后遍历左子树，最后遍历右子树；并且，在遍历左、右子树时，仍然先访问根结点，然后遍历左子树，最后遍历右子树。

前序遍历图5可得：ABCDFHEG。

中序遍历：先遍历左子树、然后访问根结点，最后遍历右子树；并且，在遍历左、右子树时，仍然先遍历左子树，然后访问根结点，最后遍历右子树。

中序遍历图5可得：BAFHDCGE。

后序遍历：先遍历左子树、然后遍历右子树，最后访问根结点；并且，在遍历左、右子树时，仍然先遍历左子树，然后遍历右子树，最后访问根结点。

后序遍历图5可得：BHFDGECA。

【考点15】顺序查找

顺序查找是从表的一端开始，依次扫描表中的各个元素，并与所要查找的数进行比较。

在下列两种情况下也只能采用顺序查找：

（1）如果线性表为无序表，则不管是顺序存储结构还是链式存储结构，只能用顺序查找。

（2）即使是有序线性表，如果采用链式存储结构，也只能用顺序查找。

【考点16】二分查找

二分查找的条件：（1）用顺序存储结构 (2)线性表是有序表。

查找的步骤：详见教材第16页。

对于长度为n的有序线性表，在最坏情况下，二分法查找只需比较log2n次，而顺序查找需要比较n次。

【考点17】排序

1、交换排序

（1）冒泡排序法，在最坏的情况下，冒泡排序需要比较次数为n(n - 1)/2。

（2）快速排序法 ，在最坏的情况下，快速排序需要比较次数为n(n - 1)/2。

2、插入类排序法：

（1）简单插入排序法，最坏情况需要n(n-1)/2次比较；

（2）希尔排序法，最坏情况需要O(n1.5)次比较。（大写O是算法复杂度的表示方法）

3、选择类排序法：

（1）简单选择排序法，最坏情况需要n(n-1)/2次比较；

（2）堆排序法，最坏情况需要O(nlog2n)次比较。

相比以上几种(除希尔排序法外)，堆排序法的时间复杂度最小。

第二章 程序设计基础

【考点1】程序设计方法与风格

形成良好的程序设计风格需注意：(详见教材第19页)。

1、源程序文档化； 2、数据说明的方法； 3、语句的结构； 4、输入和输出。

注释分序言性注释和功能性注释。

语句结构清晰第一、效率第二。

【考点2】结构化程序设计方法的四条原则

1、自顶向下； 2、逐步求精； 3、模块化； 4、限制使用goto语句。

【考点3】结构化程序的基本结构

顺序结构：是最基本、最普通的结构形式，按照程序中的语句行的先后顺序逐条执行。

选择结构：又称为分支结构，它包括简单选择和多分支选择结构。

循环结构：根据给定的条件，判断是否要重复执行某一相同的或类似的程序段。循环结构对应两类循环语句：先判断后执行的循环体称为当型循环结构；先执行循环体后判断的称为直到型循环结构。

【考点4】面向对象的程序设计及面向对象方法的优点

面向对象的程序设计以对象为核心，强调对象的抽象性，封装性，继承性和多态性。

面向对象方法的优点

（1）人类习惯的思维方法一致；（2）稳定性好；（3）可重用性好；

（4）易于开发大型软件产品；（5）可维护性好。

【考点5】对象及其特点

对象（object）：面向对象方法中最基本的概念，可以用来表示客观世界中的任何实体，对象是实体的抽象。

对象的基本特点：

（1）标识惟一性；（2）分类性；（3）多态性；（4）封装性；（5）模块独立性好。

【考点6】属性，类和实例

属性：即对象所包含的信息，它在设计对象时确定，一般只能通过执行对象的操作来改变。

类：是具有相似属性与操作的一组对象。类是关于对象性质的描述。类是对象的抽象，对象是其对应类的一个实例。

【考点7】消息及其组成

消息：是一个实例与另一个实例之间传递的信息。对象间的通信靠消息传递。它请求对象执行某一处理或回答某一要求的信息，它统一了数据流和控制流。

消息的组成包括：

(1)接收消息的对象的名称；（2）消息标识符，也称消息名；（3）零个或多个参数。

【考点8】继承和多态

继承：是使用已有的类定义作为基础建立新类的定义技术，广义指能够直接获得已有的性质和特征，而不必重复定义他们。

继承具有传递性，一个类实际上继承了它上层的全部基类的特性。

继承分单继承和多重继承。单继承指一个类只允许有一个父类，即类等级为树形结构；多重继承指一个类允许有多个父类。

多态性：是指同样的消息被不同的对象接受时可导致完全不同的行动的现象

第三章 软件工程基础

【考点1】软件定义与软件特点

软件指的是计算机系统中与硬件相互依存的另一部分，包括**程序、数据和相关文档的完整集合。**

名称	描述
程序	软件开发人员根据用户需求开发的、用程序设计语言描述的、适合计算机执行的指令序列
数据	使程序能正常操纵信息的数据结构
文档	与程序的开发、维护和使用有关的图文资料

软件的特点：

软件是一种逻辑实体，具有抽象性；

软件的生产与硬件不同，它没有明显的制作过程；

软件在运行、使用期间不存在磨损、老化问题；

软件的开发、运行对计算机系统具有依赖性，受计算机系统的限制，这导致了软件移植的问题；

软件复杂性高，成本昂贵；

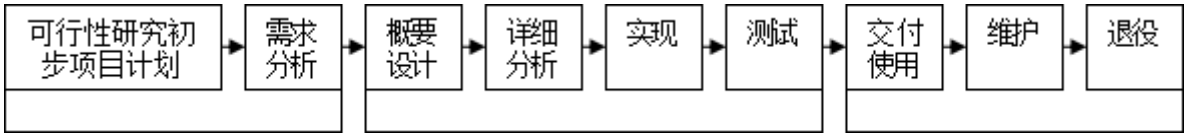
软件开发涉及诸多的社会因素。

根据应用目标的不同，软件可分应用软件、系统软件和支撑软件（或工具软件）。

名称	描述
应用软件	为解决特定领域的应用而开发的软件，如办公自动化软件
系统软件	计算机管理自身资源，提高计算机使用效率并为计算机用户提供各种服务的软件，如操作系统
支撑软件（或工具软件）	支撑软件是介于两者之间，协助用户开发软件的工具性软件。

【考点2】软件的生命周期

软件生命周期是指软件产品从提**出、实现、使用维护到停止使用退役的整**个过程。可分为软件定义，软件开发及软件维护3个阶段。软件生命周期中，能够准确确定软件系统必须做什么和必须具备哪些功能的阶段是：需求分析。



【考点3】软件危机和软件工程的概念

软件危机泛指在计算机软件的开发和维护过程中遇到的一系列严重的问题，集中表现在成本，质量。生产效率等几个方面。

所谓软件工程是指采用工程的概念、原理、技术和方法指导软件的开发与维护。是建立并使用完善的工程化原则，以较经济的手段获得，能在实际机器上有效运行的可靠软件的一系列方法；软件工程的主要思想强调在软件开发过程中需要应用工程化原则。软件工程的核心思想是把软件当作一个工程产品来处理。

软件工程包括3个要素：方法，工具和过程

名称	描述

名称	描述
方法	方法是完成软件工程项目的技术手段
工具	工具支持软件的开发、管理、文档生成
过程	过程支持软件开发的各个环节的控制、管理

【考点4】软件工程过程

软件工程过程是把软件转化为输出的一组彼此相关的资源活动，包含4种基本活动：

（1）P(plan)——软件规格说明；（2）D(do)——软件开发；

（3）C(check)——软件确认；（4）A(action)——软件演进。

【考点5】软件开发技术和软件工程管理

软件工程的理论和技术性研究的内容主要包括软件开发技术和软件工程管理。

软件开发技术包括软件开发方法学、开发过程、开发工具和软件工程环境，其主体内容是软件开发方法学。

软件开发方法包括分析方法，设计方法和程序设计方法。

软件工程管理包括软件管理学，软件工程经济学，软件心理学等。

软件管理学包括人员组织，进度安排，质量保证，配置管理，项目计划等。

软件工程经济学是研究软件开发中成本的估算，成本效益的方法和技术。

【考点6】软件工程的原则

软件工程的原则：抽象，信息隐蔽，模块化，局部化，确定性，一致性，完备性，可验证性（详细见教材第28-29页）。

【考点7】需求分析概述

需求分析阶段的工作：需求获取，需求分析，编写需求规格说明书，需求评审。

需求分析方法有：

（1）结构化需求分析方法；

①面向数据结构的Jackson方法（ISD）；

②面向数据流的结构化分析方法（SA）；

③面向数据结构的结构化数据系统开发方法（DSSD）；

（2）面向对象的分析的方法（OOA）。

从需求分析建立的模型的特性来分：静态分析和动态分析。

【考点8】结构化方法和结构化分析方法

1、结构化方法包括结构化分析方法，结构化设计方法，结构化编程方法。

结构化方法中，软件功能分解属于总体设计阶段。

2、结构化分析方法的概念

结构化分析方法是面向数据流自顶而下逐步求精进行需求分析的方法。

结构化分析方法在软件需求分析阶段的应用。

3、结构化分析的常用工具

数据流图（DFD-Data Flow Diagram）：是结构化分析方法中用于系统逻辑模型的一种工具。它以图形的方式描绘在系统中流动和处理的过程。

数据流图中四种基本的符号。

箭头：表示数据流，数据流是数据在系统中传播的路径。

圆或椭圆：表示加工，加工又称为数据处理，是对数据流进行某些操作或变换。

双横：表示数据存储（数据源）。数据存储又称为文件，指暂时保存的数据，它可以是数据库文件或任何形式的数据组织。

方框：源、潭。表示数据的源点或终点。它是软件系统外部环境中的实体，统称外部实体

数据字典（DD）：它是结构分析方法的核心，是用来描述系统中所用到的全部数据和文件的文档,作用是对DFD中出现的被命名的图形元素进行确切解释。

数据字典由以下4类元素组成

（1）数据流 （2）数据流分量 （3）数据存储 （4）处理

判定树（决策树）：是一种描述加工的图形工具，适合描述时候处理中具有多个判断，而且每个决策与若干条件有关。

判定表：与判定树类似，也是一种描述加工的图形工具。如果一个加工逻辑有多个条件、多个操作，并且在不同的条件组合下执行不同的操作，那么可以使用判定表来描述。

【考点9】软件需求规格说明书

软件需求规格说明书（SRS，Software Requirement Specification）是需求分析阶段得出的最主要的文档。软件需求规格说明书的特点：有正确性、无歧义性、完整性、可验证性、一致性、可理解性、可修改性和可追踪性。其中最重要的是**无歧义性。**

【考点10】软件设计的基本概念

软件设计是确定系统的物理模型。

软件设计是开发阶段最重要的步骤，是将需求准确地转化为完整的软件产品或系统的唯一途径。

从技术观点上看，软件设计包括软件结构设计、数据设计、接口设计、过程设计。

（1）结构设计定义软件系统各主要部件之间的关系；

（2）数据设计将分析时创建的模型转化为数据结构的定义；

（3）接口设计是描述软件内部、软件和协作系统之间以及软件与人之间如何通信；

（4）过程设计则是把系统结构部件转换为软件的过程性描述。

从工程管理角度来看，软件设计分两步完成：**概要设计和详细设计。**

（1）概要设计将软件需求转化为软件体系结构、确定系统级接口、全局数据结构或数据库模式；

（2）详细设计确立每个模块的实现算法和局部数据结构，用适当方法表示算法和数据结构的细节。

【考点11】软件设计的基本原理

1、软件设计中应该遵循的基本原理和与软件设计有关的概念：

模块化：把程序划分成独立命名且可独立访问的模块，每个模块完成一个子功能。

抽象化：抽出事物的本质特性而暂时不考虑它们的细节。

信息隐藏和局部化：信息隐蔽是指在一个模块内包含的信息（过程或数据），对于不需要这些信息的其他模块来说是不能访问的，实现信息隐蔽依靠对象的封装。

模块独立性：模块独立性是指每个模块只完成系统要求的独立的子功能，并且与其他模块的联系最少且接口简单。模块的独立程度是评价设计好坏的重要度量标准。

【考点12】耦合性和内聚性

衡量软件的模块独立性是用耦合性和内聚性两个定性的度量标准。

耦合性：是对一个软件结构内不同模块之间互联程度的度量。耦合性的强弱取决于模块间接口的复杂程度。

内聚性：是一个模块内部各个元素间彼此结合的紧密程度的度量。

一个模块的内聚性越强则该模块的模块独立性越强。一个模块与其他模块的耦合性越强则该模块的模块独立性越弱。

在结构程序设计中，模块划分的原则是模块内具有高内聚度，模块间具有低耦合度。

耦合和内聚的种类（详见教材第35页）。

耦合度由低到高：非直接耦合，数据耦合，标记耦合，控制耦合，外部耦合，公共耦合，内容耦合。

内聚性由强到弱：功能内聚，顺序内聚，通信内聚，过程内聚，时间内聚，逻辑内聚，偶然内聚。

【考点13】结构化设计方法

结构化分析方法是面向数据流自顶而下，逐步求精进行需求分析的方法，基本思想将软件设计成由相对独立，单一功能的模块组成的结构，与结构分析方法衔接使用，以数据流图为基础得到软件的模块结构，适用于变换型结构和事物型结构的目标系统。

1、概要设计的任务：（1）划分出组成系统的物理元素 （2）设计软件的结构

2、概要设计的工具：

结构图（SC-Structure Chart）也称程序结构图，在结构图中，模块用一个矩形表示，箭头表示模块间的调用关系。可以用带注释的箭头表示模块调用过程中来回传递的信息。还可用带实心圆的箭头表示传递的是控制信息，空心圆箭心表示传递的是数据。

结构图的基本形式：基本形式、顺序形式、重复形式、选择形式。

结构图有四种模块类型：传入模块、传出模块、变换模块和协调模块。

程序结构图中的专业术语：

名称	描述
深度	表示控制的层数
上级模块，从属模块	上，下两层模块a和b，且有a调用b，则a是上级模块，b是从属模块
宽度	整体控制跨度（最大模块的层）的表示
扇入	调用该模块的模块个数
扇出	一个模块直接调用的其他模块数
原子模块	树中位于叶子节点的模块

3、面向数据流的设计方法

任何软件系统都可以用数据流图表示，典型的数据流类型有两种：变换型和事务型。

变换型系统结构图由输入、中心变换、输出三部分组成。

4、设计的准则

（1）提高模块独立性。

（2）模块规模适中。

（3）深度，宽度，扇出和扇入适当。如果深度过大，则说明有的控制模块可能简单了，如果宽度过大，则说明系统的控制过于集中，扇出过大说明模块过分复杂，需要控制和协调过多的下级模块，应适当加中间层次，扇出过小可以把模块进一步分解成若干小模块，或合并到上级模块中，扇入越大则共享该模块的上级数目越多。好的软件设计结构通常顶层高扇出，中间扇出较少，底层高扇入。

（4）使模块的作用域在该模块的控制域内。

（5）减少模块的接口和界面的复杂性。

（6）设计成单入口，单出口的模块。

（7）设计功能可预测的模块。

详细设计常用的设计工具（工程设计工具）：图形工具，表格工具和语言工具。

图形工具：

程序流程图：箭头表示控制流，方框表示加工步骤，菱形表示逻辑条件。

N-S图：有五种基本图形。

PAD图：问题分析图，有五种基本图型。

表格工具：判定表。

语言工具：PDL——过程设计语言（结构化的英语和伪码）。

【考点14】软件测试的目标和准则

软件测试的目标：发现程序中的错误。

软件测试的准则：

（1）所有测试都是应追溯到需求。

（2）严格执行测试计划，排除测试的随意性。

（3）充分注意测试中的群集表现。程序中存在错误的概率与该程序中已发现的错误数成正比。

（4）程序员应避免检查自己的程序。

（5）穷举测试不可能。穷举测试是把程序所有可能的执行路径都进行检查，即使小规模的程序的执行路径数也相当大，不可能穷尽，说明测试只能证明程序有错，不能证明程序中无错。

（6）妥善保存测试计划，测试用例出错统计和最终分析报告。

【考点15】软件测试方法

从是否需要执行被测软件的角度分为静态测试和动态测试;按功能分为白盒测试和黑盒测试

1、静态测试和动态测试

静态测试包括代码检查、静态结构分析、代码质量度量。不实际运行软件，主要通过人工进行。

动态测试是通过运行软件来检验软件中的动态行为和运行结果的正确性。动态测试的关键是使用设计高效、合理的测试用例。测试用例就是为测试设计的数据，由测试输入数据（输入值集）和预期的输出结果（输出值集）两部份组成。测试用例的设计方法一般分为两类：黑盒测试方法和白盒测试方法。

2、白盒测试和黑盒测试

（1）白盒测试

白盒测试也称为结构测试或逻辑测试，是把程序看成装在一只透明的白盒子里，测试者完全了解程序的结构和处理过程。它根据程序的内部逻辑来设计测试用例，检查程序中的逻辑通路是否都按预定的要求正确地工作。

白盒测试的基本原则：

（1）保证所测模块中每一独立路径至少执行一次。

（2）保证所测模块所有判断的每一分支至少执行一次。

（3）保证所测模块每一循环都在边界条件和一般条件下至少各执行一次。

（4）验证所有内部数据结构的有效性。

（5）按照白盒测试的基本原则，“白盒”法是穷举路径测试。

白盒测试的方法：逻辑覆盖，基本路经测试。

（2）黑盒测试

黑盒测试也称功能测试或数据驱动测试，是把程序看成一只黑盒子，测试者完全不了解，或不考虑程序的结构和处理过程。它根据规格说明书的功能来设计测试用例，检查程序的功能是否符合规格说明的要求。

黑盒测试的方法：等价划分法，边界值分析法，错误推测法。

【考点16】软件测试的实施

软件测试过程分4个步骤，***即单元测试、集成测试、验收测试和系统测试。***

单元测试是对软件设计的最小单位——模块进行正确性检验的测试，单元测试的根据是源程序和详细设计说明书，单元测试的技术可以采用静态分析和动态测试。

单元测试期间对模块进行的测试：模块接口，局部数据结构，重要的执行通路，出错处理通路，边界条件。

驱动模块相当于被测模块的主程序，它接收测试数据，并传给所测模块，输出实际测试结果

桩模块通常用于代替被测模块调用的其他模块，其作用仅做少量的数据操作，是一个模拟子程序。

集成测试是测试和组装软件的系统化技术，主要目的是发现与接口有关的错误，集成测试的依据是概要设计说明书。

集成测试的方法：非增量方式组装和增量方法组装。

增量方式包括自顶而下的增量方式，自底而上的增量方式和混合增量方式。

确认测试的任务是验证软件的功能和性能，确认测试的实施首先运用黑盒测试方法，对软件进行有效性测试，即验证被测软件是否满足需求规格说明确认的标准。

检查软件产品是否符合需求定义的过程是：确认测试。

系统测试是通过测试确认的软件，作为整个基于计算机系统的一个元素，与计算机硬件、外设、支撑软件、数据和人员等其他系统元素组合在一起，在实际运行（使用）环境下对计算机系统进行一系列的集成测试和确认测试。

系统测试的具体实施一般包括：功能测试、性能测试、操作测试、配置测试、外部接口测试、安全性测试等。

【考点17】程序调试

在对程序进行了成功的测试之后将进入程序调试（通常称Debug，即排错）。

程序的调试任务是诊断和改正程序中的错误。

程序调试和软件测试的区别：

（1）软件测试是尽可能多地发现软件中的错误，而程序调试先要发现软件的错误，然后借助于一定的调试工具去执行找出软件错误的具体位置。

（2）软件测试贯穿整个软件生命期，调试主要在开发阶段。

程序调试的基本步骤：

（1）错误定位。从错误的外部表现形式入手，研究有关部分的程序，确定程序中出错位置，找出错误的内在原因；

（2）修改设计和代码，以排除错误；

（3）进行回归测试，防止引进新的错误。

软件调试可分为静态调试和动态调试。静态调试主要是指通过人的思维来分析源程序代码和排错，是主要的设计手段，而动态调试是辅助静态调试的。

主要的调试方法有：

（1）强行排错法；（2）回溯法；（3）原因排除法，包括演绎法，归纳法和二分法。

第四章 数据库设计基础

【考点1】数据库的基本概念

数据（Data）是数据库存储的基本对象，是描述事物的符号记录。

数据库（DB）是长期储存在计算机内、有组织的、可共享的大量数据的集合，它具有统一的结构形式并存放于统一的存储介质内，是多种应用数据的集成，并可被各个应用程序所共享，所以数据库技术的根本目标是解决数据共享问题。

数据库管理系统（DBMS）是数据库的管理机构，负责数据库中的数据组织、数据操纵、数据维护、控制及保护和数据服务等。数据库管理系统是数据库系统的核心。数据库系统包含数据库和数据库管理系统。

数据库管理系统的功能：

（1）数据模式定义：即为数据库构建其数据框架；

（2）数据存取的物理构建：为数据模式的物理存取与构建提供有效的存取方法与手段；

（3）数据操纵：为用户使用数据库的数据提供方便，如查询、插入、修改、删除等以及简单的算术运算及统计；

（4）数据的完整性、安全性定义与检查；

（5）数据库的并发控制与故障恢复；

（6）数据的服务：如拷贝、转存、重组、性能监测、分析等。

为完成数据库管理系统的功能，数据库管理系统提供相应的数据语言：

数据定义语言（DDL）：负责数据模式定义和数据物理存取构建。

数据操纵语言（DML）：负责数据的操纵。

数据控制语言（DCL）：负责数据完整性，安全性的定义与检查以及并发控制，故障恢复等功能。

数据语言按使用方式具有两个结构形式：交互式命令语言（自含型和自主型语言）和宿主型语言。

数据库管理员（DBA）的工作：数据库设计，数据库维护，改善系统性能，提高系统效率。

数据库系统（DBS）是指在计算机系统中引入数据库后的系统，一般由数据库、数据库管理系统、应用系统、数据库管理员和用户构成。

数据库应用系统（DBAS）是数据库系统再加上应用软件及应用界面这三者所组成，具体包括：数据库、数据库管理系统、数据库管理员、硬件平台、软件平台、应用软件、应用界面。

【考点2】数据管理的发展和基本特点

数据管理技术的发展经历了三个阶段：**人工管理阶段、文件系统阶段和数据库系统阶段，数据独立性最高的阶段是数据库系统阶段。**

人工管理阶段特点：（1）计算机系统不提供对用户数据的管理功能（2）数据不能共享（3）不单独保存数据。

文件系统阶段的缺陷：（1）数据冗余（2）不一致性（3）数据联系弱。

数据库系统的发展阶段：第一代的网状、层次数据库系统；第二代的关系数据库系统；第三代的以面向对象模型为主要特征的数据库系统。

数据库系统的基本特点：

（1）数据的高集成性 （2）数据的高共享性和低冗余性 （3）数据高独立性 （4）数据统一管理与控制。

数据独立性是数据与程序间的互不依赖性，即数据库中的数据独立于应用程序而不依赖于应用程序。

数据的独立性一般分为物理独立性与逻辑独立性两种。

（1）物理独立性：当数据的物理结构（包括存储结构、存取方式等）改变时，其逻辑结构，应用程序都不用改变。

（2）逻辑独立性：数据的逻辑结构改变了，如修改数据模式、增加新的数据类型、改变数据间联系等，用户的应用程序可以不变。

【考点3】数据系统的内部结构体系

1、数据统系统的三级模式：

（1）概念模式，也称逻辑模式，是对数据库系统中全局数据逻辑结构的描述，是全体用户公共数据视图。一个数据库只有一个概念模式。

（2）外模式，外模式也称子模式，它是数据库用户能够看见和使用的局部数据的逻辑结构和特征的描述，一个概念模式可以有若干个外模式。

（3）内模式，内模式又称物理模式，它给出了数据库物理存储结构与物理存取方法。一个数据库只有一个内模式。

内模式处于最底层，它反映了数据在计算机物理结构中的实际存储形式，概念模式处于中间层，它反映了设计者的数据全局逻辑要求，而外模式处于最外层，它反映了用户对数据的要求。

2、数据库系统的两级映射 （详见教材第55页）

两级映射保证了数据库系统中数据的独立性。

（1）概念模式到内模式的映射。该映射给出了概念模式中数据的全局逻辑结构到数据的物理存储结构间的对应关系；

（2）外模式到概念模式的映射。概念模式是一个全局模式而外模式是用户的局部模式。一个概念模式中可以定义多个外模式，而每个外模式是概念模式的一个基本视图。

【考点4】数据模型的基本概念

数据模型按不同的应用层次分为：

概念数据模型：简称概念模型，是一种面向客观世界，面向用户的模型，不涉及具体的硬件环境和平台也与具体的软件环境无关的模式，它是整个数据模型的基础。

逻辑数据模型：又称数据模型，它是一种面向数据库的模型。分为层次模型，网状模型，关系模型和面向对象模型，其中层次模型和网状模型统称为非关系模型。层次模型用树型结构表示实体之间联系的模型。

物理数据模型：又称物理模型，它是一种面向计算机物理表示的模型。

【考点5】E—R模型

1、E-R模型的基本概念

（1）实体：现实世界中的事物可以抽象成为实体，实体是概念世界中的基本单位，它们是客观存在的且又能相互区别的事物。

（2）属性：现实世界中事物均有一些特性，这些特性可以用属性来表示。

（3）码：唯一标识实体的属性集称为码。

（4）域：属性的取值范围称为该属性的域。

（5）联系：在现实世界中事物间的关联称为联系。

两个实体集间的联系实际上是实体集间的函数关系，这种函数关系可以有下面几种：一对一的联系、一对多或多对一联系、多对多。

2、E-R模型的的图示法

E-R模型用E-R图来表示，E-R图包含了表示实体集、属性和联系的方法。

（1）实体的表示：用矩形表示实体集，在矩形内写上该实体集的名字。

（2）属性的表示：用椭圆形表示属性，在椭圆形内写上该属性的名称。

（3）联系的表示：用菱形表示联系，菱形内写上联系名。

【考点6】层次模型和网状模型

层次模型是有根的定向有序树，是数据库系统中最早出现的数据模型。网状模型对应的是有向图。

层次模型和网状模型各自应满足的条件

模型名称	满足的条件

模型名称	满足的条件
层次模型	（1）有且只有一个结点没有双亲结点，这个结点称为根结点（2）根以外的其他结点有且只有一个双亲结点
网状模型	（1）允许一个以上的结点无双亲（2）一个结点可以有多于一个的双亲

【考点7】关系模型及相关概念

关系模式采用二维表来表示，由关系数据结构，关系操纵和关系完整性约束3部分组成，在关系数据库中，用来表示实体间联系的是关系。

关系：一个关系对应一张二维表。一个关系就是一个二维表，但是一个二维表不一定是一个关系。

元组：表中的一行即为一个元组。

属性：表中的一列即为一个属性，给每一个属性起一个名称即属性名。

分量：元组中的一个属性值，是不可分割的基本数据项。

域：属性的取值范围。

在二维表中惟一标识元组的最小属性值称为该表的键或码。二维表中可能有若干个健，它们称为表的候选码或候选健。从二维表的所有候选键选取一个作为用户使用的键称为主键或主码。表A中的某属性集是某表B的键，则称该属性值为A的外键或外码。

关系操纵：数据查询、数据的删除、数据插入、数据修改。

关系模型允许定义三类数据约束，它们是**实体完整性约束、参照完整性约束以及用户定义的完整性约束。**其中实体完整性约束、参照完整性约束必须满足的完整性约束条件。参照完整性约束不允许关系应用不存在的元组。实体完整性约束要求关系的主键中属性值不能为空，这是数据库完整性的最基本要求。

【考点8】关系代数

关系代数是一种抽象的查询语言，关系代数的运算对象是关系，运算结果也是关系。运算对象，运算符和运算结果是运算的三大要素。**集合运算符，专门的运算符，算术比较符和逻辑运算符。**

关系模型的基本运算：（1）插入 （2）删除 (3)修改 （4）查询（包括投影、选择、笛卡尔积运算）还有扩充运算交、除、连接及自然连接运算。

关系代数的5个基本操作中并，差，交，笛卡尔积是二目运算。

设关系R和S具有相同的关系模式

1、并：R和S的并是由属于R或属于S的所有元组构成的集合。

2、差：R和S的差是由属于R但是不属于S的元组构成的集合

3、笛卡尔积：设R和S的元数分别为r和s，R和S的笛卡尔积是一个（r+s）元的元组集合，每个元组的前r个分量来自R的一个元组，后s个分量来自S的一个元组。运**算后得到的新表的元组数是R*S，属性是r+s。**

4、交：属于R又属于S的元组构成的集合。

5、投影：一元运算，对一个关系进行垂直切割，消去某些列，并重新按排列的顺序。

6、选择：一元运算，根据某些条件对关系进行水平分割。即选择符合条件的元组。

7、除：给定关系R（X，Y）和S（Y，Z），其中X，Y，Z是属性组，R中的Y和S中Y可以有不同的属性名，但必须出自相同的域集。

8、连接：也称θ连接运算，是一种二元运算，它的操作是从两个关系的笛卡尔积中选取属性间满足一定条件的元组，以合并成一个大关系。连接运算包括等值连接和不等值连接。连接运算后得到的新表的属性是运算前表中属性相加。即多于原来关系中属性的个数。

9、自然连接：自然连接满足的条件是（1）两关系间有公共域（2）通过公共域的相等值进行连接。

【考点9】数据库设计和管理

数据库设计中有两种方法，面向数据的方法和面向过程的方法。

面向数据的方法是以信息需求为主，兼顾处理需求；面向过程的方法是以处理需求为主，兼顾信息需求。由于数据在系统中稳定性高，数据已成为系统的核心，因此面向数据的设计方法已成为主流。

数据库设计目前一般采用生命周期法，即将整个数据库应用系统的开发分解成目标独立的若干阶段。它们是：需求分析阶段、概念设计阶段、逻辑设计阶段、物理设计阶段。

一个低一级范式的关系模式，通过模式分解可以转化为若干个高一级范式的关系模式的集合，这种过程就叫规范化。

概念结构设计是将需求分析阶段得到的用户需求抽象为信息结构即概念模型的过程，它是整个数据库设计的关键。

逻辑结构设计的任务是将E—R图转换成关系数据模型的过程。

数据库的物理结构是指数据库在物理设备上的存储结构和存取方法。它依赖于给定的计算机系统。

常用的存取方法：索引方法，聚簇方法和HASH方法。

数据库管理的内容：

（1）数据库的建立，它是数据库管理的核心，包括数据模式的建立和数据加载。

（2）数据库的重组。

（3）数据库安全性控制。

（4）数据库的完整性控制，数据库的完整性是指数据的正确性和相容性。

（5）数据库的故障恢复。

（6）数据库监控

输入输出

有以下程序段

```
int j;
float y;
char name[50];
scanf(“%2d%f%s”,&j,&y,name);
```

若从键盘上输入55 566 7777qbc后，y的值为（566.0）

循环

```
#include<stdio.h>
```

```
main()
{
int x =8;
for(;x>0;x--)
{
if(x%3)  8 5 4 2
{
printf(“%d”,x--);
continue;
}
printf(“%d”,--x);
}
}
```

函数

```
void f(int v,int w)
{
int t;
t=v;
v=w;
w=t;
}
main()
{
int c=1,y=3,z=2;
if(x>y)
f(x,y);
else if(y>z)
f(y,z);
else
f(x,z);
printf(“%d,%d,%d\n”,x,y,z);
}
```

字符串的操作

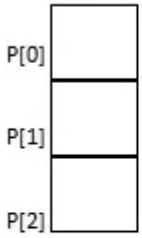
```
#include<stdio.h>
main()
{
char s[]=”012xy\08s34f4w2”;
int i,n=0;
for(i=0;s[i]!=0;i++)
 if(s[i]>=’0’ &&s[i]<=’9’) n++;
printf(“%d\n”,n);
}
```

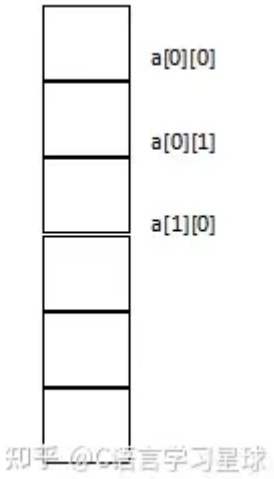
指针基本操作

```
#include<stdio.h>
#include<stdlib.h>
main()
{
int *a,*b,*c;
a = b = c=(int *)malloc(sizeof(int));
*a = 1;*b=2,*c = 3;
a=b;
printf(“%d,%d,%d\n”,*a,*b,*c);
}
```

- l b = *p++;//指针变量p会先与++结合，相当于先取指针变量p所指向的存储单元中的值赋给变量b，然后使指针变量p自增1
- l 指针数组：一般定义形式如下： 类型名 *指针数组名[常量表达式]

如，int *p[3]; //因为[]的优先级高于*号，因此，p首先与【】结合，构成p【3】，说明了p是一个数组名，在它前面的*号说明数组p中的每个元素只能存放指针





数组及指针

```
#include<stdio.h>
main()
{
int x[3][2]={0}，i;
for(i=0;i<3;i++)
 scanf(“%d”,x[i]);
printf(“%3d%3d%3d\n”,x[0][0],x[0][1],x[1][0]);
}
```

若运行时输入：2 4 6<回车>，则输出结果是（2 0 4）

结构体知识点补充：

```
struct student
{
char name[10];
char sex;
int age;
```

```
float score;
};
Struct student s1,*ps,stu[3];
eg.
struct S
{
int a;
int b;
}data[2]={10,100,20,200};
main()
{
sturct s p =data[1];
printf(“%d\n”,(++p.a));
}
```

链表：

结构体中含有指向本结构体类型的指针成员，例如下面

```
#include "stdio.h"
#include"stdlib.h"
struct link
{
 char data;
 struct link *next;
};
typedef struct link LINK;
LINK *create_list()
{
char ch;
LINK *h,*s,*r;
h = (LINK*)malloc(sizeof(LINK));
r = h;
printf("请输入一些字母（空格也被认为是字母，回车结束）");
ch = getchar();
while(ch !='\n')
{
    s = (LINK*)malloc(sizeof(LINK));
    s->data=ch;
    printf("%c",ch);
    r->next=s;
    r = s;
    ch=getchar();
}
r->next = NULL;
return h;
}
void print_list(LINK *h)
{
LINK *p;
p = h->next;
if(p==NULL)
    printf("The List is NULL\n");
else
while(p!= NULL)
```

```
{
    printf("%c",p->data);
    p=p->next;
}
printf("\n\n");
}
int add_node_at_front(LINK *head,char newData)
{
LINK *p;
p =(LINK *)malloc(sizeof(LINK));
p->data = newData;
if(head==NULL)
{
head = p;
p->next = NULL;
}
else
{
p->next = head;
head ->next = p;
}
return 1;
}
void del_node(LINK *head,char dataToDel)
{
LINK *p = head,*pPre=head;
if(p==NULL)
return;
printf("come to del_node,head->data is %c,deldata is %c\n",head-
>data,dataToDel);
while(p!=NULL)
{
 if(p->data == dataToDel)
{
if(p==head)
{
 printf("come to del_node 1\n");
    free(p);
    head = pPre=p=NULL;
 if(head == NULL)
        printf("head is null ");
 else
        printf("head is %d",head);
}
else
{
   pPre->next = p->next;
free(p);
p=NULL;
}
break;
}
pPre=p;
p = p->next;
}
```

```
}
LINK *find_node(LINK *head,char dataToFind)
{
LINK *p = head;
while(p!= NULL)
{
 if(p->data == dataToFind)
{
 return p;
}
p = p->next;
}
return NULL;
}
int modify_node(LINK *head,char oldData,char newData)
{
LINK *p = head;
p = find_node(head,oldData);
if(p != NULL)
{
    p->data = newData;
return 1;
}
return 0;
}
main()
{
 char c,cNew;
    LINK *head,*p;
    head = create_list();
    printf("%d",head);
    print_list(head);
    c = getchar();
    del_node(head,c);
    print_list(head);
}
```

冒泡排序

冒泡排序每一趟排序把最大的放在最右边。

比如：

87 12 56 45 78

87和12交换：12 87 56 45 78

87和56交换： 56 87 45 78

87和45交换： 45 87 78

87和78交换： 78 87

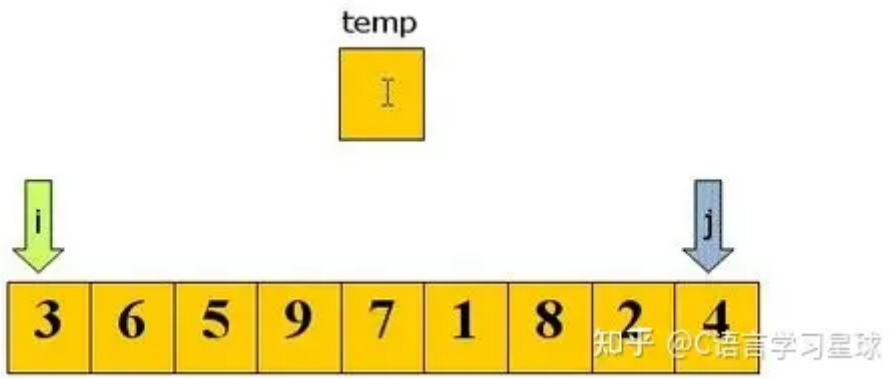
到此第一趟排序结束，接下来的每一趟排序都是这样。

```
#include<stdio.h>
void Print(int *num, int n)
```

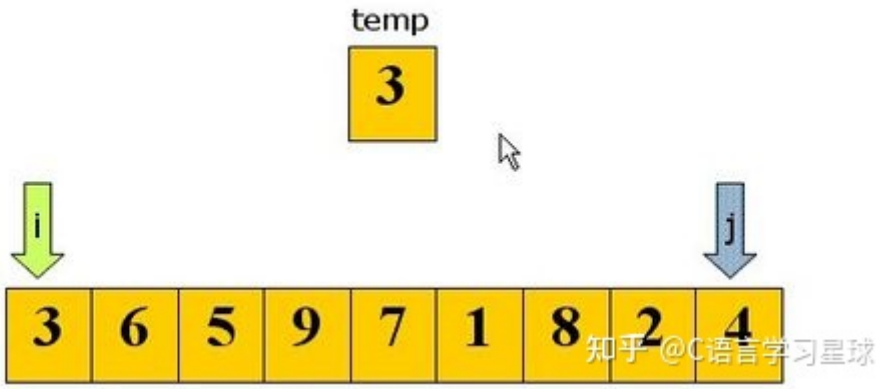
```
{
int i;
for(i = 0; i < n; i++)
printf("%d ", num[i]);
puts("\n");
return;
}
void Bubble_Sort(int *num, int n)
{
int i, j;
for(i = 0; i < n; i++)
{
for(j = 0; i + j < n - 1; j++)
{
if(num[j] > num[j + 1])
{
int temp = num[j];
num[j] = num[j + 1];
num[j + 1] = temp;
}
Print(num, n);
}
}
return;
}
int main()
{
int num[8] = {87, 12, 56, 45, 78};
Bubble_Sort(num, 5);
return 0;
}
```

快速排序：

一、假设我们给一个int数组进行排序，数组中数字初始序列为int a[9]={3,6,5,9,7,1,8,2,4}



二、分析快速排序的原理前，我们先声明一些东西，首先设置一个临时变量用来存放随机取出数组中的一个数，一般我们取数组的第一个元素也就是说temp=a[0]，同时设置两个游标分别指向数组第一个元素和最后一个元素。



三、算法的基本运算步骤为：

1、依次比较数组的后游标所指与temp的大小，如果temp<a[j]，则j--，直到遇到第一个temp>a[j]，则停止移动，将a[j]赋值给a[i]

四、算法的基本运算步骤为：

2、依次比较数组的前游标所指与temp的大小，如果temp>a[i]，则i++，直到遇到第一个temp<a[i]，则停止移动，将a[i]赋值给a[j]

\5. 5

五、算法运算步骤为：

3、判断i是否等于j，如果不相等则循环1、2步，直到i等于j，则完成一次快速排序。

END