全国计算机等级考试二级 C 语言公共基础知识

第一章数据结构与算法

1.1 算法

算法：是指解题方案的准确而完整的描述。

算法不等于程序，也不等计算机方法，程序的编制不可能优于算法的设计。

算法的基本特征：是一组严谨地定义运算顺序的规则，每一个规则都是有效的，是明确的，此顺序将在有限的次数下终止。特征包括：

（1）可行性；

（2）确定性，算法中每一步骤都必须有明确定义，不充许有模棱两可的解释，不允许有多义性；

（3）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，包括合理的执行时间的含义；

（4）拥有足够的情报。

算法的基本要素：一是对数据对象的运算和操作；二是算法的控制结构。

指令系统：一个计算机系统能执行的所有指令的集合。

基本运算和操作包括：算术运算、逻辑运算、关系运算、数据传输。

算法的控制结构：顺序结构、选择结构、循环结构。

算法基本设计方法：列举法、归纳法、递推、递归、减斗递推技术、回溯法。

算法复杂度：算法时间复杂度和算法空间复杂度。

算法时间复杂度是指执行算法所需要的计算工作量。

算法空间复杂度是指执行这个算法所需要的内存空间。

1.2 数据结构的基本基本概念

数据结构研究的三个方面：

（1）数据集合中各数据元素之间所固有的逻辑关系，即数据的逻辑结构；

（2）在对数据进行处理时，各数据元素在计算机中的存储关系，即数据的存储结构；

（3）对各种数据结构进行的运算。

数据结构是指相互有关联的数据元素的集合。

数据的逻辑结构包含：

（1）表示数据元素的信息；

（2）表示各数据元素之间的前后件关系。

数据的存储结构有顺序、链接、索引等。

线性结构条件：

（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。

1.3 线性表及其顺序存储结构

线性表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，而由多个记录构成的线性表又称为文件。

非空线性表的结构特征：

（1）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）线性表中所有元素的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。
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：插入、删除。 （详见 14--16 页）

1.4 栈和队列

栈是限定在一端进行插入与删除的线性表，允许插入与删除的一端称为栈顶，不允许插入与删除的另一端称为栈底。
栈按照“先进后出”（FILO）或“后进先出”（LIFO）组织数据，栈具有记忆作用。用 top 表示栈顶位置，用 bottom 表示栈底。
栈的基本运算：（1）插入元素称为入栈运算；（2）删除元素称为退栈运算；（3）读栈顶元素是将栈顶元素赋给一个指定的变量，此时指针无变化。
队列是指允许在一端（队尾）进入插入，而在另一端（队头）进行删除的线性表。Rear 指针指向队尾，front 指针指向队头。
队列是“先进行出”（FIFO）或“后进后出”（LILO）的线性表。
队列运算包括（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。
循环队列：s=0 表示队列空，s=1 且 front=rear 表示队列满

1.5 线性链表

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。
链式存储方式即可用于表示线性结构，也可用于表示非线性结构。
线性链表，HEAD 称为头指针，HEAD=NULL（或 0）称为空表，如果是两指针：左指针（Llink）指向前件结点，右指针（Rlink）指向后件结点。
线性链表的基本运算：查找、插入、删除。

1.6 树与二叉树

树是一种简单的非线性结构，所有元素之间具有明显的层次特性。
在树结构中，每一个结点只有一个前件，称为父结点，没有前件的结点只有一个，称为树的根结点，简称树的根。每一个结点可以有多个后件，称为该结点的子结点。没有后件的结点称为叶子结点。
在树结构中，一个结点所拥有的后件的个数称为该结点的度，所有结点中最大的度称为树的度。树的最大层次称为树的深度。
二叉树的特点：（1）非空二叉树只有一个根结点；（2）每一个结点最多有两棵子树，且分别称为该结点的左子树与右子树。
二叉树的基本性质：
（1）在二叉树的第 k 层上，最多有 2^{k-1}(k≥1)个结点；
（2）深度为 m 的二叉树最多有 2^m-1 个结点；
（3）度为 0 的结点（即叶子结点）总是比度为 2 的结点多一个；
（4）具有 n 个结点的二叉树，其深度至少为[$\log_2 n$]+1,其中[$\log_2 n$]表示取 $\log_2 n$ 的整数部分；

（5）具有 n 个结点的完全二叉树的深度为$[\log_2 n]+1$；
（6）设完全二叉树共有 n 个结点。如果从根结点开始，按层序（每一层从左到右）用自然数 1，2，…n 给结点进行编号（k=1,2…n），有以下结论：
①若 k=1，则该结点为根结点，它没有父结点；若 k>1，则该结点的父结点编号为 INT(k/2)；
②若 2k≤n，则编号为 k 的结点的左子结点编号为 2k；否则该结点无左子结点（也无右子结点）；
③若 2k+1≤n，则编号为 k 的结点的右子结点编号为 2k+1；否则该结点无右子结点。
满二叉树是指除最后一层外，每一层上的所有结点有两个子结点，则 k 层上有 2k-1 个结点深度为 m 的满二叉树有 2m-1 个结点。
完全二叉树是指除最后一层外，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。
二叉树存储结构采用链式存储结构，对于满二叉树与完全二叉树可以按层序进行顺序存储。
二叉树的遍历：
（1）前序遍历（DLR），首先访问根结点，然后遍历左子树，最后遍历右子树；
（2）中序遍历（LDR），首先遍历左子树，然后访问根结点，最后遍历右子树；
（3）后序遍历（LRD）首先遍历左子树，然后访问遍历右子树，最后访问根结点。

1.7 查找技术

顺序查找的使用情况：
（1）线性表为无序表；
（2）表采用链式存储结构。
二分法查找只适用于顺序存储的有序表，对于长度为 n 的有序线性表，最坏情况只需比较 log2n 次。

1．8 排序技术

排序是指将一个无序序列整理成按值非递减顺序排列的有序序列。
交换类排序法：（1）冒泡排序法，需要比较的次数为 n(n-1)/2；　（2）快速排序法。
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。
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 $O(n\log_2 n)$次比较.

第二章程序设计基础

2．1 程序设计设计方法和风格

如何形成良好的程序设计风格
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。

2．2 结构化程序设计

结构化程序设计方法的四条原则是：1. 自顶向下；2. 逐步求精；3.模块化；4.限制使用 goto 语句。
结构化程序的基本结构和特点：
（1）顺序结构：一种简单的程序设计，最基本、最常用的结构；
（2）选择结构：又称分支结构，包括简单选择和多分支选择结构，可根据条件，判断应该选择哪一条分支来执行相应的语句序列；
（3）重复结构：又称循环结构，可根据给定条件，判断是否需要重复执行某一相同程序段。

2．3 面向对象的程序设计

面向对象的程序设计：以 60 年代末挪威奥斯陆大学和挪威计算机中心研制的 SIMULA 语言为标志。

面向对象方法的优点：

（1）与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。

对象是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。

面向对象的程序设计方法中的对象是系统中用来描述客观事物的一个实体，是构成系统的一个基本单位，由一组表示其静态特征的属性和它可执行的一组操作组成。

属性即对象所包含的信息，操作描述了对象执行的功能，操作也称为方法或服务。

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。

类是指具有共同属性、共同方法的对象的集合。所以类是对象的抽象，对象是对应类的一个实例。

消息是一个实例与另一个实例之间传递的信息。

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。

继承是指能够直接获得已有的性质和特征，而不必重复定义他们。

继承分单继承和多重继承。单继承指一个类只允许有一个父类，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

3．1 软件工程基本概念

计算机软件是包括程序、数据及相关文档的完整集合。

软件的特点包括：

（1）软件是一种逻辑实体；

（2）软件的生产与硬件不同，它没有明显的制作过程；

（3）软件在运行、使用期间不存在磨损、老化问题；

（4）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；

（5）软件复杂性高，成本昂贵；

（6）软件开发涉及诸多的社会因素。

软件按功能分为应用软件、系统软件、支撑软件（或工具软件）。

软件危机主要表现在成本、质量、生产率等问题。

软件工程是应用于计算机软件的定义、开发和维护的一整套方法、工具、文档、实践标准和工序。
软件工程包括 3 个要素：方法、工具和过程。
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和活动，包含 4 种基本活动：
（1）P——软件规格说明；
（2）D——软件开发；
（3）C——软件确认；
（4）A——软件演进。
软件周期：软件产品从提出、实现、使用维护到停止使用退役的过程。
软件生命周期三个阶段:软件定义、软件开发、运行维护，主要活动阶段是：
（1）可行性研究与计划制定；
（2）需求分析；
（3）软件设计；
（4）软件实现；
（5）软件测试；
（6）运行和维护。
软件工程的目标和与原则：
目标：在给定成本、进度的前提下，开发出具有有效性、可靠性、可理解性、可维护性、可重用性、可适应性、可移植性、可追踪性和可互操作性且满足用户需求的产品。
基本目标：付出较低的开发成本；达到要求的软件功能；取得较好的软件性能；开发软件易于移植；需要较低的费用；能按时完成开发，及时交付使用。
基本原则：抽象、信息隐蔽、模块化、局部化、确定性、一致性、完备性和可验证性。
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。
软件工程管理包括：软件管理学、软件工程经济学、软件心理学等内容。
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。
软件工程原则包括抽象、信息隐蔽、模块化、局部化、确定性、一致性、完备性和可验证性。

3．2 结构化分析方法

结构化方法的核心和基础是结构化程序设计理论。
需求分析方法有（1）结构化需求分析方法；　（2）面向对象的分析的方法。
从需求分析建立的模型的特性来分：静态分析和动态分析。
结构化分析方法的实质：着眼于数据流，自顶向下，逐层分解，建立系统的处理流程，以数据流图和数据字典为主要工具,建立系统的逻辑模型。
结构化分析的常用工具
（1）数据流图；　（2）数据字典；　（3）判定树；　（4）判定表。
数据流图：描述数据处理过程的工具，是需求理解的逻辑模型的图形表示，它直接支持系统功能建模。
数据字典：对所有与系统相关的数据元素的一个有组织的列表，以及精确的、严格的定义，使得用户和系统分析员对于输入、输出、存储成分和中间计算结果有共同的理解。
判定树：从问题定义的文字描述中分清哪些是判定的条件，哪些是判定的结论，根据描述材料中的连接词找出判定条件之间的从属关系、并列关系、选择关系，根据它们构造判定树。
判定表：与判定树相似，当数据流图中的加工要依赖于多个逻辑条件的取值，即完成该加工的一组动作是由于某一组条件取值的组合而引发的，使用判定表描述比较适宜。
数据字典是结构化分析的核心。

软件需求规格说明书的特点：
（1）正确性；
（2）无岐义性；
（3）完整性；
（4）可验证性；
（5）一致性；
（6）可理解性；
（7）可追踪性。

3．3 结构化设计方法

软件设计的基本目标是用比较抽象概括的方式确定目标系统如何完成预定的任务，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。

从技术观点来看，软件设计包括软件结构设计、数据设计、接口设计、过程设计。

结构设计：定义软件系统各主要部件之间的关系。

数据设计：将分析时创建的模型转化为数据结构的定义。

接口设计：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。

过程设计：把系统结构部件转换成软件的过程描述。

从工程管理角度来看：概要设计和详细设计。

软件设计的一般过程：软件设计是一个迭代的过程；先进行高层次的结构设计；后进行低层次的过程设计；穿插进行数据设计和接口设计。

衡量软件模块独立性使用耦合性和内聚性两个定性的度量标准。

在程序结构中各模块的内聚性越强，则耦合性越弱。优秀软件应高内聚，低耦合。

软件概要设计的基本任务是：
（1）设计软件系统结构；　（2）数据结构及数据库设计；
（3）编写概要设计文档；　（4）概要设计文档评审。

模块用一个矩形表示，箭头表示模块间的调用关系。

在结构图中还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。

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。

事务型数据流的特点是：接受一项事务，根据事务处理的特点和性质，选择分派一个适当的处理单元，然后给出结果。

详细设计：是为软件结构图中的每一个模块确定实现算法和局部数据结构，用某种选定的表达工具表示算法和数据结构的细节。

常见的过程设计工具有：图形工具（程序流程图）、表格工具（判定表）、语言工具（PDL）。

3．4 软件测试

软件测试定义：使用人工或自动手段来运行或测定某个系统的过程，其目的在于检验它是否满足规定的需求或是弄清预期结果与实际结果之间的差别。

软件测试的目的：发现错误而执行程序的过程。

软件测试方法：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：是基本计算机的测试，主要包括白盒测试方法和黑盒测试方法。
白盒测试：在程序内部进行，主要用于完成软件内部操作的验证。主要方法有逻辑覆盖、基本基路径测试。
黑盒测试：主要诊断功能不对或遗漏、界面错误、数据结构或外部数据库访问错误、性能错误、初始化和终止条件错，用于软件确认。主要方法有等价类划分法、边界值分析法、错误推测法、因果图等。
软件测试过程一般按 4 个步骤进行：单元测试、集成测试、验收测试（确认测试）和系统测试。

3．5 程序的调试

程序调试的任务是诊断和改正程序中的错误，主要在开发阶段进行。
程序调试的基本步骤：
（1）错误定位；
（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。
软件调试可分表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。主要调试方法有：
（1）强行排错法；
（2）回溯法；
（3）原因排除法。

第四章 数据库设计基础

4．1 数据库系统的基本概念

数据：实际上就是描述事物的符号记录。
数据的特点：有一定的结构，有型与值之分，如整型、实型、字符型等。而数据的值给出了符合定型的值，如整型值 15。
数据库：是数据的集合，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共享。
数据库存放数据是按数据所提供的数据模式存放的，具有集成与共享的特点。
数据库管理系统：一种系统软件，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，是数据库的核心。
数据库管理系统功能：
（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、安生性定义与检查；
（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。
为完成以上六个功能，数据库管理系统提供以下的数据语言：
（1）数据定义语言：负责数据的模式定义与数据的物理存取构建；
（2）数据操纵语言：负责数据的操纵，如查询与增、删、改等；
（3）数据控制语言：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。
数据语言按其使用方式具有两种结构形式：交互式命令(又称自含型或自主型语言)宿主型语

言（一般可嵌入某些宿主语言中）。
数据库管理员：对数据库进行规划、设计、维护、监视等的专业管理人员。
数据库系统：由数据库（数据）、数据库管理系统（软件）、数据库管理员（人员）、硬件平台（硬件）、软件平台（软件）五个部分构成的运行实体。
数据库应用系统：由数据库系统、应用软件及应用界面三者组成。
文件系统阶段：提供了简单的数据共享与数据管理能力，但是它无法提供完整的、统一的、管理和数据共享的能力。
层次数据库与网状数据库系统阶段 ：为统一与共享数据提供了有力支撑。
关系数据库系统阶段
数据库系统的基本特点：数据的集成性 、数据的高共享性与低冗余性 、数据独立性（物理独立性与逻辑独立性）、数据统一管理与控制。
数据库系统的三级模式：
（1）概念模式：数据库系统中全局数据逻辑结构的描述，全体用户公共数据视图；
（2）外模式：也称子模式与用户模式。是用户的数据视图，也就是用户所见到的数据模式；
（3）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。
数据库系统的两级映射：
（1）概念模式到内模式的映射；
（2）外模式到概念模式的映射。

4.2 数据模型

数据模型的概念：是数据特征的抽象，从抽象层次上描述了系统的静态特征、动态行为和约束条件，为数据库系统的信息表与操作提供一个抽象的框架。描述了数据结构、数据操作及数据约束。
E-R 模型的基本概念
（1）实体：现实世界中的事物；
（2）属性：事物的特性；
（3）联系：现实世界中事物间的关系。实体集的关系有一对一、一对多、多对多的联系。
E-R 模型三个基本概念之间的联接关系：实体是概念世界中的基本单位，属性有属性域，每个实体可取属性域内的值。一个实体的所有属性值叫元组。
E-R 模型的图示法：（1）实体集表示法； （2）属性表法； （3）联系表示法。
层次模型的基本结构是树形结构，具有以下特点：
（1）每棵树有且仅有一个无双亲结点，称为根；
（2）树中除根外所有结点有且仅有一个双亲。
从图论上看，网状模型是一个不加任何条件限制的无向图。
关系模型采用二维表来表示，简称表，由表框架及表的元组组成。一个二维表就是一个关系。在二维表中凡能唯一标识元组的最小属性称为键或码。从所有侯选健中选取一个作为用户使用的键称主键。表 A 中的某属性是某表 B 的键，则称该属性集为 A 的外键或外码。
关系中的数据约束：
（1）实体完整性约束：约束关系的主键中属性值不能为空值；
（2）参照完全性约束：是关系之间的基本约束；
（3）用户定义的完整性约束：它反映了具体应用中数据的语义要求。

4.3 关系代数

关系数据库系统的特点之一是它建立在数据理论的基础之上，有很多数据理论可以表示关系模型的数据操作，其中最为著名的是关系代数与关系演算。
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）

4.4 数据库设计与管理

数据库设计是数据应用的核心。

数据库设计的两种方法：

（1）面向数据：以信息需求为主，兼顾处理需求；

（2）面向过程：以处理需求为主，兼顾信息需求。

数据库的生命周期：需求分析阶段、概念设计阶段、逻辑设计阶段、物理设计阶段、编码阶段、测试阶段、运行阶段、进一步修改阶段。

需求分析常用结构析方法和面向对象的方法。结构化分析（简称 SA）方法用自顶向下、逐层分解的方式分析系统。用数据流图表达数据和处理过程的关系。对数据库设计来讲，数据字典是进行详细的数据收集和数据分析所获得的主要结果。

数据字典是各类数据描述的集合，包括 5 个部分：数据项、数据结构、数据流（可以是数据项，也可以是数据结构）、数据存储、处理过程。

数据库概念设计的目的是分析数据内在语义关系。设计的方法有两种

（1）集中式模式设计法（适用于小型或并不复杂的单位或部门）；

（2）视图集成设计法。

设计方法：E-R 模型与视图集成。

视图设计一般有三种设计次序：自顶向下、由底向上、由内向外。

视图集成的几种冲突：命名冲突、概念冲突、域冲突、约束冲突。

关系视图设计：关系视图的设计又称外模式设计。

关系视图的主要作用：

（1）提供数据逻辑独立性；

（2）能适应用户对数据的不同需求；

（3）有一定数据保密功能。

数据库的物理设计主要目标是对数据内部物理结构作调整并选择合理的存取路径，以提高数据库访问速度有效利用存储空间。一般 RDBMS 中留给用户参与物理设计的内容大致有索引设计、集成簇设计和分区设计。

数据库管理的内容：

（1）数据库的建立；

（2）数据库的调整；

（3）数据库的重组；

（4）数据库安全性与完整性控制；

（5）数据库的故障恢复；