**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**

1、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。

2、算法的基本特征：

1）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；

2）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；

3）可行性，算法原则上能够精确地执行；

4）拥有足够的情报。

3、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。

4、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。

5、算法的基本控制结构：顺序，选择，循环。

6、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**

1、算法效率的度量——算法的复杂度：时间复杂度和空间复杂度。

1）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。

2）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**

1、数据结构可分为数据的逻辑结构和存储结构。

1）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。

2）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。

2、存储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。

3、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**

1、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。

2、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**

1、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。

2、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。

3、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。

4、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。

5、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

单链表的结构（图1）

……

数据域 指针域

数据域 指针域

数据域 指针域

……

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。

L D R

……

图2 双链表的结构

L D R

L D R

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。

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。

4、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。

5、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。

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。

**【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

1、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。

2、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。

3、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。

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：

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。

**【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**

1、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。

2、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。

3、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。

4、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 |

1、软件的特点：

1）软件是一种逻辑实体，具有抽象性；

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。

2、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。

可行性研究初步项目计划

需求

分析

概要

设计

详细

分析

实现

测试

维护

退役

交付使用

定义阶段 开发阶段 维护阶段

**【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 |
| 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**

1、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。

2、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。

3、软件开发方法包括分析方法，设计方法和程序设计方法。

4、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。

5、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。

6、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。

**【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

（1）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。

1）箭头：表示数据流，数据流是数据在系统中传播的路径。

2）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。

3）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。

4）方框：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

（2）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

（3）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。

（4）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。

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 |
| 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。