

# 结构化开发方法

# 概述

- 结构化方法由**结构化分析、结构化设计、结构化程序设计**构成，它是一种面向数据流的开发方法。结构化分析是根据分解与抽象的原则，按照系统中数据处理的流程，**用数据流图来建立系统的功能模型，从而完成需求分析工作**。结构化设计是根据模块独立性准则、软件结构优化准则将数据流图转换为软件的体系结构，用软件结构图来建立系统的物理模型，实现系统的概要设计。结构化程序设计使用3种基本控制结构构造程序，任何程序都可以由顺序、选择和重复3种基本控制结构构造。
- 结构化方法的核心思想是“自顶向下，逐步分解”。特别适合于数据处理领域的问题，但是**不适合**解决大规模的、特别复杂的项目，且难以适应需求的变化。

# 结构化分析

- (1) 研究“物质环境”。首先，应画出当前系统（可能是非计算机系统，或是半计算机系统）的**数据流图**，说明系统的输入、输出数据流，说明系统的数据流情况，以及经历了哪些处理过程。在这个数据流图中，可以包括一些非计算机系统中数据流及处理的命名，例如部门名、岗位名、报表名等。这个过程可以帮助分析员有效地理解业务环境，在与用户的充分沟通与交流中完成。
- (2) 建立系统逻辑模型。当物理模型建立完成之后，接下来的工作就是画出相对于真实系统的等价逻辑数据流图。在前一步骤建立的数据流图的基础上，将所有自然数据流都转成等价的逻辑流，例如，将现实世界的报表存储在计算机系统里的文件里；又如将现实世界中“送往总经理办公室”改为“报送报表”。


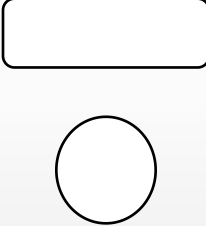
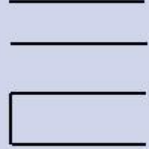

# 结构化分析

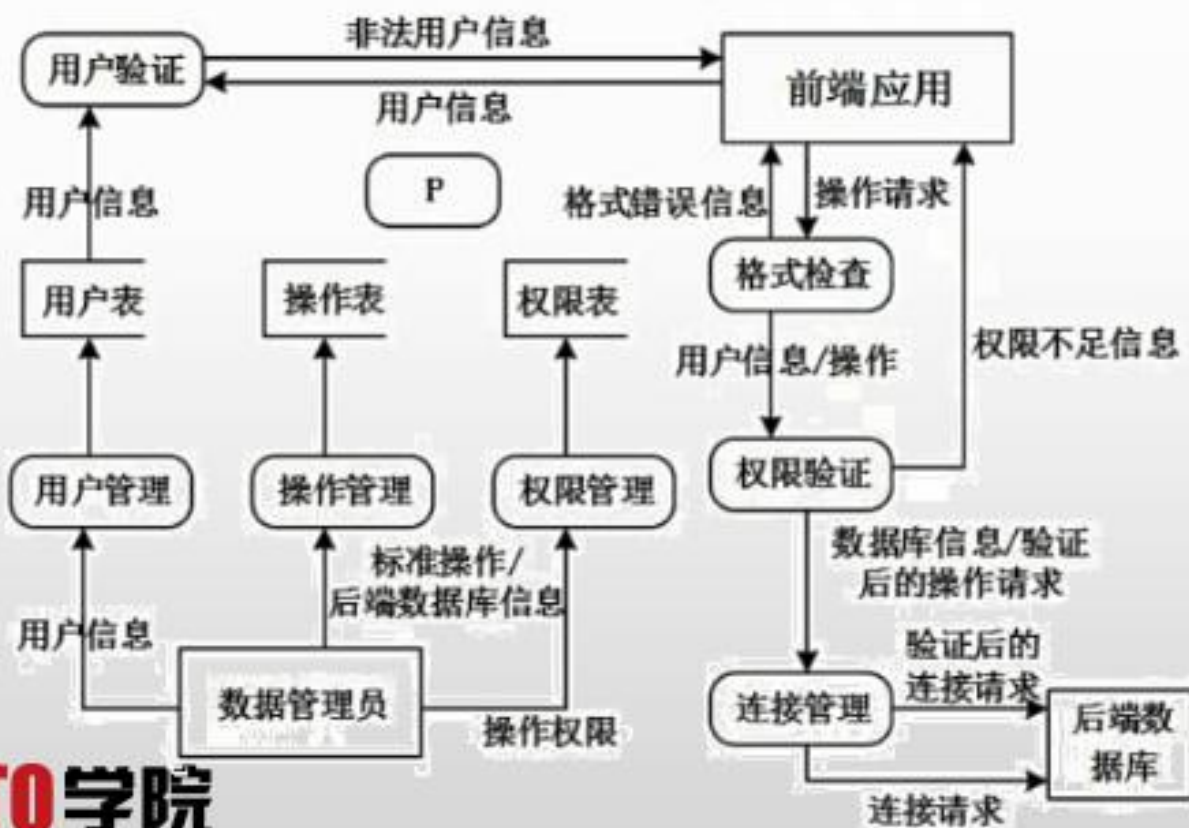
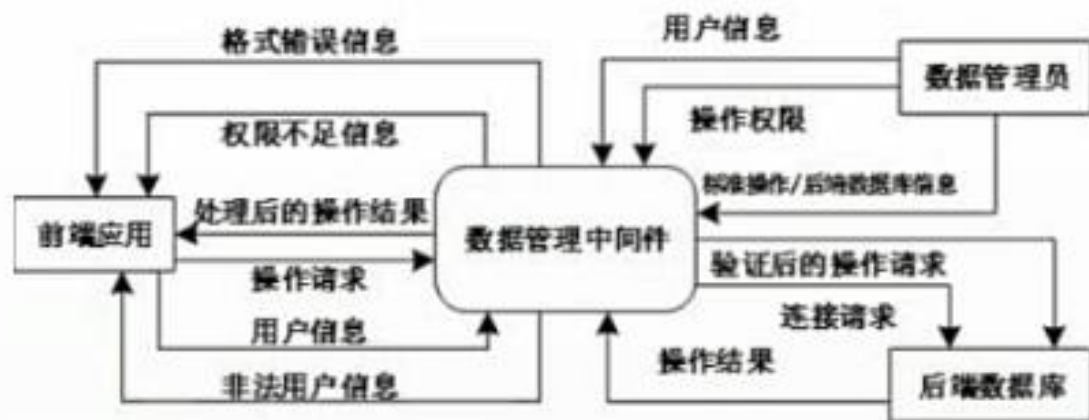
- (3) 划清人机界限。最后，确定在系统逻辑模型中，哪些将采用自动化完成，哪些仍然保留手工操作。这样，就可以清晰地划清系统的范围。
- 结构化分析与面向对象分析方法之间的最大差别是：结构化分析方法把系统看作一个过程的集合体，包括人完成的和电脑完成的；而面向对象方法则把系统看成一个相互影响的对象集。结构化分析方法的特点是利用数据流图来帮助人们理解问题，对问题进行分析。
- 结构化分析一般包括以下工具：**数据流图 (Data Flow Diagram, DFD)**、**数据字典 (Data Dictionary, DD)**、**结构化语言**、**判定表**、**判定树**。

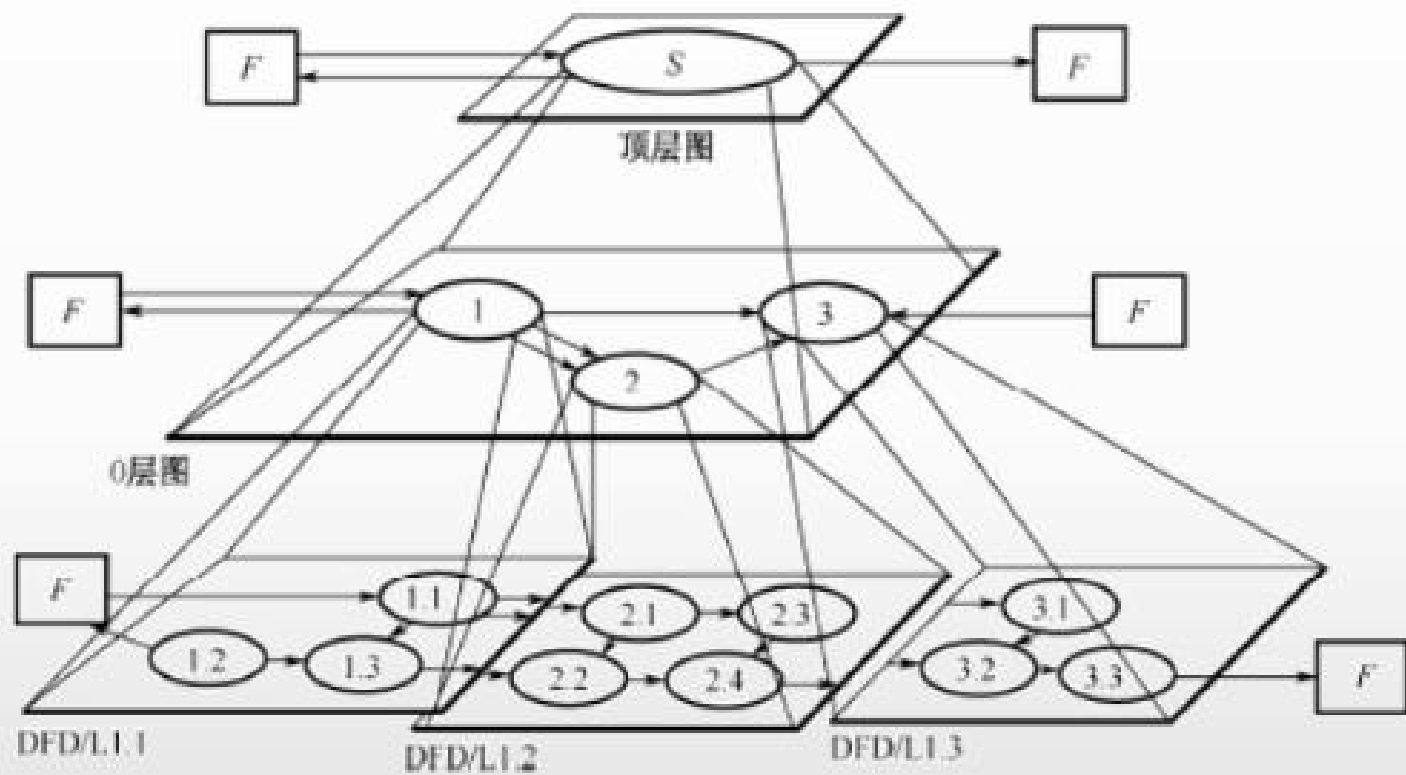
# 结构化分析

- 结构化系统分析方法从总体上来看是一种强烈依赖数据流图的自顶向下的建模方法。它不仅是需求分析技术，也是完成需求规格化的有效技术手段。
- 结构化方法的分析结果由以下几部分组成：一套分层的数据流图、一本数据词典、一组小说明（也称加工逻辑说明）、补充材料。

# 数据流图

元素	说明	说明
数据流	由一组固定成分的数据组成，表示 <b>数据的流向</b> 。每个数据流用一个定义明确的名字表示，以反映数据流的含义。	
加工	描述了输入数据流到输出数据流之间的变换， <b>输入数据流经过什么处理后变成了输出数据流</b> 。	
数据存储	以记录文件或记录表的形式来存储数据。	
外部实体	存在于软件系统之外的人员或组织。数据的发源地与归宿地。	

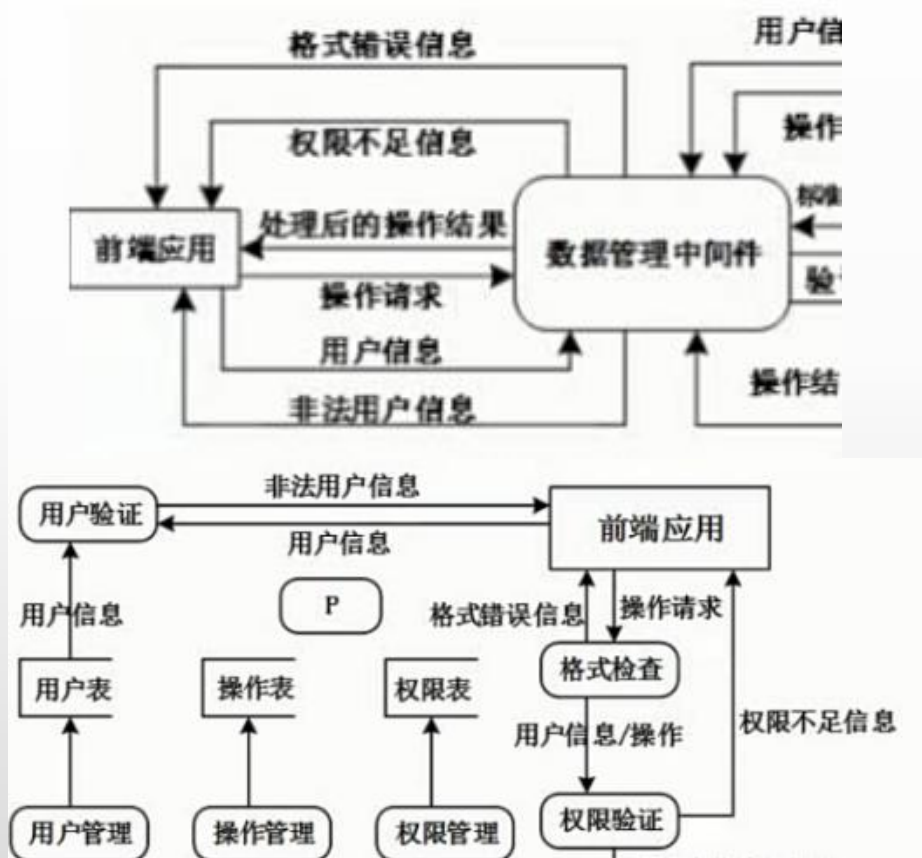






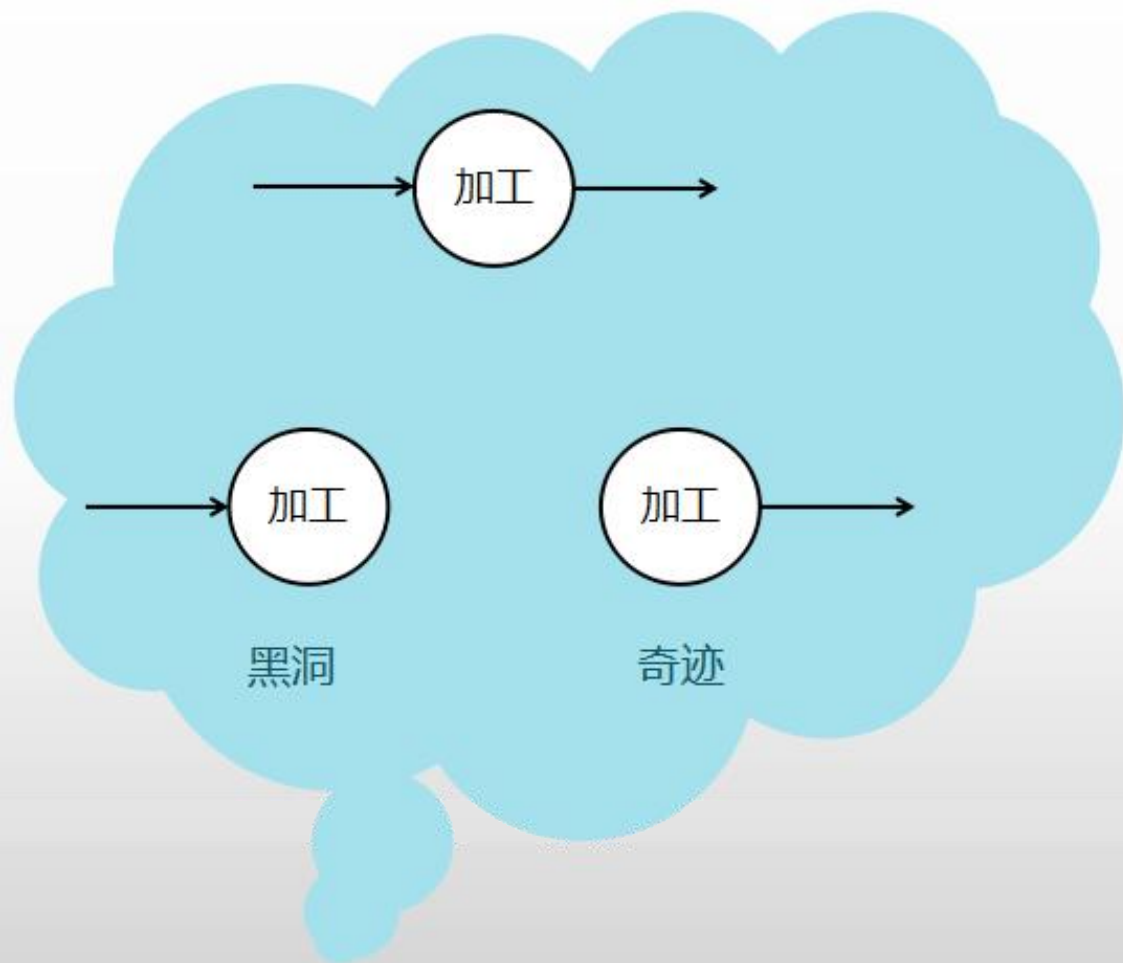
# 数据流图的平衡原则

- 1、父图与子图的平衡



# 数据流图的平衡原则

- 2、子图内部的平衡



# 数据字典

- 数据字典技术是一种很实用、有效的表达数据格式的手段。它是对所有与系统相关的 数据元素的一个有组织的列表和精确严格的定义，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机有共同的理解。
- 数据字典有以下4类条目：**数据流、数据项、数据存储和基本加**

符号	含义	举例说明
=	被定义为	
+	与	$x=a+b$ ，表示x由a和b组成
[..., ...]或 [... ...]	或	$x=[a, b]$ ， $x=[a b]$ ，表示x由a或由b组成
{...}	重复	$x=\{a\}$ ，表示x由0或多个a组成
(...)	可选	$x=(a)$ ，表示a可以在x中出现，也可以不出现

- 客户基本信息=客户编号+客户名称+身份证号码+手机+家庭电话

# 典型真题

- 结构化分析的输出不包括（ ）。
- A.数据流图
- B.数据字典
- C.加工逻辑
- D.结构图

# 典型真题

- 试题分析
- 《软件设计师教程（第5版）》P325页：结构化方法的分析结果由以下几部分组成：一套分层的数据流图、一本数据词典、一组小说明（也称加工逻辑说明）、补充材料。
- 因此本题选择D选项，结构图不属于结构化分析的输出。
- 试题答案：D

# 典型真题

- 某航空公司拟开发一个机票预订系统，旅客预订机票时使用信用卡付款。付款通过信用卡公司的信用卡管理系统提供的接口实现。若采用数据流图建立需求模型，则信用卡管理系统是（ ）。
- A.外部实体
- B.加工
- C.数据流
- D.数据存储

# 典型真题

- 试题分析
- 数据流图中的基本图形元素包括数据流、加工、数据存储和外部实体。其中，数据流、加工和数据存储用于构建软件系统内部的数据处理模型，而外部实体表示存在于系统之外的对象，用来帮助用户理解系统数据的来源和去向。外部实体包括：人/物、外部系统、组织机构等。
- 试题答案：A

# 典型真题

- 某医院预约系统的部分需求为：患者可以查看医院发布的专家特长介绍及其就诊时间；系统记录患者信息，患者预约特定时间就诊。用DFD对其进行功能建模时，患者是（ ）；用ERD对其进行数据建模时，患者是（ ）。
- A.外部实体                      B.加工
- C.数据流                          D.数据存储
- A.实体                              B.属性
- C.联系                              D.弱实体



# 典型真题

- 1、患者不涉及加工，为外部实体。
- 2、患者有其信息，所以为实体。
- 试题答案：A、A

# 结构化设计

- 结构化设计包括体系结构设计、接口设计、数据设计和过程设计等任务。它是一种面向数据流的设计方法，是以结构化分析阶段所产生的成果为基础，进一步自顶而下、逐步求精和模块化的过程。
- 在结构化方法中，模块化是一个很重要的概念，它将一个待开发的软件分解成为若干个小的简单部分--模块，每个模块可以独立地开发、测试。这是一种复杂问题的"分而治之"原则，其目的是使程序的结构清晰、易于测试与修改。
- 具体来说，模块是指执行某一特定任务的数据结构和程序代码。通常将模块的接口和功能定义为其外部特性，将模块的局部数据和实现该模块的程序代码称为内部特性。而在模块设计时，最重要的原则就是实现信息隐蔽和模块独立。模块通常具有连续性，也就意味着作用于系统的小变动将导致行为上的小变化，同时规模说明的小变动也将影响到一小部分模块。

- **1.抽象化**

- 对软件进行模块设计的时候，可以有不同的抽象层次。在最高的抽象层次上，可以使问题所处环境的语言描述问题的解法。而在较低的抽象层次上，则采用过程化的方法。抽象化包括对过程的抽象、对数据的抽象和对控制的抽象。
- (1) 过程的抽象。在软件工程过程中，从系统定义到实现，每进展一步都可以看做是对软件解决方案的抽象化过程的一次细化。在从概要设计到详细设计的过程中，抽象化的层次逐次降低。当产生源程序时到达最低的抽象层次。
- (2) 数据抽象。数据抽象与过程抽象一样，允许设计人员在不同层次上描述数据对象的细节。
- (3) 控制抽象。控制抽象可以包含一个程序控制机制而无须规定其

- **2.自顶向下，逐步细化**

- 将软件的体系结构按自顶向下方式，对各个层次的过程细节和数据细节逐层细化，直到用程序设计语言的语句能够实现为止，从而最后确立整个的体系结构。最初的说明只是概念性地描述了系统的功能或信息，但并未提供有关功能的内部实现机制或有关信息的内部结构的任何信息。设计人员对初始说明仔细推敲，进行功能细化或信息细化，给出实现的细节，划分出若干成分。然后再对这些成分，施行同样的细化工作。随着细化工作的逐步展开，设计人员就能得到越来越多的细节。

-

- **3.信息隐蔽**
- 信息隐蔽是开发整体程序结构时使用的法则，即将每个程序的成分隐蔽或封装在一个单一的设计模块中，并且尽可能少地暴露其内部的处理。通常我们将难的决策、可能修改的决策、数据结构的内部连接，以及对它所做的操作细节、内部特征码、与计算机硬件有关的细节等隐蔽起来。
- 通过信息隐蔽可以提高软件的可修改性、可测试性和可移植性，它也是现代软件设计的一个关键性原则。

- **4.模块独立**

- 模块独立是指每个模块完成一个相对独立的特定子功能，并且与其他模块之间的联系最简单。保持模块的高度独立性，也是在设计时的一个很重要的原则。通常我们用耦合（模块之间联系的紧密程度）和内聚（模块内部各元素之间联系的紧密程度）两个标准来衡量，**我们的目标是高内聚、低耦合**。通过信息隐蔽可以提高软件的可修改性、可测试性和可移植性，它也是现代软件设计的一个关键性原则。

- 在结构化设计中，系统由多个逻辑上相对独立的模块组成，在模块划分时需要遵循如下原则：
- **(1) 模块的大小要适中。**系统分解时需要考虑模块的规模，过大的模块可能导致系统分解不充分，其内部可能包括不同类型的功能，需要进一步划分，尽量使得各个模块的功能单一；过小的模块将导致系统的复杂度增加，模块之间的调用过于频繁，反而降低了模块的独立性。一般来说，一个模块的大小使其实现代码在1~2页纸之内，或者其实现代码行数在50~200行之间，这种规模的模块易于实现和维护。

- **(2) 模块的扇入和扇出要合理。**一个模块的扇出是指该模块直接调用的下级模块的个数；扇出大表示模块的复杂度高，需要控制和协调过多的下级模块。扇出过大一般是因为缺乏中间层次，应该适当增加中间层次的控制模块；扇出太小时可以把下级模块进一步分解成若干个子功能模块，或者合并到它的上级模块中去。一个模块的扇入是指直接调用该模块的上级模块的个数；扇入大表示模块的复用程度高。设计良好的软件结构通常顶层扇出比较大，中间扇出较少，底层模块则有大扇入。一般来说，系统的平均扇入和扇出系数为3或4，不应该超过7，否则会增大出错的概率。

•



- **(3) 深度和宽度适当。**深度表示软件结构中模块的层数，如果层数过多，则应考虑是否有些模块设计过于简单，看能否适当合并。宽度是软件结构中同一个层次上的模块总数的最大值，一般说来，宽度越大系统越复杂，对宽度影响最大的因素是模块的扇出。在系统设计时，需要权衡系统的深度和宽度，尽量降低系统的复杂性，减少实施过程的难度，提高开发和维护的效率。
- 模块的扇入指模块直接上级模块的个数。模块的直属下级模块个数即为模块的扇出。

- 模块的**内聚**类型通常可以分为**7种**，根据内聚度**从高到低排序**。

内聚类型	描 述
功能内聚	完成一个单一功能，各个部分协同工作，缺一不可
顺序内聚	处理元素相关，而且必须顺序执行
通信内聚	所有处理元素集中在一个数据结构的区域上
过程内聚	处理元素相关，而且必须按特定的次序执行
瞬时内聚	所包含的任务必须在同一时间间隔内执行（如初始化模块）
逻辑内聚	完成逻辑上相关的一组任务
偶然内聚	完成一组没有关系或松散关系的任务

- 模块的**耦合**类型通常也分为**7种**，根据耦合度**从低到高排序**。

耦合类型	描 述
非直接耦合	没有直接联系，互相不依赖对方
数据耦合	借助参数表传递简单数据
标记耦合	一个数据结构的一部分借助于模块接口被传递
控制耦合	模块间传递的信息中包含用于控制模块内部逻辑的信息
外部耦合	与软件以外的环境有关
公共耦合	多个模块引用同一个全局数据区
内容耦合	一个模块访问另一个模块的内部数据；一个模块不通过正常入口转到另一模块的内部；两个模块有一部分程序代码重叠；一个模块有多个入口

# 典型真题

- 以下关于模块化设计的叙述中，正确的是（ ）。
- A.尽量考虑低内聚、高耦合，保持模块的相对独立性
- B.通过信息隐蔽可以提高软件的可修改性、可测试性和可移植性，它也是现代软件设计的一个关键性原则。
- C.模块的规模要大
- D. 模块的扇入指模块直接下级模块的个数。模块的直属上级模块个数即为模块的扇出。

# 典型真题

- 试题分析：
- 略
- 试题答案： B

# 典型真题

- 耦合是模块之间的相对独立性（互相连接的紧密程度）的度量。耦合程度不取决于（ ）。
- A.调用模块的方式
- B.各个模块之间接口的复杂程度
- C.通过接口的信息类型
- D.模块提供的功能数

# 典型真题

- 试题分析
- 耦合性也叫块间联系。指软件系统结构中各模块间相互联系紧密程度的一种度量。模块之间联系越紧密，其耦合性就越强，模块之间越独立则越差，模块间耦合的高低取决于模块间接口的复杂性，调用的方式以及传递的信息。
- 试题答案：D

# 典型真题

- 某企业管理信息系统中，采购子系统根据材料价格、数量等信息计算采购的金额，并给财务子系统传递采购金额、收款方和采购日期等信息，则这两个子系统之间的耦合类型为（ ）耦合。
- A.数据
- B.标记
- C.控制
- D.外部

- **试题分析**
- **非直接耦合：**两个模块之间没有直接关系，它们之间的联系完全是通过主模块的控制和调用来实现的。
- **数据耦合：**一个模块访问另一个模块时，彼此之间是通过简单数据参数（不是控制参数、公共数据结构或外部变量）来交换输入、输出信息的。
- **标记耦合：**这个记录是某一数据结构的子结构，而不是简单变量。其实传递的是这个数据结构的地址；
- **控制耦合：**如果一个模块通过传送开关、标志、名字等控制信息，明显地控制选择另一模块的功能，就是控制耦合。



- **外部耦合：**一组模块都访问同一全局简单变量而不是同一全局数据结构，而且不是通过参数表传递该全局变量的信息，则称之为外部耦合。
- **公共耦合：**若一组模块都访问同一个公共数据环境，则它们之间的耦合就称为公共耦合。公共的数据环境可以是全局数据结构、共享的通信区、内存的公共覆盖区等。
- **内容耦合：**如果发生下列情形，两个模块之间就发生了内容耦合
  - (1) 一个模块直接访问另一个模块的内部数据；
  - (2) 一个模块不通过正常入口转到另一模块内部；
  - (3) 两个模块有一部分程序代码重叠(只可能出现在汇编语言中)；
  - (4) 一个模块有多个入口。

# 典型真题

- 本题属于数据耦合，采购子系统模块给财务子系统模块传递数据。
- 试题答案：A

# 典型真题

- 模块A、B和C有相同的程序块，块内的语句之间没有任何联系，现把改程序块取出来，形成新的模块D，则模块D的内聚类型为（ ）  
内聚。以下关于该内聚类型的叙述中，不正确的是（ ）。

- A.巧合                      B.逻辑
- C.时间                      D.过程
- A.具有最低的内聚性
- B.不易修改和维护
- C.不易理解
- D.不影响模块间的耦合关系

# 典型真题

- 试题分析
- 功能内聚：完成一个单一功能，各个部分协同工作，缺一不可。
- 顺序内聚：处理元素相关，而且必须顺序执行。
- 通信内聚：所有处理元素集中在一个数据结构的区域上。
- 过程内聚：处理元素相关，而且必须按特定的次序执行。
- 瞬时内聚：所包含的任务必须在同一时间间隔内执行（如初始化模块）。
- 逻辑内聚：完成逻辑上相关的一组任务。
- 偶然内聚：完成一组没有关系或松散关系的任务。

# 典型真题

- 巧合内聚就是偶然内聚。偶然内聚由于内容都是不相关的，所以必然导致它与外界多个模块有关联，这也使得模块间的耦合度增加。
- 试题答案：A、D

技术成就梦想