# 第一节 基本任务：

从“做什么”变换到“怎么做”的物理模型。

1. 将复杂的系统划分成模块
2. 确定每个模块的功能
3. 确定模块的调用关系
4. 确定模块间的接口
5. 评价模块结构的质量

# 第二节 软件设计的基本原理

1. 模块化

模块：程序中数据说明，可执行语句等程序对象的集合，或单独命名和编址的元素，如高级语言的函数，子程序等。

模块的基本属性：

接口：模块的输入输出

功能

逻辑

状态：模块调用和被调用的关系

模块化指解决一个复杂问题自顶向下逐层把软件系统分成若干模块过程。

1. 抽象

认识复杂现象过程中的思维工具。即抽象出事物本质的共同特性暂不考虑它的细节。

1. 信息隐蔽

抽象确定组成软件的过程实体。信息隐蔽定义和实施对模块过程细节和局部数据结构的存取限制。

1. 模块独立性

只完成独立的子功能，并与其他模块的联系少接口简单。

衡量标准：耦合性和内聚性

1. 耦合性：模块间的联系
2. 无直接耦合：不传递任何信息
3. 数据耦合：有调用关系，只传递简单的数据值
4. 标记耦合：传递数据结构（其实是地址）
5. 控制耦合：一个模块掉另一个，通过传递控制变量信息有选择执行块内功能
6. 公共耦合：多个模块的公共数据
7. 内容耦合：直接或通过非正常入口使用另一个模块的内部数据
8. 内聚性：模块内的联系
9. 偶然内聚：元素之间没有任何联系
10. 逻辑内聚：几个逻辑上相识的功能组合，通过参数决定执行的功能
11. 时间内聚：同时执行的动作组合在一起
12. 通信内聚：在同一个数据结构上的操作
13. 顺序内聚：模块元素都关于同一功能，前一功能的输出是下一功能的输出
14. 功能内聚：共同完成一个任务，缺一不可

# 第三节 软件结构优化准则

1. 软件结构图
2. 注意

1.调用只能从上到下，表示控制关系，不用画箭头也可以。（说明是方法调方法）

1. 优化准则
2. 划分模块时做到高内聚，低耦合，保持模块相当独立。
3. 若干模块耦合过高，每个模块内功能不复杂，可将它们合并，以减少信息传递和公共区的引用。
4. 多个相关模块，对它们功能进行分析，消去重复功能。

(2)一个模块的作用范围应在其控制之内，否则增加数据的传输和模块的耦合，不易维护。

，且要和影响模块的层次上尽量靠近，不然也会额外的数据传输。中间模块不需要这些数据。

模块的作用范围：影响所以模块的范围，包括自己和下属。

模块的控制范围：本身及下属（间接或直接从属）的集合。

改进方法：上移判断点，或下移受判断的。软件结构的深度，宽度，扇入扇出适当。

# 第四节 面向数据流的设计方法

1. 数据流的类型

变化型

事务型