# 3危机原因

# 忽略软件需求分析

# 轻视软件维护

# 重视程序而忽视软件配置其余成分 7基本原理

# 用分阶段的生命周期计划严格管理

# 坚持进行阶段评审

# 实行严格的产品控制

# 采用现代程序设计技术

# 结果应能清楚地审查

# 开发小组的人员应该少而精

# 承认不断改进软件工程实践的必要性 9方法要素

# 软件工程方法学包含三个要素：方法、工具、过程

目前使用得最广泛的软件工程方法学：传统方法学和面向对象方法学。P9

# 软件过程：

# 瀑布模型：阶段间具有顺序性和依赖性、保证质量、保证前面没有错误、

# 快速原型模型：优点：在正式开发之前先进行原型设计，用户认为原型ok，才开始进行开发，

# 有助于保证用户的真实需求被满足

# 增量模型：先把初始产品给用户，让用户的反馈去驱动下一步的产品优化

# 螺旋模型：使用原型及其他方法来降低风险，风险驱动规模与进度。

# 喷泉模型：体现面向对象软件开发过程迭代和无缝的特性

# RUP （Rational统一过程）：有RUP就选RUP。 35三种可行性研究

# 技术可行性：现有技术能不能实现

# 经济可行性：这个产品的经济效益能够超过它的开发运营成本吗

# 操作可行性：系统的操作方式在这个用户组织内行得通吗 41数据流图​（DFD）会画 58获取需求方法

1.访谈 2.面向数据流自顶向下求精 3.简易的应用规格说明技术4.快速建立软件原型

# 97 耦合与内聚

# 耦合：耦合是对一个软件结构内不同模块之间互连程度的度量。

# 模块间无连接，则耦合程度最低；但软件系统中模块间不可能无连接。

# 耦合程度低到高：

# 数据耦合：通过参数交换信息，交换的信息仅是数据。低耦合，系统中必然存在的耦合

# 控制耦合：传递的信息中有控制信息。中等程度耦合，可以分解后用数据耦合替代。

# 特征耦合：把整个数据结构作为参数传递，但只用到其中一部分数据。有安全风险

# 公共环境耦合：多个模块通过一个公共数据环境相互作用。当共享的数据很多时，用公共环境要比参数传 递方便

# 内容耦合：最高程度耦合，坚决避免使用：A访问了B的内部数据、A可通过非正常入口进入B内部、AB 代码重叠、A有多个入口

# 尽量使用数据耦合，少用控制耦合和特征耦合，限制公共环境耦合的范围，完全不用内容耦合。

# 内聚：内聚表示一个模块内各个元素彼此结合的紧密程度

# 低内聚：不好

# 偶然内聚：有一些关系，但很少

# 逻辑内聚：逻辑上有一些关系

# 时间内聚：一个模块内包含的任务必须在同一段时间内执行（初始化）

# 中内聚：

# 过程内聚：模块内处理元素相同，必须以特定次序执行（流程图）

# 通信内聚：模块中所有元素都用同一个输入数据和同一个输出数据

# 高内聚：好

# 顺序内聚：模块内处理元素和同一个功能密切相关，处理必须顺序执行

# 功能内聚：模块内所有处理元素都属于一个整体，完成单一功能

# 136 给出一段程序，计算其环形复杂度，复杂度代表有多少条独立路径

# 151方法步骤​

# 测试方法：两种

# 黑盒测试（功能测试）：不考虑程序内部结构和处理过程，在程序接口上进行测试，只检查程序功能 是否能按照规格说明书的规定正常使用，能否正确输出信息，运行过程能否保持外部 信息的完整性。

# 白盒测试（结构测试）：在知道程序的结构和处理算法后，对程序内部的逻辑进行测试，检查执行通 路是否能按照预定要求正确工作。

# 测试步骤：

# 模块测试：对模块内的代码和逻辑进行测试

# 子系统测试：对几个模块形成的子系统之间的逻辑与连接进行测试

# 系统测试：对几个子系统之间的通信和连接进行测试

# 验收测试：与系统测试类似，但是这种测试有用户参与

# 平行运行：新版本和旧版本同步运行

# 161两种测试（确认测试）

# ​Alpha测试：由用户在开发者的场所进行，并且在开发者对用户的“指导”下进行测试

# Beta测试：由软件的最终用户们在一个或多个客户场所进行

# 162 白盒测试：重点看逻辑覆盖

# 逻辑覆盖包括：

# 语句覆盖：每条语句至少覆盖一次

# 判定覆盖：每条语句至少覆盖一次，每个判定的1和0都要至少执行一次

# 条件覆盖：每条语句至少覆盖一次，判定表达式中的每个条件都取得各种可能结果

# 判定/条件覆盖：判定覆盖不一定包含条件覆盖，条件覆盖也不一定包含判定覆盖；判定表 达式中每个条件都取可能的值，判定表达式也要取1和0的值

# 条件组合覆盖：判定表达式中条件的各种可能组合都至少出现一次

# 点覆盖：每条语句都覆盖，类似语句覆盖

# 边覆盖：类似判定覆盖，每条线经过一次

# 路径覆盖：每一条路径经过一次

171 黑盒测试的方法：等价划分、边界值分析、错误推断。P171

# 172等价划分

# 是一种黑盒测试技术，把输入域划分为若干个数据类，选取少量有代表性的输入数据作为测试数据，用较小的代价暴露较多的程序错误。如果规定了输入值范围，则有一个有效类，和两个无效类；

# 175边界值分析：边界情况时程序最容易发生错误。边界值分析法应该选择更好等于、邵晓宇和稍大于等价类边界值的数据作为测试数据，而不是选取每个等价类内的典型值或任意值作为测试数据。

# 190维护：软件开发过程工作量最大的是维护

# 完善性维护占50%-66%

# 改正性维护站17%-21%

# 适应性维护占18%-25%

# 其他维护4%

**UML的中文名称是统一建模语言，英文名称是Unified Modeling Language，它是一种语言而不是方法。**

# 194可维护性因素

# 决定软件可维护性的因素：

# 可理解性

# 可测试性

# 可修改性

# 可移植性

# 可重用性

# 210 对象的基本特点：

# 以数据为中心

# 对象是主动的

# 实现了数据封装

# 本质上具有并行性

# 模块独立性好

# 216三种模型

# 面向对象建模的三种模型：

# 对象模型：描述系统数据结构 数据结构

# 动态模型：描述系统控制结构 执行操作

# 功能模型：描述系统功能 完成数据值的变化

# 305项目管理

# 管理：通过计划、组织和控制等一系列活动，合理地配置和使用各种资源，以达到既定目标的过程

# 代码行技术：用代码行数估算程序大小

# 功能点技术：对软件学习域特征和软件复杂性的评估结果，估算软件规模

# 308 工作量估算方法：

# 静态单变量模型

# 动态多变量模型

# 331能力成熟度模型

# 由于问题是由人们管理软件过程的方法不当引起的，所以新软件技术的运用并不会自动提高软件的生产率和质量。能力成熟度模型有助于软件开发机构建立一个有规律的、成熟的软件过程。

# CMM的五个成熟度级别反映出一个软件机构蔚来达到从一个无需的混乱的软件过程计划到一种有序的、有纪律的且成熟的软件过程的目的，必须经历的过程改进活动的途径。 CMM的五层模型，每一层的特点。举例：描述的是已定义级的特点，但却说是可重复级的特点

1. 初始级：1级成熟度的软件机构，其过后才能能力是不可预测的，其软件过程是不稳定的，产品质量只能根据相关人员的个人工作能力而不是软件机构的过程能力来预测。
2. 可重复级：2级成熟度软件机构，软件项目的策划和跟踪是稳定的，已经为一个有纪律的管理过程提供了可重复以前成功实践的项目环境。软件项目过程活动处于项目管理体系的有效控制之下，执行着基于以前项目的准则且合乎现实的计划。
3. 已定义级：3级成熟度的软件机构，无论是管理活动还是工程活动都是稳定的。软件开发的成本和进度以及产品的功能和质量都受到控制，而且个软件产品的质量具有可追溯性
4. 已管理级
5. 优化级

数据流图的画法。看一下画的步骤，自顶向下画。数据流的命名。P40

给出一段程序，计算其环形复杂度。复杂度代表有多少条独立路径。P136

NABCD分析（与此有关）

1. N need 需求
2. A approach 做法
3. B benefit 好处
4. C conpetitors 竞争
5. D delivery 交付

开放性题目：与自己小组做的项目有关，比如说用了什么框架，前后端交互的语言等。