## Ch01

**Q1: 什么是软件？软件的特点是什么**？

A1: 软件 = 程序+数据+文档。特点:1)抽象性、2）可复制性、3）无折旧性、4）受硬件制约、5）未完全摆脱手工工艺、6）开发成本高

**Q2: 什么是软件工程？**

A2: 软件工程是一门指导计算机软件开发和维护的工程学科。它采用工程的概念、原理、技术、方法来开发和维护软件。把经过时间考验而证明正确的管理技术同当前能够得到的最好的技术方法结合起来。

**Q3: 简述软件工程的基本原则以及七条基本原理？**

A3: **基本原则**：1）选取适宜的开发风范、2）采用合适的设计方法、3）提供高质量的工程支持、4）有效的软件工程管理

**基本原理：**1）用分阶段的生命周期计划严格管理

2）坚持阶段评审

3）实行严格的产品控制

4）采用现代程序设计技术

5）结果应能清楚地审查

6）开发小组人员应少而精

7）承认不断改进实现软件工程实践的必要性

**Q4:软件生存周期分哪几个阶段？简述每个阶段的任务。**

**软件定义：**

1. 问题定义:问题是什么
2. 可行性研究: 是否有可行的解
3. 需求分析:系统必须做什么

**软件开发：**

1. 总体设计:概括地说，系统应如何解决这个问题
2. 详细设计:怎么具体实现这个系统
3. 编码/单元测试: 正确的程序模块
4. 综合测试:符合要求的软件

**软件维护：**

1. 维护:持久地满足用户需求的软件

**Q5: 简述CMM的五个等级**

初始级、可重复级、已定义级、已管理级、优化级

**Q6:软件工程的三要素是什么？**

A6:过程+工具+方法

**Q7:软件工程的研究内容**

A7: 软件工程主要研究内容包括**软件开发技术**和**软件管理技术**两大方面的问题。软件开发技术的三要素是过程、工具和方法；管理技术包括计划、组织、控制、领导和激励等。

瀑布模型，演化模型（增量，螺旋，原型），喷泉模型

瀑布模型：顺序型，本阶段完成才能开始下个阶段，全部逻辑设计完成才能物理实现。

文档方面：每个阶段都需要完成文档，每次完成必须进行复审。

优点：强迫开发人员采取规范方法和严谨过程+严格规定了每个阶段工作成果+每个阶段必须检查。

缺点：文档驱动模型，用户很晚才能看见软件+不允许或者限制需求变更+不适合模糊开发。

适合项目：需求明确，解决方案清晰。

原型模型：

过程：快速分析+构造原型+运行和评价+修改和改进

特点：通过不断构建和完成来确定需求，适合模糊开发+可以减少不确定性和风险+每个阶段目标不同+用户可以很早看见软件+可以经过用户多次评价修改运行等，不断的得到双方认可

优点：适合模糊开发+可以做培训环境，用户培训和开发同时参与+给予用户改变设想的机会+降低风险，使开发柔性更大+使系统更容易维护+对用户更友好+开发费用减少，时间缩短。

缺点：对不熟悉专业的开发者容易把次要当成主要+每次用户的修改容易使主要部分越改越少，次要部分越改越大+资源规划和管理比较难，随时更新文档也会带来一定麻烦+为了得到用户满意的原型，忘记考虑用户环境和原型环境。

适用环境：实验工具，作为解决不确定性的工具，辅助分析用户需求的工具，充作同步培训工具（主要是作为辅助性的工具）。

不适用：嵌入式软件，实时控制软件，科技数值计算软件。

螺旋模型：1）风险驱动的，有利于减少风险；2）维护和开发没有本质区别，维护是模型的另一个周期；3）减少测试不足带来的风险；4）用户和开发人员密切合作；5）用户可以更早看到产品。可以不断对产品进行评估。

缺点：1）过程复杂，风险分析有时成本过高；2）过多依赖用户，项目是否成功很大程度上取决用户的责任心和水平。

适用项目：内部的大规模软件的开发，不太适合合同软件。一般只适用于大规模软件的开发。

增量模型：（渐增模型）是一种非整体开发的模型。是一种进化式的开发过程。

基本思想

* + 假设软件需求可以分段获取和实现，把软件产品分解为一系列增量来分步开发，分批地逐步向用户提交产品。首先构造系统的核心功能，然后逐步增加功能和完善性能。

模型优点

* + 逐步交付软件产品，使用户可以在短时间内看到和使用软件。
  + 使用户有充分时间学习和适应新产品。
  + 减少投资风险。

模型缺点

* + 增量划分困难；策略是先进行总体分析和设计，以减少风险。
  + 增量集成要求软件体系结构必须是开放式的。

适合项目类型

* + 大部分需求明确，但需求可能发生变更。
  + 庞大或功能复杂的项目。
  + 对于市场或用户把握不准，需要逐步了解的项目。

喷泉模型：喷泉模型认为软件生命周期的各个阶段是相互重叠和多次反复的。主要用于面向对象方法中。

其特点如下：

１**.** 开发过程有分析、系统设计、软件设计和实现**4**个阶段。

２.各阶段相互重叠，它反映了软件过程并行性的特点。

３.以分析为基础，资源消耗成塔型。

４.反映了软件过程迭代性的自然特性，从高层返回低层无资源消耗。

５.强调增量开发，整个过程是一个迭代的逐步提炼的过程。

Ch02

**系统工程的任务**:1)识别用户的要求

2):系统建模和模拟

3):成本估算和进度安排

4):可行性分析

5):生成系统规格说明

**可行性研究的目的**:用最小的代价在最短的时间内确定该项目是否值得去解决，是否存在可行的解决方案。

**可行性研究的内容**:技术上可行、经济上可行、操作上可行、社会上可行

**可行性研究的任务**:技术可行性研究、经济可行性研究、操作可行性研究、社会政策上允许的可行性研究、开放方案的选择

**可行性研究的两种描述手段**: SFD(系统流程图)、DFD(数据流图)

**软件成本的估算方法**: 1)基于已完成的类似项目进行估算

2):基于分解技术进行估算

3):基于经验估算模型的估算: IBM(静态单变量)、Putnam (动态多变量)、CoCoMo模型(层次模型 基本的CoCoMo模型是静态单变量的)

**效益度量的指标**:1)货币的时间价值

2):投资回收期

3):纯收入

4):投资回收率

## Ch03

**Q1:简述软件需求内容？**

A1: 系统需求: 1)功能需求、2):非功能需求、3):领域需求

**Q2:简述需求工程的活动**

A2: 分为需求开发和需求管理。

需求开发:1)问题获取、2)分析、3)编写规格说明、4)验证

需求管理:1)变更控制、2)版本控制、3)需求跟踪、4)需求状态跟踪

**Q3:常用的需求分析方法**

1. 功能分解法
2. 面向数据流的结构化分析方法（SA）
3. 面向数据结构的分析方法
4. 信息建模法
5. 面向对象的分析方法（OOA）

**Q3: 需求分析的操作原则**

1. 必须能够表示和理解问题的信息域
2. 必须能够定义软件将完成的功能
3. 必须能够表示软件的行为
4. 必须划分描述数据、功能和行为的模型
5. 分析过程应该从要素信息移向细节信息

**Q4:需求工程的指导性原则**

1. 在开始建立分析模型前，先充分理解问题
2. 开发原型，使得用户能够了解如何进行人机交互
3. 记录每个需求的起源及原因
4. 使用多个需求视图
5. 给需求赋予优先级
6. 努力删除歧义性

## Ch04

**软件设计分为总体设计和详细设计**

**常用的设计方法有：SD法，Jackson法、OOD法、HIPO法、Parnas法、Warnier法**

**软件设计的任务:**：以需求说明书为依据，完成系统结构、数据结构、程序结构等设计，最后得到软件设计说明书。

**软件设计的目标：**构造一个高内聚低耦合的软件模型，提高可维护性，提高可理解性，提高效率。

**软件设计准则：**

1）抽象化与逐步求精

2）模块化准则

3）信息隐蔽准则

4）模块独立性准则

**耦合、内聚与独立性的关系：**

1）内聚与耦合密切相关，强耦合的模块意味着弱内聚，强内聚的模块意味着低耦合。

2）耦合与内聚都是模块独立性的定性标准，都反映模块独立性的良好程度。但耦合是直接的主导因素，内聚则辅助耦合共同对模块独立性进行衡量。

**部件级设计完成的工作：**

1. 为每个部件确定采用的算法，选择某种适当的工具表达散发的过程，编写部件的详细过程性描述
2. 确定每一部件内部使用的数据结构
3. 在部件级设计结束时，应该把上述结果写入部件级设计说明书，并且通过复审形成正式文档，作为下一阶段的工作依据。

**部件级设计常用的3种描述方式：**

1. 图形描述：程序流程图、结构化流程图、问题分析图PAD
2. 语言描述：PDL
3. 表格描述

**常见的软件体系结构：**

单主机结构、C/S结构、B/S结构

## Ch05

1. **数据流的流向：**1）从源流向加工 2）从加工流向宿 3）从加工流向文件 4）从文件流向加工 5）从加工流向加工
2. **数据字典里包括的条目：**1)数据流 2）文件 3）数据项 4）加工 5）源或宿
3. **简述结构化设计的基本步骤：**
4. 建立初始结构图
   * 1. 将整个软件看作一个大的功能模块，通过功能分解不断将其分解成若干个较小的功能模块，直至得到一组不必再分解的模块(结构图中的底层模块)
     2. 将SA所得到的DFD映射成初始的结构图(SC)
5. 对结构图进行改进

可根据设计准则和启发式设计策略对初始结构图进行改进

1. 书写设计文档

书写设计规格说明，特别要为每个模块书写模块的功能、接口、约束和限制等

1. 设计评审
2. **简述数据流图映射到结构图的基本步骤：**

1）复审和精化数据流图

2）确定数据流图的类型(变换型、事务型)

3）将DFD映射成初始结构图：采用变换分析或事务分析技术，将DFD映射成初始结图

4）改进初始结构图

## CH07&&Ch08

1. **面向对象 = 对象+分类+继承+通过消息的传递**

**2、面向对象的分析过程：**

1）获得客户对系统的需求：包括标识场景和用况，以及建造需求模型

2）用基本的需求为指南，来选择类和对象

3）定义类的结构层次

4）建造对象-关系模型

5）建造对象-行为模型

6）利用用况/场景来复审分析模型

**3、面向对象的设计过程：**

1）系统设计

2）对象设计

3）消息设计

4）复审

**4、典型的面向对象的方法：**1）Coad & Yourdon 方法 2）OMT方法（James Rumbaugh 创立的 Object Model Technology） 3）Booch方法 4）OOSE方法(Jacobson创立的)

1. **UML = Unified Modeling Language**
2. **创建用况模型的一般步骤：**1）定义系统 2）确定执行者 3）确定用况（用况名称，参与的执行者，前置条件，事件流，后置条件或加特殊需求） 4）描述用况 5）定义用况间的关系 6）确认模型

**7、用况之间的关系有哪四种？** 关联（直线）、包含（虚线实头《extend》）、扩展（虚线实头《include》）、泛化（实线空白头）

**8、类之间的关系有哪四种？** 关联、依赖、泛化、实现

确定类+描述模板+确定关联

**9、什么是动态建模？**

动态建模用来描述系统的动态行为，显示对象在系统运行期间不同时刻的动态交互。

**10、UML中用哪些图来建立动态模型？**

状态机图、活动图、顺序图、通信图

**11、什么是状态机图？**

状态机图是对类描述的补充，它说明该类的对象所有可能的状态以及哪些事件将导致状态的改变。

**12、简述顺序图与通信图的异同**

1）顺序图和通信图都属于交互图，用来描述对象之间的动态关系

2）顺序图强调消息的时间顺序，通信图强调参与交互的对象的组织关系

3）顺序图和通信图在语义上是等价的，两者可以相互转换

## CH11

1. **人机界面设计中考虑人的因素：** 1）人对感知过程的认识；2）用户的技能和行为方式；3）人体测量学对设计的影响
2. **人机界面风格：** 1）语言界面；2）图形用户界面；3）直接操纵用户界面；4）多媒体用户界面；5）多通道用户界面。
3. **简述人机界面设计过程：** 1）用户、任务和环境分析及建模；2）界面设计；3）界面构造；4）界面确认
4. **人机界面设计中设计的模型：** 1）软件工程师创建的设计模型；2）人机工程师创建的用户模型；3）终端用户在脑海里对界面产生的映像，称为用户的模型或系统感觉；4）系统实现者创建的系统映像（注： 当系统映像和系统感觉是一致的，用户就会很舒服）
5. **人机界面设计中任务分析的途径与方法：** 采用逐步精华的方法和面向对象的方法，剖析原有应用系统，分析系统需求规格说明。
6. **界面设计活动：**

1）定义界面对象和动作 分析用户场景，屏幕布局，制定一份指导工作的文档

2）设计问题：系统响应时间+用户求助设施+错误信息处理+命令标记

3）**黄金原则**：让用户拥有控制权+减少用户的记忆负担+保持界面一致

**7、**有效的设计评估应包括**专家评审**和**可用性测试**两方面（可用性测试的缺点：强调首次使用+测试时间不能太长，所以只能设计部分界面，很难确定长时间使用后的）

### CH12

1. **程序设计语言涉及到的定义**：语法+语义+语用
2. **程序设计语言的基本成分：**数据成分+运算成分+控制成分+传输成分
3. **程序设计语言的三种特性：**心理特性+应用特性+工程特性
4. **程序语言的选择：**优先选择高级语言；必要时使用低级语言（三个原因：运行时间限制+储存空间限制+不提供高级语言支持）

注：可能会考选择题让自己选一个

多种语言适用的时候优先熟悉的语言

COBOL适用于商业领域、FORTRAN适用于工程和科学计算领域

Prolog、Lisp适用于人工智能领域 Smalltalk,C++适用于OO系统开发

C适用于多个应用领域

1. **编程风格：**源程序中的内部文档、数据说明、语句构造、输入/输出
2. **什么是程序设计风格? 为什么应使程序具有良好的风格？为了具有良好的设计风格，应注意哪些方面的问题?**

程序具有良好的风格可以在很大程度上提高程序的可读性，改善程序的质量。要形成良好的程序设计风格，应从源程序文档化、数据说明、语句构造、输入输出和追求效率几个方面加以注意。

## CH13

1. **软件测试的目的：**为了发现和改正错误。注：不可能发现所有的错误 也不能保证软件是完全正确的 （选择 or 判断）
2. **软件测试的原则：**1）尽量不由程序设计者进行测试

2）关键是注重测试用例的选择

3）充分注意测试中的群集现象

**3、** **软件测试的方法：静态分析法**和**动态测试法**

静态分析法包括：桌前检查**（程序员个人）**+代码会审**（小组检查常见错误讨论检查）**+步行检查**（小组运行程序检查）**（最常用的静态分析法）

动态测试法：白盒法+黑盒法

**4、什么是白盒法？常用的白盒测试方法有哪几种？**

由于需要分析了解程序的内部结构，好像一个透明的盒子，因此称为白盒法。

白盒测试法：逻辑覆盖测试+基本路径覆盖测试+数据流测试+循环测试

**5、白盒法测试的逻辑覆盖标准有哪些？**

1）语句覆盖：选择足够的测试用例，使得程序中每个语句至少都能被执行一次。

2） 判定覆盖： 执行足够的测试用例，使得程序中每个判定至少都获得一次“真”值和“假”值。

3）条件覆盖：执行足够的测试用例，使得判定中的每个条件获得各种可能的结果。

4）判定/条件覆盖：执行足够的测试用例，使得判定中每个条件取到各种可能的值，并使每个判定取到各种可能的结果。

5）条件组合覆盖： 执行足够的例子，使得每个判定中条件的各种可能组合都至少出现一次6）路径覆盖：指选择足够的测试用例，使得运行这些测试用例时，被测程序的每条可能执行到的路径都至少经过一次（如果程序中包含环路，则要求每条环路至少经过一次）

**6、什么是黑盒法？常用的黑盒测试方法有哪几种**

不考虑程序的内部结构，只根据程序功能或程序外部特性设计测试样例，像一个不透明的盒子，因此称为黑盒法。

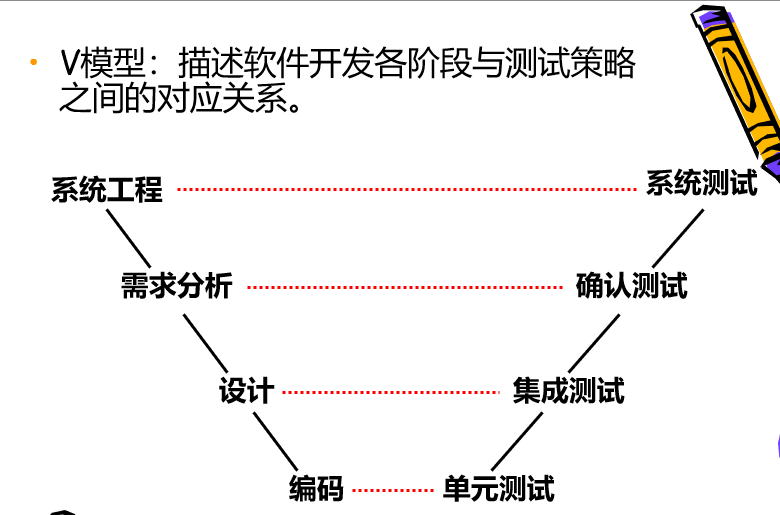
黑盒测试法：等价类划分+边界值分析+比较测试+错误猜测+因果图

**7、简述V模型中的测试策略有哪些？**

包括单元测试、集成测试、确认测试、系统测试。

1. 单元测试（常用白盒法）：针对程序中的模块或构件，主要揭露编码阶段产生的错误。
2. 集成测试：针对集成的软件系统，主要揭露设计阶段产生的错误。
3. 确认测试：是根据软件需求规约对集成的软件进行确认，主要揭露不符合需求规约的错误。
4. 系统测试：对于基于计算机系统中的软件，还需将它集成到基于计算机系统中，并进行系统测试，以揭露不符合系统工程中对软件要求的错误。

**8、画出表示软件开发各阶段与测试策略之间的对应关系的V模型**



**9、集成测试包括哪两种方式?各有什么优缺点？**

集成测试包括**非增量式集成测试**和**增量式集成测试**。通常采用增量式集成测试。

1）非增量式集成是将所有的模块一次连接起来，简单、易行，节省机时，但测试过程中难于查错，发现错误也很难定位，测试效率低。

2）增量式集成是将模块一个一个地连入系统，每连入一个模块，都要对新系统进行测试。这种组装测试方案比较非增量式，容易查出错误及进行错误定位，有利于查出模块接口部分的错误，因此测试效率高。因此通常采用增量式。

**10、增量式集成测试包括哪些方式？各有什么优缺点？**

分为**自顶向下集成**和**自底向上集成**两种。

1. 自顶向下：

优点：能够尽早发现系统主控方面的问题。

缺点：无法验证桩模块是否完全模拟了下属模块的功能。无法尽早查出底层容易出错的复杂模块中的错误，所以导致过多的回归测试。

1. 自底向上：

优点：驱动模块较容易编写桩模块，能够尽早查出底层涉及较复杂的算法和实际的I/O模块中的错误。

缺点：只有当系统所有模块全部组装完成，才能看到系统实体，才能测试系统的主控功能。

**11、三明治测试:**将自顶向下测试和自底向上测试综合。在程序结构的高层使用自顶向下策略，而低层使用自底向上策略。

**12、系统测试分为哪些？**

包括恢复测试、安全测试、压力测试、性能测试。

**13、面向对象的测试策略**

类内测试：把类作为面向对象软件的单元。传统的单元测试等价于面向对象的类内测试。包括类内的方法测试和类的行为测试。

类间测试：相当于集成测试。两种集成策略：基于线程的测试和基于使用的测试。

**14、测试完成的标准：**使用指定的测试用例设计方法产生测试样例，运行这些测试用例均为发现错误（包括发现错误后已被纠正的情况），则测试可终止。

**15、调试的目的是什么？常用的调试方法有哪些？**

调试的目的是确定错误的原因和准确位置，并加以纠正。

调试的方法：蛮力法、回溯法、原因分析法（分为归纳法和演绎法）

## CH15

2个错误认识：维护是新的开发+维护就是改错（判断？）

**1、软件维护分哪几种类型？其中哪种类型的维护最重要？**

1. 纠错性维护
2. 适应性维护
3. **改善性维护（最常见）**
4. 预防性维护

影响维护工作量的六个因素：

系统规模+程序设计语言+系统年龄+数据库技术的应用+先进的软件开发技术+其他因素

维护活动包括：建立维护组织+确定维护过程+保管维护记录+进行维护评价

1. **什么是软件的可维护性？影响可维护性的因素主要有哪些？**

软件的可维护性是指理解、改正、调整和改进软件的难易程度。

影响可维护性的因素主要有:

1. 可理解性（understandability）、
2. 可测试性（testability）、
3. 可修改性（modifiability）
4. 可移植性（portability）

**3、提高软件可维护性通常采用的方法有哪些？**

通常采用的方法有：

1. 确定质量管理目标和优先级
2. 使用提高软件质量的技术与工具
3. 选择可维护性高的程序设计语言
4. 改进程序文档
5. 进行质量保证审查

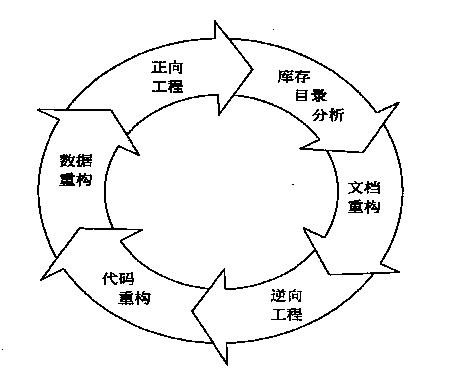
**4、再工程包括业务过程再工程和软件再工程。**

**再工程概念：在逆向原工程的基础上修改或者重构已有系统，产生系统新版本**

**目的：为遗留系统转化为可演化系统提供可行途径**

逆向工程可用分为：对用户界面的逆向工程+对数据的+对理解的

**5、画出软件再工程过程模型**



软件再工程”，是提高软件可维护性的一类重要的软件工程活动