**软件工程题库考试复习题**

**主要题型和重点**

一、选择题 20分

二、名词解释 5题\*3分=15分 题库共100道

三、简答题 4题\*5分=20分

四、论述 1题\*15分=15分

五、应用 30分

软件测试：

1.、白盒测试：基本路径（画流程图、关键路径、测试用例）；

2、决策表、因果图（表述方式、列出条件、设计用例）

建模：

1、传统结构：数据流程图（怎么画）、数据制表等；

2、UML、面向对象简答应用（use case、类图、状态图、交互图：了解）表述、目的、数据字典

**软件工程与方法复习**

# 一、选择题 （28题）

1、软件生存期模型是从软件项目需求定义开始到软件被废弃使用为止，跨越整个生存期的系统开发、运行和维护所实施的全部过程、活动和任务的结构框架。到目前为止，存在的软件生存期模型有： 演化模型 ， 螺旋模型 ， 智能模型 ， 喷泉模型 ， 瀑布模型 等。

瀑布模型 螺旋模型 快速原型模型 增量模型 喷泉模型 统一过程

2、软件需求分析方法包括原型化方法和结构分析方法。软件原型化方法是在研究分析阶段的方法和技术中产生的，但是也可用以面向软件开发的其他阶段。由于软件项目的特点和运行原形的目的的不同，原型主要有三种不同的**作用类型**： 探索型 ， 实验型 ， 进化型 。

探索型 的目的是要弄清目标系统的需求，确定所希望的特性，研究多种方案的可行性。它主要针对开发目标模糊，用户和开发者对项目都缺乏经验的情况。

实验型 的目的用于大规模开发和实现之前，考核方案是否合适，规格说明书是否可靠。

进化型 的目的不在于改进规格说明，而是将系统建造的易于变化，在改进原型的过程中，逐步将原型变成最终系统。它将原型方法的思想扩展到软件开发的全过程，适合于满足需求的变动。

1. 由于运用原型的目的和方式不同，在使用原型时可采用以下两种**不同的策略**：

（1）、废弃策略：先构造一个功能简单而且质量要求不高的模型系统，针对这个模型系统反复进行分析修改，形成比较好的设计思想，据此设计出完整、准确、一致、可靠的最终系统，系统构造完成后，原来的模型被废弃不用。它对应于 探索型 和 实验型 。

（2）、 追加策略：先构造一个功能简单而且质量要求不高的模型系统作为最终系统的核心，然后不断扩充修改，逐步追加新的要求，最后成为最终的系统。它对应于 进化型 。

3、在软件工程的设计阶段中，有三种常用的设计方法：结构化设计方法 SD 、Jackson方法和Parnas方法。SD方法侧重于 用数据流图表示系统的分解，且用数据字典和说明分别表示数据和加工的含义 ；Jackson方法侧重于 由数据结构导出模块结构 ；Parnas方法的主要思想 将可能引起变化的因素隐藏在某有关模块内部，是这些因素变化时的影响范围受到限制 。从七十年代中期到九十年代早期， SD 是最为常用的方法。 Parnas 方法只提供了重要的设计准则，没有规定出具体的工作步骤。

4、程序的正确性证明是指 用逻辑方法证明程序终止且满足给定的程序规范 ，常用的正确性证明方法之一为 Hoare 方法。在程序中插入的逻辑公式称为 Hoare公式 。当 Hoare公式 出现在循环中，并且它具有如下性质：只要第一次进入选换时 Hoare公式 为真，那么第N次进入循环时 Hoare公式 仍为真。此时，这样的 Hoare公式 就称为 断言 。如果公式 E 在执行程序段“Z：=Z\*A；Y：=Y-1”之前为真，则执行该程序段之后 E 仍保持为真。

A： 给它一组数据，检验其执行结果是否正确

用逻辑方法证明程序终止且满足给定的程序规范

逐条语句检查程序是否有错

证明程序终止

证明程序满足给定的程序规范

B： SD Jackson Hoare Parnas

C，D： Horn子句 循环不变式 断言 验证条件

Hoare公式

E： A\*Z = Y A\*Y = Z Z=AY AB = Z\* AY

Y >0 ∧ Z= AY

5、软件语言主要包括需求定义语言、功能性语言、设计性语言、程序设计性语言和文档语言等。功能性语言用来书写软件的功能规约，如 Z 语言。函数式程序设计语言和逻辑式程序设计语言都属于 **申述式** 语言。逻辑式语言的基础是 **谓词逻辑** ，如PROLOG是逻辑式程序设计语言，它的基本运算单位是 Horn子句 。函数式语言是一种面向 值 的语言。

6、软件方法学是以软件方法为研究对象的学科。

从开发风范上看，可分为 自顶向下的开发方法与自底向上的开发方法 。

从性质上看，可分为 形式方法与非形式方法 。

从适应范围来看，可分为 整体性方法与局部性方法 。

形式方法的目的是把软件作为数学来重新发现。形式方法被用来避免系统中的 歧义性、不完全性 、不一致性。

软件自动化方法是指利用计算机使软件的设计实现自动化的方法和相关的技术。软件自动化的实现途径有四种：过程途径、归纳途径、 演绎途径、**转换途径** 。

7、开发软件时，对于提高软件开发人员效率至关重要的是 程序开发环境 。软件工程中描述软件生存周期的瀑布模型一般包括计划、 需求分析 、设计、编码、 测试 、维护等几个阶段。其中，设计阶段在管理上有可以分成 概要设计 和 详细设计 两个步骤。

8、软件需求分析的任务不应包括 **结构化程序设计** 。进行需求分析可使用多种工具，但 **PAD图** 是不适用的。在需求分析中，分析员要从用户那里解决的最重要的问题是 要让软件做什么 。需求规格说明书的内容不应当包括 对算法的详细过程的描述 。该文档在软件开发中具有重要的作用，但其作用不应当包括 软件可行性分析的依据 。

9、**块间联系和块内联系是评价程序结构质量的重要标准**。联系的方式、共用信息的作用、共用信息的数量和接口的 简单性 等因素决定了块间联系的大小。在块内联系中， 功能内聚 的块内联系最强。SD方法的总原则是使每个模块执行 一个 功能，模块间传递 数据型 参数，模块通过 标准引用 语句调用其他模块。

10、1960年底Dijkstra提倡的 结构化程序设计 是一种有效的提高程序设计效率的方法。Dijkstra为了使程序结构易于理解，把基本控制结构限于顺序、选择 、 重复 3种，应避免使用 **GOTO语句**  。 结构化程序设计 不仅提高程序设计的生产率，同时也容易进行程序的 维护 。

11、程序的3种基本控制是 顺序，条件，循环 。它们的共同点是 只有一个入口和一个出口 。结构化程序设计的一种基本方法是 逐步求精法 。软件测试的目的是 发现程序中的错误 。软件调试的目的是 找出错误所在并改正之 。

12、在面向对象软件开发过程中特别重视复用。软件构件应独立于当初开发它们的应用而存在。在以后的应用开发中，可以调整这些独立构件以适应新问题的需要。因此，应使得类成为一个 可复用 的单元。这样就有一个 类 生存期问题。 类 生存期有自己的步骤，与任何特定应用的开发 无关 。按照这些步骤，可以完整地描述一个基本 实体 ，而不仅仅考虑当前正在开发的系统。系统开发的各个阶段都可能会标识新的类。随着各个新类的标识， 类 生存期引导开发工作逐个阶段循序渐进。在设计和实现类时，应尽可能利用既存类提供为当前应用所需要的功能，利用既存类的3个可能途径是： 演化 复用既存类，对既存类进行修改以得到满足要求的类，重新开始进行开发。

13、测试大型软件通常由 单元测试 、集成测试、确认 Validation 测试组成。确认测试主要寻找与软件 需求规格 说明不一致的错误。语句覆盖、判定覆盖、条件覆盖和路径覆盖都是白盒测试法设计测试用例的覆盖准则，在这些覆盖准则中最弱的准则是 语句覆盖 ，最强的准则是 路径覆盖 。此外，还有多种黑盒测试的设计测试用例方法，如 因果图 。

14、如何评价软件的质量一直是软件技术人员所关心的问题，目前已有多种软件质量模型来描述软件的质量特性。ISO/IEC 9126是国际标准化组织在1991年提出的软件质量标准。它由三个层次组成，第一层是质量特性，第二层是质量子特性，第三层是度量指标。六个质量特性是：功能性、可靠性、易使用性、效率、可维护性和 可移植性 。

其中功能性包括质量子特性 完备性、一致性、可追踪性 ；

可靠性包括质量子特性 一致性 ；

易使用性包括质量子特性 操作性、培训性、产品文档完备性 ；

可维护性包括质量子特性 一致性、可见性、产品文档完备性、清晰性、模块性、自描述性、简单性、结构性 。

15、软件复用是使用已有的软件产品 如设计、代码、文档等 来开发新的软件系统的过程。软件复用的形式大体可分为垂直式复用和水平式复用。垂直式复用是指 同一应用领域 中的复用，水平式复用是指 不同应用领域 中的复用。

为了提高构件 Component 的复用率，通常要求构件具有较好的 通用性和可变性 。为了将不同软件生产商在不同软硬件平台上开发的构件组装成一个系统，必须解决异构平台的各构件间的互操作问题。

目前国际上已出现了一些支持互操作的构件标准，典型的有国际对象管理组织OMG推荐的 **CORBA** 和Microsoft公司推出的 **DCOM**  。

16、软件测试通常可分为单元测试、集成测试、确认测试和系统测试，其中确认测试主要用于发现 需求分析 阶段的错误。在集成测试时，通常可采用自顶向下增殖式集成和自底向上增殖式集成。

在自底向上增殖式集成时，对每个被集成的模块 要设计驱动模块，但不必设计桩模块 。

对那些为众多用户开发的软件，如操作系统，编译程序，通常还要进行α测试和β测试，以发现可能只有最终用户才能发现的错误，其中，α测试是指最终用户在 开发环境下，开发人员在场 的情况下所进行的测试，β测试是指最终用户在 用户的实际使用环境下，开发人员不在场 的情况下所进行的测试。

在软件维护阶段，当修改软件后，除了进行常规的测试外，还应进行 回归 测试。

17、类常常被看作是一个抽象数据类型的实现，更合适的是把类看作是 概念 的一个模型。事实上，类是单个 语义 单元。类的用户能够操纵的操作叫做类的 界面 。类定义的其余部分给出数据定义和辅助功能定义，包括类的实现。

类的实现常常包括了其它类的实例，这些实例 应受保护不 被其它对象存取，包括同一个类的其它实例。类的实现可能还包括某些私有方法，实现它们的类可以使用，而其它任何对象都不能使用。

类，就它是一个数据值的聚合的意义来看，与Pascal中的记录或C中的结构类似，但又有区别。类扩展了通常的记录语义，可提供各种级别的 可访问性 。

18、由rumbaugh等人提出一种面向对象方法叫做对象模型化技术  **OMT**  ，既三视点技术，它要求把分析时收集的信息建立在3个模型中。

第一个模型是 对象模型 ，它的作用是描述系统的静态结构，包括构成系统的对象和类、它们的属性和操作，以及它们之间的联系。

第二个模型是 动态模型 ，它描述系统的控制逻辑，主要涉及系统中各个对象和类的时序及变化状况。 动态模型 包括两种图，即 状态迁移图 和 事件追踪图 。 状态迁移图 描述每一类对象的动态行为， 事件追踪图 描述发生于系统执行过程中的某一个特定场景。

第三个模型是 功能模型 ，它着重于描述系统内部数据的传送与处理，它由多个数据流图组成。

19、软件维护是软件生存期的最后阶段。软件工程学针对维护工作的主要目的是提高 软件的可维护性 ，降低 维护的代价 。软件的 可测试性 、 可理解性 、 可修改性 是决定软件可维护性的基本因素。

20、软件的再工程是软件的一类工程活动，它能够使人们： I **增加对软件的了解**； II 准备或直接提高软件自身的 可维护性 、 可复用性 或演化性。第II部分旨在改善软件的 静态质量 ，使得软件更容易为人们服务。纯粹是处于改善性能的代码优化 不属于 软件再工程。逆向工程属于上述软件再工程的第 I 部分。

21、从供选择的答案中选出同下列关于软件测试的各条叙述关系最密切的字句，分别填入 A 、 B 、 C 、 D 、 E 中。

代码审查 对可靠性要求很高的软件，例如操作系统，由第三者对源代码进行逐行检查。

退化测试 已有的软件改版时，由于受到变更的影响，改版强正常的功能可能发生异常，性能也可能下降。因此，对变更的软件进行测试是必要的。

白盒测试 在意识到被测试的模块的内部结构或算法的情况下进行测试。

原型 为了确认用户的需求，先做出系统的主要部分，提交给用户使用。

桩 在测试具有层次结构的大型软件时，有一种方法是从上层模块开始，由上到下进行测试。此时，有必要用一些模块替代尚未测试的下层模块。

22、在面向对象软件设计过程中，应按如下要求进行类的设计：只有类的共有界面的成员才能成为使用类的操作，这就是软件设计的 信息隐蔽 准则。

当且仅当一个操作对类的实例的用户有用时它才是类公共界面的成员，这是软件设计的 最小界面 准则。

由同属一个类的操作负担存取或加工类的数据，这是软件设计的 高内聚 准则。

两个类之间的交互应当仅设计参数表，这是软件设计的 显式信息传递 准则。

每个派生类应当做基类的特殊化来开发，而基类所具有的公共截面成为派生类的共有界面的一个子集，这是软件设计的 继承性 准则。

23、集成测试也叫做 组装测试 或 联合测试 。通常，在 单元测试 的基础上，将所有模块按照设计要求组装成为系统。子系统的集成测试特别成为 部件测试 ，它所做的工作是要找出子系统和系统需求规格说明之间的 不一致 。

24、软件可维护性是纠正软件系统出现的错误和缺陷，以及为满足新的要求进行修改、 扩充 的容易程度。目前广泛使用7个特性来衡量软件的可维护性，其中就有 可靠性 、 可理解性 、 可使用性 。其中， 可靠性 和 可理解性 主要在改正性维护中侧重应用， 可使用性 主要在适应性维护和 完善性 维护中侧重应用。

25、软件的再工程技术主要有 改进软件 、 获取、保存及扩充软件知识 和 理解软件 。分析、度量属于 理解软件 ，知识库和变换属于 获取、保存及扩充软件知识 。复用工程属于 改进软件 。

26、所谓一个构件，可以 一个类型、类或其他的工作成品。对于构件，应当按可复用的要求 设计 、 实现 、打包、编写文档。构件是 内聚的 ，并具有相当稳定的公开的 接口 。这里所说的构件是基于 面向对象 技术的。

27、软件项目进度管理有许多方法，但 **IPO**  不是常用的进度控制图示方法。在几种进度控制图示方法中， 甘特图 难以表达多个子任务间的逻辑关系，使用 PERT 不仅能表达子任务之间的逻辑关系，而且可以找出关键子任务。在 PERT 中，用带箭头的边表示 任务 ，用圆圈结点表示 事件 。

28、在考虑各种软件开发资源时， 人力 是最重要的资源。如果把软件开发所需要的资源画成一个金字塔形：在塔的上层是最基本的资源 人力 ，在底部为 工具 。 工具 包括硬件资源和软件资源。 **宿主机 、 目标机** 和其他硬件设备属于硬件资源。IPSE工具属于软件资源中的 **框架工具** 。

# 二、名词解释（100题）

## 2.软件

软件是计算机系统中与硬件相互依存的部分，它是包括程序、数据及相关文档的完整集合

## 7.软件工程

软件工程是研究和应用如何以系统化的、规范的、可度量的方法去开发、运行和维护软件，即把工程化方法应用到软件上。

## 17.软件生存周期

软件生存周期是指软件产品从考虑其概念开始到该软件产品交付使用，直至最终退役为止的整个过程，一般包括**计划、分析、设计、实现、测试、集成、交付、维护**等阶段。

## 22.软件过程

软件过程是人们用于开发和维护软件及其相关过程的一系列活动，包括软件工程活动和软件管理活动。

## 27.软件基本过程

软件过程（Software Process）是指一套关于项目的阶段、状态、方法、技术和开发、维护软件的人员以及相关Artifacts（计划、文档、模型、编码、测试、手册等）组成。软件过程可概括为三类：**基本过程类、支持过程类和组织过程类**。基本过程类包括获取过程、供应过程、开发过程、运作过程、维护过程和管理过程。

## 32.软件支持过程

支持过程类包括文档过程、配置管理过程、质量保证过程、验证过程、确认过程、联合评审过程、审计过程以及问题解决过程。

## 37.软件组织过程

组织过程类包括**基础设施过程、改进过程以及培训过程**。

## 97.过程规范（未找到答案）

对软件从立项、设计、开发、实施、维护各阶段中的每一个过程根据一定的标准进行规范。

## 38.软件度量

是对软件开发项目、过程及其产品**进行数据定义**、收集以及分析的持续性定量化过程，目的在于对此加以理解、预测、评估、控制和改善。

## 68.软件质量

与软件产品满足规定的和隐含的需求的能力有关的特性或特性的全体。或“所有描述计算机软件优秀程度的特性的组合。”

## 12.软件危机

软件危机是指在计算机软件的开发和维护过程中所遇到的一系列严重问题

## 83.软件可靠性

（1）在规定的条件下，在规定的时间内，软件不引起系统失效的概率。

（2）在规定的时间周期内，在所述条件下程序执行所要求的功能的能力。

## 47.软件能力成熟度模型

它是对于软件组织在定义、实施、度量、控制和改善其软件过程的实践中各个发展阶段的描述。CMM的核心是把软件开发**视为一个过程**，并根据这一原则对软件开发和维护进行过程监控和研究，以使其更加科学化、标准化、使企业能够更好地实现商业目标。

## 93.软件质量模型（未找到答案）

随着计算机技术及其应用的日益普及,各行各业对软件的依赖性越来越大;同时,由于软件质量问题导致的不良后果也越来越多。因此,软件质量的重要性日益突出。如何保证软件质量已经成为软件开发过程中必须贯穿始终加以考虑的重要问题。然而,质量是一个复杂且难以琢磨的概念。对于不同的人或不同的应用系统,质量的含义和要求不尽相同。并且,软件质量属性是多方面的。一个软件的质量往往涉及到许多不同的质量属性,不同类型的软件所关注的质量属性也不尽相同。因此,为了更好地理解、预测和评价软件和信息系统的质量,人们建立了各种质量模型来衡量软件的质量。

**基于软件的特性和大家所关心的质量问题，形成一套大家都能接受衡量软件质量高低的一种模型。**

## 3.项目

项目是一件事情、一项独一无二的任务，也可以理解为是在一定的时间和一定的预算内所要达到的预期目的。项目是指在一定的约束条件下（主要是限定时间、限定资源），具有明确目标的一次性任务。项目是一系列具有特定目标,有明确开始和终止日期，资金有限，消耗资源的活动和任务。

## 33.项目整体管理

也叫集成管理，保证项目各个要素相互协调所需要的管理过程，它需要在相互影响的项目目标和方案中作出平衡，以满足或超出项目干系人的需求和愿望

## 28.项目范围管理

项目范围管理是指对项目包括什么与不包括什么进行定义并控制的过程。这个过程用于确保项目组和项目干系人对做为项目结果的项目产品以及生产这些产品所用到的过程有一个共同的理解。

## 8.配置管理

配置管理(Configuration Management，CM)是通过技术或行政手段对软件产品及其开发过程和生命周期进行控制、规范的一系列措施。配置管理的目标是记录软件产品的演化过程，确保软件开发者在软件生命周期中各个阶段都能得到精确的产品配置。

## 58.项目计划评审技术

计划评审技术就是工程项目当作一种系统，用网络图或者表格或者矩阵来表示各项具体工作的先后顺序和相互关系，以时间为中心，找出从开工到完工所需要时间的最长路线，并围绕关键路线对对系统进行统筹规划，合理安排以及对各项工作的完成进度进行严密的控制，以达到用最少的时间和资源消耗来完成系统预定目标的一种计划与控制方法。

## 23.里程碑

里程碑一般是项目中完成阶段性工作的标志，标志着上一个阶段结束、下一个阶段开始，将一个过程性的任务用一个结论性的标志来描述，明确任务的起止点。一系列的起止点就构成了引导整个项目进展的里程碑。里程碑定义了当前阶段完成的标准和下一新阶段启动的条件和前提。

## 96.软件开发过程模型

软件过程模型 是指软件开发全部过程、活动和任务的结构框架。

**软件过程是为了获得高质量的软件产品所需要完成的一系列任务的框架，它规定了完成各项任务的工作步骤。**通常使用生命周期模型简洁地描述软件过程。生命周期模型规定了把生命周期划分成哪些阶段及各个阶段的执行顺序，因此，也称为过程模型。

常用的过程模型有：

瀑布模型（需求完全确定）

渐进式开发（逐步迭代）模型：原型模型、增量模型、螺旋模型

**瀑布模型的特点是**：

每个阶段都必须完成规定的文档，没有交出合格的文档就是没有完成该阶段的任务。

每个阶段结束前都要对所完成的文档进行评审，以便迟早发现问题，改正错误。

1）规定了阶段之间自上而下、相互衔接的固定次序，上一阶段的成果作为下一阶段的输入

2）每一个阶段完成之后，必须对阶段性成果进行评审，确认之后才可进入下一阶段。

瀑布模型是由文档驱动的。这个事实也是它的一个缺点，在可运行的软件产品交付给用户之前，用户只能通过文档来了解产品是什么样的。最终开发出的产品可能并不是用户真正需要的。

**快速原型模型：**快速原型模型的第一步是建造一个快速原型，实现客户或未来的用户与系统的交互，用户或客户对原型进行评价，进一步细化待开发软件的需求。通过逐步调整原型使其满足客户的要求，开发人员可以确定客户的真正需求是什么；第二步则在第一步的基础上开发客户满意的软件产品。

**螺旋模型：**它将瀑布模型和快速原型模型结合起来，强调了其他模型所忽视的风险分析，特别适合于大型复杂的系统。螺旋模型沿着螺线旋转，分4个方面的活动：**制定计划、风险分析、实施工程、客户评估。**沿螺旋线自内向外每旋转一圈，便开发出一个更为完善的、新的软件版本。螺旋模型由风险驱动，强调可选方案和约束条件从而支持软件的重用，有助于将软件质量作为特殊目标融入产品开发之中。

**增量模型：**把软件产品作为一系列的增量构件来设计、编码、集成和测试。每个构件由多个相互作用的模块构成，并且能够完成特定的功能。使用增量模型时，第一个增量构件往往实现软件的基本需求，提供最核心的功能。强调每一个**增量均发布一个可操作产品**。

喷泉模型：喷泉模型对软件复用和生存周期中多项开发活动的集成提供了支持，以面向对象的软件开发方法为基础，它适合面向对象的开发方法。它克服了瀑布模型不支持软件重用和多项开发活动集成的局限性。喷泉模型使开发过程具有迭代性和无间隙性。系统某个部分常常重复工作多次，相关功能在每次迭代中随之加入演化的系统。无间隙是指在分析、设计和实现等开发活动之间不存在明显的边界。

## 62/99.过程模型

所谓软件过程模型就是一种开发策略，这种策略针对软件工程的各个阶段提供了一套范形，使工程的进展达到预期的目的。对一个软件的开发无论其大小，我们都需要选择一个合适的软件过程模型，这种选择基于项目和应用的性质、采用的方法、需要的控制，以及要交付的产品的特点。一个错误模型的选择，将迷失我们的开发方向

## 92.极限编程

极限编程是一个轻量级的、灵巧的软件开发方法；同时它也是一个非常严谨和周密的方法。它的基础和价值观是**交流、朴素、反馈和勇气**；即，任何一个软件项目都可以从四个方面入手进行改善：**加强交流；从简单做起；寻求反馈；勇于实事求是**。XP是一种近螺旋式的开发方法，它将复杂的开发过程分解为一个个相对比较简单的小周期；通过积极的交流、反馈以及其它一系列的方法，开发人员和客户可以非常清楚开发进度、变化、待解决的问题和潜在的困难等，并根据实际情况及时地调整开发过程

## 13.配置项

软件配置项是在软件工程中过程中创建的信息，凡是纳入配置管理范畴的工作成果都是配置项；一个纯软件的CIS通常也称为软件配置。

## 18.基线

是一个软件**配置管理**的概念，它能够帮助我们在**不严重阻碍合理变更的条件下控制变更，**IEEE中这样定义基线：**已经通过正式审评和批准的规格说明或产品**，他可以作为进一步开发的基础，并且只有通过正式的变更控制规程才能修改它

## 34.预防性维护

预防性维护是指采用先进的软件工程方法对需要维护的软件或软件中的某一部分重新进行设计、编制和测试，提高软件的可维护性和可靠性等，为以后进一步改进软件打下良好基础。

为了降低设备失效或功能退化的概率，按预定的时间间隔或规定的标准进行的维护。

## 39.适应性维护

随着计算机技术的飞速发展和更新换代，软件系统所需的外部环境或数据环境可能会更新和升级，如操作系统或数据库系统的更换等。为了使软件系统适应这种变化，需要对软件进行相应的修改，这种维护活动称为适应性维护。

## 5.分布图

分布图反映了系统中**软件和硬件的物理架构**，表示系统运行时的处理节点以及节点中组件的配置。

## 10.组件图

组件图描述组件以及它们之间的关系，表示**系统的静态实现视图**。

## 15.活动图

活动图反映系统中从一个活动到另一个活动的流程，**强调对象间的控制流程**。

## 45.用例图

**用例图定义了系统的功能需求**，它完全是从系统的外部观看系统功能，并不描述系统内部对功能的具体实现。

## 40.类图

类图描述系统的**静态结构**，表示系统中的**类以及类与类之间的关系**。

## 30.状态图

状态图表示一个状态机，**强调对象行为的事件顺序**。

## 35.对象图

对象图描述了一组对象以及它们之间的关系，表示**类的对象实例**。

## 20.协作图

协作图表示一组**对象之间的动态协作关系**，反映收发消息的对象的结构组织。

## 25.时序图

时序图表示一组对象之间的动态协作关系，反映对象之间发送消息的时间顺序。

## 84/89.数据流图/上下文数据流图

数据流图：简称DFD，它从数据传递和加工角度，以图形方式来表达系统的逻辑功能、数据在系统内部的**逻辑流向和逻辑变换过程**，是**结构化系统分析方法**的主要表达工具及用于表示软件模型的一种图示方法。

## 94.业务流程图

业务流程图是一种描述系统内各单位、人员之间**业务关系、作业顺序和管理信息流向的图表**，利用它可以帮助分析人员找出业务流程中的不合理流向，它是物理模型。业务流程图主要是描述业务走向。业务流程图描述的是完整的业务流程，**以业务处理过程为中心，一般没有数据的概念**

## 90.静态视图

**是因为它不描述与时间有关的系统行为**，此种行为在其他视图中进行描述。静态视图主要是**由类及类间相互关系构成**，这些相互关系包括：**关联、泛化和各种依赖关系**，如使用和实现关系

## 95.动态视图（未找到答案）

就是用图形化的方法表示各种状态随时间变化的状态。动态视图包括：顺序图、交互图、状态图、活动图。

## 69.体系结构

**体系结构包括一组部件以及部件之间的联系。**

## 42.过程框架

定义了**若干小的框架活动**，为完整的软件开发过程建立基础

## 43.功能点分析

是面向功能的软件度量方法，**注意力集中于软件系统的“功能性”和“实用性”**

## 44.结构化程序设计

进行大型程序设计的一种方法。整个程序是由各个部分按一定的层次结构组织起来的。每一层次中规定只使用几种基本的程序结构。从最高层开始，依次向下，逐步求精。设计出的程序条理清楚、容易理解、容易调试与维护

## 48.工作分解结构

以可交付成果为导向对项目要素进行的分组，它归纳和定义了项目的整个工作范围每下降一层代表对项目工作的更详细定义。

## 49.非功能性需求

非功能需求（non-functional requirement）是从各个角度对系统的约束和限制，反映了应用对软件系统质量和特性的额外要求。

## 53.COCOMO模型

构造性成本模型，是一种利用经验模型进行成本估算的方法。

它是一种精确、易于使用的，**基于模型的成本估算方法**。最早由勃姆（Boehm）于 1981 年提出。从本质上说是一种参数化的项目估算方法，参数建模是把项目的某些特征作为参数，通过建立一个数字模型预测项目成本（类似于居住面积作为参数计算的整体的住房成本）。

## 52.个体软件过程

就是为使软件工程师更好地工作而设计的一个框架，它指出如何估计和计划工作，如何按照这些计划来跟踪自己的性能，以及如何提高程序的质量。

## 57.团队软件过程

团队软件过程是为开发软件产品的开发团队提供指导，TSP的早期实践侧重于帮助开发团队改善其质量和生产率，以使其更好的满足成本及进度的目标。加上PSP帮助高绩效的工程师在一个团队中工作，来开发有质量保证的软件产品，生产安全的软件产品，改进组织中的过程管理。

## 60.主动对象

主动对象内部包含一个**线程**，可以自动完成动作或改变状态。主动对象是内部拥有自己的控制线程的对象。

## 63.RMMM计划（Risk Mitigation, Monitoring and Management Plan）

翻译为：风险缓解，监测和管理计划，又称软件项目风险管理，是软件项目管理的重要内容。在进行软件项目风险管理时，要辩识风险，评估它们出现的概率及产生的影响，然后建立一个规划来管理风险。风险管理的主要目标是预防风险。

软件项目风险是指在软件开发过程中遇到的预算和进度等方面的问题以及这些问题对软件项目的影响。软件项目风险会影响项目计划的实现，如果项目风险变成现实，就有可能影响项目的进度，增加项目的成本，甚至使软件项目不能实现。

## 67.统一过程

是软件工程的过程。**它提供了在开发组织中分派任务和责任的纪律化方法。**它的目标是在可预见的日程和预算前提下，确保满足最终用户需求的高质量产品。

统一过程模型是一种“**用例驱动，以体系结构为核心，迭代及增量**”的软件过程框架，由UML方法和工具支持。

## 50.统一建模语言UML

**它是一个支持模型化和软件系统开发的图形化语言**,为软件开发的所有阶段提供模型化和可视化支持，包括由需求分析到规格，到构造和配置。是一种面向对象的建模语言，**它是运用统一的、标准化的标记和定义实现对软件系统进行面向对象的描述和建模。**

## 70.消息

消息，软件对象之间进行交互作用和通讯是利用消息的。在面向对象的程序设计中，消**息是指一个类实例和另一个类实例之间传递的信息**。

## 73.结构复杂性度量（未找到答案）

软件度量的根本目的是为了管理的需要,利用度量来改进软件过程。人们是无法管理不能度量的事物。没有对软件过程的可见度就无法管理;而没有对见到的事物有适当的度量或适当的准则去判断、评估和决策,也无法进行优秀的管理。

结构复杂性度量就是为了管理需要，通过工程化的方法去衡量软件内部的结构的复杂性。

## 74.决策表

决策表又称判断表，是一种呈**表格状的图形工具**，**适用于描述处理判断条件较多，各条件又相互组合、有多种决策方案的情况**。精确而简洁描述复杂逻辑的方式，将多个条件与这些条件满足后要执行动作相对应。但不同于传统程序语言中的控制语句，决策表能将多个独立的条件和多个动作直接的联系清晰的表示出来

## 78.文本复杂性度量（未找到答案）

软件的复杂性。分为四类：结构、数据结构、算法、文档复杂性度量。

## 79.数据字典

精确地、严格地定义了**每个与系统相关的数据元素**，并以字典式顺序将它们组织起来，使得用户和分析员对所有的输入、输出、存储成分和中间计算有共同的理解。

## 88.错误播种（植入）模型（未找到答案）

是一种软件错误模拟技术，即在软件系统中人为的播种一定数量的已知类型的错误或错误数目，之后通过让他人检测，看看检测出来的错误数中包含多少植入的错误，从而间接地计算出软件中包含的错误数或者测试人员的水平。

当然也可以通过播种特定位置特定类型的错误来评估软件的性能，研究特定类型的软件错误的特性。

## 100.领域分析

领域分析的目的是**支持软件重用**，它产生的是描述了特定领域中各系统共性的领域模型。

领域分析的过程由领域分析的前期准备、分析领域知识和形成重用成分三部分组成。

面向对象分析方法是领域分析的一个很好工具。

## 15.面向对象方法特征有哪些？

封装性、继承性、多态性

**1、封装性**

封装是一种信息隐蔽技术，它体现于类的说明，是对象的重要特性。封装使数据和加工该数据的方法（函数）封装为一个整体，以实现独立性很强的模块，使得用户只能见到对象的外特性（对象能接受哪些消息，具有那些处理能力），而对象的内特性（保存内部状态的私有数据和实现加工能力的算法）对用户是隐蔽的。封装的目的在于把对象的设计者和对象者的使用分开，使用者不必知晓行为实现的细节，只须用设计者提供的消息来访问该对象。

**2、继承性**

继承性是子类自动共享父类之间数据和方法的机制。它由类的派生功能体现。一个类直接继职其它类的全部描述，同时可修改和扩充。

继承性具有传达室递性。继承分为单继承（一个子类只有一父类）和多重继承（一个类有多个父类）。类的对象是各自封闭的，如果没继承性机制，则类对象中数据、方法就会出现大量重复。继承不仅支持系统的可重用性，而且还促进系统的可扩充性。

**3、多态性**

对象根据所接收的消息而做出动作。同一消息为不同的对象接受时可产生完全不同的行动，这种现象称为多态性。利用多态性用户可发送一个通用的信息，而将所有的实现细节都留给接受消息的对象自行决定，如是，同一消息即可调用不同的方法。例如：Print消息被发送给一图或表时调用的打印方法与将同样的Print消息发送给一正文文件而调用的打印方法会完全不同。多态性的实现受到继承性的支持，利用类继承的层次关系，把具有通用功能的协议存放在类层次中尽可能高的地方，而将实现这一功能的不同方法置于较低层次，这样，在这些低层次上生成的对象就能给通用消息以不同的响应。在OOPL中可通过在派生类中重定义基类函数（定义为重载函数或虚函数）来实现多态性。

综上可知，在OO方法中，**对象和传递消息分别表现事物及事物间相互联系的概念**。类和继承是是适应人们一般思维方式的描述范式。方法是允许作用于该类对象上的各种操作。这种对象、类、消息和方法的程序设计范式的基本点在于对象的封装性和类的继承性。通过封装能将对象的定义和对象的实现分开，通过继承能体现类与类之间的关系，以及由此带来的动态联编和实体的多态性，从而构成了面向对象的基本特征。

## 55.面向对象

首先根据客户需求抽象出业务对象；**然后对需求进行合理分层，构建相对独立的业务模块；**之后设计业务逻辑，**利用多态、继承、封装、抽象的编程思想，**实现业务需求；最后通过整合各模块，**达到高内聚、低耦合的效果**，从而满足客户要求。

## 85.对象/类

类是对象的抽象，而对象是类的具体实例。类是抽象的，不占用内存，而对象是具体的，占用存储空间。

## 54.信息隐蔽

信息隐蔽指在设计和确定模块时,使得一个模块内包含信息(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。在面向对象方法中，信息隐蔽是通过对象的**封装性来实现的。信息隐蔽的概念与模块的独立性直接相关。**

## 80.封装

隐藏**对象的属性和实现细节**，仅对外公开接口,控制在程序中属性的读和修改的访问级别。

封装就是将抽象得到的数据和行为（或功能）相结合，形成一个有机的整体，也就是将数据与操作数据的源代码进行有机的结合，形成“类”，**其中数据和函数都是类的成员**。

## 75.继承

继承是指一个**对象**直接使用另一**对象**的**属性和方法**。

## 65.多态性

同一操作作用于不同**的类的实例**，将产生不同的执行结果，即不同类的**对象**收到相同的消息时，得到不同的结果。对象根据所接受的消息而做出动作，同样的消息被不同的对象接收时可能导致完全不同的行为，这种现象称为多态性。

## 59.内聚

内聚是模块功能强度（**一个模块内部各个元素彼此结合的紧密程度**）的度量。

## 64.耦合

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。

## 4.通信聚合

通信内聚：访问相同数据的所有操作被定义在一个类中。一般来说，这些类只着眼于数据的查询。访问和存储

## 9.信息聚合（未找到答案）

就是把目前所掌握的信息资料合到一块，围绕某主题，把极度分散、高度相关、前后依存的信息碎片，迅速、及时的整合成完整的、有参考价值的信息。

## 14.数据聚合（未找到答案）

结合来自不同节点的数据，通过去除冗余，减少信息的传输量，从而达到节省能量的目的。

## 19.控制耦合

当操作A调用操作B，并且向B传递了一个**控制标记**时，就会发生此种耦合

## 24.标记耦合

当**类B被声明**为类A某一操作中的**一个参数类型**时会**发生此种耦合**。由于**类B现在作为类A定义的一部分，**所以修改系统就会变得更为复杂

## 29.数据耦合

**当操作需要传递较长的数据参数时就会发生此种耦合**。随着类和构件之间通信“带宽”的增长以及接口复杂性的增加，测试和维护就会越来越困难

## 94.软件测试

**按是否查看程序内部结构分为：**

**黑盒测试**：只关心输入和输出的结果

着眼于软件的外部特性，而不开了软件的内部逻辑结构和处理过程，只依据程序的需求规格说明书，检查程序的功能是否符合他的功能说明，故又称为功能测试。

主要检查：程序的功能是否正确或完善；数据的输入能否正确接收，输出是否正确；能否保证外部信息的完整性。

方法有：等价类划分法，边界值法，错误推测法，因果图法，判定表法，状态迁移法，正交试验法

**白盒测试**：去研究里面的源代码和程序结构

白盒测试也称结构测试或逻辑驱动测试，它是知道产品内部工作过程，可通过测试来检测产品内部动作是否按照规格说明书的规定正常进行，按照程序内部的结构测试程序，检验程序中的每条通路是否都有能按预定要求正确工作，而不顾它的功能。

**语句覆盖**：设计足够的测试用例，使**每个语句**至少执行一次

**判定覆盖**：设计足够的测试用例，使**每判定**的取真分支和取假分支至少执行一次

**条件覆盖**：设计足够的测试用例，使每个判断中**每个条件的可能取值至少满足一次**

**条件组合覆盖**：设计足够的测试用例，使所有判定中**各条件判断结果的所有组**合至少出现一次。

**路径覆盖**：设计足够多的测试用例，覆盖程序中所有可能的路径。覆盖程度最高

**按是否运行程序分为：**

**静态测试**：是指不实际运行被测软件，而只是静态地检查扫描程序代码、界面或文档可能存在的错误的过程。

**动态测试**：是指实际运行被测程序，输入相应的测试数据，检查输出结果和预期结果是否相符的过程

**按阶段划分：**

**单元测试：**对软件中的最小可测试单元进行检查和验证。

**集成测试**：是单元测试的下一阶段，是指将通过测试的单元模块组装成系统或子系统，再进行测试，重点测试不同模块的接口部门。集成测试就是用来检查各个单元模块结合到一起能否协同配合，正常运行。

**系统测试**：指的是将整个软件系统看做一个整体进行测试，包括对功能、性能，以及软件所运行的软硬件环境进行测试。系统测试的主要依据是《系统需求规格说明书》文档。

**验收测试**：指的是在系统测试的后期，以用户测试为主，或有测试人员等质量保障人员共同参与的测试，它也是软件正式交给用户使用的最后一道工序。

其他还有回归测试、冒烟测试、随机测试

黑盒测试包括功能测试和性能测试；

功能测试有：逻辑功能测试、界面测试、易用性测试、安装测试、兼容测试；

性能测试有：一般性能测试、稳定性测试、压力测试、负载测试

## 76.测试配置（未找到答案）

配置测试用于测试和验证软件，在不同的软件和硬件配置中进行运行。配置测试就是测试软件是否和系统的其他与之交互的元素之间兼容，如浏览器、操作系统、硬件等，验证被测软件在不同的软件和硬件配置中的运行情况。

## 71.压力测试

在软件工程中，压力测试是对系统不断施加压力的测试，是通过确定一个系统的瓶颈或者不能接收的性能点，来获得系统能提供的最大服务级别的测试。

## 61.确认测试

检查已实现的软件是否满足了需求规格说明中所确定的各种需求，以及软件配置是否完全、正确。（过程：功能性测试---软件配置复查---验收测试----α测试和β测试）

## 91.Beta测试

Beta测试由软件的最终用户们在一个或多个客房场所进行。与Alpha测试不同，开发者通常不在Beta测试的现场，因Beta测试是软件在开发者不能控制的环境中的“真实”应用。用户Beta测试过程中遇到的一切问题（真实在或想像的），并且定期把这些问题报告给开发者。接收到在Beta测试期间报告的问题之后，开发者对软件产品进行必要的修改，并准备向全体客户发布最终的软件产品

## 36.基本路径测试

是在程序控制流图的基础上，通过分析**控制构造的环路复杂性**，导出基本可执行路径集合，从而设计测试用例的方法。设计出的测试用例要保证在测试中程序的每个可执行语句至少执行一次

## 41.黑盒测试

黑盒测试也称功能测试或数据驱动测试，它是在已知产品所应具有功能的情况下，通过测试来检测每个功能是否都能正常使用。

黑盒测试也称功能测试，它是通过测试来检测每个功能是否都能正常使用。在测试中，把程序看作一个不能打开的黑盒子，在完全不考虑程序内部结构和内部特性的情况下，在程序接口进行测试，它只检查程序功能是否按照需求规格说明书的规定正常使用，程序是否能适当地接收输入数据而产生正确的输出信息。黑盒测试着眼于程序外部结构，不考虑内部逻辑结构，**主要针对软件界面和软件功能进行测试。**

## 26.等价类划分

等价类划分法是一种典型的、重要的**黑盒测试方法**，它将程序所有可能的输入数据（有效的和无效的）划分成若干个等价类。然后从每个部分中选取具有代表性的数据当做测试用例进行合理的分类，测试用例由有效等价类和无效等价类的代表组成，从而保证测试用例具有完整性和代表性。

## 31.边界值测试

针对**各种边界情况**设计测试用例

## 96.因果图

因果图是根据分析时对各种条件可能产生的结果的一种对照分析法。 **用于软件黑盒测试.**

鱼骨图是由日本管理大师石川馨先生所发明出来的，故又名石川图。鱼骨图是一种发现问题“根本原因”的方法。其特点是简捷实用，深入直观。它看上去有些象鱼骨，问题或缺陷（即后果）标在"鱼头"外。在鱼骨上长出鱼刺，上面按出现机会多寡列出产生生产问题的可能原因。鱼骨图有助于说明各个原因之间如何相互影响。它也能表现出各个可能的原因是如何随时间而依次出现的。这有助于着手解决问题。

## 46.白盒测试

## 98.基于时间的缺陷到达模式（未找到答案）

是一种软件过程质量测试方法，是基于时间的缺陷到达模式，缺陷到达模式可以提供更多的信息，有时即使得到的整体缺陷率是一样的，但其质量差异可能较大，原因就是缺陷到达的时间不同，越多的缺陷到达越早，则测试过程质量就越好。

# 三、简答题（100题）

## 1.简述软件安全性与软件可靠性的异同。

相同点：软件安全性与软件可靠性均与软件失效有关.

不同点：软件可靠性使用统计分析的方法来确定软件失效发生的可能性,而失效的发生未必导致灾难或灾祸.软件安全则考察失效会导致灾难发生的条件.

## 2.简述软件开发的可行性研究的目的和内容。

可行性研究的目的在于**研究解决问题的必要性和可能性**

研究的内容：技术可行性；经济可行性；社会可行性包括；法律可行性

## 3.什么是面向数据流的设计，数据流包括哪几种？

面向数据流的设计方法把**信息流映射成软件结构**，信息流的类型决定了映射的方法，**根据系统的数据流进行的设计，又称结构化设计**

数据流包括：变换流、事务流

1．变换流

根据基本系统模型，信息通常以外部世界的形式进入软件系统，经过处理以后再以外部世界的形式离开系统。

2．事务流

基本系统模型意味着变换流，因此，原则上所有信息流都可以归结为这一类。但是当数据流是以事务为中心的，也就是说，数据沿输入通路到达一个处理T，这个处理根据输入数据的类型在若干个动作序列中选出一个来执行。这类数据流应该划为一类特殊的数据流，称为事务流。

## 4.什么时候测试结束，为什么？

通过在测试过程中进行风险、测试、覆盖率、缺陷和信心这5个方面的评估，并和测试计划中的要求进行比较，若能满足要求,就结束测试.

在软件消亡之前，如果没有测试的结束点，那么软件测试就永无休止，永远不可能结束。软件测试的结束点，要依据自己公司具体情况来制定，不能一概而论！个人认为测试结束点由以下几个条件决定：：

1）．基于“测试阶段”的原则

2）．基于“测试用例”的原则

3）．基于“缺陷收敛趋势”的原则

4）．基于“缺陷修复率”的原则

5）．基于“验收测试”的原则

6)．基于“覆盖率”的原则

7)．基于“项目计划”的原则

8)．基于“缺陷度量”的原则

9)．基于“质量成本”的原则

10).基于“测试行业经验”的原则

## 5.什么是对象，对象有哪几种形式？

**在系统中用来描述客观事物的一个实体**，它是构成系统的一个基本单元，由一组属性和一组对属性进行操作的服务组成。

对象就是一个包含数据以及与这些数据有关的操作的集合。每个实体都是对象

## 6.什么是McCabe复杂性度量，如何计算？

是利用程序模块的程序图中环路的个数，来计算程序的复杂性的。为此，也称为环路复杂度计算法。

计算方法：

V(G)=m-n+p

V(G)是有向图G中环路数，m是图G中弧数，n是图G中结点数，p**是图G中强连通分量个数。**

## 7.软件项目管理的主要方面是什么？

1）软件项目计划与组织

2）软件项目成本管理

3）软件项目进度控制

4）软件质量保证

5）软件配置管理

6）生成软件项目管理文档

## 8.怎样完成从数据流图到程序结构图的转换？

从以下4个步骤进行:

1.重画数据流图 .

2.区分有效（逻辑）输入、有效（逻辑）输出和**中心变换部分**

3.进行一级分解，设计上层模块

4.进行二级分解，设计中、下层模块

## 9.如何估算程序中的错误？

可用下面的估算方法来计算:

1)用Shooman模型估算ET

2)用植入法估算固有错误总数

3)用**最小二乘法估算**固有错误

## 10.什么是类，类与对象间是什么关系？

类是一种**复杂的数据类型，**它是将不同类型的数据和与这些数据相关的操作封装在一起的集合体。类是对某一类对象的抽象,而对象是某一种类的实例

## 11.软件的缺陷为什么在软件开发和维护过程中会扩大？

## 12.成本估算方法有哪几种？

成本估算方法 就方法论而言，有两种基本的成本估算方法：自顶向下和自底向上。

这两种方法都要求采用某种方法做出估算,可以利用的估算方法有:

1、工作分解结构方法（WBS）

2、专家判断

3、 基于问题的估算

4、基于经验模型的估算

## 13.面向数据流的设计方法包含那些步骤？

1) 确定数据流的类型：变换流还是事务流

2) 划定流界

3) 将数据流图转换为软件结构

4) 通过设计复审和启发式策略精化所得到软件结构

## 14.什么叫测试用例，如何设计测试用例

测试用例（Test Case）是为某个特殊目标而编制的一组测**试输入、执行条件以及预期结果，**以便测试某个程序路径或核实是否满足某个特定需求。

测试种类：白盒测试、黑盒测试、静态测试、动态测试测试种类的组合：静态白盒测试、静态黑盒测试、动态白盒测试、动态黑盒测试

## 16.什么是CMM关键过程域，举例说明其含义。

关键过程域是指一系列相互关联的操作活动，这些活动反映了一个软件组织改进过程时必须集中精力改进的几个方面。换句话说，关键过程域标识了达到某个成熟度等级时所必须满足的条件。

如软件成熟度模型中第 4 级（管理级）有 2 个关键过程域，**主要的任务是为软件过程和软件产品建立一种可以理解的定量的方式。**

## 17.影响软件成本的主要因素有哪些具体方面？

软件的规模、复杂度、开发工具、开发人员的业务素质、软件工程管理水平及开发工作环境等。

## 18.画数据流图原则是什么？

（1）明确系统界面。

（2）自顶向下逐层扩展。

（3）合理布局。

（4）数据流图绘制过程，就是系统的逻辑模型的形成过程，必须始终与用户密切接触，详细讨论，不断修改，也要和其他系统建设者共同商讨一求一致意见。

## 19.什么叫基本路径测试，如何确定基本路径？

实际程序中路径是个很庞大的数字，所有路径覆盖是不现实的，测试中把路径数压缩在一定范围内，称为基本路径测试。

**在程序控制流图的基础上，通过分析控制构造的环路复杂性测度，导出基本可执行路径的基本集合。**

## 20.简述USE/CASE的作用。

**测试用例：**

在不展现一个系统或子系统内部结构的情况下，对系统或子系统的某个连贯的功能单元的定义和描述。其实Use Case就是对系统功能的描述而已，不过一个Use Case描述的是整个系统功能的一部分，这一部分一定要是在逻辑上相对完整的功能流程。

测试用例是为某个特殊目标而编制的一组测试输入、执行条件以及预期结果，以便测试某个程序路径或核实是否满足某个特定需求。

**测试用例的作用：**

1，通过一个用例来证明被测软件的某功能符合需求说明书中规定的要求，可以通过设计正反两方面的测试用例来验证

2，可以保证一个软件被测试的有效性，使测试人员知道哪些些功能以被测，哪些功能还需要测试，从而避免漏测，重复测，提高测试效率

3，指导测试的工作，保证测试是有计划的实施，而不是随意性的测试

4，为公司留下财富，为后期软件维护提高帮助，为公司新人进来提供指导，在测试的时候可以尽量把人为因素的影响减少到最小。保障软件测试质量的稳定

5，可以做为评估测试结果的，为编写测试报告提供依据。

6，分析缺陷的标准，通过收集缺陷，对比测试用例和缺陷数据库，分析确证是漏测还是缺陷复现。漏测反映了测试用例的不完善，应立即补充相应测试用例，最终达到逐步完善软件质量。而已有相应测试用例，则反映实施测试或变更处理存在问题。

## 21.软件质量的特性有哪些，怎样度量？

功能性、可靠性、易用性、效率、可维护性、可移植性。

通**过产品质量、过程中质量和软件维护质量**度量等三种方法来度量。

## 23.数据流图作用是什么？

DFD是**描述数据处理的工具**。 数据流图从数据传递和加工的角度，以图形的方式刻画数据流从输入到输出的移动变换过程。从而便于需求分析.

## 24.简述回溯测试和作用。

回溯法采用试错的思想，它尝试分步的去解决一个问题。在分步解决问题的过程中，当它通过尝试发现现有的分步答案不能得到有效的正确的解答的时候，它将取消上一步甚至是上几步的计算，再通过其它的可能的分步解答再次尝试寻找问题的答案。回溯法通常用最简单的递归方法来实现，在反复重复上述的步骤后可能出现两种情况：

1、找到一个可能存在的正确的答案

2、在尝试了所有可能的分步方法后宣告该问题没有答案

作用：

1、找出满足约束条件的所有解。

## 25.面向对象设计涉及那几个主要活动？

分析活动、设计活动、编码活动

## 26.Boehm和McCall等人从哪些方面来确定软件质量，他们有什么异同？

软件质量就是“软件与明确的和隐含的定义的需求相一致的程度”。具体地说，软件质量是软件符合明确叙述的功能和性能需求、文档中明确描述的开发标准、以及所有专业开发的软件都应具有的隐含特征的程度。 影响软件质量的主要因素，这些因素是从管理角度对软件质量的度量。可划分为三组，分别反应用户在使用软件产品时的三种观点。正确性、健壮性、效率、完整性、可用性、风险（产品运行）；可理解性、可维修性、灵活性、可测试性（产品修改）；可移植性、可再用性、互运行性（产品转移）。

McCall等人将质量模型分为三层因素、衡量准则、度量,并对软件质量因素进行了研究,认为软件质量是正确性、可靠性、效率等构成的函数,而正确性、可靠性、效率等被称为软件质量因素,或软件质量特征,它表现了系统可见的行为化特征。每一因素又由一些准则来衡量,而准则是跟软件产品和设计相关的质量特征的属性。

Boehm模型是由Boehm等在1978年提出来的质量模型,在表达质量特征的层次性上它与McCall模型是非常类似的。不过,它是基于更为广泛的一系列质量特征,它将这些特征最终合并成19个标准。Boehm提出的概念的成功之处在于它**包含了硬件性能的特征**,这在McCall模型中是没有的。但是,其中与McCall模型类似的问题依然存在。

## 27.什么是软件的逆向工程和再工程？

软件逆向工程是通过反汇编和调试等手段分析计算机程序的二进制可执行代码，从而获得程序的算法细节和实现原理的技术

软件再工程是对既存软件系统进行调查，**并将其重构为新形式代码的开发过程**。

## 28.简述数据字典的作用

数据字典是指对数据的数据项、数据结构、数据流、数据存储、处理逻辑、外部实体等进行定义和描述，其目的是对数据流程图中的各个元素做出详细的说明。就是用来描述数据流和加工的具体含义

数据字典（Data dictionary）是一种用户可以访问的记录数据库和应用程序源数据的目录。主动数据字典是指在对数据库或应用程序结构进行修改时，其内容可以由DBMS自动更新的数据字典。被动数据字典是指修改时必须手工更新其内容的数据字典。

数据字典最重要的作用是**作为分析阶段的工具**。任何字典最重要的用途都是供人查询对**不了解的条目的解释**，在结构化分析中，数据字典的作用是给数据流图上每个成分加以定义和说明。换句话说，**数据流图上所有的成分的定义和解释的文字集合就是数据字典，**而且在数据字典中建立的一组严密一致的定义很有助于改进分析员和用户的通信。

## 29.纠错技术中的归纳法和演绎法的中心思想是什么？

1.归纳法，指的是从许多个别事例中获得一个较具概括性的规则。这种方法主要是从收集到的既有资料，加以抽丝剥茧地分析，最后得以做出一个概括性的结论。演绎法，则与归纳法相反，是从既有的普遍性结论或一般性事理，推导出个别性结论的一种方法。由较大范围，逐步缩小到所需的特定范围。

2.归纳法是从特殊到一般，优点是能体现众多事物的根本规律，且能体现事物的共性。缺点是容易犯不完全归纳的毛病。演绎法是从一般到特殊，优点是由定义根本规律等出发一步步递推，逻辑严密结论可靠，且能体现事物的特性。缺点是缩小了范围，使根本规律的作用得不到充分的展现。归纳法和演绎法在应用上并不矛盾，有些问题可采用前者，有些则采用后者。而更多情况，将两者结合着应用，则能收到更好的效果。

## 30.面向对象设计如何体现抽象信息、隐藏和模块化这三个概念的？

通过类的引入, 封装,把数据结构和操作这些数据的方法紧密地结合在一起所构成的模块.对象的封装性实现隐藏。

## 31.影响软件质量的因素分哪两大类？

(1) 可以直接度量的因素， 如单位时间内千行代码（KLOC）中所产生的错误数。

(2) 只能间接度量的因素， 如可用性或可维护性。

## 32.如何衡量软件成本，各表示什么含义？

成本估算和成本管理是软件项目管理的核心任务之一。在制定项目计划时，就必须对项目需要的人力及其他资源、项目持续时间和项目成本做出估算。如果新项目和以往的项目类似，估算可以参考以前的成本费用。 现在已有一些用于软件成本估算的技术可供借鉴。这些估算技术各有其优缺点，但以下几方面是共同的：

–事先建立软件的工作范围；

–以软件度量（经验度量、相似工程类比的度量）为基础做出估算

–把项目分解为可单独进行估算的小块

成本估算的工具和技术：

1.类比估算法：也叫自上而下估算法，他是使用以前相似项目的实际成本作为目前项目成本估算的根据。

2.自下而上估算法：是估计各个工作项或活动，并将单个工作项汇总成整体项目估算的一种方法，又是成为基于活动成本法（ABC法）。

3.工作分布估算法：使用项目阶段百分比来估算。

4.参数模型法：是在数学模型中应用项目特征（参数）估算项目成。

5.启发式估算：使用历史数据和统计关系来估算

6.分阶段估算：按阶段逐一估算项目（最适合高风险项目的估算技巧）

7.计算机化工具。

## 33.需求分析的任务是什么？

1、确定对系统的综合要求：

2、**分析系统的数据要求**

**3、导出系统的逻辑模型**

4、修正项目开发计划

5、开发原型系统

## 35.举例说明什么是类的整体部分结构

如建立一个时钟类，这个类中的属性如秒、分钟、小时等，加上对属性的操作如秒记时，分钟记时，小时记时等，就是一个类的整体部分结构。

## 36.软件质量的含义是什么？

软件在使用时，能适合用户需要的目标程度。

## 37.如何安排软件开发进度，应该考虑哪些问题？

识别进度计划所有者

决定任务和里程碑

排序工作活动

任务历时评估

整合任务计划

审查批准进度计划

考虑：

1. 任务分配、人力资源分配、时间分配要与工程进度相协调
2. 任务分解与并行化

3、工作量分布

目前，程序评估与审查技术（PERT）和关键路径方法（CPM）是两种比较常用的项目进度安排方法

## 38.什么是结构化分析，有什么特点？

是面向数据流进行需求的分析，遵循自顶向下、逐步求精原则。

主要特点是快速、自然和方便。成功率较高，发展较为成熟；简单、易掌握，适应于瀑布模型；特别适合于数据处理领域中的应用，对规模大的项目，特别复杂的应用不太适应。难于解**决软件重用问题**，难于适应需求的变化。

## 39.什么是边界值分析，与等价类划分有什么不同？

边界值分析法就是对输入或输出的边界值进行测试的一种黑盒测试方法。

（1）、边界值分析不是从某等价类中随便挑一个作为代表，而是使这个等价类的每个边界都要作为测试条件。

（2）、边界值分析不仅考虑输入条件，还要考虑输出空间产生的测试情况。

## 40.举例说明什么是类的组装结构关系

如建立一个时钟类，这个类中的属性如秒、分钟、小时等，加上对属性的操作如秒记时，分钟记时，小时记时等，就构成一个类的组装结构关系。

## 41.原型开发由哪些步骤？

（1）、确定原型开发目标

（2）、功能选择

（3）、原型构造

（4）、评价原型

## 42.什么是软件配置管理，有什么作用？

简单而言就是**管理软件的变化**,它应用于整个软件工程过程，通常由相应的工具、过程和方法学组成。

实施有效的软件配置管理可以解决以下软件开发中的常见问题：

1、开发人员未经授权修改代码或文档;

2、人员流动造成企业的软件核心技术泄密;

3、找不到某个文件的历史版本;

4、无法重现历史版本；

4、无法重新编译某个历史版本，使维护工作十分困难；

5、“合版本”时，开发冻结，造成进度延误；

6、、软件系统复杂,编译速度慢,造成进度延误；

7、因一些特性无法按期完成而影响整个项目的进度或导致整个项目失败；

8、已修复的Bug在新版本中出现；

9、配置管理制度难于实施;

10、分处异地的开发团队难于协同,可能会造成重复工作,并导致系统集成困难；

## 43.原型开发技术的目的，特点是什么？

解决用户与开发者之间的鸿沟,以原型(软件产品的样品)为共同语言，实现用户与开发者双向沟通。

容易地确定系统的性能，确认各项主要系统服务的可应用性，确认系统设计的可行性，确认系统作为产品的结果。

软件开发人员和用户之间的理解偏差在功能展现时显露出来在原型设计中发现需求的不完善和不一致可迅速地展现一个简单的应用系统通过原型系统可以导出系统需求原型系统可以支持用户培训和系统测试。

## 44.简述在确定等价划分测试用例设计中遵循的原则。

如果输入条件规定了取值范围，或者值的个数，则可以确定**一个有效等价类和两个无效等价类；**

如果输入条件规定了输入值的集合，或者是规定了“必须如何”的条件，这时可以确立一个有效等价类和一个无效等价类；

如果输入条件是一个布尔量，则可以确立一个有效等价类和一个无效等价类；

如果规定了输入数据的一组值，而且程序要对每一个输入值分别进行处理，这时要对每一个规定的输入值确立一个等价类，而对于这组值之外的所有值确立一个等价类；

如果规定了输入数据必须遵守的规则，则可以确立一个有效等件类（即遵守规则的数据）和若干无效等价类（从不同角度违反规则的数据）；

如果确知以划分的等价类中的各元素在程序中的处理方式不同，则应进一步划分成更小的等价类

## 45.领域分析的目标是什么，依据是什么？

专注于理解问题的真实本质设计出一套准确简洁的、可以理解的和正确的真实世界的模型。业务过程会谈

## 46.原型开发包括那几类，目的是什么？

（1）、抛弃式原型开发是弄清需求和为风险评估提供额外的信息。

（2）、演化式原型开发先给出一个系统的最初实现，让用户去使用和评价，不断进行细化和完善，经过多次反复形成最后完善的应用系统。

（3）、递增式原型开发减少修改的范围，实现易于控制和管理。

## 47.基线在配置管理中有什么作用？

为了有效地控制变动

## 48.简述总体设计的一般过程？

总体设计又称为概要设计或初步设计。总体设计过程通常由两个主要阶段组成：

系统设计:确定系统的具体实现方案。结构设计:确定软件结构。9个步骤：

设想供选择的方案；选取合理的方案；推荐最佳方案；功能分解；设计软件结构；设计数据库；制定测试计划；书写文档；审查和复审

## 49.什么是软件测试中逻辑覆盖，一般软件测试至少应保证哪些覆盖？

是以程序内部的逻辑结构为基础的测试技术，是通过对程序逻辑结构的遍历实现程序的覆盖，这一方法要求测试人员对程序的逻辑结构有清楚的了解.语句覆盖.

## 50.类的开发有几个途径，如何进行类的开发？

一是新建基类，二是通过类间的继承来进行类的开发。

## 51.原型具有什么特点？

（1）、遵循了人们认识事物的客观规律，易于掌握和接受

（2）、将模拟的手段引入系统分析的初始阶段，沟通了人们（用户和开发人员）的思想，缩短了用户和系统分析人员之间的距离，解决了结构化方法中最难于解决的一环。强调用户参与、描述、运行、沟通。

（3）、充分利用最新的软件工具，摆脱了传统的方法，使系统开发的时间、费用大大地减少，效率、技术等方面都大大地提高。强调软件工具支持。

## 52.软件维护的基本内容是什么？

**改正性维护**

是指改正在系统开发阶段已发生而系统测试阶段尚未发现的错误。这方面的维护工作量要占整个维护工作量的17%～21%。所发现的错误有的不太重要，不影响系统的正常运行，其维护工作可随时进行：而有的错误非常重要，甚至影响整个系统的正常运行，其维护工作必须制定计划，进行修改，并且要进行复查和控制。

**适应性维护**

是指使用软件适应信息技术变化和管理需求变化而进行的修改。这方面的维护工作量占整个维护工作量的18%～25%。由于目前计算机硬件价格的不断下降．各类系统软件屡出不穷，人们常常为改善系统硬件环境和运行环境而产生系统更新换代的需求；企业的外部市场环境和管理需求的不断变化也使得各级管理人员不断提出新的信息需求。这些因素都将导致适应性维护工作的产生。进行这方面的维护工作也要像系统开发一样，有计划、有步骤地进行。

**完善性维护**

　　这是为扩充功能和改善性能而进行的修改，主要是指对已有的软件系统增加一些在系统分析和设计阶段中没有规定的功能与性能特征。这些功能对完善系统功能是非常必要的。另外，还包括对处理效率和编写程序的改进，这方面的维护占整个维护工作的50%～60%，比重较大．也是关系到系统开发质量的重要方面。这方面的维护除了要有计划、有步骤地完成外．还要注意将相关的文档资料加入到前面相应的文档中去。

**预防性维护**

　　为了改进应用软件的可靠性和可维护性，为了适应未来的软硬件环境的变化，应主动增加预防性的新的功能，以使应用系统适应各类变化而不被淘汰。例如将专用报表功能改成通用报表生成功能，以适应将来报表格式的变化。这方面的维护工作量占整个维护工作量的4%左右。

## 53.解释体系结构中深度、宽度、扇出、扇入对软件的影响。

**深度表示软件结构中控制的层数.**

宽度是软件结构中同一个层次上的模块总数的最大值

一个模块的扇入是指直接调用该模块的上级模块的个数.

一个模块的扇出是指该模块直接调用的下级模块的个数.

设计原则:低扇出高扇入，扇入大表示模块的重用性高,利用率高.扇出大表示模块的复杂度高.所以要高扇入低扇出.

## 54.为什么说在确认测试阶段发现需求理解的错误，要修改就要付出更大的代价？

## 55.什么是OOA建模语言，它应该包括哪些方面？

Object-Oriented Analysis（面向对象分析方法）是确定需求或者业务的角度，按照面向对象的思想来分析业务。

分析模型

1、对象模型:对用例模型进行分析,把系统分解成互相协作的分析类,通过类图/对象图描述对象/对象的属性/对象间的关系,是系统的静态模型

2、动态模型:描述系统的动态行为,通过时序图/协作图描述对象的交互,以揭示对象间如何协作来完成每个具体的用例,单个对象的状态变化/动态行为可以通过状态图来表达

3、功能模型(即用例模型à作为输入)。

UML

## 56.为什么要进行阶段评审？

评审项目的状态，并确定项目是否应该进入下一阶段。

项目的资源状况（人力、资金）

项目的风险状况

项目的技术进展

项目的规模

里程碑状态（进度）

## 57.软件维护分几类，每类的内容是什么？

## 58.什么叫模块化，为什么要划分模块？

指解决一个复杂问题时自顶向下逐层把系统划分成若干模块的过程

第一，把一个系统分解成各个不同的子模块，不同的开发者专注于对其中某一模块的开发，一方面实现了劳动的分工，另一方面也提高了自由软件开发的效率。

第二，对于开发者而言，基于模块化的自由软件开发具有更大的吸引力，其在参与开发过程中可以得到更高的期望收益。

第三，在非模块化的软件开发过程中，存在着严重的“搭便车”现象，当一个开发者选择参与开发，其余的开发者就会选择“搭便车”，最终会导致软件的供给不足；在基于模块化的开发过程中，所有的开发者都更倾向于参与开发不同的模块，从而实现整个系统的开发。

## 59.比较在组装测试中，自顶向下与自底向上、深度优先与宽度优先的优缺点。

**自顶向下**

**优点:**

<1>较早地验证主要的控制和判断点。

<2>可以首先实现和验证一个完整的软件功能，增强信心；

<3>减少了测试驱动程序开发和维护的费用;

<4>可以和开发设计工作一起并行执行集成测试，能够灵活的适应目标环境；

<5>容易进行故障隔离和错误定位。

**缺点：**桩模块的开发和维护费用大。

**自底向上**

**优点**

<1>减少了桩模块的工作量;

<2>容易对错误进行定位。

**缺点**

<1>直到最后一个模块加进去之后才能看到整个系统的框架；

<2>只有到测试过程的后期才能发现时序问题和资源竞争问题；

<3>驱动模块的设计工作量大;

<4>不能被及时发现高层模块设计上的错误。

自底向上的组装测试方案是工程实践中最常用的测试方法。相关技术也较为成熟。它的优点很明显:管理方便、测试人员能较好地锁定软件故障所在位置。但它对于某些开发模式不适用，如使用XP开发方法，它会要求测试人员在全部软件单元实现之前完成核心软件部件的组装测试。尽管如此，自底向上的组装测试方法仍不失为一个可供参考的组装测试方案。

自顶向下集成的优点在于能尽早地对程序的主要控制和决策机制进行检验，因此较早地发现错误。缺点是在测试较高层模块时，低层处理采用桩模块替代，不能反映真实情况，重要数据不能及时回送到上层模块，因此测试并不充分。

自底向上集成方法不用桩模块，测试用例的设计亦相对简单，但缺点是程序最后一个模块加入时才具有整体形象。它与自顶向综合测试方法优缺点正好相反。

## 60.简述RUP（Rational Unified Process）的基本内容

RUP（统一软件开发过程，统一软件过程)是一个面向对象且基于**网络**的程序开发方法论。

**内容：**迭代式开发、管理需求、体系结构、可视化建模、验证软件质量、控制软件变更

## 100.简要说明RUP中阶段、核心工作流和迭代的关系。

RUP（统一软件开发过程，统一软件过程)是一个面向对象且基于网络的程序开发方法论。

RUP中的软件生命周期在时间上被分解为四个顺序的阶段，分别是：初始阶段（Inception）、细化阶段（Elaboration）、构造阶段（Construction）和交付阶段（Transition）。每个阶段结束于一个主要的里程碑（Major Milestones）；每个阶段本质上是两个里程碑之间的时间跨度。在每个阶段的结尾执行一次评估以确定这个阶段的目标是否已经满足。如果评估结果令人满意的话，可以允许项目进入下一个阶段。

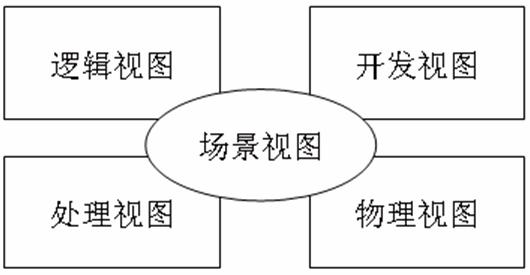
RUP中有9个核心工作流，分为6个核心过程工作流和3个核心支持工作流。尽管6个核心过程工作流可能使人想起传统瀑布模型中的几个阶段，**但应注意迭代过程中的阶段是完全不同的，这些工作流在整个生命周期中一次又一次被访问。9个核心工作流在项目中轮流被使用，在每一次迭代中以不同的重点和强度重复。**

## 70.什么是RUP（Rational Unified Process）的4+1视图？

RUP，统一软件开发过程，是一个面向对象且基于网络的程序开发方法论。

要开发出用户满意的软件并不是件容易的事，软件架构师必须全面把握各种各样的需求、权衡需求之间有可能的矛盾之处，从不同视图进行架构设计，来分门别类地将不同需求一一满足

Philippe Kruchten提出的4+1视图方法



该方法的不同架构视图承载不同的架构设计决策，支持不同的目标和用途：

逻辑视图：当采用面向对象的设计方法时，逻辑视图即对象模型。

开发视图：描述软件在开发环境下的静态组织。

处理视图：描述系统的并发和同步方面的设计。

物理视图：描述软件如何映射到硬件，反映系统在分布方面的设计。

## 61.简述模型在软件开发中的作用。

软件开发是一个复杂的过程,建立模型可降低复杂度.建立模型可以使开发者有效地交流.软件开发模型能清晰、直观地表达软件开发全过程，明确规定了要完成的主要活动和任务，用来作为软件项目工作的基础。

## 62.软件维护有哪些特点？

1、结构化维护与非结构化维护差别巨大

2、维护的代价高昂

3、维护的问题很多

下面列出和软件维护有关的部分问题：

（1）理解别人写的程序通常非常困难，而且困难程度随着软件配置成分的减少而迅速增加。如果仅有程序代码没有说明文档，则会出现严重的问题。

（2）需要维护的软件往往没有合格的文档，或者文档资料显著不足。认识到软件必须有文档仅仅是第一步，容易理解的并且和程序代码完全一致的文档才真正有价值。

（3）当要求对软件进行维护时，不能指望由开发人员给我们仔细说明软件。由于维护阶段持续的时间很长，因此，当需要解释软件时，往往原来写程序的人已经不在附近了。

（4）绝大多数软件在设计时没有考虑将来的修改。除非使用强调模块独立原理的设计方法学，否则修改软件既困难又容易发生差错。

　　（5）软件维护不是一项吸引人的工作。形成这种观念很大程度上是因为维护工作经常遭受挫折。

上述种种问题在现有的没采用软件工程思想开发出来的软件中，都或多或少地存在着。不应该把一种科学的方法学看做万应灵药，但是，软件工程至少部分地解决了与维护有关的每一个问题。

## 63.模块化的三个重要特征是什么，阐明各自的作用。

1.相对独立性，可以对模块单独进行设计、制造、调试、修改和存储，这便于由不同的专业化企业分别进行生产；

2.互换性，模块接口部位的结构、尺寸和参数标准化，容易实现模块间的互换，从而使模块满足更大数量的不同产品的需要；

3.通用性，有利于实现横系列、纵系列产品间的模块的通用，实现跨系列产品间的模块的通用。

## 64.单元测试时，为什么需要开发驱动模块和桩模块？

这是因为在开发设计的初级阶段，完成一个单元的开发任务后，这个单元相关联上面、下面的模块尚未开发出来，为了要测试所开发单元，所以要驱动模块和桩模块

## 65.什么是Use Cases Model，主要作用是什么？

**是系统既定功能及系统环境的模型**

用例模型最重要的作用是将系统行为传达给客户或最终用户。

## 66.什么是需求工程？

需求工程是指应用已证实有效的技术、方法进行需求分析，确定客户需求，帮助分析人员理解问题并定义目标系统的所有外部特征的一门学科。

它通过合适的工具和记号系统地描述待开发系统及其行为特征和相关约束，形成需求文档，并对用户不断变化的需求演进给予支持。RE可分为系统需求工程（如果是针对由软硬件共同组成的整个系统）和软件需求工程（如果仅是专门针对纯软件部分）。软件需求工程是一门分析并记录软件需求的学科，它把系统需求分解成一些主要的子系统和任务，把这些子系统或任务分配给软件，并通过一系列重复的分析、设计、比较研究、原型开发过程把这些系统需求转换成软件的需求描述和一些性能参数。

需求工程是一个不断反复的需求定义、文档记录、需求演进的过程，并最终在验证的基础上冻结需求。80年代，HerbKrasner定义了需求工程的五阶段生命周期：需求定义和分析、需求决策、形成需求规格、需求实现与验证、需求演进管理。近来，MatthiasJarke和KlausPohl提出了三阶段周期的说法：获取、表示和验证。

## 67.简述软件的维护过程。

1、维护组织的建立

2、维护事件流的确定

3、维护报告文档的书写和审查

4、维护组织——明确维护责任、减少混乱

5、每个维护要求都通过维护管理员转交给相应的系统管理员去评价。系统管理员对维护任务做出评价之后，由变化授权人决定应该进行的活动。

## 68.简述层次方框图与软件结构图的异同点。

相同：形式类似。

不同：

1,层次图描绘软件的**层次结构.层次方框图描绘的是数据结构**。

2,层次图的**方框**表示模块或子模块。层次方框图的方框表示数据结构整体或其子集。

3,层次图的**连线**表示**调用关系**，层次方框图表**示组成关系**。

## 69.简述黑盒子测试和白盒子测试及其适用性。

黑盒子测试是一种把程序看做一个黑盒子，完全不考虑程序内部结构和处理过程的测试方法。黑盒子测试是对程序的接口进行测试，它只检验程序功能是否能够要找预定的那样正常的工作。

黑盒子测试力图发现以下问题：

1。功能是否不正确或者不符合要求

2。界面是否错误

3。数据结构错误或外部数据库访问错误

4。性能不符合

白盒测试也称结构测试或逻辑驱动测试，它是按照程序内部的结构测试程序，通过测试来检测产品内部动作是否按照设计规格说明书的规定正常进行，检验程序中的每条通路是否都能按预定要求正确工作。 这一方法是把测试对象看作一个打开的盒子，测试人员依据程序内部逻辑结构相关信息，设计或选择测试用例，对程序所有逻辑路径进行测试，通过在不同点检查程序的状态，确定实际的状态是否与预期的状态一致。

对于白盒子和黑盒子测试，他们两者是不能互相替代的，白盒子测试主要是早阶段进行的，黑盒子测试时后阶段进行的。

## 71.简述软件分析员在系统分析中的任务和作用。

系统分析员（system analyst）是对大型、复杂的信息系统建设任务中，承担分析、设计和领导实施的领军人物。 要做好与客户之间的关系，同时对客户的需求要正确的理解，要选择合适的开发技术，同时做好与客户间沟通交流，学会说服对方。

工作内容：

1、参与用户需求调研、负责系统体系结构、功能、性能的分析和总体设计工作；

2、负责项目的开发流程管理，进行项目的计划、管理、跟进工作；

3、参与并指导开发工程师完成系统详细设计和开发工作，解决相应业务、技术难题；

4、参与编制用户手册、协助客户的系统软件、硬件平台的安装实施工作；

5、制定项目文档格式，编写项目管理规范要求的相关文档

作用：系统分析工作是解决一个问题的工作，目**标是将一个对计算机应用系统的需求转化成实际的物理实现，**

## 72.软件的可维护性与哪些因素有关，如何提高软件的可维护性？

主要有5个：1可理解性2可测试性3可修改性4可移植性5可重用性

方法：

1. 建立明确的软件质量目标。
2. 采用现代化的开发方法。
3. 建立明确的质量保证审查。
4. 选择可维护的语言。
5. 改进程序员的文档。

## 73.事务型软件结构图有什么特点，原因是什么？

软件结构包括构成系统的设计元素的描述、设计元素之间的交互、设计元素的组合模式以及在这些模式中的约束。一个系统由一组构件以及它们之间的交互关系组成，这种系统本身又可以成为一个更大的系统的组成元素。

**软件结构图也就是为了反映软件系统中组件之间相互关系和约束的体系结构设计图，**

称为软件体系结构图更为合适，一般通过分层次或分时间段等方式说明体系结构的各个组成部分的组合关系。

在结构化设计方法中，软件结构图主要分为变换型软件结构图和事务型软件结构图两种。

事务流

基本系统模型意味着变换流，因此，原则上所有信息流都可以归结为这一类。但是当数据流是以**事务为中心**的，也就是说，数据沿输入通路到达一个处理T，这个处理根据输入数据的类型在若干个动作序列中选出一个来执行。这类数据流应该划为一类特殊的数据流，称为事务流。

## 75.Coad/Yourdon的OOA模型包括那几个方面？

Coad和Yourdon方法可分为面向对象分析（OOA）和面向对象设计（OOD）两部分。在OOA中，**建立了概念模型，由类与对象、属性、服务、结构和主题等5个分析层次组成**。

1）类与对象。从问题域、文字出发，寻找并且标识类与对象。

2）属性。确定对象信息及其之间的关系。可分为原子概念层的单个数据，和类结构中的公有属性与特定属性。

3）服务。标识消息连接和所有服务说明。

4）结构。标识类层次结构，确定类之间的整体部分结构与通用特定结构。

5）主题。主题是比结构更高层次的模块，它与相关类在一起，控制着系统的复杂度。

在Coad和Yourdon方法的系统设计阶段也采用了OOA阶段的5个层次来描述以下设计模型的4个部分。

1）问题域。

2）用户界面。包括问题域与用户、系统外部和专用设备、磁盘文件和数据管理界面等。

3）任务管理。包括任务的运行和任务的交互管理等。

4）数据库管理。包括文件管理和数据库管理等。

## 76.面向对象生存期模型与传统的生存期模型有什么区别?

软件生存期模型也称为过程模型，是从软件项目需求定义直至软件运行维护为止，跨越整个生命周期的系统开发、运行和维护所实施的全部过程、活动和任务的结构框架。

典型的生存期模型有瀑布模型、快速模型、增量模型、螺旋模型、形式化方法模型、基于组件的开发模型。

在整个软件开发过程中将面向对象技术贯穿于整个生存周期。

## 77.软件工程的文档分哪两大类，主要作用是什么？

软件文档可以分为开发文档和产品文档两大类。

开发文档包括：《功能要求》、《投标方案》、《需求分析》、《技术分析》、《系统分析》、《数据库文档》、《功能函数文档》、《界面文档》、《编译手册》、《QA文档》、《项目总结》等。

产品文档包括：《产品简介》、《产品演示》、《疑问解答》、《功能介绍》、 《技术白皮书》、《评测报告》、《安装手册》、《使用手册》、《维护手册》、 《用户报告》、《销售培训》等。

## 78.详细设计的任务是什么？

详细设计是软件工程中软件开发的一个步骤，就是对概要设计的一个细化，就是详细设计每个模块实现算法，所需的局部结构。

详细设计包括业务对象设计、功能逻辑设计、数据库设计和界面设计等工作。详细设计是系统实现的依据，需要考虑所有的设计细节。

详细设计的主要任务是设计每个模块的实现算法、所需的局部数据结构。详细设计的目标有两个：实现模块功能的算法要逻辑上正确和算法描述要简明易懂。

## 79.为什么把软件测试的目标定义为只是发现错误？

一个软件的成型和从业者的角色密不可分，软件测试者最主要的工作是提前发现软件中的种种缺陷，发现的问题提交给开发者来进行问题定位及修正，这样的分工才让这个工作流程更加便捷，毕竟术业有专攻。从这个层面上来说软件测试的目的简单的说是发现软件中的缺陷，专业的说是质量保证，只有通过测试人员的认可这个软件才可以上市或者发布。

如果从测试驱动开发的层面来说，测试人员不仅仅只是发现问题就了之了这么简单，测试人员要对发现的缺陷问题进行分类汇总分析及问题的初步定位，帮助开发人员更好的了解开发的软件本身的优势及缺陷，开发人员会根据测试数据分析合适的调整开发适配方案，从整个软件开发进程上说不定会少走很多弯路。

## 80.什么是对象关系模型和对象行为模型，有什么不同？

对象关系模型是描述对象与对象之间的关系的模型，

对象行为模型指的是对象与对象之间相互关系的行为模型

关系模型和对象模型存在概念上的阻抗不匹配，但是在关系数据库的存储模型上是一致的，无论你从关系模型的三大范式理论出发，还是从对象模型的ORM理论出发，最终一定会得到一致的数据库表设计。

## 81.软件工程是如何克服软件危机的？

**什么是软件危机**

落后的软件生产方式无法满足迅速增长的计算机软件需求，从而导致软件开发与维护过程中出现一系列严重问题的现象。

**产生原因：**

1、开发人员方面，对软件产品缺乏正确认识，没有真正理解软件产品是一个完整的配置组成。造成开发中制定计划盲目、编程草率，不考虑维护工作的必要性。

2、软件本身方面，对于计算机系统来说，软件是逻辑部件，软件开发过程没有统一的、公认的方法论和规范指导，造成软件维护困难。

3、尤其是随着软件规模越来越大,复杂程度越来越高,原有软件开发方式效率不高、质量不能保证、成本过高、研制周期不易估计、维护困难等一系列问题更为突出，技术的发展已经远远不能适应社会需求；

**解决途径：**

软件工程诞生于60年代末期，它作为一个新兴的工程学科，主要研究软件生产的客观规律性，建立与系统化软件生产有关的概念、原则、方法、技术和工具，指导和支持软件系统的生产活动，以期达到降低软件生产成本 、改进软件产品质量、提高软件生产率水平的目标。软件工程学从硬件工程和其他人类工程中吸收了许多成功的经验，明确提出了软件生命周期的模型，发展了许多软件开发与维护阶段适用的技术和方法，并应用于软件工程实践，取得良好的效果。

在软件开发过程中人们开始研制和使用软件工具，用以辅助进行软件项目管理与技术生产，人们还将软件生命周期各阶段使用的软件工具有机地集合成为一个整体，形成能够连续支持软件开发与维护全过程的集成化软件支援环境，以期从管理和技术两方面解决软件危机问题。

此外，人工智能与软件工程的结合成为80年代末期活跃的研究领域。基于程序变换、自动生成和可重用软件等软件新技术研究也已取得一定的进展，把程序设计自动化的进程向前推进一步。在软件工程理论的指导下，发达国家已经建立起较为完备的软件工业化生产体系，形成了强大的软件生产能力 。软件标准化与可重用性得到了工业界的高度重视，在避免重用劳动，缓解软件危机方面起到了重要作用。

## 82.软件开发主要由哪些方面风险，如何进行软件风险分析？

1、由于与客户沟通不畅对客户的需求了解不足造成的风险在软件开发项目整个生命周期的中都存在的风险，主要包括需求变更风险，涉及风险，过程风险，安装及维护风险。

2、由于管理人员素质不够，经验不足，沟通不畅，任务或其分配不合理，对项目的控制力度不够造成的各种风险，主要包括进度风险，预算风险，管理能力风险，信息安全风险。

3、由于技术力量不足，开发环境工具不足造成的。主要包括技术风险，质量风险，软件设计工具风险，软件开发工具风险，员工技能风险。

4、由于公司或项目组内外部环境变化所导致的风险，主要包括人力资源风险，政策风险，市场风险，营销风险。

软件项目中的风险永远不能全部消除，而只能采用避免、减轻、和接受三种因对策略。避免：通过分析找出发生风险事件的原因，消除这些原因来避免一些特定风险事件的发生。减轻：通过降低风险事件发生的概率或得失衡量来减轻风险对项目的影响，也可采用风险转移的方法来减轻风险对项目的影响。接受：对于一些无法避免的风险，应当接收风险造成的后果或者提前设计相应的应对措施，但这需要一定的资金做后盾。

1、风险标识，2、风险估算，3、风险评价和管理。

## 83.程序的编码风格主要体现在哪几个方面？

程序的编码风格即为程序设计风格或编程风格，其主要作用是使无论是程序本人还是其他人，都能比较容易的阅读、理解及修改源代码；

主要表现在四个方面：源程序文档化、数据说明方法，表达式和语句结构及输入/输出方法；

## 84.简述软件测试中桩模块和驱动模块的作用。

桩模块（Stub）是指模拟被测试的模块所调用的模块，而不是软件产品的组成的部分。主模块作为驱动模块,与之直接相连的模块用桩模块代替。

**桩模块的作用：**在集成测试前要为被测模块编制一些模拟其下级模块功能的“替身”模块，以代替被测模块的接口，接受或传递被测模块的数据，这些专供测试用的“假”模块称为被测模块的桩模块。

传统的单元测试术语（unit testing terminology），包括了驱动模块（driver） 和桩模块（stub）。驱动模块是用来模拟被测试模块的上一级模块，相当于被测模块的主程序。它接收数据，将相关数据传送给被测模块，启用被测模块，并打印出相应的结果。

**作用：**驱动模块的目的很单纯，就是为了访问类库的属性和方法，来检测类库的功能是否正确；

## 85.如何从需求文档中识别类和对象

## 86.软件危机主要有哪些表现?

1、软件需求增长得不到满足

2、软件生产高成本，价格昂贵。

　 3、软件开发费用和进度失控。

　　4、软件需求定义不准确，易偏离用户需求。

　　5、软件质量不易保证、可靠性差。

　　6、软件可维护性差。

## 87.什么是软件维护的副作用，如何防止软件维护的副作用？

软件维护的副作用主要是在修该原有错误时会引入新的错误。

预防的措施主要是程序模块化，信息隐蔽，高内聚、低耦合等。

面向对象分析设计中使用封装、抽象、继承等办法。具体的包括：

1、按模块把修改分组；

2、自顶向下地分析所修改模块的顺序；自底向上修改有关模块。

3、每次修改一个模块；牵涉到全局性的基础模块改动必须仔细分析。

4、对于每个修改了的模块，在安排修改下一个模块之前，要确定这个修改的副作用，主要是关注修改后对调用这个模块的其他模块的功能影响。

## 88.好的软件体系结构设计应遵循哪些原则？

软件系统的体系结构设计的原则是满足合适性、结构稳定性、可扩展性、可复用性

## 89.简述渐增式测试方法与非渐增式测试方法的优劣。

渐增式测试是先从单个模块开始测试，然后每次将测试后的一个模块添加到系统中来进行下一个测试的方法。

非渐增式测试，是指先独立地测试每一模块，然后将所有这些模块连接到一起运行；一下子把所有模块放在一起，并把庞大的程序作为一个整体来测试，测试者面对的情况十分复杂。

**非渐增式与渐增式两种测试的区别**

非渐增式测试方法需要编写的测试用例较多，工作量较大；渐增式测试方法开销小。

渐增式测试方法发现模块间接口错误早；而非渐增式测试方法晚。

非渐增式测试方法发现错误，较难诊断；而使用渐增式测试方法，如果发生错误则往往和最近加进来的那个模块有关。

渐增式测试方法测试更彻底

渐增式测试方法需要较多的机器时间

使用非渐增式测试方法，可以并行测试。

## 90.UML中有哪些动态建模的工具，各表达什么内容？

三种：1、ERD 实体关系图；2、DFD 数据流图；3、STD 状态迁移图。

1、ERD：E-R Diagram（Entity-Relationship Diagram），ER图，即实体关系图；它主要是描述的数据对象间的关系。是数据建模活动的记号。就是用图形来说明一个系统实体之间的关系。

2、数据流图（Data Flow Diagram）：简称DFD，它从数据传递和加工角度，以图形方式来表达系统的逻辑功能、数据在系统内部的逻辑流向和逻辑变换过程，是结构化系统分析方法的主要表达工具及用于表示软件模型的一种图示方法。

3、状态—迁移图(STD)或状态—迁移表被用来描述系统或对象的状态,以及导致系统或对象的状态改变的事件,从而描述系统的行为.属于结构化分析方法使用工具。

## 91.软件技术审查和管理复审的作用是什么？

1、由一组评审者按照规范的步骤对软件需求、设计、代码或其他技术文档进行仔细地检查，以找出和消除其中的缺陷。

作用：

（1）发现软件在功能、逻辑、实现上的错误；

（2）验证软件符合它的需求规格；

（3）确认软件符合预先定义的开发规范和标准；

（4）保证软件在统一的模式下进行开发；

（5）便于项目管理。

2、管理复审是指系统地评价鉴定全部管理工作的一种控制方法，它侧重于管理决策方面的审核，包括对组织中关于计划、工程技术、生产、营销、人事、会计以及财务方面的审核。

管理复审的主要任务是，在软件生命周期的每个阶段结束时，对工程项目的成本、实际花费的经费、投资回收的前景、项目的进度等经济因素从管理角度进行审查。从而找出在成本、进度、质量上存在的问题，为后续改进提供问题基础。

## 92.软件的可维护性与那些软件质量特性有关？

软件可维护性的定义：软件能够被理解、校正、适应及增强功能的容易程度。

软件可维护性可用下面七个质量特性来衡量，即可理解性、可测试性、可修改性、可靠性、可移植性、可使用性和效率。对于不同类型的维护,这七种特性的侧重点也是不相同。

## 93.什么是模块独立性，如何衡量模块独立性？

模块独立性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.

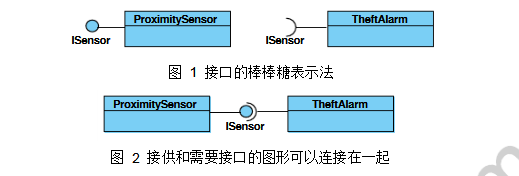
模块独立性是指模块内部各部分及模块间的关系的一种衡量标准，由内聚和耦合来度量。

## 95.UML中的接口类有什么特点，举例说明。

**接口描述了一组实现这一接口的分类器所必须共同遵守的合同，这种合同包括结构上的也包括行为上的，**此外，接口是不能实例化的。接口实现则是一种依赖关系，其描述了接口与分类器之间的一种特殊的实现关系。

图1示例了接口的棒棒糖表示方法，其中“棒棒糖”表示与之相连的类提相应的接口，也就是说相关联的类实现了这一接口。

从图中我们可以看出，ProximitySensor类实现了ISensor接口，而TheftAlarm类则需要一个ISensor接口。由于两个类一个提供，而另一个需要，因此我们可以将其连接在一起，如图2所示。



## 97.CMM与ISO9000有什么异同

1、CMM是指“能力成熟度模型”，它是对于软件组织在定义、实施、度量、控制和改善其软件过程的实践中各个发展阶段的描述。CMM的核心是把软件开发视为一个过程，并根据这一原则对软件开发和维护进行过程监控和研究，以使其更加科学化、标准化、使企业能够更好地实现商业目标。

2、ISO9000是指质量管理体系标准，它不是指一个标准，而是一种标准的统称。

**ISO与CMM的联系及区别**

**联系:**两者都共同着眼于质量和过程管理，目前2000版的ISO更多的和CMMI有直接对应的关系,甚至是大量的CMMI4和CMMI5级的要求.

**区别:** CMM是专门针对软件产品开发和服务,而ISO9000涉及的范围则相当宽。

CMM强调软件开发过程的成熟度,即过程的不断改进和提高.而ISO9000则强调可接收的质量体系的最低标准.

## 98.什么是耦合，如何衡量模块的耦合度？

耦合是程序结构中各个模块之间相互关联的度量。

它取决于各个模块之间的接口的复杂程度、调用模块的方式以及哪些信息通过接口

## 99.软件测试分几种类型，主要解决什么问题？

参照94题

# 四、论述题（20题）

## 1.论述面向对象方法对传统方法的优势和问题

答：面向对象方法是一种把面向对象的思想应用于软件开发过程中，指导开发活动的系统方法，简称OO (Object-Oriented)方法，是建立在“对象”概念基础上的方法学。对比传统方法，它的优点是：所开发的程序是面向对象程序，直接描述客观世界的对象及其相互关系；具有模块性、继承性和类比性；**可广泛支持对象的重用**；十分有利于开发过程的质量控制；**极大地减小了系统维护的复杂程度**。缺点是：面对大型系统时，若缺乏整体系统设计划分，易造成系统结构不合理、各部分关系失调等问题；**只能在现有业务基础上进行分类整理，不能从科学管理角度进行理顺和优化**；需要一定的软件支持环境。

## 2.什么是面向对象开发过程，讨论各阶段任务和要点

答：面向对象开发过程就是指利用面向对象方法进行的软件开发的完整流程，主要分为四个阶段：分析、设计、编程、测试。面向对象分析包括需求分析和需求模型化两个部分。其主要作用是明确用户的需求，并用标准化的面向对象模型来规范地表达这一需求，最后形成面向对象的分析模型。它是系统设计的依据。面向对象设计的任务是把分析阶段得到的问题**域需求转变成符合要求的、抽象的系统实现方案**，主要分为两个阶段：**系统设计和对象设计。**要点设计针对问题域子系统、人-机交互子系统、任务管理子系统、数据管理子系统、类中的服务和关联。编程和测试则分别是由开发人员编写程序具体代码并在系统代码开发完成后对系统整体性能和功能进行测试。

## 3.用覆盖的观点讨论面向对象的软件测试策略

答：软件测试覆盖包括分支覆盖，语句覆盖、条件覆盖等。

语句覆盖：它要求被测程序的每一可执行语句在测试中尽可能都检验过;

分支覆盖：要求程序中所有判定的分支尽可能得到检验;

条件覆盖：当判定式中含有多个条件时，要求每个条件的取值均得到检验；

还有其他的覆盖方式则是将这三种覆盖进行组合，使得测试覆盖更加全面，可以覆盖到程序中所有可能的路径。由此可以看出只有对程序内部十分了解才能进行适度有效的覆盖测试。但是贯穿在程序内部的逻辑存在着不确定性和无穷性，尤其对于大规模复杂软件。因此我们不能穷举所有的逻辑路径，即使穷举也未必会带来好运（穷举不能查出程序逻辑规则错误，不能查出数据相关错误，不能查出程序遗漏的路径）。所以正确使用覆盖（白盒）测试，就要先从代码分析入手，根据不同的代码逻辑规则、语句执行情况，选用适合的覆盖方法。任何一个高效的测试用例，都是针对具体测试场景的。逻辑测试不是片面的测试正确的结果或是测试错误的结果，而是尽可能全面地覆盖每一个逻辑路径。这样才能使得测试的工作量与准确性达到一个合理的平衡。

## 4.结合软件工程要素，论述面向对象方法的思想

答：软件工程的方法（开发方法）、工具（支持方法的工具）、过程（管理过程）称为软件工程的三要素。方法支撑过程和工具；过程和工具促进方法学的研究。

面向对象方法（Ohject--Oriented Method，OOM）是一种运用对象、类、消息传递、继承、封装、聚合、多态性等概念来构造软件系统的软件开发方法。

它的优点是：所开发的程序是面向对象程序，直接描述客观世界的对象及其相互关系；具有模块性、继承性和类比性；可广泛支持对象的重用；十分有利于开发过程的质量控制；极大地减小了系统维护的复杂程度。

缺点是：面对大型系统时，若缺乏整体系统设计划分，易造成系统结构不合理、各部分关系失调等问题；只能在现有业务基础上进行分类整理，较难从科学管理角度进行理顺和优化；需要一定的软件支持环境。

因此，通过三要素的关系可以看出，面向对象方法决定了它需要一定的开发工具和严格合理的管理过程来支持，同时，在这两个条件得到满足的情况下，面向对象的方法将能够极大的方便、优化工程的开发以及管理维护。

## 5.论述需求工程过程，说明各阶段关系

答：需求工程过程包括需求开发和需求管理两个部分，需求开发的一般过程分为需求获取、需求分析、编写需求规格说明书（SRS）、需求验证四个阶段；

需求管理则主要包括需求基线的建立、需求变更控制以及需求跟踪等活动。需求获取是通过与用户的交流，对现有系统的观察及对任务进行分析，从而开发、捕获和修订用户的需求；需求分析是通过之前需求获取所得到的信息，为最终用户所看到的系统建立一个概念模型，作为对需求的抽象描述；由需求模型构件生成精确的形式化的描述，即需求规格说明书，它将作为用户和开发者之间的一致协议；

需求分析的结果应该通过评审、测试等手段验证它的正确性、完整性和一致性，这就是需求验证。

贯穿于整个过程中，需求管理中最基本的任务则是明确需求，并使所有相关人员达成共识；建立需求跟踪能力联系链，确保所有用户需求被正确地应用，并且在需求发生变更时，能够完全地控制其影响范围，始终保持产品与需求的一致性。

## 6.什么是软件可维护性，讨论软件工程与可维护性关系

答：软件可维护性即维护人员对该软件进行维护的难易程度,具体包括理解、改正、改动和改进该软件的难易程度。

软件工程是一门研究用工程化方法构建和维护有效的、实用的和高质量的软件的学科。

软件工程的目标是：在给定成本、进度的前提下，开发出具有适用性、有效性、可修改性、可靠性、可理解性、可维护性、可重用性、可移植性、可追踪性、可互操作性和满足用户需求的软件产品。软件交付使用后，需要能够对它进行修改，以改正潜伏的错误，改进性能和其它属性，使软件产品适应环境的变化等。软件维护费用在软件开发费用中占有很大的比重。因此，**可维护性是软件工程中一项十分重要的目标**。

## 7.论述软件工程中的文档类型和作用

答：按照文档产生和使用的范围，软件文档大致可分为三类：开发文档、管理文档、用户文档。

开发文档：这类文档是在软件开发过程中，作为软件开发人员前一阶段工作成果的体现和后一阶段工作依据的文档。包括软件需求说明书、数据要求说明书、概要设计说明书、详细设计说明书、可行性研究报告、项目开发计划。

管理文档：这类文档是在软件开发过程中，由软件开发人员制定的需提交人员的一些工作计划或工作报告。使管理人员能够通过这些文档了解软件开发项目安排、进度、资源使用和成果等。包括项目开发计划、测试计划、测试报告、开发进度月报及项目开发总结。

用户文档：这类文档是软件开发人员为用户准备的有关该软件使用、操作、维护的资料。包括用户手册、操作手册、维护修改建议、软件需求说明书。

## 8.论述需求工程的过程和方法，讨论其有效性

需求工程包括需求开发和管理，而需求开发又包括这几个过程：需求获取，需求分析，需求规格说明和需求验证。

在需求开发之前，还需要有一个知识培训的过程，需求工程也是一个项目工程，因此也包括了项目的管理。

对于这些过程，有以下方法可以采用。

1、知识培训：需求分析员培训、用户培训、开发人员培训；

2、需求获取：项目范围确定、用户确定、用例确定、系统事件和响应、获取方法、检查完善；

3、需求分析：绘制关联图、原型开发、需求优先顺序别、需求建模、数据字典创建、子系统。

在知识培训中，通过对三方的培训，可以使分析员提高沟通能力，理解并掌握需求工程技能，使用户明白需求重要性，能够准确描述需求，使开发人员对用户应用有所了解。

通过需求获取中的方法，可以明确项目目标、确定用户群，通过用例迅速明确项目功能需求，这些可以通过会议讨论，观察用户工作过程，问答式对话，诱发式需求诱导等实现。

而在需求分析阶段则是对用户的需求获取之后的一个粗加工过程，需要对需求进行推敲和润色以使所有涉众都能准确理解需求。 分析过程首先需要对需求进行检查，以保证需求的正确性和完备性，然后将高层需求分解成具体的细节，创建开发原型，完成需求从需求获取人员到开发人员的过渡。

## 9.讨论软件测试与软件可靠性的关系，如何计算可靠性

答：软件测试的目的是发现软件错误，排除软件缺陷，提升软件可靠性。软件可靠性可定义为在规定的条件下，在规定的时间内，软件不引起系统失效的概率或在规定的时间周期内，在所述条件下程序执行所要求的功能的能力。

据此，可以在测试时间内，对软件各个模块进行测试统计，得到每个功能块的可靠性，功能串联时可靠性R=R1\*R2\*•••RN，并联时R=1-（1-R1）（1-R2）•••（1-RN）。混联则综合两式进行计算。

## 10.什么是软件可测试性，讨论软件工程与软件可测试性

答：软件的可测试性是指软件**发现故障并隔离、定位其故障的能力特性**，以及在一定的时间和成本前提下，进行测试设计、测试执行的能力。软件可测试性就是一个计算机程序能够被测试的容易程度。软件工程是一门研究用工程化方法构建和维护有效的、实用的和高质量的软件的学科。

软件工程的目标是：在给定成本、进度的前提下，开发出具有适用性、有效性、可修改性、可靠性、可理解性、可维护性、可重用性、可移植性、可追踪性、可互操作性和满足用户需求的软件产品。

而软件在完成后，必须保证其能够防止因概念、设计和结构等方面的不完善造成的软件系统失效，具有挽回因操作不当造成软件系统失效的能力，也就是要达成软件可靠性。软件测试正是保证软件可靠性的一项非常重要的技术保障。因此，在软件工程中，必须在软件设计时就考虑好软件的可测试性，这样才能在软件完成后通过测试，调整改进软件的不足，满足其可靠性要求

## 11.论述软件测试的基本任务，方法和策略及其在现代软件工程的地位

测试的目的（来源老师PPT）：

1）.测试是为了证明程序有错，而不是证明程序无错误；

2）.一个好的测试用例在于能够发现至今未发现的错误；

3）.一个成功的测试是发现了至今未发现的错误的测试。

测试的方法：

1）测试方法概述

2）白盒测试方法

3）黑盒测试方法

4）其他测试方法

测试的策略：

按阶段进行测试是一种基本的测试策略

1）单元测试

2）集成测试

3）系统测试

4）确认测试

软件测试在软件工程中的地位：覆盖整个生命周期。从需求阶段已介入，编写测试计划、测试用例、测试总结等。

## 12.论述软件测试与软件质量的关系

软件质量是指软件产品的特性可以满足用户的功能、性能需求的能力。软件过程是人们通常所说的软件生命周期中的活动，一般包括软件需求分析、软件设计、软件编码、软件测试、交付、安装和软件维护。随着软件过程的开始，软件质量也逐渐建立起来。软件过程的优劣决定了软件质量的高低，好的过程是高效高质量的前提。人员和过程是决定软件质量的关键因素。高质量的人员和好的过程应该得到好的产品。

软件系统的开发包括一系列生产活动，其中由人带来的错误因素非常多，错误可能出现在程序的最初需求分析阶段，设计目标可能是错误的或描述不完整，也可能在后期的设计和开发阶段，因为人员之间的交流不够，交流上有误解或者根本不进行交流，所以尽管人们在开发软件的过程中使用了许多保证软件质量的方法和技术，单开发出的软件中还会隐藏许多错误和缺陷。**可见，只有通过严格的软件测试，才能很好的提高软件质量，而软件质量并不是依靠软件测试来保证的，软件的质量要靠不断的提高技术水平和改进软件开发过程来保证，软件测试只是一种有效的提高软件质量的技术手段，而不是软件质量的安全网。**

软件测试能够找出软件缺陷，确保软件产品满足需求。但是测试不是质量保证，二者并不等同。测试可以查找错误并进行修改，从而提高软件产品的质量。软件测试避免错误以求高质量，并且还有其他方面的措施以保证质量问题,如软件质量保证。

正规的软件测试系统主要包括：制定测试计划、测试设计、实施测试、建立和更新测试文档。而软件质量保证的工作主要为：制定软件质量要求、组织正式度量、软件测试管理、对软件的变更进行控制、对软件质量进行度量、对软件质量情况及时记录和报告。软件质量保证的职能是向管理层提供正确的可行信息，从而促进和辅助设计流程的改进。软件质量保证的职能还包括监督测试流程，这样测试工作就可以被客观地审查和评估，同时也有助于测试流程的改进。**二者的不同之处在于软件质量保证工作侧重对软件开发流程中的各个过程进行管理与控制，杜绝软件缺陷的产生。而测试则是对已产生的软件缺陷进行修复。**

## 14.针对敏捷方法的十二条核心实践，结合软件工程问题论述其有效性

PPT参考答案：

1. On-Site Customer （现场客户） : 客户是Team成员，在开发现场和开发人员一起工作。

2. 计划项目 （Planning Game）

3. 频繁地小规模发布软件（Small Releases）

4. 简单设计 （Simple Design）

5. 测试驱动开发 （Test Driven Development）

6. 持续集成 （Continuous Integration）

7. 集体拥有代码（Collective Code Ownership）

8. 编程规范（Coding Standards）

9. 重构（Refactoring）

10. System Metaphor（系统隐喻）

11. Pair Programming（结对编程）

12. 平稳的工作效率（Sustainable Pace）

敏捷开发中有12条原则，它们是敏捷实践区别于重型过程的特征所在。

1.我们最优先要做的是通过尽早的、持续的交付有价值的软件来使客户满意。

2.即使到了开发的后期，也欢迎改变需求。敏捷过程利用变化来为客户创造竞争优势。

3.经常性的交付可以工作的软件，交付的间隔可以从几周到几个月，交付的时间间隔越短越好。

4.在整个项目开发期间，业务人员和开发人员必须天天都在一起工作。

5.围绕被激励起来的人来构建项目。给他们提供所需要的环境和支持，并且信任他们能够完成工作。

6.在团队内部，最具有效果并且富有效率的传递信息的方法，就是面对面的交谈。

7.工作的软件是首要进度度量标准。

8.敏捷过程提可持续的开发速度。责任人、开发者和用户应该能够保持一个长期的、恒定的开发速度。

9.不断地关注优秀的技能和好的设计会增强敏捷能力。

10.简单——使未完成的工作最大化的艺术——是根本的。

11.最好的构架、需求和设计出自与自组织的团队。

12.每隔一定时间，团队会在如何才能更有效地工作方面进行反省，然后相应地对自己的行为进行调整。

## 15.针对软件危机，论述传统软件工程方法和敏捷方法的解决思路

PPT参考：

软件危机出现于20 世纪60 年代末，软件危机是指在计算机软件的开发和维护过程中遇到的一系列严重问题。比如：

1）软件开发的成本和进度难以准确估计，延迟交付甚至取消项目的现象屡见不鲜

2）软件存在着错误多、性能低、不可靠、不安全等质量问题

3）软件维护极其困难，而且很难适应不断变化的用户需求

为了解决“软件危机”，许多计算机和软件科学家参照技术过程的一般模式提出了软件生存期概念（Life cycle）及其瀑布模型（Waterfall Model）。计算机软件的生存周期描述了软件孕育、诞生、成长、成熟、衰亡的生存过程：

制定计划、需求分析、软件设计、程序编制、软件测试、运行维护。

但是，传统软件工程方法推行多年以来，人们对于软件危机的“恐惧”仍没有丝毫减弱，相反随着软件系统的急速膨胀而增强，表现为：对软件开发成本和进度的估计常常不准确，开发成本超出预算，实际进度比预定计划一再拖延的现象并不罕见；用户对“已完成”系统不满意的现象也经常发生；软件产品的质量往往靠不住，Bug一大堆，补丁一个接一个等等。于是，无论是产业界还是理论界，都开始对传统软件工程学思想产生怀疑，甚至背叛，**于是以“人”为核心的敏捷开发方法就是在这样的背景下产生的**，它背叛了传统软件工程学中以“过程”为核心，把设计和开发尽可能分开，尽量弱化“人”在整个工程中地位的思想。

敏捷开发实际上包括了许多优秀的软件开发习惯。首先，这种方法改变了软件测试的流程，在编写代码前进行测试，减少了开发风险；此外，通过这种方法，可以对软件进行持续集成，即每个小时都在集成，任何部件间的冲突都可以随时解决；这种方法的“动态规划”和“重构”做法，意味着开发者可以对软件架构进行持续改进，可以根据用户的需求随时进行改进，而利用传统的软件开发方法则很难对软件的架构进行调整，同时也会造成成本的大幅增加。

## 16.论述过程、方法和工具在软件工程实践中的关系

软件工程过程是软件系统、产品的定义、设计、实现以及维护的过程。

软件工程方法包括系统工程、系统分析方法和建模、系统概要设计、 软件体系结构、用户界面、 构件级设计方法。

## 17.针对不同软件方法，论述项目管理要点

软件研究人员也在不断探索新的软件开发方法。至今已形成了八类软件开发方法.

一、Parnas方法

此方法在1972年提出的。主要针对软件在可维护性和可靠性方面存在着严重问题，主要提出了信息隐蔽原则：在概要设计时列出将来可能发生变化的因素，并在模块划分时将这些因素放到个别模块的内部。第二条原则是在软件设计时应对可能发生的种种意外故障采取措施。软件是很脆弱的，很可能因为一个微小的错误而引发严重的事故，所以必须加强防范。。

二、SASA方法

1978年，**提出了结构化方法**，即SASD方法，也可称为面向功能的软件开发方法或面向数据流的软件开发方法，首先用结构化分析（SA）对软件进行需求分析，然后用结构化设计（SD）方法进行总体设计，最后是结构化编程（SP）。

三、**面向数据结构的**软件开发方法

1、Jackson方法 ：从目标系统的输入、输出数据结构入手，导出程序框架结构，再补充其它细节，就可得到完整的程序结构图。此方法有时也称为面向数据结构的软件设计方法。

2、Warnier方法

与Jackson方法类似。差别有三点：一是它们使用的图形工具不同，分别使用Warnier图和Jackson图；另一个差别是使用的伪码不同；最主要的差别是在构造程序框架时，Warnier方法仅考虑输入数据结构，而Jackson方法不仅考虑输入数据结构，而且还考虑输出数据结构。

四、问题分析法

基本思想是：考虑到输入、输出数据结构，指导系统的分解，在系统分析指导下逐步综合。这一方法的具体步骤是：

从输入、输出数据结构导出基本处理框；分析这些处理框之间的先后关系； 按先后关系逐步综合处理框，直到画出整个系统的PAD图。PAM方法的另一个优点是使用PAD图。这是一种二维树形结构图，是到目前为止最好的详细设计表示方法之一，远远优于NS图和PDL语言。

五、面向对象的软件开发方法

随着OOP（面向对象编程）向OOD（面向对象设计）和OOA（面向对象分析）的发展，最终形成面向对象的软件开发方法OMT(Object Modelling Technique）。这是一种自底向上和自顶向下相结合的方法，而且它以对象建模为基础，从而不仅考虑了输入、输出数据结构，实际上也包含了所有对象的数据结构。

六、可视化开发方法

用户界面在软件系统中所占的比例也越来越大，有的甚至高达60～70%。产生这一问题的原因是图形界面元素的生成很不方便。可视化开发就是在可视开发工具提供的图形用户界面上，通过操作界面元素，诸如菜单、按钮、对话框、编辑框、单选框、复选框、列表框和滚动条等，由可视开发工具自动生成应用软件。

七、ICASE

提高人类的劳动生产率，提高生产的自动化程度，一直是人类坚持不懈的追求目标。软件开发也不例外。

八、软件重用和组件连接

软件重用（Reuse）又称软件复用或软件再用。软件人工制品可以是源代码片断、子系统的设计结构、模块的详细设计、文档和某一方面的规范说明等。所以软件重用是利用已有的软件成份来构造新的软件。它可以大大减少软件开发所需的费用和时间，且有利于提高软件的可维护性和可靠性。

## 18.论述软件项目特点和管理要点

软件项目的特点包括：发展快，经验失效快，复杂度高，主要依靠人的脑力劳动，过程的可见性差，结果难于测试，市场模式不完善。

项目管理的基本内容人力资源管理信息化项目是知识密集型的项目，与项目组人力资源关系密切，项目组成员的构成、责任心、能力和稳定性对信息化项目的质量以及是否成功有决定性的影响。项目主管应有效地使用人力资源，明确主要开发人员的职责和任务，提前做好有关人员力量的搭配，尽量使各个工程阶段中人员的变动不要太大。

成本管理首先，对项目按系统、子系统以及生命周期分解，分别估算出各个子系统在各个阶段的成本，然后再把这些成本汇总，估算出整个项目的成本。在实施过程中，对实际的费用进行控制，使项目成本不至于超出预算。

质量管理信息化项目的质量管理不仅仅是项目完成后的检查，而应包括在信息系统实施过程中的全面质量控制，保证在整个开发过程中各个阶段性工作成果质量。项目小组应组织有关的管理者和员工对阶段性工作成果进行审查和测试，这又称为里程碑式审查。通过保证各阶段性成果的质量，最终保证整体系统的质量。

知识资产管理 在信息化项目实施过程中，不可避免地要将许多工作成果制成文档或电子化文档。为了获得高质量的文档，需要将此工作制度化、规范化、标准化。另外，这些工作成果中牵涉到企业的知识资产和业务信息。必须设置专人保管，并建立借阅制度，防范这些工作成果和业务信息流出企业。

风险管理信息化项目涉及到组织变革的风险，经济风险，心理风险等。在项目推进过程中应提前对这些风险有所认识，一旦有关问题发生时，可从容冷静地处理解决。

沟通和范围管理 信息化项目经常涉及到与多个公司的合作，例如，将部分工程委托给其他公司实施。在合作过程中，一定要将各自的工作范围和责任范围书面化，并不断进行信息交换和阶段性成果的确认。进度管理 在实际的信息化项目推进中往往有各种各样的变动，完全按照计划日程推进工程是不现实的。当项目的进度计划需要调整时，调整的重点应放在对近期内即将发生的活动加强控制；或者对工期估计最长或预算估计最大的活动进行调整。

## 19.论述软件估算的困难和解决方法

软件成本估算是对将要开发的或正在开发的软件项目所需要的工作量和工作进度作出预测，从而产生出一组在可接受误差范围内的近似规划。

成本估算的困难：

1） 需求信息的复杂性：与其他有些传统项目不同，信息系统要满足的人的主观需要。由于人的复杂性，给信息系统带来了无数的难以确定的因素。而且，随着项目的进展，许多具体情况的明确，项目的成本估算也会相应的有所变化。

2）开发技术和工具的不断变化：开发工具软件的不断升级，技术方案不断更新，这些技术的进步让信息系统项目可以提供功能越来越强，但是给信息系统项目的成本估算带来困难。

3）缺乏类似项目估算数据可供参考：有效的项目成本估算是建立在大量的同类项目的成本估算的基础上的。没有大量的同类项目的经验，信息系统项目的成本估算也就非常困难了。

4）缺乏专业和富有经验的人才。

5）信息系统研发人员技术能力的差异：不同人员的不同态度、经验和能力都会造成不同人员的截然不同的效率，这也给信息系统的成本估算带来了困难。

6.）管理层的压力与误解。

成本估算的方法：

1) 工作分解结构方法（WBS）

2) 专家判断

3) 类比估算

4) 其他估算方法：德尔菲法(Delphi technique)、COCOMO模型、特征点(feature point)、对象点(object point)、3-D功能点(3-D function points，数据、功能、控制)、Bang度量(DeMarco‘s bang metric)、模糊逻辑(fuzzy logic)、标准构件法(standard component)等

5) 基于问题的估算

## 20.什么是软件性能，论述软件开发中如何提高软件的性能

软件的性能是软件的一种非功能特性，它关注的不是软件是否能够完成特定的功能，而是在完成该功能时展示出来的及时性。由于感受软件性能的主体是人，不同的人对于同样的软件能有不同的主观感受，而且不同的人对于软件性能关心的视角也不同。包括系统响应时间和应用延迟时间、吞吐量、并发用户数、资源利用率。

# 五、应用题（20题）

## 1.假设一家工厂的采购部每天需要一张定货报表，报表按零件编号排序，表中列出所有需要再次定货的零件。对于每个需要再次定货的零件应该列出下述数据：零件编号，零件名称，定货数量，当前价格，主要供应者，次要供应者。零件入库或出库称为事务，通过放在仓库中的显示终端把事务报告给定货系统。当某种零件的库存数量少于库存量临界值时就应该再次定货。

要求：（1）画出系统的数据流图（15分）；

*答: 没有指明要哪一级别的数据流图, 那就按照题目描述中的颗粒度来画.*

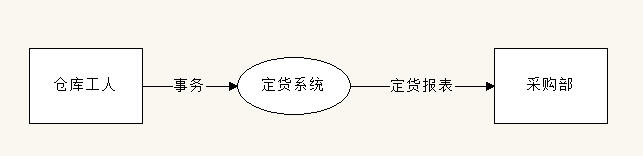
*为了方便理解, 先把顶层数据流图画出来.*

*顶层数据流图的样子, 就是一群人吃火锅. 图里面只有三个元素: 人, 火锅, 人与火锅的关系.*

*火锅: 就是软件, 写上XX系统. XX在题目中找. 比如本题目中, 就是定货系统.*

*人: 就是软件的使用者. 具体有哪些人, 在题目中找. 比如本题中, 有采购部和仓库工人.*

*人与火锅的关系: 就是使用者和软件之间的数据交换, 他们是有方向的. 仓库工人与软件的关系, 就是仓库工人报告事务. 采购部与软件的关系, 就是采购部拿到定货报表.*

**

*然后把中间的火锅展开, 就形成了0层数据流图:*

*具体展开办法, 就是根据各个使用者, 在题目中寻找他们办事的步骤:*

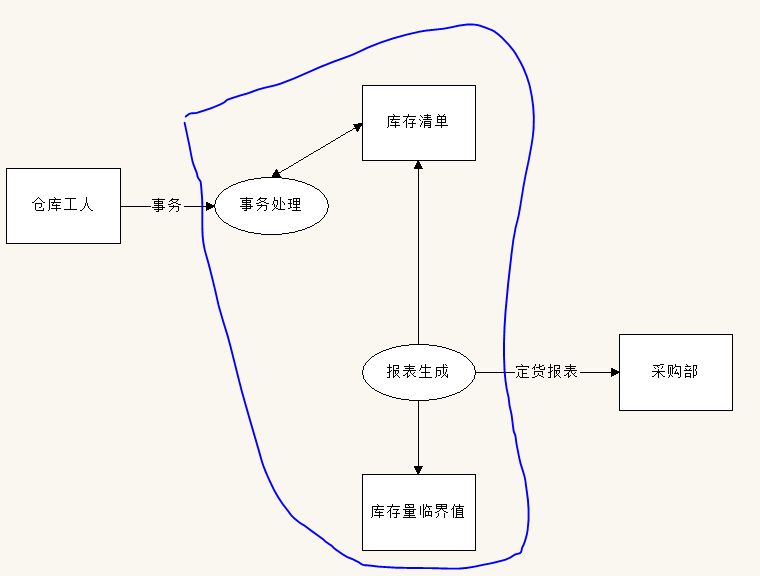
*比如仓库工人, 他要发送事务给定货系统, 但是具体发送给哪个东西呢, 要在题目中找. 题目中找不到, 那就认为发给了一个叫[事务处理]的步骤(如果题目中能找到, 那就使用题目中给的名字).*

*这个[事务处理], 要干的事情就是维护[库存清单], 也就是某某东西还有多少个.*

*这个[库存清单], 是一个需要保存的数据文件, 称为存储文件.*

*这个[事务处理], 要读写[库存清单].*

*比如采购部, 他要从系统中拿定货报表. 题目中也没有说是哪个步骤提供了定货报表, 那就把那个步骤定义为[报表生成]. 这个[报表生成], 根据题目的说法, 要读取库存清单, 还要读取库存量临界值, 然后才能做出一个报表. 也就是说, 这个步骤, 要读取[库存清单]和[库存量临界值]两个存储文件.*

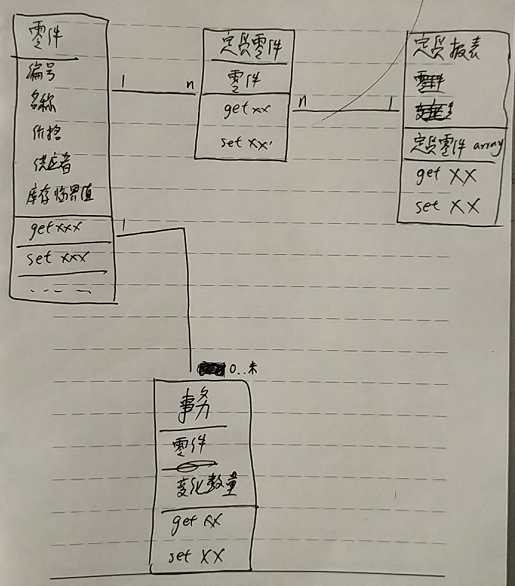
**

*蓝色框中的部分, 就是顶层数据流图中火锅的展开.*

（2）如果采用面向对象分析方法，请识别可能的类，画出类图，说明可能的属性和操作（15分）。

答:

识别类的方法, 就是寻找题目中的名词作为类, 其动词作为方法, 至于属性, 则根据上下文自己判断:



注意, 重数n代表数目不定, 具体是0..\*还是1..\*呢, 并不明确.

考试的时候, 如果你能确定, 那就写清楚了. 如果不确定0行不行, 那就写n

其中每一个属性都有对应的get和set方法, 就略去不写了, 考试的时候要加上.

## 2.现在有一个医院病房监护系统，用户提出的系统功能要求如下：

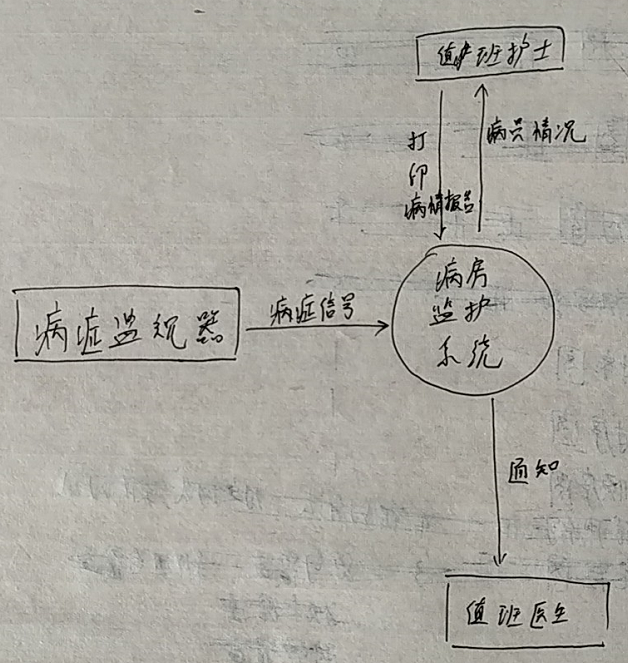
在医院病房监护系统中，病症监视器安置在每个病房，将病人的病症信号实时传送到中央监视系统进行分析处理。

在中心值班室里，值班护士使用中央监视系统对病员的情况进行监控，根据医生的要求随时打印病人的病情报告，系统会定期自动更新病历。

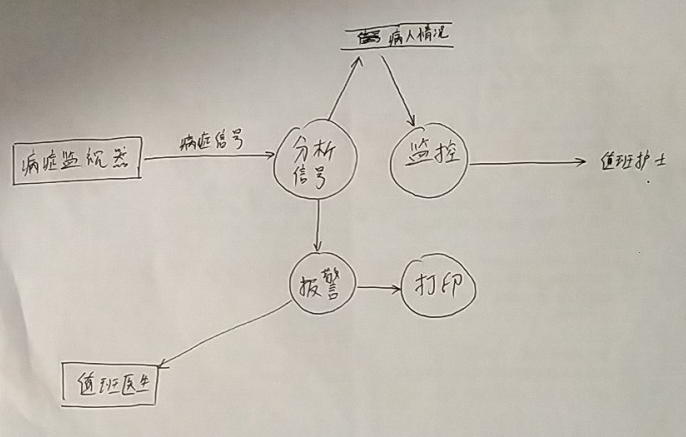
当病症出现异常时，系统会立即自动报警，通知值班医生及时进行处理，同时立即打印病人的病情报告和更新病历。

要求：（1）请画出数据流图（10分），并采用UML建模语言画出用例图（10分）。

*答: 顶层数据流图(实际考试直接画0层就可以了):*

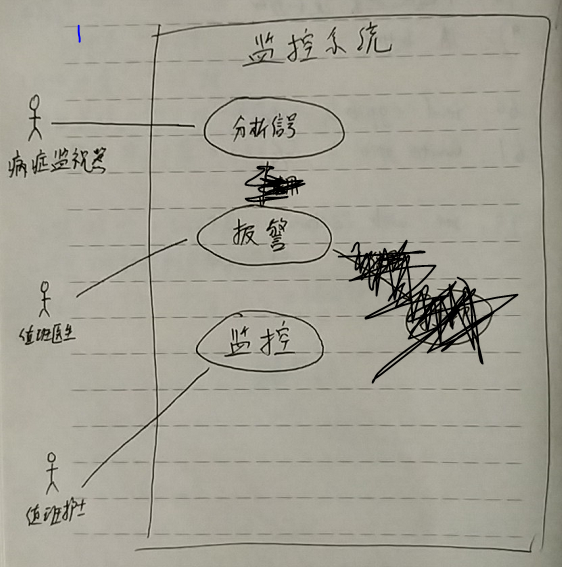
**

*0层数据流图:*



*用例图,(最简单情况下)包含三部分: 参与者, 用例, 关系.*

*参与者, 就是软件的使用者. 用例, 就是从使用者的角度, 软件能提供的一个相对独立的小功能. 关系, 就是在参与者和用例之间画一条线, 表示两者有关系.*



原来答案中用例和用例之间有使用关系, 这是错的.

用例和用例之间的关系只能是无/扩展/包含/泛化

（2）以医院病房监护系统为例，分析比较结构化开发方法与面向对象的开发方法的主要特点，结合该实例说明，为什么面向对象的开发方法更加优越（10分）？

答：面向对象的方法具有以下主要特点：

1. 对软件开发过程所有阶段进行综合考虑。
2. 软件生存期各阶段所使用的方法、技术具有高度的连续性，用符合人类认识世界的思维方式来分析、解决问题。

在医院病房监护系统中，一但建立了系统的USE/CASE模型后，无论在需求分析、软件设计，甚至实现阶段，所建立的静态模型和动态模型都具有相同的描述方式，描述模型的一致性，不仅为开发者带来方便，也更直观的描述系统的特性。

而结构化方法，是以数据流、数据封闭性为基础，从对数据进行加工的角度进行需求分析，建立分层的DFD 图（如上图）。而在软件总体设计阶段，则用模块结构图来进行描述， 详细设计阶段又是新的描述方式（如N-S 图、PAD 图等），总之，由于不同阶段有不同的描述方式，需要进行多次转换，不仅为开发者带来诸多不便，也影响了对系统直观、准确的描述。

## 3.连锁超市进货操作流程为：厂家持“定货单”及“厂家送货单”送货，库管科检查货物。

如果来货相符，库管科做收货确认，（包括：在“定货单”之“点核数量”栏手工填写收货数量；在电脑中找出“定货单”，在“点核数量”栏输入收货数量；在“收货日报表”上记录该收货；在“定货单”、“厂家送货单”上加盖库管科公章），然后库管科打印商品条码，收送货组与卖场办理移交手续，卖场收货人在“定货单”上签名盖章，日末，库管可将单据汇总，连同“收货日报表”分别送卖场、财会，并自留一份，第二天电脑科打印“验收进货单”，分送卖场、采购、财会。

如果来货不符，通知负责该次定货之采购主任，采购主任填写“定货修正单”，报采购部主管签名同意，“定货修正单”交库管科，库管科从电脑中提出原“定货单”，按“定货修正单”内容进行修改，将修改后之新“定货单”打印（一式三份），和原“定货单”连在一起，最后进行上述来货相符后的工作。

要求：

1. 画出该系统的的数据流图（15分）；

答:

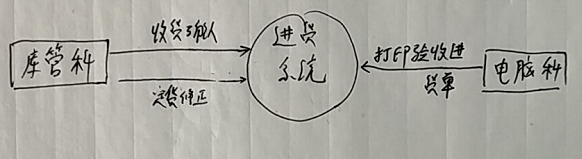
题目描述内容很多, 包含了很多非软件的操作, 比如盖章, 签名, 送货等. 这些都和软件无关, 不应该体现在数据流图中. 把和软件无关的内容去掉以后, 是这样的:

~~厂家持“定货单”及“厂家送货单”送货，库管科检查货物。~~

~~如果来货相符~~，库管科做收货确认，（包括：~~在“定货单”之“点核数量”栏手工填写收货数量~~；在电脑中找出“定货单”，在“点核数量”栏输入收货数量；~~在“收货日报表”上记录该收货；在“定货单”、“厂家送货单”上加盖库管科公章），然后~~库管科打印商品条码，~~收送货组与卖场办理移交手续，卖场收货人在“定货单”上签名盖章，日末，库管可将单据汇总，连同“收货日报表”分别送卖场、财会，并自留一份，第二天~~电脑科打印“验收进货单”，~~分送卖场、采购、财会。~~

~~如果来货不符，通知负责该次定货之采购主任，采购主任填写“定货修正单”，报采购部主管签名同意，“定货修正单”交库管科，~~库管科从电脑中提出原“定货单”，按“定货修正单”内容进行修改，~~将修改后之新“定货单”打印（一式三份），和原“定货单”连在一起，~~最后进行上述来货相符后的工作。

则顶层数据流图:



注意, 只有库管科和电脑科是在用软件, 其他人都是人玩人, 和软件无关.

0层数据流图:



1. 给出该系统主要处理、数据存储或外部实体的数据字典（15分）。

定货单=订货单编码+商品+下单数量+供货商+点核数量

验收进货单=商品+数量+供货商+订货单编码

商品=编码+名称+{供货商}

如果还有时间, 就把其他的人工部分的也写上吧. 多写的应该不扣分.

厂家送货单=送货单编码+订货单编码+商品+数量+供货商

收货日报表= 日期+商品+数量+收货单编码+收货人

定货修正单= 定货单编码+修改内容

## 4.设计一个简单的学生选课系统，系统能够注册符合要求的学生，学生注册以后就能选课。选课过程如下：

（1）浏览限选课名列表，然后根据课程的任课教师、时间、地点和人数限制决定是否选课成功；

**（2）任课教师也可以根据情况，提供新的课程或修改过去课程的说明；**

（3）教务管理人员统一管理注册的情况，可以删除、修改学生、教师和课程的权限。

要求：

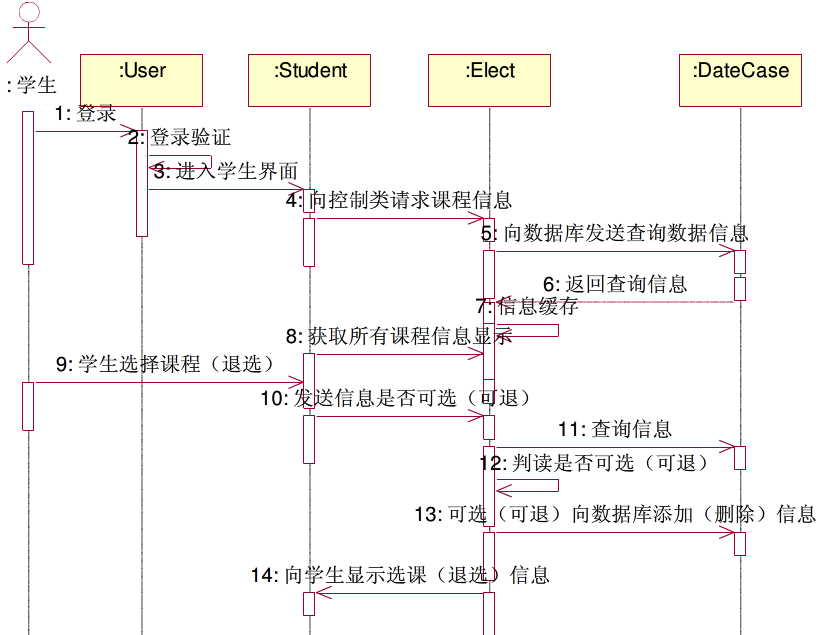
（1）请设计出功能模型，用用例（use-case）图表示（10分）。

（2）请设计出对象模型，要求画出对象图，标出多重性、角色、限定词和关系（10分）；

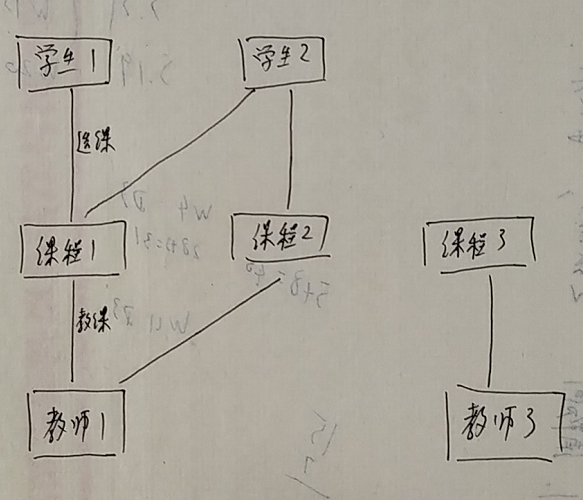
（3）请设计出动态模型，要求画册出学生成功注册后选课的时序图（10分）；

答: 用例图:





对象图:



## 5.某铁路售票系统，其相关角色可简化为乘客、售票员和管理人员。业务过程如下：

* 售票窗口售票，乘客先指名车次和乘车时间，确认有余票后销售。
* 售票员根据乘客提供的信息查询余票，并卖出车票或取消本次购票，等待下一次的交易。
* 在购票期间，乘客可以按取消键取消本次购票。
* 管理人员可以取出钱币，并重置系统余额。
* 管理人员可以打印每天的交易账单。

其中部分数据结构的组成如下：

购票申请单 = 乘客身份证号 + 姓名 + 目的地 + 价格

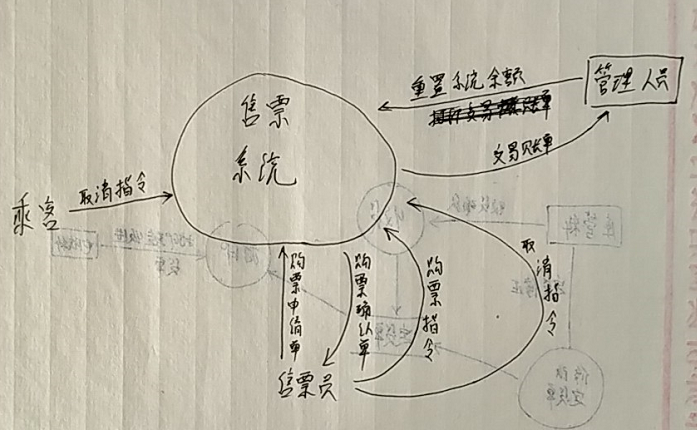
购票确认单 = 身份证号 + 姓名 + 目的地 + 是否有余票

每日账单 = 当日初余额 + 当日末余额 + 日期 + 售票张数

要求：

1. 请画出分层数据流图（10分），并写出第一层数据流图的数据字典（5分）。

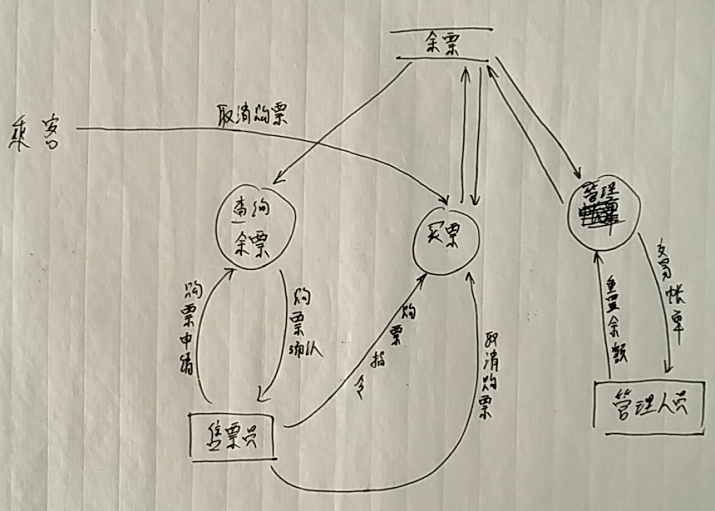
答: 顶层数据流图:



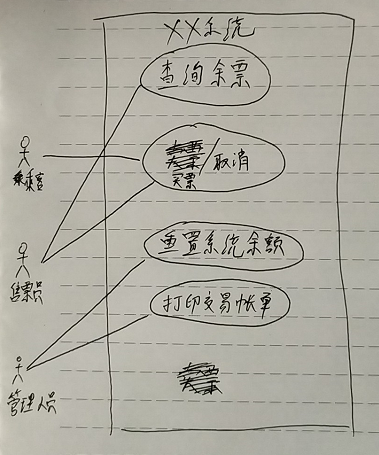
注意, 其实交易账单没必要发送给管理人员, 而是管理人员发送一个[打印交易账单]的指令给系统. 但是题目既然提到了[每日账单]的数据结构, 那就只好配合一下.

[取消指令]也没必要加, 但是提到了, 那就加上.

0层数据流图:



1. **请采用UML的用例图描述系统功能需求，并识别出可能的类（15分）。**



**从用例图识别类, 那就是把所有的参与者分别当作一个类. 把每个参与者关联的用例作为那个类的方法:**

**乘客类: 方法有买票, 取消买票**

**售票员类: 方法有买票, 取消买票, 查询余票**

**管理人员类: 方法有重置系统余额, 打印交易账单**

## 6.某个银行的存取款业务处理系统有以下功能：

* 其相关角色可简化为储户和银行工作人员；
* 储户将填好的存／取款单和存折交给银行工作人员输入系统；
* 系统要求进行以下处理：

（1）业务分类处理：系统审查存／取款单，不合格则退回；合格则确定本次业务的性质(存款或取款)；

（2）存款处理：系统将存款单上的存款金额分别记录在存折和账目文件中；记录现金账；打印存款通知单给储户；最后将存折还给储户；

（3）取款处理：系统将取款单上的取款金额记录在账目文件和存折中并修改现金账。

要求：

（1）请画出分层数据流图（10分），并写出第一层数据流图的数据字典（5分）。

（2）请采用UML的用例图(use case diagram)描述系统功能需求，并识别出可能的类（15分）。

答:

1. 数据字典:

存款单=流水号+储户+金额

取款单=流水号+储户+金额

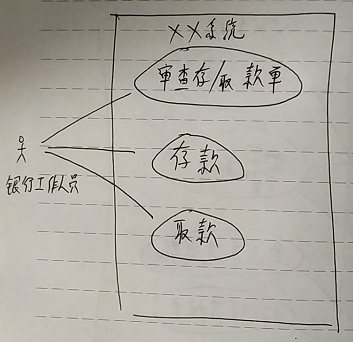
储户=身份证号码+姓名+账号

账目文件=时间+储户+金额+操作类型

操作类型=存款|取款

存款通知单=时间+流水号+存款单流水号+储户+金额

(2)用例图:



根据用例图, 识别出来的类有银行工作人员, 其方法有审查存/取款单,存款,取款

另外, 存款单和取款单也可以认为是类

## 7.某公司准备开发公共汽车刷卡计费系统，具体需求如下：

（1）司机在每到达一站时，点击到站按钮，通知系统已到达一站；

（2）乘客上下车时都要刷卡。上车刷卡通知系统该乘客已上车。在下车刷卡时，系统根据该乘客乘坐的站数，计算票价，并从卡中的余额中扣除。

（3）乘务员可查询某乘客的卡是否在上车时刷过，也可查询每个班次的运营情况（如，载客量和运营额）。

（4）管理员可以通过系统调节费率，记费方式等。

要求：

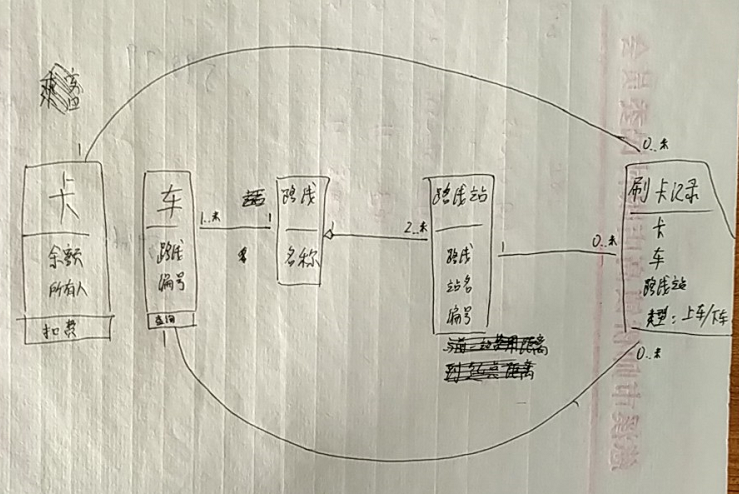
（1）请根据以上对该系统主要需求的描述，并尽可能地考虑一些特殊的情况，进行需求分析，建立该系统完整的用例图，并附用例的文本说明（10分）。

（2）请建立该系统的OOA类图（10分）。

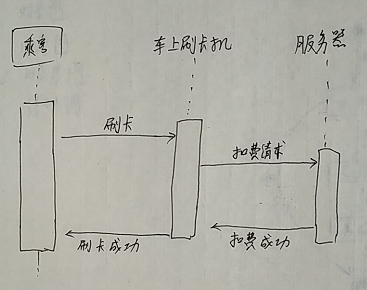
（3）用顺序图描述乘客下车刷卡时，系统中有关的对象的交互序列（10分）。

答:

类图:



顺序图:



## 8.现在有一系统正在进行集成测试，模块A提供的文件接口能够读取txt格式的文本文件，不论该文件是否被其它进程使用。请补充完成如下测试用例中的（1）、（2）、（3）和（4）部分。

|  |  |
| --- | --- |
| 测试用途编号 | FILE01\_READFILE |
| 测试项目 | 测试模块A提供的文件接口 |
| 测试标题 | 文件B正在被其他进程执行读/写操作，通过A模块的文件接口读取文件B中的数据 |
| 重要级别 | 高 |
| 预置条件 | **（ 1 ）（5分）** |
| 输入 | **（ 2 ）（5分）** |
| 操作步骤 | **（ 3 ）（15分）** |
| 预期输出 | **（ 4 ）（5分）** |

答:

预置条件:无

输入:txt格式的文本文件的路径

操作步骤:调用模块A提供的文件接口

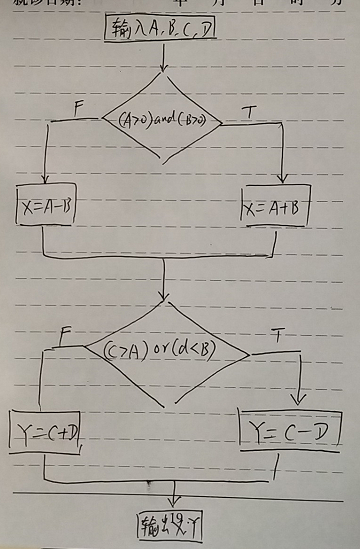
预期输出:能够正确读取, 不管该文件是否被其他进程使用

## 9.依据下面给出的N—S图画出其流程图，在此基础上设计最少测试用例，实现语句覆盖和条件组合覆盖。



答:

流程图:



实现语句覆盖的测试用例:

所谓语句覆盖, 就是每个语句都被执行到. 在流程图上, 就是每条线都要走过.

那么最简单的办法, 就是设计两个测试用例, 他们在每个判断上的结果是一真一假. 这两个测试用例合起来, 就是把每条线都走了.

令A=1, B=1, 那么第一个判断就是T. 令C=2, D=2 则第二个判断是T

令A=-1, B=-1, 那么第一个判断是F. 令C=-2, D=0, 则第二个判断是F

这两个测试用例就视线了语句覆盖

实现条件组合覆盖的测试用例:

所谓条件组合覆盖, 就是对于流程中的每一个判断, 要把所有的条件组合都走一次.

比如第一个判断 (A>0) and (B>0), 包含了两个子判断(A>0)和(B>0). 两个子判断都可真可假, 合起来就是4种可能性. 把这4种可能性都覆盖了, 就实现了条件组合覆盖.

第二个判断, 也有4种可能性.

那么最理想的情况, 构造4个测试用例, 每个测试用例, 分别覆盖了两个判断种的一个条件组合, 那么合起来, 就把所有的条件组合覆盖到了.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | A>0 | B>0 | C>A | D<B |
| A=1 B=1 C=2 D=0 | T | T | T | T |
| A=1 B=-1 C=2 D=-2 | T | F | T | F |
| A=-1 B=1 C=-2 D=0 | F | T | F | T |
| A=-1 B=-1 C=-2 D=0 | F | F | F | F |
|  |  |  |  |  |
|  |  |  |  |  |

## 10.有一款8位微机，其十六进制常数定义为：以0x或0X开头的数是十六进制整数，其值的范围是-7f～7f（表示十六进制的大小写字母不加区别），如0X13，0X6A，-0X3c。根据上述条件使用等价划分法设计测试用例。

答: 有效范围是-7F~7F, 那么等价类有三个, 一个是有效范围内的, 比如0x7F.

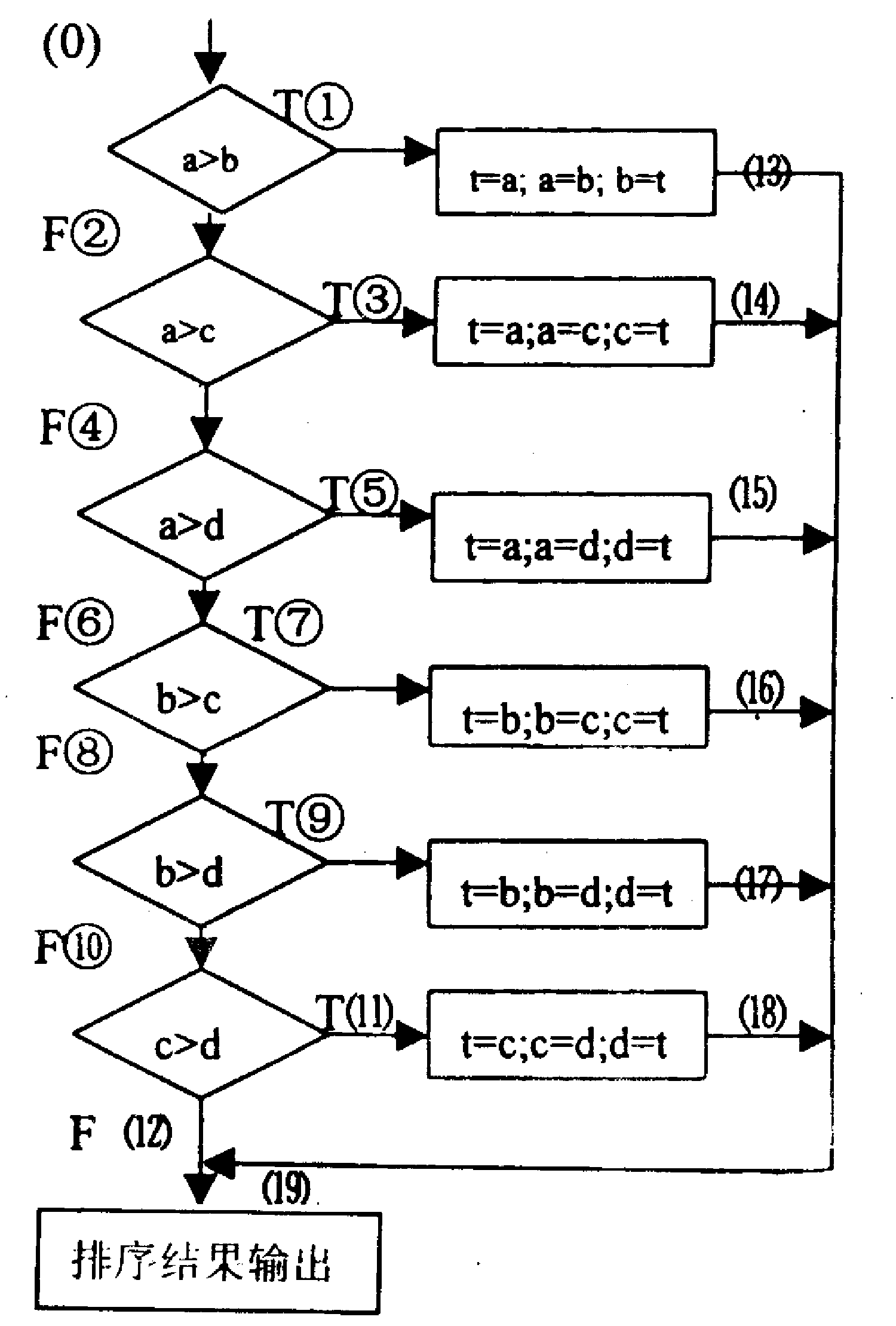
一个是有效范围左边的, 比如-0x80, 一个是有效范围右边的, 比如是0x80

## 11.下面是某程序的流程图

（1）计算它的环路复杂性（10分）。

环路复杂性数量=判断节点数+1 = 6+1 = 7

（2）为了完成基本路径测试，求它的一组独立的路径（20分）。



答:

所谓基本路径测试, 就是实现语句覆盖(概念不同, 形式一样, 考试的时候不要这么说. 可以这么理解). 就是让流程图里面每一个地方都能走到.

实现了基本路径测试的一组路线, 成为一组独立路径.

了解了概念以后, 题目就很好做了:

如果流程图里有N个T/F判断, 那么就需要N+1个独立路径. 这个可以用来验证是否缺了或者多了一个独立路径.

独立路径：至少沿一条新的边移动的路径

路径1：0->1->13->19

路径2：0->2->3->14->19

路径3：0->2->4->5->15->19

路径4：0->2->4->->6->7->16->19

路径5：0->2->4->->6->8->9->17->19

路径6：0->2->4->->6->8->10->11->18->19

路径7：0->2->4->->6->8->10->12

## 12.某网站开发新的邮件系统供用户使用，现需要做系统测试。请以“输入正确的用户邮件地址和错误的密码”为例，完成如下的测试用例中（1）、（2）、（3）和（4）部分。

测试用例：

* 测试目标：（ 1 ）（5分）；
* 测试环境：Windows XP 操作系统和浏览器IE8；
* 输入数据：（ 2 ）（5分）；
* 步骤：（ 3 ）（15分）；
* 期望结果（ 4 ）（5分）。

答:

* 测试目标：（ 检验客户输入正确邮箱地址和错误密码时候会出现的现象 ）（5分）；
* 测试环境：Windows XP 操作系统和浏览器IE8；
* 输入数据：（ 用户名，密码 ）（5分）；
* 步骤：（ ①输入正确用户邮件地址②输入错误密码③提交观察结果）（15分）；
* 期望结果（ 系统反馈输入密码错误提示，不会进入邮箱 ）（5分）。

## 13.下列是一段伪码程序：

START

INPUT （A，B，C）

IF A>5

THEN X=10

ELSE X=1

ENDIF

IF B>10

THEN Y=20

ELSE Y=2

ENDIF

IF C<15

THEN Z=30

ELSE Z=3

ENDIF

PRINT(X,Y,Z)

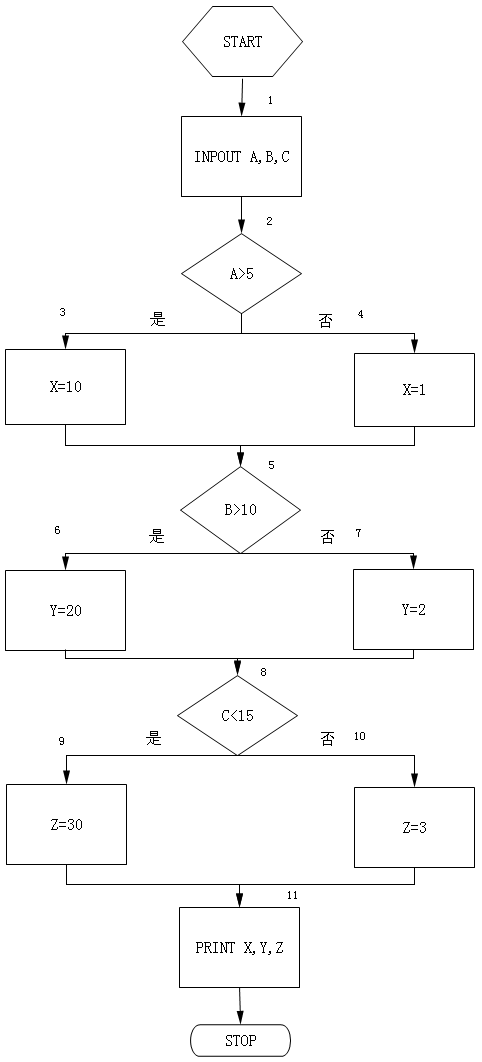
STOP

问题：（1）画出控制流程图（10分）

（2）采用基本路径测试，给出独立的路径（10分）

（3）写出路径覆盖用例（10分）

答：（1）控制流程图：



1. 独立路径

所谓一套独立路径, 就是他们实现了语句覆盖. 路径的数量的上限, 是环路复杂度, 也就是判断节点数量+1.

在这个例子中, 上限没有达到, 只需要两个 用例就能实现语句覆盖.

1,2,3,5,6,8,9,11

1,2,4,5,7,8,10,11

（3）路径覆盖用例

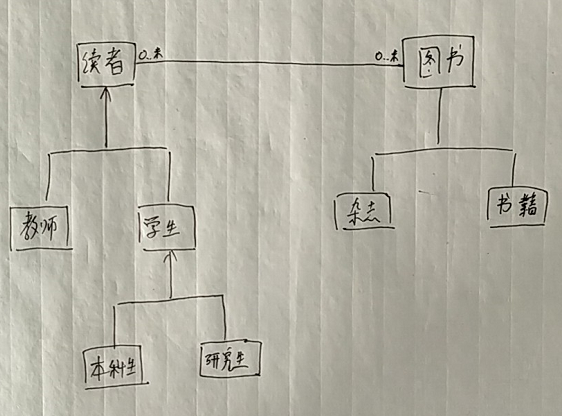
略.

总的来说, 就是3个判断分别可能是T或者F. 组合起来就是8种可能.

把8种可能都来一遍, 就覆盖了. 也就是要构造8个测试用例, 分别对应这8种可能.

## 14.图书馆管理系统中，读者包括教师、学生（学生中有本科生、研究生、）等等，图书包括杂志、各类书籍，请用UML建立UseCase图（5分），读者和图书的类图（10分），图书的状态图（5分），以及借还书过程的活动图（10分）。

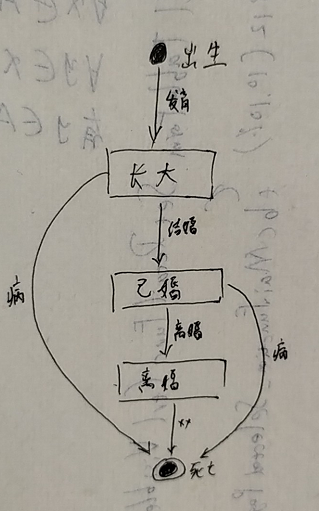
答: 类图:

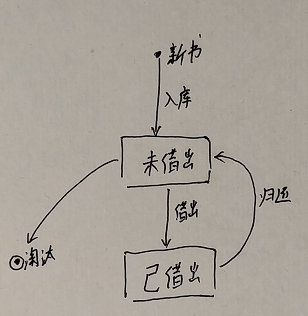


注意, 只写了类名, 属性和方法省略了, 考试的时候随便加几个吧

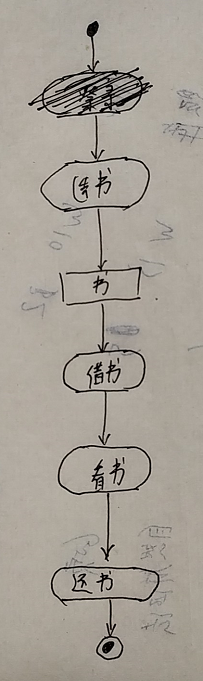
状态图:

所谓状态图, 就是把一个东西的各种状态列出来, 然后画线和文字表示各种状态之间的跳转. 还有个开始状态(实心圆圈表示)和结束状态(实心圆圈+外面一个圆表示). 比如人的状态图:





活动图:



## 15.由键盘输入一个足够大的数，判断该数是否为素数（除一和本身外不能被其他数整除）。

要求：

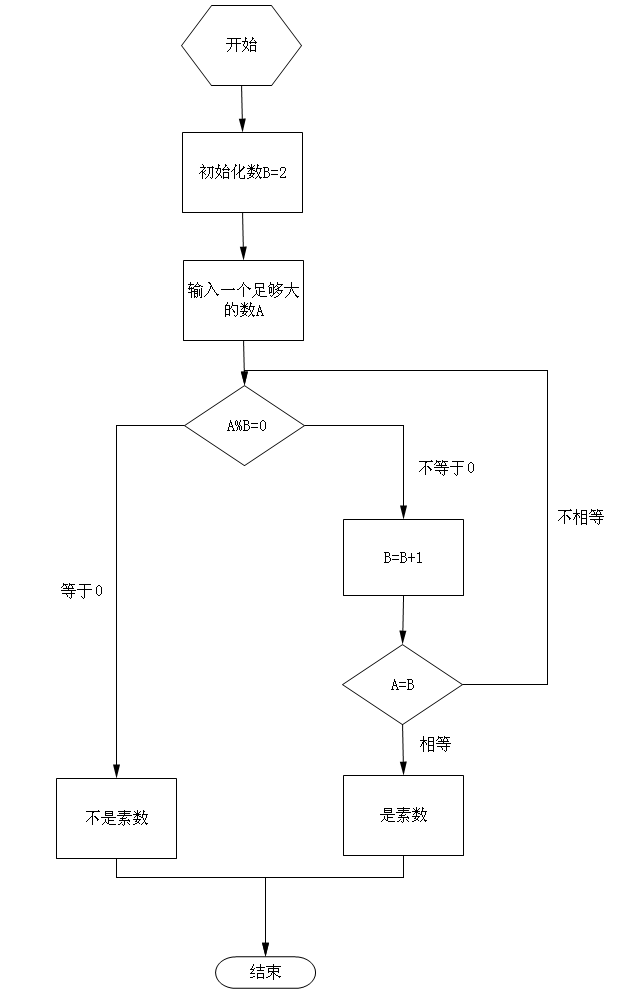
（1）画出实现该算法的程序流程图，画出控制流图（5分）；

（2）写出判定测试和条件测试的测试用例（5分）；

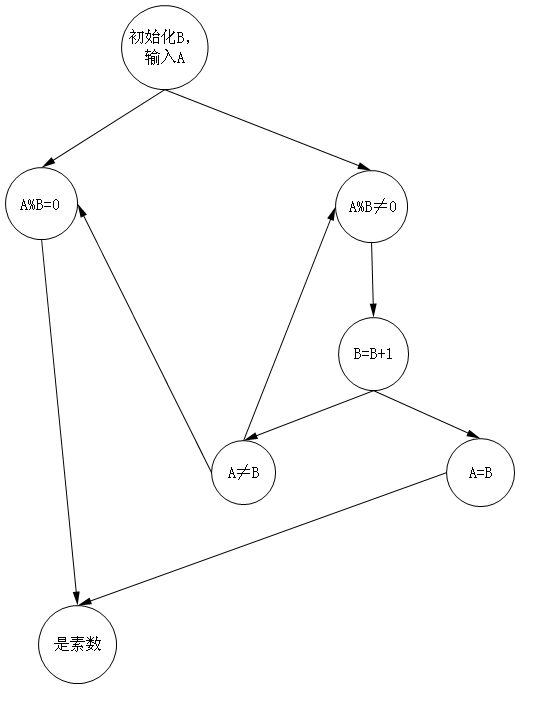
（3）如果采用基本路径测试法，写出测试用例（10分）；

（4）如果对循环进行测试，应考虑那些测试点，写出测试用例（10分）。

答：（1）流程图



控制流图：



（2）判定测试用例：

判定表：

|  |  |  |  |
| --- | --- | --- | --- |
|  | 1 | 2 | 3 |
| 条件 | 能否被1整除？ | Y | Y |
| 能否被本身整除？ | Y | Y |
| 能否被除了1和本身外的数整除？ | Y | N |
| 动作 | 是素数 |  |  |
| 不是素数 |  |  |

测试用例：

A1=100，能被1整除，能被100整除，能被2、4、5…….整除，不是素数

A2=101，能被1整除，能被101整除，不能被其他数整除，是素数

条件测试用例：A=200，能被1整除，能被200整除，能被2、4、5…整除，不是素数

（3）基本路径测试用例：

①输入A=10001，,10001%2≠0，B=B+1=3，10001%3≠0，B=B+1=4……，10001%10001=0，A=B，结论：A是素数

②输入A=2，2%2=0，结论：A不是素数

③输入A=15,15%2≠0，B+B+1=3，15%3=0,A≠B，结论：A不是素数

（4）循环测试：

应该考虑①B为初始值，A=B的点②B=B+1，A=B的点③B=B +1,A≠B的点

## 16.某考务处理系统具有以下功能：

输入报名单；自动编制准考证号；输出准考证；输入成绩清单；输出成绩通知单；输出成绩分布表；输入合格标准、输出录取通知单；试题难度分析,并输出试题难度分析表。

其中部分数据流的组成如下所示:

报名单 = 报名号 + 姓名 + 通信地址

考生名册 = 报名号 + 准考证号 + 姓名 + 通信地址

成绩册 = 准考证号 + { 课程号 + 成绩 } (其中{W}表示W重复多次)

准考证 = 报名号 + 姓名 + 准考证号

要求：

（1）画出顶层数据流图和底层数据流图（20分）；

（2）画出USE/CASE图（10分）。

## 17.某房屋租赁服务系统主要功能如下：系统对出租的房屋信息、房主信息、需要租房的人员信息，以及租房人和房主的会面情况进行管理和维护。

房主信息包括：房主编码、姓名、地址、电话号码和密码；

房屋信息包括：房主编码、房屋地址、类型（单间/套间）、适合住宿的人数、房租、是否可以出租。

租房人信息包括：编码、姓名、现住址、电话号码、出生日期、性别、密码。

每当房屋信息发生变化时，房主可以通过系统维护信息，但需交纳一定的费用，由系统自动给出费用信息。

租房人可通过系统查询现有的可租用的房屋，但必须先在系统中注册。若租房人希望租用某房屋，则需要发出租房请求，请求中包含房屋的详细信息，系统将安排租房人与房主会面的时间和地点，并将会面信息通知租房人和房主，会面信息包括会面时间、地点以及会面双方的基本信息，系统将记录会面信息。

要求：

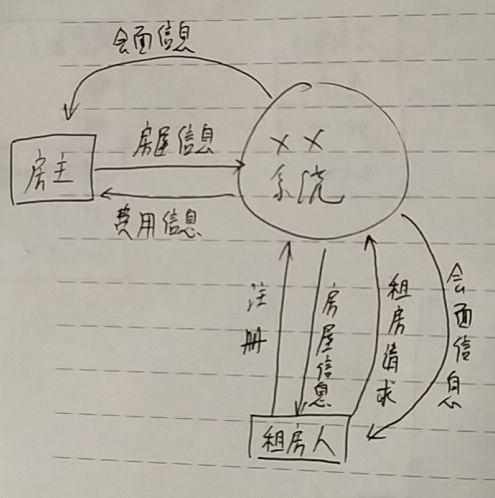
（1）画出上述过程的分层的数据流图（顶层和底层，中间层可忽略）。（15分）

（2）给出顶层数据流图的数据字典。（5分）

（3）应用Use/Case图表达系统的交互。（10分）

答:

1. 顶层数据流图:



还缺了一个注册结果, 从XX系统发送到租房人

(2)

房屋信息=房主编码+房屋地址+类型+适合住宿的人数+房租+是否可以出租

费用信息=时间+房主编码+房屋地址+金额

注册请求=姓名+现住址+电话号码+出生日期+性别+密码

注册结果=租房人编码(假设永远成功)

租房请求=租房人编码+房主编码+房屋地址+类型+适合住宿的人数+房租+是否可以出租

会面信息=时间+地点+房主编码+租房人编码

## 18.某医院病人住院与治疗结帐业务处理过程及说明如下：

（1）病人经医生建议提出申请住院治疗，选择住院病房标准，经核查病房档案文件，有相应床位时可办理住院。

（2）办理时由病人填写住院登记卡，内容包括：病历号，姓名，病床号，支付方式和其他内容。支付方式可选择现金支付或银行转帐两种，但每个病人只能选择一种方式。病床租出后在病床档案中作“使用”标志。

（3）病人在住院期间除治疗外医院还提供其他收费服务，如用餐、保健等。所有费用按病床号记入计费文件中，病人出院时一起结算。

（4）病人出院时，医院根据计费文件和住院登记卡的支付方式办理结帐手续。并修改病床档案文件中的标志。

主要文件和档案格式如下：

病床档案：病床号，病床标准，日租金，是否空闲。

住院登记卡：病历号，姓名，病床号，支付方式，入住日期，…。

计费文件：病床号，服务项目，费用，日期。

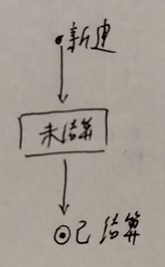
请画出病人住院与结帐业务的分层数据流图（15分）。

采用面向对象方法，请你给出“计费文件”类的可能操作（只要给出操作名）（5分），并画出“计费文件”的状态图（10分）。

答: 根据前面的流程描述, 理解为: 一个人在一次住院期间, 只有一个计费文件, 里面包含了所有相关的消费记录. 那么[计费文件]类的操作有:get病人, set病人, get欠费总额,add消费记录

但是后面又提到计费文件的格式, 从格式上看, 把计费文件解释为[单次消费]. 那么操作就是四个属性的get和set.

计费文件状态图:



## 19.某制造企业的物料出入库管理的工作流程叙述如下：

1．出库工作流程

（1）领料人提交领料单（每一种物料有一张领料单）；

（2）仓库保管员根据领料计划单检验该领料单是否有效；

（3）若经检验没有相应的领料计划，则通知领料人该领料单无效；

（4）若领料单有效，仓库保管员根据领料单上的物料代码核对是否有足够的库存；

（5）若没有足够的库存，仓库保管员向领料人发缺货单；

（6）若有足够的库存，仓库保管员在领料单上签字，并登记出库单，修改物料主文件中的现有库存数；相应的物料出库，物料清单交领料人。

2．入库工作流程

（1）采购员提交入库申请单（每一种物料有一张入库申请单）；

（2）仓库保管员根据采购计划单验收入库申请单；

（3）若验收发现没有相应的采购计划，则仓库保管员向采购员发无效申请单：

（4）若验收合格，则仓库保管员向检验员申请物料检验；检验员根据检验结果填写物料检验单；

（5）如果物料或供货方不合格，则向采购员发出退货单；

（6）如果检验合格，则仓库保管员登记入库单，修改物料主文件中的现有库存数，相应物料入库。

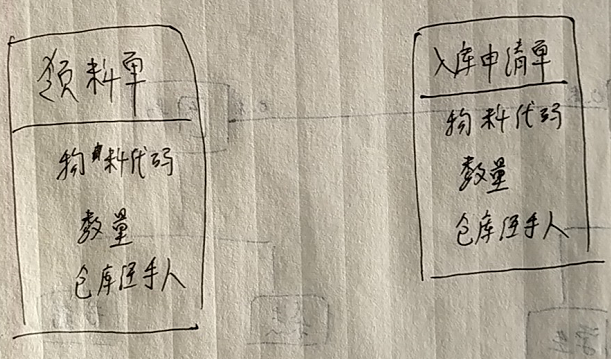
要求：

（1）画出上述业务分层的数据流图。（15分）

（2）如果采用面向对象的方法，画出USE/CASE图（10分），其中“领料单”和“入库申请单”可以定义为两个类，请给出这两个类至少应具有的属性。（5分）

答:

两个类的属性:



## 20.某大学校园一卡通管理系统包括以下功能：

（1）办理新卡，新入校的教师、学生或丢失卡者可提交院系盖章的证明申请办理新卡，系统根据人事处和学生处提供的名单核实办理，同时更新持卡基本信息文件。

（2）存款，师生可向卡上存现金或通过指定账户向卡上划拨款，并修改明细账文件和账户文件。

（3）消费，可在校园内用一卡通消费，每次消费记入明细帐文件。

（4）查询对账，师生可查询每次存入或消费时的时间和金额。

（5）挂失，师生可根据证件提出挂失申请，挂失后该卡不能消费

要求：

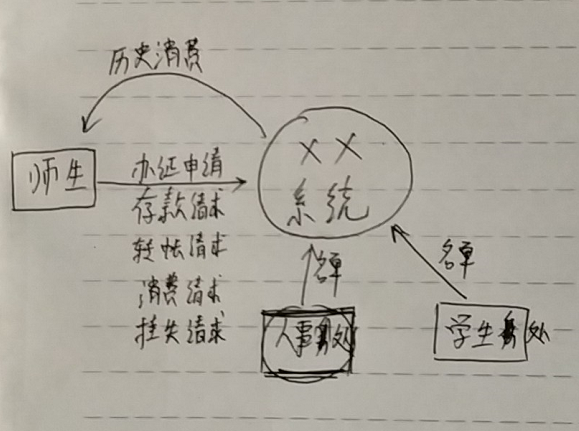
（1）请根据所述分析业务，并画出数据流图。要求功能完整，逻辑合理（15分）;

（2）给出顶层数据流图的数据字典（5分）;

（3）画出USE/CASE图（10分）。

答:

1. 顶层数据流图:



(2)

数据字典:

办证申请=身份证号码

存款请求=身份证号码+金额

转账请求=出钱身份证号码+收钱身份证号码+金额

消费请求=身份证号码+金额+说明

挂失请求=身份证号码

名单={身份证号码}

历史消费={单次历史消费}

单次历史消费=身份证号码+时间+金额+说明

说明: 这20题应用题, 好多没有答案, 好多答案错误(主观判断), 所以我提供了答案. 有些题没有给答案, 因为类似题型已经在前面出现过.

有些答案给的是简化版, 严格来说是会扣分的. 但是如果你能搞出完整版来, 你也没必要看我这份了. 完整版需要更多的理解, 会需要多投入很多时间, 收到的效益很低, 所以没必要.

孙超