名词解释

1. **数据词典——是描述数据信息的集合，它对数据流图中的各个元素按规定格式进行详细的描述和确切的解释,是数据流图的补充工具。**
2. **数据流图——他以图形的方式反映系统的数据流程**
3. 白盒测试——**按照程序内部的结构测试程序，检验程序中的每条路径是否都能按预定要求正确工作。有两种测试法既逻辑覆盖测试法和路径测试法**
4. 黑盒测试——**按照程序的功能测试程序，检验与程序功能有关的输入、输出与程序执行是否正确。有四种方法既等价分类法、边界值分析法、错误猜测法和因果图法**
5. **完善性维护——**为了适应用户业务和机构的发展变化而对软件的功能、性能进行修改、扩充的过程称为完善性维护。因为各种用户的业务和机构在相当长的时期内不可能是一成不变的，所以功能、性能的增加是不可避免的，而且这种维护活动在整个维护工作中所占的比重很大
6. 软件可靠性——指在给定的时间内，程序按照规定的条件成功地运行的概率
7. 软件配置——是一个软件在生存周期内，他的各种形式、各种版本的文档与程序的总称
8. **软件再工程——运用逆向工程、重构等技术，在充分理解原有软件的基础上，进行分解、综合、并重新构建软件，用于提高软件的可理解性、可维护性可复用性或演化性。**
9. **α测试——**是在一个受控的环境下，由用户在开发者的“指导”下进行的的测试，由开发者负责记录错误和使用中出现的问题。
10. **β测试——**是由软件的最终用户（多个）在一个或多个用户场所来进行。由用户负责记下遇到的所有问题，包括主观认定的和真实的问题，定期向开发者报告，开发者在综合用户的报告之后进行修改，最后将软件产品交付给全体用户使用。
11. **聚集关系——**表示类或对象之间的整体与部分的关系
12. **泛化关系——**表示类或对象之间的一般与特殊的关系
13. 内聚**——一个模块内部各个元素彼此结合的紧密程度**的度量。
14. 耦合**——一**一个软件结构内不同模块之间互连程度的度量。

名词解释：

一章：

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。

产生软件危机的原因：一方面与软件本身的特点有关，另一方面也和软件开发与维护方法不正确有关。

软件工程：是指导计算机软件开发和维护的一门工程学科。采用工程的概念、原理、技术和方法来开发与维护软件，把经过时间考验而证明正确的管理技术和当前能够得到的最好的技术方法结合起来，以经济地开发出高质量的软件并有效地维护它。

软件工程的 7 条基本原理：（1）用分阶段的生命周期计划严格管理;(2) 坚持进行阶段评审;(3) 实行严格的产品控制;(4) 采用现代程序设计技术;(5) 结果可以清楚地审查;(6) 开发小组的人员应该少而精;(7) 承认不断改进软件工程实践的必要性.

软件工程方法学 3 要素：方法、工具、过程

软件过程：是为了获得高质量软件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。

软件生命周期的概念：有软件定义、软件开发和软件维护 3 个小时期组成

软件生命周期 8 个阶段的主要任务：（1）问题定义：“需要解决的问题是什么？”(2) 可行性研究：“寻求可行的解决方案？”(3) 需求分析：“解决这些问题需要系统做什么？”(4) 总体设计（概要设计）：“应该怎样实现目标系统？”(5) 详细设计（模块设计）：“如何具体地实现这个系统？”(6) 编码和单元测试：“写代码，测试每个模块！”(7) 综合测试：“通过各类测试和调试来完善软件”(8) 软件维护：“通过各种必须的维护活动使系统持久满足用户的需要！”

二章：

可行性研究的五个方案：技术可行性，经济可行性，操作可行性，法律可行性，社会效益

可行性研究过程：1.复查系统规模与目标、2.研究目前的系统、3. 导出新系统的高层逻辑模型、4. 进一步定义问题、5. 导出和评价供选择的解法、6. 推荐行动方针、7. 草拟开发计划、8.书写文档提交审查

系统流程图：用来描述物理系统的工具。系统流程图表达的是数据在系统各部件之间流动的

情况，而不是对数据进行加工处理的控制过程。即：系统流程图≠程序流程图。

数据流图：用来描述逻辑系统的工具。数据流图(DFD)是一种图形化技术，它描绘信息流和数据从输入移动到输出的过程中所经受的变换，即数据流图描绘数据在软件中流动和被处理的逻辑过程。

三章：

需求分析在软件生命周期中位置：最后一个阶段；**任务**：完整、准确、清晰、具体地确定系统所要完成的工作。

软件系统的综合要求:功能需求,性能需求,可靠性和可用性需求，出错处理需求，接口需求，约束，逆向需求，将来可能提出的要求

获取需求的方法：访谈、面向数据流自顶向下求精、简易的应用规格说明技术、快速建立软件原型

3 种分析模型：数据模型（ER 图），功能模型（DFD），行为模型（状态转换图）

需求分析阶段的主要图形工具：层次方框图（描绘数据的层次结构）；Warnier 图（描绘数据的层次结构）；IPO 图（IPO 图是输入、处理、输出图的简称）

五章：

总体设计的两个阶段：（1）系统设计阶段（2）结构设计阶段

总体设计的设计原理：模块化，抽象，逐步求精，信息隐藏和局部化，*模块独立（耦合，内聚）

耦合：是对一个软件结构内不同模块之间互连程度的度量；包括：

（1）数据耦合——如果两个模块彼此间通过参数交换信息，而且交换的信息仅仅是数据

（2）控制耦合——如果传递的信息中有控制信息(尽管有时这种控制信息以数据的形式出现)

（3）特征耦合——整个数据结构作为参数传递而被调用的模块只需要使用其中一部分数据元素

（4）公共环境耦合——两个或多个模块通过一个公共数据环境相互作用

（5）内容耦合——如果出现下列情况之一，两个模块间就发生了内容耦合

低——高

3．内聚：标志一个模块内各个元素彼此结合的紧密程度；包括：

（1）偶然内聚——如果一个模块完成一组任务，这些任务彼此间即使有关系，关系也是很松散的。

(2)逻辑内聚——如果一个模块完成的任务在逻辑上属于相同或相似的一类。

(3)时间内聚——如果一个模块包含的任务必须在同一段时间内执行。

（4）过程内聚——如果一个模块内的处理元素是相关的，而且必须以特定次序执行。

（5)通信内聚——如果模块中所有元素都使用同一个输入数据和(或)产生同一个输出数据。

（6）顺序内聚——如果一个模块内的处理元素和同一个功能密切相关，而且这些处理必须顺序执行(通常一个处理元素的输出数据作为下一个处理元素的输入数据)。

（7）功能内聚——如果模块内所有处理元素属于一个整体，完成一个单一的功能。

低——中——高

7 条启发规则：改进软件结构，提高模块独立性，模块规模适中，*深度、宽度、扇出和扇入合理（**深度**表示软件结构中控制的层数；**宽度**是软件结构内同一个层次上的模块总数的最大值；**扇出**：调用其它的模块数（3-4）；**扇入**：被上一级模块调用数（越多越好）），*模块的作用域应在控制域范围内(**模块的作用域定义**为受该模块内一个判定影响的所有模块的集合。**模块的控制域是**这个模块本身以及所有直接或间接从属于它的模块的集合。)，尽量降低模块接口的复杂程度，设计单入口、单出口的模块，模块功能可以预测

六章：

过程设计的工具：程序流程图(程序框图)，盒图(N-S 图)，PAD 图（问题分析图），*判定表，判定树，过程设计语言

面向数据结构的设计方法：（1）Jackson 图（程序中数据元素彼此间的逻辑关系只有三类：顺序结构，选择结构，重复结构）(2) *改进的 Jackson 图(3)* Jackson 方法

程序的处理过程：(1)确定输入和输出数据结构;(2)找有对应关系的输入输出数据单元;(3)从数据结构图导出程序结构图 (均用 Jackson 图表示);(4)列出所有操作和条件，并且把它们分配到程序结构图的适当位置;(5)用伪码表示程序。

七章：

软件测试的定义或目标：测试是为了证明程序有错，而不是证明程序无错误；一个好的测试用例在于它能发现至今未发现的错误；一个成功的测试是发现了至今未发现的错误的测试。

测试步骤：模块测试，子系统测试，系统测试，验收测试，平行运行

逻辑覆盖：（1）语句覆盖 （2）判定覆盖 (3）条件覆盖(4) 判定－条件覆盖 (5)条件组合覆盖 (6) 路径覆盖（7）点覆盖（8) 边覆盖

黑盒测试步骤：等价划分，边界值分析，错误推测，因果图法

白盒测试（结构测试）技术：分析程序内部一每个分支通路。**过程**：按照程序内部的逻辑测试程序，检测程序中主要执行通路是否按预定要求正确工作。

软件可靠性的定义：程序在给定的时间间隔内，按照规格说明书的规定成功运行的概率。

软件可用性的定义：程序在给定的时间点，按照规格说明书的规定，成功运行的概率。

八章：

软件维护的定义：在软件已经交付使用之后，为了改正错误或满足新的需要而修改软件的过程。

软件维护的分类：改正性维护（17%~21%)，适应性维护(17%~21%)，完善性维护(50%~66%)，预防性维护（4%~5%）

十三章：

项目管理的概念：就是通过计划、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。

软件工作量的估算：静态单变量模型（E=A+B×(ev)C)，动态多变量模型，COCOMO2 模型

制定项目的进度工具：Gantt（甘特）图，工程网络

估算软件规模：（1）**代码行技术**：L = $\frac{a+4m+b}{6}$ 其中 L 为估计的程序规模，a 为程序的最小规模、b 为程序的最大规模、m 为最可能的规模。

优点：代码是所有软件开发项目都有的“产品”，而且很容易计算代码行数。

缺点：源程序仅是软件配置的一个成分，用他的规模代表整个软件的规模似乎不太合理，用不同语言实现同一个软件所需要的代码行数并不相同，这种方法不适用于非过程语言。

三、名词解释题每小题 3 分，共 15 分。

31．软件生存周期模型：是描述软件开发过程中各种活动如何执行的模型。

32．数据字典（DD)：数据字典是用来定义数据流图中的各个成分的具体含义的。它以一种准确的、无二义性的说明方式为系统的分析、设计及维护提供了有关元素的一致的定义和详细的描述。

33．内聚性：内聚性是模块独立性的衡量标准之一，它是指模块的功能强度的度量，即一个模块内部各个元素彼此结合的紧密程度的度量。

34．JSP 方法：JSP 方法是面向数据结构的设计方法，其定义了一组以数据结构为指导的映射过程，它根据输入，输出的数据结构，按一定的规则映射成软件的过程描述，即程序结构。

35．多态性：指相同的操作或函数、过程可作用于多种类型的对象上并获得不同结果。或（不同的对象，收到同一消息可以产生不同的结果。）

1．程序的可维护性：为满足用户新的需求，或当环境发生了变化，或运行中发现了新的错误时，对一个已投入运行的软件进行相应诊断和修改所需工作量的大小。

2．容错技术：对那些无法避开的差错，使其影响减少至最小的技术。也就是说，当错误发生时，尽可能地不影响其它的系统元素，或是把用户的影响限制在某些容许的范围内。

3.结构化维护:如果维护工作是从评价完整的软件配置开始入手，确定软件的重要结构特点、性能特点以及接口特点；估量要求的改动将带来的影响，并且计划实施途径。然后首先修改设计并且对所做的修改进行仔细审查。接下来编写相应的源程序代码；使用在测试说明书中包含的信息进行回归测试；最后，把修改后的软件再次交付使用。

4.软件生存周期是指从提出软件开发要求开始，直到该软件报废不用为止的整个时期。这个时期又分为若干个阶段，对软件生产的管理和进度控制有重要作用，使软件的开发有相应的模式、流程、工序和步骤。

5.模块独立性:是模块化、抽象和信息隐蔽的直接产物。每个模块只要完成独立的功能，与其它模块联系越少，则模块的独立性就越强。通过模块与模块之间的耦合性和模块内部的内聚性来衡量模块的独立性。

1．数据流图：是描述数据处理过程的工具。它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。

2．软件维护是软件生命周期的最后一个阶段，是在软件已经交付给用户使用之后，为了改正软件错误或满足新的需要而修改软件的过程。它包括四种类型的维护活动：改正型维护、适应型维护、预防型维护和完善型维护。

3．软件测试是一个为了寻找软件错误而运行程序的过程。目的就是为了发现软件中的错误。一个好的测试用例是指很可能找到迄今为止尚未发现的错误的用例。一个成功的测试是指揭示了迄今为止尚未发现的错误的测试。

4．程序的可维护性：为满足用户新的需求，或当环境发生了变化，或运行中发现了新的错误时，对一个已投入运行的软件进行相应诊断和修改所需工作量的大小。

5.软件生存周期:是指从提出软件开发要求开始，直到该软件报废不用为止的整个时期。这个时期又分为若干个阶段，对软件生产的管理和进度控制有重要作用，使软件的开发有相应的模式、流程、工序和步骤。