

全国高等教育自学考试指定教材辅导

# 软 件 工 程

## 应试指导及模拟试题

全国高等教育自学考试命题研究组 编

教材依据 陆丽娜 主编

中国大地出版社

书名:软件工程应试指导及模拟试题

作者:全国高等教育自学考试命题研究组

出版社:中国大地出版社

ISBN:7-80097-498-7 /TP393

出版日期:2002年1月

# 前 言

国家教育部考试中心于 2002 年开始,正式执行自学考试新计划,同时使用新编的大纲和教材。

参加自考的学生渴求在考前能通过应试指导的帮助及模拟试题的演练,全面检查自己所学的知识是否扎实,考试大纲所要求的内容是否掌握,已经理解的知识能否完整、确切、简明地进行书面表述,并借此增强考生分析和解决实际问题的能力,帮助考生顺利通过考试。因此,为配合广大考生参加考试,并能顺利过关,我们利用多年积累的自考教学辅导资源和经验,全面系统地剖析了各门专业课程新大纲和教材的内容体系,组织编写了一套“全国高等教育自学考试应试指导及模拟试题”丛书,推向全国,以满足考生之急需,适应社会之需要。

本书在编写过程中,严格按照考试大纲的要求,以指定教材为基础,包括了所有考试的知识点,并着重突出重点和难点,充分体现了“在考察课程主体知识的同时,注重考查能力尤其是应用能力”的新的命题指导思想。

本书以习题为主,完全按照指定教材的结构,以章为单位。每章设“考试要求”、“知识重点”、“反馈测试题解”三部分。“考试要求”主要是考试大纲所规定的本章考核要求。“知识重点”主要是对该章的重点、要点内容的总结归纳;“反馈测试题解”则根据考试大纲对各知识点不同能力层次的要求,将知识及知识点下的细目以各种主要考试题型的形式编写,覆盖全部考核内容,适当突出重点章节,并且加大重点内容的覆盖密度,所有试题均附详细解答;书后附有模拟试卷及 2001 年度试题,供考生检验自己学习情况,建议在规定时间内完成。本书由苏亚菲主编。

欢迎广大读者对本丛书提出宝贵意见,以便我们今后工作中得以改进。

全国高等教育自学考试命题研究组

2002 .3



# 目 录

软件工程考试概述 .....	( 1 )
第一章 绪 论 .....	( 6 )
考试要求 .....	( 6 )
知识重点 .....	( 7 )
反馈测试题解 .....	( 8 )
第二章 软件可行性研究与项目开发计划 .....	(22)
考试要求 .....	(22)
知识重点 .....	(22)
反馈测试题解 .....	(24)
第三章 软件需求分析 .....	(34)
考试要求 .....	(34)
知识重点 .....	(35)
反馈测试题解 .....	(38)
第四章 软件概要设计 .....	(58)
考试要求 .....	(58)
知识重点 .....	(58)
反馈测试题解 .....	(62)
第五章 软件详细设计 .....	(93)
考试要求 .....	(93)
知识重点 .....	(93)
反馈测试题解 .....	(95)
第六章 软件编码.....	(113)
考试要求.....	(113)
知识重点.....	(113)
反馈测试题解.....	(115)
第七章 软件测试.....	(123)
考试要求.....	(123)
知识重点.....	(123)
反馈测试题解.....	(127)
第八章 软件维护.....	(147)
考试要求.....	(147)
知识重点.....	(147)
反馈测试题解.....	(148)

第九章 软件开发的增量模型..... (160)

    考试要求..... (160)

    知识重点..... (160)

    反馈测试题解..... (164)

第十章 面向对象的方法..... (177)

    考试要求..... (177)

    知识重点..... (178)

    反馈测试题解..... (180)

第十一章 软件质量与质量保证..... (203)

    考试要求..... (203)

    知识重点..... (204)

    反馈测试题解..... (206)

第十二章 软件工程管理..... (220)

    考试要求..... (220)

    知识重点..... (221)

    反馈测试题解..... (222)

第十三章 软件开发环境..... (236)

    考试要求..... (236)

    知识重点..... (236)

    反馈测试题解..... (238)

软件工程考前模拟试题(一)..... (250)

软件工程考前模拟试题(一) 参考答案..... (253)

软件工程考前模拟试题(二)..... (257)

软件工程考前模拟试题(二) 参考答案..... (261)

软件工程考前模拟试题(三)..... (264)

软件工程考前模拟试题(三) 参考答案..... (267)

软件工程考前模拟试题(四)..... (273)

软件工程考前模拟试题(四) 参考答案..... (277)

软件工程考前模拟试题(五)..... (282)

软件工程考前模拟试题(五) 参考答案..... (285)

2001 年 10 月全国高等教育自学考试软件工程试题及参考答案 ..... (288)

# 软件工程考试概述

软件工程是高等教育自学考试计算机及应用专业(独立本科段)考试计划中的专业课程。计算机及应用专业的培养目标是在各个领域建立计算机应用系统,软件开发是建立计算机应用系统的重要环节,因此必须掌握软件工程的基本概念、基本原理、基本方法与技术、基本过程。

软件工程是计算机及应用专业的一门工程性课程,主要讲述建造软件系统的方法、技术、流程、工具、规范等,本课程的任务是使应考者掌握软件工程的基本概念、基本原理、实用的开发方法和技术;了解软件工程各领域的发展动向;如何用工程化的方法开发软件项目,以及在开发过程中应遵循的流程、准则、标准和规范。

本课程是一门实践性很强的课程,它是各种软件开发经验的总结与提炼,应考者不但应注重概念、原理、方法、技术的掌握,也应注重方法、技术的实际应用。

学习本课程要求有一定的程序设计经验,因此,应至少学习过一门程序设计语言课程。本课程还涉及到数据结构、数据库、操作系统等的一些知识和概念,因此先导课程为数据结构、数据库原理、操作系统。

从总体了解软件工程的产生、软件生产发展史、软件生存周期、各种方法和生存周期模型、软件工程面临的问题;系统掌握软件开发最基本的内容:可行性研究和软件计划、需求分析、概要设计、详细设计、编码、测试、维护,系统掌握这些阶段的目标、任务、特点、步骤和文档;掌握增量模型的基本内容、基本思想、运行机制、开发过程和步骤;掌握结构化方法的基本思想,开发过程和步骤,应遵循的原则和准则,能够应用相应的图形表示工具开发小型软件项目;初步掌握面向对象方法的基本思想、基本概念、基本模型,面向对象分析、面向对象设计、面向对象实现的任务、内容和步骤,能够应用相应的图形工具;了解 Jackson 方法的基本思想、开发过程和步骤,能够应用相应的图形表示工具;了解软件工程的质量保证中的各种基本概念和方法,了解软件工程管理中的思想和方法,了解软件开发环境与工具;要求切实掌握课程内容的基本思想、基本概念、基本方法,能够应用相应的图形表示工具进行小型项目的开发,特别是实用方法与技术的应用。

在全面系统学习的基础上掌握基本理论、基本知识、基本方法。本课程从软件开发、维护和软件管理等方面系统地阐述了软件工程的基本概念和常用的方法,各章之间既有联系又有很大区别,有的还有相对独立性。应考者应首先全面系统地学习各章,记忆应当识记的基本概念、名词,深入理解基本理论,弄懂基本方法的内涵;其次要认识各章之间的联系,注意区分相近的概念和相似的问题,并掌握它们之间的联系;再次在全面系统学习的基础上掌握重点,有目的地深入学习重点章节,但切忌在没有全面学习教材的情况下孤立地去抓重点。

把学习软件工程理论和应用软件工程的方法结合起来。应考者应在学习软件工程理论的同时掌握软件工程方法。首先要弄懂各种方法所包含的内容和各组成要素之间的关系;其次要学会正确运用和应用这些方法去分析和解决有关的软件问题。重视理论联系

实际,结合软件开发全过程的实践来进行学习。应考者在学习应把课程的内容同实际软件开发联系起来,进行对照比较,分析研究,以增强感性认识,将知识转化为能力,提高自己分析问题与解决问题的能力。

本课程的命题考试要覆盖到各章,并适当突出重点章节,体现本课程的内容重点。本课程在试题中对不同能力层次要求的分数比例,一般为:识记占 20%;领会占 30%;简单应用占 30%;综合应用占 20%。试题难易度可分为易、较易、较难和难四个等级。每份试卷中,不同难易度试题的分数比例为:易占 20%;较易占 30%;较难占 30%;难占 20%。必须注意,试题的难易度与能力层次不是一个概念,在各能力层次中都会存在不同难度的问题,切勿混淆。本课程考试试卷采用的题型,一般有:名词解释、填空、单项选择、简答、应用等五种题型。本课程考试方式为闭卷、笔试,考试时间为 150 分钟。试卷份量应以中等水平的考生在规定时间内完成全部试题为标准。评分采用百分制,60 分及格。

(一)名词解释

例 1. 软件开发环境

答:软件开发环境是指在计算机的基本软件的基础上,为了支持软件的开发而提供的一组工具软件系统。在 1985 年第八届国际软件工程会议上,一个由 IEEE 和 ACM 支持的国际工作小组提出的关于“软件开发环境”的定义是:“软件开发环境是相关的一组软件工具集合,它支持一定的软件开发方法或按照一定的软件开发模型组织而成。”

例 2. 软件概要设计

答:进入了设计阶段,要把软件“做什么”的逻辑模型变换为“怎么做”的物理模型,即着手实现软件的需求,并将设计的结果反映在“设计规格说明书”文档中,所以软件设计是一个把软件需求转换为软件表示的过程,最初这种表示只是描述了软件的总的体系结构,称为软件概要设计或结构设计。

例 3. 黑盒法

答:该方法把被测试对象看成一个黑盒子,测试人员完全不考虑程序的内容结构和处理过程,只在软件的接口处进行测试,依照需求规格说明书,检查程序是否满足功能要求。因此,黑盒测试又称为功能测试或数据驱动测试。

(二)填空

例 4. 利用“数据字典的定义式中出现的符号”,试解释以下定义的数据流组成及数据项:

姓名 = { 字母 }<sub>2</sub><sup>18</sup>,表示:\_\_\_\_\_。

终点 = [ 上海 | 北京 | 西安 ],表示:\_\_\_\_\_。

答: 姓名中最少出现 2 次“字母”,最多出现 18 次“字母”

终点可以是上海或北京或西安

例 5. 数据流图有四种基本图形符号:箭头表示\_\_\_\_\_;圆或椭圆表示\_\_\_\_\_,或可以用\_\_\_\_\_符号表示;双杠表示\_\_\_\_\_,还可以用\_\_\_\_\_符号表示;方框表示数据的\_\_\_\_\_或\_\_\_\_\_还可以用\_\_\_\_\_符号表示。

答:数据流    加工    

编号
----

    数据存储    

编号
----

    源点    终点    

--

例 6. 按国际 GB8576 - 88 的“计算机软件产品开发文件编制指南”规定,软件设计文档可分为“\_\_\_\_\_”、“\_\_\_\_\_”、“\_\_\_\_\_”。



(三) 单项选择题

例 7. 软件维护活动总的工作量由  $M = P + K \cdot \exp(C - D)$  表示, 对于该式的错误说法是 (     )

- A. 若 C 越大, D 越小, 那么维护工作量将成指数增加
- B. C 增加表示从而使得软件为非结构化设计
- C. D 表示维护人员不是原来的开发人员
- D. K 表示生产性活动工作量

答: D

例 8. 以下说法错误的是 (     )

- A. PAD 图支持逐步求精的设计方法
- B. 程序流程图往往反映的是最后的结果
- C. 程序流程图容易造成非结构化的程序结构
- D. PAD 图支持结构化的程序设计原理
- E. 程序流程图不易表示数据结构
- F. 程序流程图清晰反映了逐步求精的过程

答: F

例 9. 对于软件测试时需要的三类信息, 以下完全正确的解释是 (     )

- A. 软件配置: 指需求规格说明书、设计说明书、测试用例等
- B. 测试配置: 指测试方案、测试驱动程序、源程序等。
- C. 测试工具: 指计算机辅助测试的有关工具

答: C

例 10. 对于原型的使用建议, 以下说法不正确的是 (     )

- A. 开发周期很长的项目, 能够使用原型
- B. 在系统的使用可能变化较大, 不能相对稳定时, 能够使用原型
- C. 缺乏开发工具, 或对原型的可用工具不了解的时候, 能够使用原型
- D. 开发者对系统的某种设计方案的实现无信心或无十分的把握, 能够使用原型

答: C

(四) 简答题

例 11. 写出过程设计语言的三种重复结构格式。

答:    FOR 结构

```
FOR i= 1 TO n
    循环体
ENDFOR
```

WHILE 结构

```
WHILE 条件
    循环体
ENDWHILE
```

UNTIL 结构

```
REPEAT
```

循环体  
UNTIL 条件

例 12. 请叙述集成化 CASE 的五级模型。

答: 1990 年 Wasserman 讨论软件工程环境的集成时, 提出一个五级模型:

平台集成。“平台”或是一个单一的计算机或操作系统或是一个网络系统。平台集成是指工具或工作台在相同的平台上运行。

数据集成。指不同软件工程能相互交换数据。因而, 一个工具的结果能作为另一个工具的输入。最简单的数据集成形式是基于一个共享文件集的集成。

表示集成。表示集成或用户界面集成意指一个系统中的工具使用共同的风格, 以及采用共同的用户交互标准集。工具有一个相似的外观。目前, 表示集成有如下三种不同级别: 窗口系统集成、命令集成、交互集成。

控制集成。控制集成支持工作台或环境中一个工具对系统中其它工具的访问。除了能启动和停止其它工具外, 一个工具能调用系统中另一个工具所提供的服务, 这些服务可通过一个程序接口来访问。

过程集成。过程集成意指 CASE 系统嵌入了关于过程活动、阶段、约束和支持这些活动所需的工具的知识。

例 13. 提高可维护性的方法有哪些?

- 答: 建立明确的软件质量目标。
- 使用先进的软件开发技术和工具。
- 建立明确的质量保证。
- 选择可维护性的程序设计语言。
- 改进程序的文档。

(五)应用题

例 14. 请用判定表画出以下问题的行为逻辑。

人们往往根据天气情况决定出门时的行装: 天气可能下雨, 也可能不下雨; 天气可能变冷, 也可能不变冷。如果天气要下雨, 出门时带上雨伞; 如果天气变冷, 出门时要穿上大衣。

答:

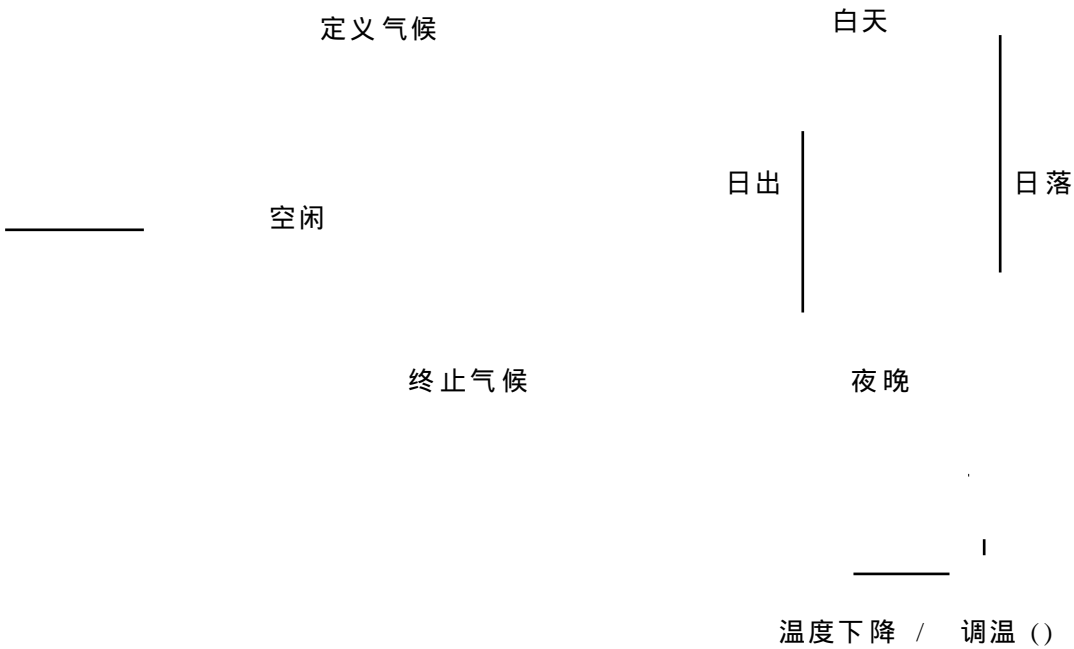
天气情况	下雨		不下雨	
	变冷	不变冷	变冷	不变冷
带雨伞				
穿大衣				

例 15. 在温室管理系统中, 有一个环境控制器类, 当没有种植作物时处于空闲状态。一旦种上作物, 就要进行温度控制, 定义气候, 即在什么时期应达到什么温度。当处于夜晚时, 由于温度下降, 要调用调节温度过程, 以便保持温度; 太阳出来时, 进入白天状态, 由于温度升高, 要调用调节温度过程, 保持要求的温度。当日落时, 进入夜晚状态。当作物收获, 终止气候的控制, 则进入空闲状态。建立环境控制器类的状态图。

答: 温室管理系统中的环境控制器类的状态图如图所示。

温度上升 / 调温 ()

-

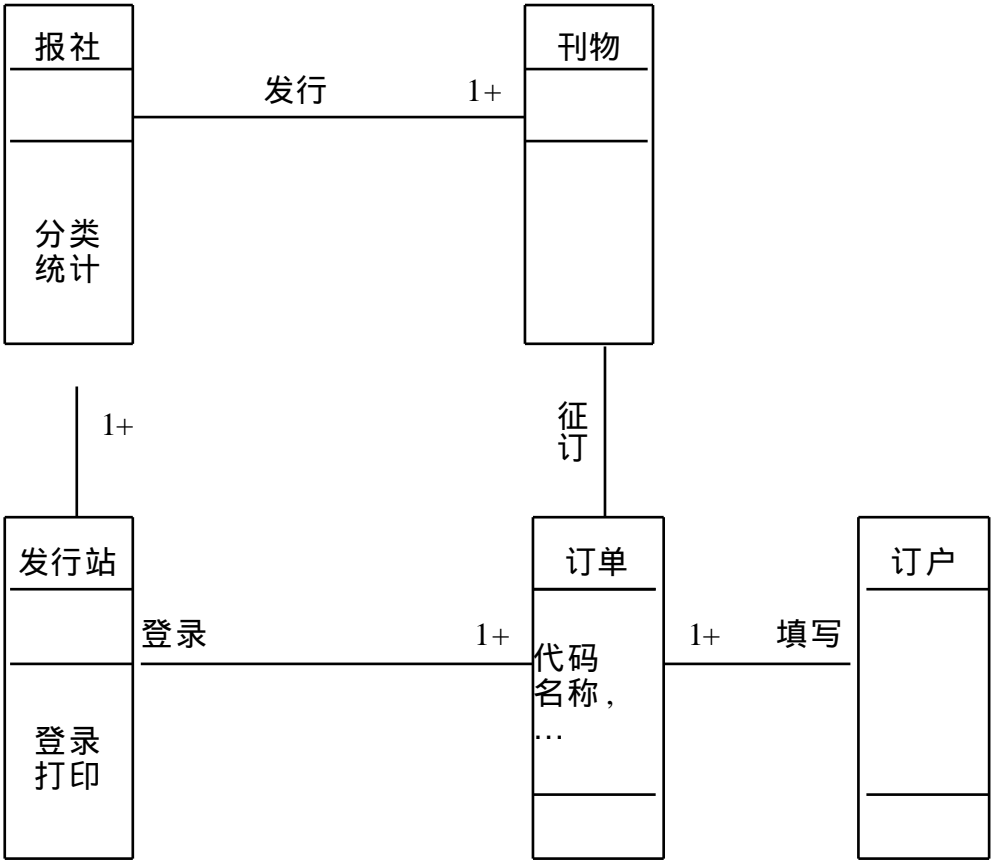


温室管理系统环境控制器类的状态图

例 16. 某报社采用面向对象技术实现报刊征订的计算机管理系统,该系统基本需求如下:

- (1) 报社发行多种刊物,每种刊物通过订单来征订,订单中有代码、名称、订期、单价、份数等项目,订户通过填写订单来订阅报刊。
  - (2) 报社下属多个发行站,每个站负责收集登录订单、打印收款凭证等事务。
  - (3) 报社负责分类并统计各个发行站送来的报刊订阅信息。
- 请就此需求建立对象模型。

答:



# 第一章 绪 论

本章总的要求是:从总体上了解软件工程的基本概念和内容;软件工程过程和生存期的基本概念和内容;软件开发的各种方法和生存周期模型。

了解软件的特点,软件生产发展的三个阶段,各阶段的特点,软件危机的产生及其表现形式;初步了解软件的生存周期模型、开发方法和工具。

理解软件工程的定义、性质、特点、目标;理解软件生存期各阶段的特点和内容。

深刻理解软件危机产生的原因,以及软件工程面临的各种问题。

## 考试要求

### 1. 软件工程的产生

软件的特点,要求达到识记层次。

软件生产的发展,要求达到识记层次。

软件危机的产生、表现、原因,要求达到领会层次。

软件工程的定义、性质、目标、内容、面临的问题,要求达到领会层次。

### 2. 软件工程过程和软件生存周期

软件工程过程概念,要求达到识记层次。

软件生存周期概念,要求达到识记层次。

### 3. 软件生存周期模型、方法和工具

#### (1) 软件生存周期模型

软件生存周期模型的定义、重要性,要求达到识记层次。

软件生存周期模型的作用、准则,要求达到识记层次。

瀑布模型、增量模型、螺旋模型、喷泉模型、变换模型、基于知识的模型,要求达到识记层次。

#### (2) 软件开发方法

软件开发目标,要求达到识记层次。

方法的作用和重要性,要求达到识记层次。

结构化方法、Jackson 方法、维也纳方法、面向对象方法,要求达到识记层次。

#### (3) 软件开发工具

工具的重要性,要求达到识记层次。

工具箱,要求达到识记层次。

开发环境,要求达到识记层次。

计算机辅助软件工程,要求达到识记层次。

# 知识重点

## (一) 软件工程的产生

软件的生产经过了三个阶段:程序设计时代,程序系统时代,软件工程时代。

软件发展第二阶段的末期,一些复杂的、大型的软件开发项目提出来了,但是,软件开发技术的进步一直未能满足发展的要求。在软件开发中遇到的问题找不到解决的办法,使问题积累起来,形成了尖锐的矛盾,导致了软件危机。

为了克服软件危机,人们提出“软件工程”的概念,要用工程化的思想来开发软件。

## (二) 软件工程的定义

软件工程是用科学知识和技术原理来定义、开发、维护软件的一门综合性的交叉学科。

软件工程的目的是成功地建造一个大型软件系统,所谓成功是要达到以下几个目标:付出较低的开发成本;达到要求的软件功能;取得较好的软件性能;开发的软件易于移植;需要较低的维护费用;能按时完成开发任务;及时交付使用;开发的软件可靠性高。

软件工程研究的主要内容是软件开发技术和软件开发管理两个方面。在软件开发技术中,主要研究软件开发方法、软件开发过程、软件开发工具和环境。在软件开发管理中,主要是研究软件管理学、软件经济学、软件心理学等。

## (三) 软件工程过程

- 1.获取过程。定义需方按合同获取一个系统、软件产品或服务活动。
- 2.供应过程。定义供方向需方提供合同中的系统、软件产品或服务所需的活动。
- 3.开发过程。定义开发者和机构为了定义和开发软件或提供服务所需的活动。此过程包括需求分析、设计、编码、集成、测试、软件安装和验收等活动。
- 4.操作过程。定义操作者和机构为了在规定的运行环境中为其用户运行一个计算机系统所需要的活动。
- 5.维护过程。定义维护者和机构为了修改和管理软件,使它处于良好运行状态所需要的活动。
- 6.管理过程。定义软件工程过程中各项管理活动,包括:项目开始和范围定义,项目管理计划,实施和控制,评审和评价,项目完成。
- 7.支持过程。支持过程对项目的生存周期过程给予支持。它有助于项目的成功并能提高项目的质量。

## (四) 软件生存周期

软件生存周期是指一个软件从提出开发要求开始直到该软件报废为止的整个时期。把整个生存周期划分为若干阶段,使得每个阶段有明确的任务,使规模大、结构复杂和管

理复杂的软件开发变得容易控制和管理。

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。软件生存周期模型确立了软件开发和演绎中各阶段的次序限制以及各阶段活动的准则,确立开发过程所遵守的规定和限制,便于各种活动的协调,便于各种人员的有效通信,有利于活动重用,有利于活动管理。目前有若干种软件生存周期模型。例如瀑布模型、增量模型、螺旋模型、喷泉模型、变换模型和基于知识的模型等。

### (五) 软件开发方法

软件开发方法是一种使用早已定义好的技术集及符号表示习惯来组织软件生产的过程。方法一般表述成一系列的步骤,每一步骤都与相应的技术和符号相关。

软件开发的目标是在规定的投资和时间内,开发出符合用户需求的高质量的软件。为了达到此目的,需要成功的开发方法。软件开发方法是克服软件危机的重要方面之一。从软件工程诞生以来,已经提出了多种软件开发方法和技术,结构化方法、Jackson 方法、维也纳开发方法(VDM)、面向对象的开发方法等对软件工程及软件产业的发展起到了不可估量的作用。

## 反馈测试题解

### 一、名词解释

#### 1. 文档

答:有关计算机程序的功能、设计、编制、使用的文字或图形资料。

#### 2. 软件

答:计算机程序及其说明程序的各种文档。

#### 3. 结构化分析

答:结构化分析是根据分解与抽象的原则,按照系统中数据处理的流程,用数据流图来建立系统的功能模型,从而完成需求分析工作。结构化设计是根据模块独立性准则、软件结构准则将数据流图转换为软件的体系结构,用软件结构图来建立系统的物理模型,实现系统的概要设计。结构化程序设计是根据结构程序设计原理,将每个模块的功能用相应的标准控制结构表示出来,从而实现详细设计。

#### 4. 软件工程

答:软件工程有多种定义,其中一种是:用科学知识和技术原理来定义、开发、维护软件的一门学科。

#### 5. 需求分析

答:需要分析阶段的任务不是具体地解决问题,而是准确地确定软件系统必须做什么,确定软件系统必须具备哪些功能。

用户了解他们所面对的问题,知道必须做什么,但是通常不能完整、准确地表达出来,也不知道怎样用计算机解决他们的问题。软件开发人员知道怎样用软件完成人们提出的各种功能要求,但是,对用户的具体业务和需求不完全清楚,这是需求分析阶段的困难所在。

系统分析员要和用户密切配合,充分交流各自的理解,充分理解用户的业务流程,完整地、全面地收集、分析用户业务中的信息和处理,从中分析出用户要求的功能和性能,完整地、准确地表达出来。这一阶段要给出软件需求说明书。

## 6. 概要设计

答:在概要设计阶段,开发人员要把确定的各项功能需求转换成需要的体系结构。在该体系结构中,每个成分都是意义明确的模块,即每个模块都和某些功能需求相对应,因此,概要设计就是设计软件的结构,明确该结构由哪些模块组成,这些模块的层次结构是怎样的,这些模块的调用关系是怎样的,每个模块的功能是什么。同时还要设计该项目的应用系统的总体数据结构和数据库结构,即应用系统要存储什么数据,这些数据是什么样的结构,它们之间有什么关系等。

## 7. 详细设计

答:详细设计阶段就是为每个模块完成的功能进行具体描述,要把功能描述转变为精确的、结构化的过程描述。即该模块的控制结构是怎样的,先做什么,后做什么,有什么样的条件判定,有什么重复处理等等,并用相应的表示工具把这些控制结构表示出来。

## 8. 编码

答:编码阶段就是把每个模块的控制结构转换成计算机可接受的程序代码,即写成以某种特定程序设计语言表示的“源程序清单”。当然,写出的程序应是结构好,清晰易读,并且与设计相一致。

## 9. 测试

答:测试是保证软件质量的重要手段,其主要方式是在设计测试用例的基础上检验软件的各个组成部分。测试分为单元测试、集成测试、确认测试。集成测试是查找各模块的功能和结构上存在的问题。集成测试是将各模块按一定顺序组装起来进行测试,主要是查找各模块之间接口上存在的问题。确认测试是按需求说明书上的功能逐项测试,发现不满足用户需求的问题,决定开发的软件是否合格,能否交付用户使用。

## 10. 维护

答:软件维护是软件生存周期中时间最长的阶段。已交付的软件投入正式使用后,便进入软件维护阶段,它可以持续几年甚至几十年。软件运行过程中可能由于各方面的原因,需要对其进行修改。其原因可能是运行中发现了软件隐含的错误而需要修改;也可能是为了适应变化了的软件工作环境而需要做适当变更;也可能是因为用户业务发生变化而需要扩充和增强软件的功能等。

## 11. 增量模型

答:瀑布模型是一种整体开发模型。在开发过程中,用户看不到软件是什么样子,只有开发完成后,整个软件全部展现在用户面前。这时如果用户发现有不满意的地方,为时已晚。

增量模型是一种非整体开发的模型。软件在该模型中是“逐渐”开发出来的,开发出一部分,向用户展示一部分,可让用户及早看到部分软件,及早发现问题。或者先开发一个“原型”软件,完成部分主要功能,展示给用户并征求意见,然后逐步完善,最终获得满意的软件产品。该模型具有较大的灵活性,适合于软件需求不明确,设计方案有一定风险的软件项目。

## 12. 喷泉模型

答:喷泉模型是一种以用户需求为动力,以对象作为驱动力的模型,适合于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。系统某些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、实现等开发活动之间不存在明显的边界。

## 13. 基于知识的模型

答:基于知识的模型又称智能模型,它把瀑布模型和专家系统结合在一起。该模型在开发的各个阶段上都利用相应的专家系统来帮助软件人员完成开发工作,使维护在系统需求说明一级上进行。为此,建立了各阶段所需要的知识库,将模型、相应领域知识、软件工程知识分别存入数据库。以软件工程知识为基础的生成规则构成的专家系统与含有应用领域知识规则的其他专家系统相结合,构成了该应用领域的开发系统。

该模型还处于研究实验阶段,还未达到实用阶段。

#### 14. 变换模型

答:这是一种适合于形式化开发方法的模型。从软件需求形式化说明开始,经过一系列变换,最终得到系统的目标程序。

#### 15. 维也纳开发方法(VDM)

答:这是一种形式化的开发方法,软件的需求用严格的形式语言描述,然后把描述模型逐步变换成目标系统。VDM是一个基于模型的方法,它以指称语义为基础。它的主要思想是将软件系统当作模型来给予描述,把软件的输入、输出看作模型对象,把这些对象在计算机内的状态看作该模型在对象上的操作。它的目的是从软件系统最高一级抽象直到最后生成目标的每一步都给予形式化说明,以此提高软件的可靠性。自70年代初提出以来,它已成为一种大型系统软件的形式化开发方法,具有较大的潜力,在欧洲及北美有相当大的影响,到80年代已将它应用到工程开发上。

#### 16. 面向对象的开发方法

答:面向对象开发方法的基本出发点是尽可能按照人类认识世界的方法和思维方式来分析和解决问题。客观世界是由许多具体的事物、事件、概念和规则组成,这些均可被看成对象,面向对象方法正是以对象作为最基本的元素,它也是分析问题、解决问题的核心。由此可见,面向对象方法自然符合人类的认识规律。计算机实现的对象与真实世界的对象有一一对应的关系,不必做任何转换,这就使面向对象易于为人们所理解、接受和掌握。面向对象开发方法包括面向对象分析、面向对象设计、面向对象实现。面向对象开发方法有Booch方法、Coad方法和OMT方法等。为了统一各种面向对象方法的术语、概念和模型,1997年推出了统一建模语言,即UML(Unified Modeling Language)语言。它是面向对象的标准建模语言,通过统一的语义和符号表示,使各种方法的建模过程和表示统一起来,将成为面向对象建模的工业标准。

#### 17. 结构化方法

答:结构化方法由结构化分析、结构化设计、结构化程序设计构成。它是一种面向数据流的开发方法。该方法简单实用,应用较广,技术成熟。

#### 18. 软件开发方法

答:软件开发方法是一种使用早已定义好的技术集及符号表示习惯来组织软件生产的过程。

#### 19. 工具箱

答:最初的软件工具是以工具箱的形式出现的,一种工具支持一种开发活动,然后将各种工具简单组合起来就构成工具箱。但是,工具箱的工具界面不统一,工具内部无联系,工具切换由人工操作。因此,它们对大型软件的开发和维护的支持能力是有限的,即使可以使用众多的软件工具,但由于这些工具之间相互隔离、独立存在,无法支持一个统一的软件开发和维护过程。

#### 20. 软件开发环境

由于工具箱存在的问题,人们在工具系统的整体化及集成化方面展开一系列研究工作,使之形成完整的软件开发环境。其目的是使软件工具支持整个生存周期,不仅能支持软件开发和维护中的个别阶段,而且能支持从项目开发计划、需求分析、设计、编码、测试到维护等所有阶段,做到不仅支持各阶段中的技术工作,还要支持管理和操作工作,保持项目开发的高度可见性、可控制性和可追踪性。

#### 21. 软件的特点

答:它是一种逻辑产品;软件的生产主要是研制;软件产品不会用坏;软件生产主要是脑力劳动;软件的成本高。

#### 22. 瀑布模型

答:瀑布模型是将软件生存周期各个活动规定为依线性顺序连接的若干阶段的模型。它包括可行性分析、项目开发计划、需求分析、概要设计、详细设计、编码、测试和维护。它规定了由前至后、相互衔



接的固定次序,如同瀑布流水,逐级下落。

### 23. 软件生存周期模型

答:模型是为了理解事物而对事物做出一种抽象,它忽略不必要的细节,它也是事物的一种抽象形式、一个规划、一个程式。

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。

### 24. 软件生存周期

答:软件生存周期是指一个软件从提出开发要求开始直到该软件报废为止的整个时期。把整个生存周期划分为若干阶段,使得每个阶段有明确的任务,使规模大、结构复杂和管理复杂的软件开发变得容易控制和管理。

## 二、填空题

1 面向对象开发方法有\_\_\_\_\_、\_\_\_\_\_和\_\_\_\_\_。

答:Booch 方法      Coad 方法      OMT 方法

2 面向对象开发方法包括\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_三部分。

答:面向对象分析    面向对象设计    面向对象实现

3 维也纳开发方法(VDM)是一种形式化的开发方法,软件的需求用\_\_\_\_\_描述,然后把描述模型逐步变换成目标系统。

答:严格的形式语言

4 结构化方法总的指导思想是\_\_\_\_\_。它的基本原则是功能的\_\_\_\_\_与\_\_\_\_\_。它是软件工程最早出现的开发方法,特别适合于\_\_\_\_\_的问题。

答:自顶向下、逐步求精    分解    抽象    数据处理领域

5 从软件工程诞生以来,已经提出了多种软件开发方法,如\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_,它们对软件工程及软件产业的发展起到了不可估量的作用。

答:结构化方法    Jackson 方法    维也纳开发方法(VDM)    面向对象的开发方法

6 变换模型是一种适合于\_\_\_\_\_方法的模型。从\_\_\_\_\_开始,经过一系列\_\_\_\_\_,最终得到系统的目标程序。

答:形式化开发    软件需求形式化说明    变换

7 喷泉模型是一种以用户需求为动力,以\_\_\_\_\_作为驱动力的模型,适合于\_\_\_\_\_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\_\_\_\_\_和\_\_\_\_\_。

答:对象    面向地象    迭代性    无间隙性

8 螺旋模型将开发过程分为几个螺旋周期,在每个螺旋周期内分为四个工作步骤。第一,\_\_\_\_\_。确定目标,选定实施方案,明确开发限制条件。第二,\_\_\_\_\_。分析所选主案,识别风险,通过原型消除风险。第三,\_\_\_\_\_。实施软件开发。第四,\_\_\_\_\_。评价开发工作,提出修改意见,建立下一个周期的计划。

答:制定计划    风险分析    开发实施    用户评估

9 \_\_\_\_\_是一种非整体开发的模型。软件在该模型中是“逐渐”开发出来的,开发出一部分,向用户展示一部分,可让用户及早看到部分软件,及早发现问题。或者先开发一个“原型”软件,完成部分主要功能,展示给用户并征求意见,然后逐步完善,最终获得满意的软件产品。

答:增量模型

10 螺旋模型将\_\_\_\_\_与\_\_\_\_\_结合起来,加入了两种模型均忽略了的风险分析,弥补了

这两种模型的不足。

答:瀑布模型 增量模型

11 瀑布模型是将软件生存周期各个活动规定为依线性顺序连接的若干阶段的模型。它包括\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_,它规定了由前至后、相互衔接的固定次序,如同瀑布流水,逐级下落。

答:可行性分析、项目开发计划 需求分析 概要设计 详细设计 编码 测试 维护

12 软件生存周期是指一个软件从提出开发要求开始直到该软件报废为止的整个时期。通常,软件生存周期包括\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_等活动,可以将这些活动以适当方式分配到不同阶段去完成。

答:可行性分析和项目开发计划 需求分析 概要设计 详细设计 编码 测试 维护

13 .软件 工程 过程 包含 的 七 个 过 程 是 \_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_。

答:获取过程 供应过程 开发过程 操作过程 维护过程 管理过程 支持过程

14 .\_\_\_\_是描述软件开发过程中各种活动如何执行的模型。

答:软件生存周期模型

15 .软件工程需要解决的问题,有\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_和\_\_\_\_。

答:软件费用 软件可靠性 软件可维护性 软件生产率 软件重用

16 .软件工程研究的主要内容是软件开发技术和软件开发管理两个方面。在软件开发技术中,主要研究\_\_\_\_、\_\_\_\_和\_\_\_\_。在软件开发管理中,主要是研究\_\_\_\_、\_\_\_\_、\_\_\_\_。

答:软件开发方法 软件开发过程 软件开发工具和环境 软件管理学 软件经济学 软件心理学

17 .软件工程研究的主要内容是\_\_\_\_和\_\_\_\_两个方面。

答:软件开发技术 软件开发管理

18 .软件生产的发展,到现在为止,经过了三个阶段,即\_\_\_\_,\_\_\_\_,\_\_\_\_。

答:程序设计时代 程序系统时代 软件工程时代

19 .软件工程要用\_\_\_\_的方法建立软件开发中的各种模型和各种算法,如可靠性模型,说明用户需求的形式化模型等。

答:数学

20 .计算机科学中的研究成果均可用于软件工程,但计算机科学着重于\_\_\_\_,而软件工程着重于\_\_\_\_。

答:原理和理论 如何建造一个软件系统

21 .软件工程要用工程科学中的观点来进行\_\_\_\_、\_\_\_\_、\_\_\_\_。

答:费用估算 制定进度 制定计划和方案

22 .软件工程是用科学知识和技术原理来\_\_\_\_、\_\_\_\_、\_\_\_\_软件的一门学科。

答:定义 开发 维护

23 软件工程是一门综合性的交叉学科,它涉及\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_等领域。

答:计算机科学 工程科学 管理科学 数学

24 .JSP 方法是以\_\_\_\_为驱动的,适合于小规模的项目。

答:数据结构

25 .为了克服软件危机,人们从其他产业的工程化生产得到启示,于是在 1968 年北大西洋公约组

织的工作会议上首先提出“\_\_\_\_\_”的概念,提出要用工程化的思想来开发软件,从此,软件生产进入软件工程时代。

答:软件工程

26. \_\_\_\_\_是一个完整的系统开发方法。首先建立现实世界的模型,再确定系统的功能需求,对需求的描述特别强调操作之间的时序性。它是以事件作为驱动的,它是一种基于进程的开发方法,所以应用于时序特点较强的系统,包括数据处理系统和一些实时控制系统。

答:JSD方法

27. 通常,\_\_\_\_\_,可使用瀑布模型、增量模型、螺旋模型进行开发;\_\_\_\_\_可使用瀑布模型、增量模型进行开发;\_\_\_\_\_一般是采用喷泉模型,也可用瀑布模型、增量模型进行开发;而形式化的维也纳方法只能用变换模型进行开发。

答:结构化方法 Jackson 方法 面向对象的开发方法

28. \_\_\_\_\_为软件开发提供了一种有效的管理模式。根据这一模式制定开发计划,进行成本预算,组织开发力量,以项目的阶段评审和文档控制为手段有效地对整个开发过程进行指导,所以它是以文档作为驱动、适合于软件需求很明确的软件项目的模型。

答:瀑布模型

29. \_\_\_\_\_一般是指为了支持软件人员开发和维护活动而使用的软件。例如项目估算工具、需求分析工具、设计工具、编码工具、测试工具和维护工具等。

答:软件工具

30. 目前有若干种软件生存周期模型。例如\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_,变换模型和基于知识的模型等。

答:瀑布模型 增量模型 螺旋模型 喷泉模型

31. CASE 技术是\_\_\_\_\_和\_\_\_\_\_的结合,它不同于以前的软件技术,因为它强调了解决整个软件开发过程的效率问题,而不是解决个别阶段的问题。由于跨越了软件生存周期各个阶段,着眼于软件分析和设计以及实现和维护的自动化,从软件生存周期的两端解决了生产率问题。

答:软件工具 软件开发方法

32. JSP(Jackson Structure Programming)方法首先描述问题的输入、输出数据结构,分析其对应性,然后推出相应的程序结构,从而给出问题的\_\_\_\_\_。

答:软件过程描述

33. 简单实用、应用广泛、技术成熟的开发方法是\_\_\_\_\_。

答:结构化方法

34. 概要设计主要是把各项功能需求转换成系统的\_\_\_\_\_。

答:体系结构

35. 最基本、应用广泛、以文档为驱动、适用于开发功能明确的软件项目生存期模型是\_\_\_\_\_。

答:瀑布模型

36. 结构化方法是一种面向数据流的开发方法。由结构化分析、\_\_\_\_\_,结构化程序设计构成。

答:结构化设计

37. 构成一个完整计算机系统的两部分是硬件与\_\_\_\_\_。

答:软件

38. 软件工程研究的主要内容是软件开发管理和\_\_\_\_\_两个方面。

答:软件开发技术

39. 将软件生存周期各个活动规定为依线性顺序联接的若干阶段的模型是\_\_\_\_\_模型。

答:瀑布

40. 软件工具是支持软件开发人员的开发和维护活动而使用的\_\_\_\_\_。

答：软件

### 三、选择题

1 软件生存周期法中,用户的参与主要在 ( )

- A .软件定义期                      B 软件开发期  
C .软件维护期                      D 整个软件生存周期过程中

答: A

2 软件工程是一种( )分阶段实现的软件程序开发方法。

- A .自底向上                      B .自顶向下                      C .逐步求精                      D .面向数据流

答: B

3 软件生存周期模型的模型中,( )适合于大型软件的开发,它吸收了软件工程“演化”的概念。

- A .喷泉模型    B .基于知识的模型  
C .变换模型    D .螺旋模型

答:D

4 .( )是计算任务的处理对象和处理规则的描述。

- A .软件                      B .硬件                      C .文档                      D .程序

答:D

5 软件工程中描述生存周期模型的瀑布模型一般包括计划、( )、设计、编码、测试、维护等几个阶段。

- A .需求分析                  B .需求调查                  C .可行性分析                  D .问题定义

答:A

6 软件生存周期包括可行性分析和项目开发计划、需求分析、概要设计、详细设计、编码、( )、维护等活动。

- A .应用                      B .检测                      C .测试                      D .以上答案都不正确

答:C

7. ( )是有关计算机程序功能、设计、编制、使用的文字或图形资料。

- A 程序                      B 文档                      C 软件                      D 数据

答: B

8. 目前有若干种软件生存周期模型。例如瀑布模型、增量模型、螺旋模型。其中基于知识的模型也称 ( )

- A. 演化模型      B. 智能模型      C. 变换模型      D. 喷泉模型

答: B

9 软件生存周期模型有多种,下列选项中,( )不是软件生存周期模型。

- A 瀑布模型      B 增量模型      C 功能模型      D 螺旋模型

答：C

10. ( )是计算机程序及其说明程序的各种文档。

- A 软件                      B 数据                      C 文档                      D 程序

答:A

11. 软件生存周期中时间最长的是( )阶段。

- A. 需求分析      B. 概要设计      C. 测试      D. 维护

答:D

12. 软件是一种( )产品。

- A. 物质                      B. 逻辑                      C. 有形                      D. 消耗

答:B

13. 软件产品的开发主要是 ( )

- A. 复制                      B. 再生产                      C. 拷贝                      D. 研制

答:D

14. 准确地解决“软件系统必须做什么”是( )阶段的任务。

- A. 可行性研究              B. 需求分析              C. 详细设计              D. 编码

答:B

15. 与计算机科学的理论研究不同,软件工程是一门( )学科。

- A. 理论性                      B. 原理性                      C. 工程性                      D. 心理性

答:C

16.( )把瀑布模型和专家系统结合在一起,在开发的各个阶段上都利用相应的专家系统来帮助软件人员完成开发工作。

- A. 增量模型                      B. 螺旋模型                      C. 喷泉模型                      D. 智能模型

答:D

17.( )方法是以数据结构为驱动的,适合于小规模的项目。

- A. JSP                      B. JSD                      C. VDM                      D. Jackson

答:A

18.( )方法的基本出发点是尽可能按照人类认识世界的方法和思维方式来分析和解决问题。

- A. 结构化                      B. Jackson  
C. 维也纳开发                      D. 面向对象开发

答:D

19. 软件工程是计算机科学中的一个分支,其主要思想是在软件生产中用( )的方法代替传统手工方法。

- A. 工程化                      B. 现代化                      C. 科学                      D. 智能化

答:A

20.( )方法是一种面向数据结构的开发方法。

- A. 结构化                      B. Jackson  
C. 维也纳开发                      D. 面向对象开发

答:B

21. 需求分析阶段的任务是 ( )

- A. 具体地解决问题                      B. 确定软件系统必须做什么  
C. 设计软件的结构

答:B

22. 软件工程要用( )科学中的方法和原理进行软件生产的管理。

- A. 管理                      B. 工程                      C. 数学                      D. 计算机

答:A

23.( )是一种以用户需求为动力,以对象作为驱动力的模型,适合于面向对象的开发方法。

- A. 增量模型                      B. 螺旋模型                      C. 喷泉模型                      D. 智能模型

答:C

24. ( )方法是一种数据流的开发方法。

- A. 结构化                      B. Jackson                      C. 维也纳开发                      D. 面向对象开发

答:A

25. ( )是一种适合于形式化开发方法的模型。

- A. 螺旋模型                      B. 喷泉模型                      C. 智能模型                      D. 变换模型

答:D

26. 软件工程要用( )科学中的观点来进行费用估算、制定进度、制定计划和方案。

- A. 管理                      B. 工程                      C. 数学                      D. 计算机

答:B

27. 在软件开发中通常要花费( )的代价进行测试和排错。

- A. 20%                      B. 40 %                      C. 70%                      D. 5 %

答:B

28. 软件生存周期是借用( )中产品生存周期的概念而得来的。

- A. 工程                      B. 需求                      C. 计算机学科                      D. 数学

答:A

29. 软件工程要用( )的方法建立软件开发中的各种模型和各种算法。

- A. 管理                      B. 工程                      C. 数学                      D. 计算机

答:C

30. 支持过程对项目的( )过程给予支持。

- A. 操作                      B. 生存周期                      C. 开发                      D. 供应

答:B

31. 详细设计阶段的任务是 ( )

- A. 把功能描述转变为精确的、结构化的过程描述  
B. 设计软件的结构,明确该结构由哪些模型组成  
C. 把每个模块的控制结构转换成计算机可接受的程序代码

答:A

32. ( )方法是一个基于模型的方法。

- A. 结构化                      B. Jackson                      C. 维也纳开发                      D. 面向对象开发

答:C

33. 软件产品的生产主要是脑力劳动,软件产品的成本主要体现在软件的( )上。

- A. 复制                      B. 开发方式                      C. 开发和研制                      D. 磨损、消耗

答:C

34. 软件是一种逻辑产品,软件产品具有无形性,它是通过( )体现它的功能和作用的。

- A. 复制                      B. 计算机的执行  
C. 开发和研制                      D. 软件费用

答:B

35. ( )方法是以事件作为驱动的,它是一种基于进程的开发方法。

- A. JSP                      B. JSD                      C. VDM                      D. Jackson

答:B

36. ( )是将软件生存周期各个活动规定为依线性顺序连接的若干阶段的模型。

- A. 瀑布模型                      B. 增量模型                      C. 螺旋模型                      D. 喷泉模型

答:A

37. 准确地解决“ 软件系统必须做什么 ”是( )阶段的任务。  
A. 可行性研究                  B. 需求分析                  C. 详细设计                  D. 编码

答:B

38. 软件生存周期中时间最长的是( )阶段。  
A. 需求分析                  B. 概要设计                  C. 测试                  D. 维护

答:D

39. 软件开发费用只占整个软件系统费用的 ( )  
A. 1/ 2                  B. 1/ 3                  C. 1/ 4                  D. 2/ 3

答:B

40. 软件开发中大约要付出( ) % 的工作量进行测试和排错。  
A. 20                  B. 30                  C. 40                  D. 50

答:C

41. 软件重用的单位是 ( )  
A. 软件模块                  B. 性能                  C. 系统                  D. 功能

答:A

42. 软件工程着重于 ( )  
A. 理论研究                  B. 原理探讨  
C. 建造软件系统                  D. 原理的理论

答:C

43. 软件是一种( )产品。  
A. 物质                  B. 逻辑                  C. 工具                  D. 文档

答:B

44. 将每个模块的控制结构转换成计算机可接受的程序代码是( )阶段的任务。  
A. 编码                  B. 需求分析                  C. 详细设计                  D. 测试

答:A

45. 作坊式小团体合作生产方式的时代是( )时代。  
A. 程序设计                  B. 软件生产自动化                  C. 程序系统                  D. 软件工程

答:C

46. 软件工程与计算机科学性质不同,软件工程着重于 ( )  
A. 理论研究                  B. 原理探讨  
C. 建造软件系统                  D. 原理的理论

答:C

四、简答题

1 软件产品的特性是什么？

答: 软件产品具有以下一些独特的特性？

(1) 软件是一种逻辑产品,是看不见摸不着的,因而具有无形性,它是脑力劳动的结晶,它以程序和文档的形式出现,保存在作为计算机存储器的磁盘和光盘介质上,通过计算机的执行才能体现它的功能和作用。

(2) 软件产品的生产主要是研制,软件产品的成本主要体现在软件的开发和研制上,软件开发研制

完成后,通过复制就产生了大量软件产品。

(3)软件产品不会用坏,不存在磨损、消耗问题。

(4)软件产品的生产主要是脑力劳动,还未完全摆脱手工开发方式,大部分产品是“定做”的。

(5)软件费用不断增加,软件成本相当昂贵。软件的研制工作需要投入大量的、复杂的、高强度的脑力劳动,它的成本非常高。

## 2.什么是软件生存周期?它有哪些活动?

答:软件生存周期是指一个软件从提出开发要求开始直到该软件报废为止的整个时期。把整个生存周期划分为若干阶段,使得每个阶段有明确的任务,使规模大、结构复杂和管理复杂的软件开发变得容易控制和管理。

软件生存周期的各阶段有不同的划分。在划分软件生存周期阶段时,应遵循的一条基本原则是各阶段的任务应尽可能相对独立,同一阶段各项任务的性质尽可能相同,从而降低每个阶段任务的复杂程度,简化不同阶段之间的联系,有利于软件项目开发的组织管理。通常,软件生存周期包括可行性分析和项目开发计划、需求分析、概要设计、详细设计、编码、测试、维护等活动,可以将这些活动以适当方式分配到不同阶段去完成。

## 3.什么是软件生存周期模型?有哪些主要模型?

答: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。软件生存周期模型确立了软件开发和演绎中各阶段的次序限制以及各阶段活动的准则,确立开发过程所遵守的规定和限制,便于各种活动的协调,便于各种人员的有效通信,有利于活动重用,有利于活动管理。

主要的软件生存周期模型有瀑布模型、增量模型、螺旋模型、喷泉模型、变换模型和基于知识的模型。

## 4.什么是软件开发方法?有哪些主要方法?

答:软件开发方法是一种使用早已定义好的技术集及符号表示习惯来组织软件生产的过程。方法一般表述成一系列的步骤,每一步骤都与相应的技术和符号相关。软件开发方法是克服软件危机的重要方面之一。

从软件工程诞生以来,人们重视了软件开发方法的研究,已经提出了多种软件开发方法和技术,对软件工程及软件产业的发展起到了重要作用的方法有:结构化方法、Jackson 方法、维也纳开发方法(VDM)、面向对象的开发方法等。

## 5.软件工程面临的问题是什么?

答:软件工程面临的问题有软件费用、软件可靠性、软件可维护性、软件生产率和软件重用等。

(1)软件费用。软件生产基本上仍处于手工状态,软件是知识高度密集的技术的综合产物,人力资源远远不能适应软件这种迅速增长的社会要求,所以软件费用上升的势头必然还将继续下去。

(2)软件可靠性。在软件开发中,通常要花费 40% 的代价进行测试和排错,就这样还不能保证以后不再发生错误,为了提高软件可靠性,就要付出足够的代价。

(3)软件维护。统计数据表明:软件的维护费用占整个软件系统费用的  $\frac{2}{3}$ ,而软件开发费用只占整个软件系统费用的  $\frac{1}{3}$ 。因此,软件工程面临如何提高软件的可维护性,减少软件维护的工作量的问题。

(4)软件生产率。计算机的广泛应用使得软件的需求量大幅度上升,而软件的生产又处于手工开发的状态,软件生产率低下,使得各国都感到软件开发人员不足,这种趋势仍旧继续下去。

(5)软件重用。提高软件的重用性,对于提高软件生产率、降低软件成本有重要意义。当前的软件开发存在着大量重复的劳动,耗费了不少的人力资源。软件重用是软件工程中的一个重要研究课题。

## 6.软件工程目标是什么?

答:软件工程是一门工程性学科,目的是成功地建造一个大型软件系统,所谓成功是要达到以下几



个目标:付出较低的开发成本;达到要求的软件功能;取得较好的软件性能;开发的软件易于移植;需要较低的维护费用;能按时完成开发任务,及时交付使用;开发的软件可靠性高。

#### 7. 软件工程内容有哪些?

答:软件工程研究的主要内容是软件开发技术和软件开发管理两个方面。在软件开发技术中,主要研究软件开发方法、软件开发过程、软件开发工具和环境。在软件开发管理中,主要是研究软件管理学、软件经济学、软件心理学等。

#### 8. 软件工程面临的问题有哪些?

答:摆在软件工程面前有许多需要解决的棘手问题,如软件费用、软件可靠性、软件可维护性、软件生产率和软件重用等。

##### (1) 软件费用

由于软件生产基本上仍处于手工状态,软件是知识高度密集的技术的综合产物,人力资源远远不能适应软件这种迅速增长的社会要求,所以软件费用上升的势头必然还将继续下去。

##### (2) 软件可靠性

软件可靠性是指软件系统能否在既定的环境条件下运行并实现所期望的结果。在软件开发中,通常要花费 40% 的代价进行测试和排错,就这样还不能保证以后不再发生错误,为了提高软件可靠性,就要付出足够的代价。

##### (3) 软件维护

统计数据表明:软件的维护费用占整个软件系统费用的  $\frac{2}{3}$ ,而软件开发费用只占整个软件系统费用的  $\frac{1}{3}$ 。之所以有如此大的花费,因为已经运行的软件还需排除隐含的错误,新增加的功能要加入进去,维护工作又是非常困难的,效率又是非常低下的。因此,如何提高软件的可维护性,减少软件维护的工作量,也是软件工程面临的主要问题之一。

##### (4) 软件生产率

计算机的广泛应用使得软件的需求量大幅度上升,而软件的生产又处于手工开发的状态,软件生产率低下,使得各国都感到软件开发人员不足,这种趋势仍旧继续下去。所以,如何提高软件生产率是软件工程又一重要问题。

##### (5) 软件重用

提高软件的重用性,对于提高软件生产率、降低软件成本有重要意义。当前的软件开发存在着大量重复的劳动,耗费了不少的人力资源。软件的重用有各种级别,软件规格说明、软件模块、软件代码、软件文档等都可以是软件重用的单位。软件重用是软件工程中的一个重要研究课题,软件重用的理论和技术课题至今尚未彻底解决。

#### 9. 软件开发的目标是什么?

答:软件开发的目标是在规定的投资和时间内,开发出符合用户需求的高质量的软件。为了达到此目的,需要成功的开发方法。

#### 10. 结构化方法总的指导思想是什么?

答:结构化方法总的指导思想是自顶向下、逐步求精。它的基本原则是功能的分解与抽象。它是软件工程中最早出现的开发方法,特别适合于数据处理领域的问题。相应的支持工具较多,发展较为成熟。

#### 11. 软件工程过程有哪些内容?

答:软件工程过程规定了获取、供应、开发、操作和维护软件时,要实施的过程、活动和任务。其目的是为各种人员提供一个公共的框架,以使用相同的语言进行交流。

这个框架由几个重要过程组成,这些主要过程含有用来获取、供应、开发、操作和维护软件所用的基本的、一致的要求。该框架还有用来控制和管理软件的过程。各种组织和开发机构可以根据具体情况

进行选择 and 剪裁。可在一个机构的内部或外部实施。

软件工程过程没有规定一个特定的生存周期模型或软件开发方法,各软件开发机构可为其开发项目选择一种生存周期模型,并将软件工程过程所含的过程、活动和任务映射到该模型中。也可以选择和使用软件开发方法来执行适合于其软件项目的活动和任务。软件工程过程包含如下七个过程。

- (1)获取过程。定义需方按合同获取一个系统、软件产品或服务的活动。
- (2)供应过程。定义供方向需方提供合同中的系统、软件产品或服务所需的活动。
- (3)开发过程。定义开发者和机构为了定义和开发软件或提供服务所需的活动。此过程包括需求分析、设计、编码、集成、测试、软件安装和验收等活动。
- (4)操作过程。定义操作者和机构为了在规定的运行环境中为其用户运行一个计算机系统所需要的活动。
- (5)维护过程。定义维护者和机构为了修改和管理软件,使它处于良好运行状态所需要的活动。
- (6)管理过程。定义软件工程过程中各项管理活动,包括:项目开始和范围定义,项目管理计划,实施和控制,评审和评价,项目完成。
- (7)支持过程。支持过程对项目的生存周期过程给予支持。它有助于项目的成功并能提高项目的质量。

12. 需求规格说明书的内容有哪些？

答: (1)引言

目的

项目背景

参考资料

术语

(2)项目概述

目标

用户的特点

假定与约束

(3)具体要求

功能需求

外部接口需求

性能需求

软件属性需求

数据需求

产品化需求

(4)运行环境规定

设备

支持软件

接口

控制

13. 软件危机的表现有哪些？

答: (1)经费预算经常突破,完成时间一再拖延。由于缺乏软件开发的经验和软件开发数据的积累,使得开发工作的计划很难制定。主观盲目制定计划,执行起来与实际情况有很大差距,使得开发经费一再突破。由于对工作量估计不足,对开发难度估计不足,进度计划无法按时完成,开发时间一再拖延。

(2)开发的软件不能满足用户要求。开发初期对用户的要求了解不够明确,未能得到明确表达。开

发工作开始后,软件人员和用户又未能及时交换意见,使得一些问题不能及时解决,导致开发的软件不能满足用户的要求,因而开发失败。

(3)开发的软件可维护性差。开发过程没有统一的、公认规范,软件开发人员按各自的风格工作,各行其是。开发过程无完整、规范的文档,发现问题后进行杂乱无章的修改。程序结构不好,运行时发现错误也很难修改,导致可维护性差。

(4)开发的软件可靠性差。由于在开发过程中,没有确保软件质量的体系和措施,在软件测试时,又没有严格的、充分的、完全的测试,提交给用户的软件质量差,在运行中暴露出大量的问题。这种不可靠的软件,轻者会影响系统正常工作,重者会发生事故,造成生命财产的重大损失。

#### 14. 软件的特点有哪些?

答:(1)软件是一种逻辑产品,它与物质产品有很大的区别。软件产品是看不见摸不着的,因而具有无形性,它是脑力劳动的结晶,它以程序和文档的形式出现,保存在计算机存储器的磁盘和光盘介质上,通过计算机的执行才能体现它的功能和使用。

(2)软件产品的生产主要是研制,软件产品的成本主要体现在软件的开发和研制上,软件开发研制完成后,通过复制就产生了大量软件产品。

(3)软件产品不会用坏,不存在磨损、消耗问题。

(4)软件产品的生产主要是脑力劳动,还未完全摆脱手工开发方式,大部分产品是“定做”的。

(5)软件费用不断增加,软件成本相当昂贵。软件的研制工作需要投入大量的、复杂的、高强度的脑力劳动,它的成本非常高。

#### 15. 软件危机的原因有哪些?

答:(1)软件的规模越来越大,结构越来越复杂。随着计算机应用的日益广泛,需要开发的软件规模日益庞大,软件结构也日益复杂。

(2)软件开发管理困难而复杂。由于软件规模大,结构复杂,又具有无形性,因此导致管理困难,进度控制困难,质量控制困难,可靠性无法保证。

(3)软件开发费用不断增加。软件生产是一种智力劳动,它是资金密集、人力密集的产业,大型软件投入人力多,周期长,费用上升很快。

(4)软件开发技术落后。

(5)生产方式落后。仍然采用个体手工方式开发,根据个人习惯爱好工作,无章可循,无规范可依据,靠言传身教方式工作。

(6)开发工具落后,生产率提高缓慢。

#### 16. 软件生产的发展有哪几个阶段?

答:自从第一台计算机诞生以后,就开始了软件的生产,到现在为止,经过了三个阶段。即程序设计时代,程序系统时代,软件工程时代。

#### 17. 软件工程性质是什么?

答:软件工程是一门综合性的交叉学科,它涉及计算机科学、工程科学、管理科学、数学等领域。

计算机科学中的研究成果均可用于软件工程,但计算机科学着重于原理和理论,而软件工程着重于如何建造一个软件系统。

软件工程要用工程科学中的观点来进行费用估算、制定进度、制定计划和方案。

软件工程要用管理科学中的方法和原理进行软件生产的管理。

软件工程要用数学的方法建立软件开发中的各种模型和各种算法,如可靠性模型,说明用户需求的形式化模型等。

## 第二章 软件可行性研究与项目开发计划

本章总的要求是:深刻理解可行性研究的必要性,掌握可行性研究的任务及可行性研究的具体步骤。

了解系统流程图的作用及符号表示。

理解可行性研究报告与项目开发计划的内容。

### 考试要求

#### 1. 可行性研究

可行性研究的任务,要求达到识记层次。

可行性研究的具体步骤,要求达到领会层次。

#### 2. 系统流程图,要求达到识记层次。

#### 3. 成本——效益分析

投资回收率达到识记层次。

回收期要求达到识记层次。

纯收入要求达到识记层次。

#### 4. 项目开发计划,要求达到识记层次。

### 知识重点

#### (一)可行性研究的任务

##### 1. 技术可行性

对要开发项目的功能、性能、限制条件进行分析,确定在现有的资源条件下,技术风险有多大,项目是否能实现。它一般要考虑的情况包括:

(1)开发的风险:在给出的限制范围内,能否设计出系统并实现必须的功能和性能?

(2)资源的有效性:可用于开发的人员是否存在问题?可用于建立系统的其它资源是否具备?

(3)技术:相关技术的发展是否支持这个系统?

(4)开发人员在评估技术可行性时,一旦估计错误,将会出现灾难性后果。

##### 2. 经济可行性

进行开发成本的估算以及了解取得效益的评估,确定要开发的项目是否值得投资开发。

经济可行性研究范围较广,包括成本——效益分析、公司经营长期策略、开发所需的成本和资源、潜在的市场前景。

### 3 .社会可行性

要开发的项目是否存在任何侵犯、妨碍等责任问题,要开发项目的运行方式在用户组织内是否行得通,现有管理制度、人员素质、操作方式是否可行。

社会可行性所涉及的范围也比较广,它包括:合同、责任、侵权、用户组织的管理模式及规范,其他一些技术人员常常不了解的陷阱等。

### (二)可行性研究的具体步骤

典型的可行性研究有下列步骤:

- 1 .确定项目规模和目标
- 2 .研究正在运行的系统
- 3 .建立新系统的高层逻辑模型
- 4 .导出和评价各种方案
- 5 .推荐可行的方案
- 6 .编写可行性研究报告

### (三)可行性研究报告的主要内容

一个可行性研究报告的主要内容如下:

- 1 .引言。说明编写本文档的目的;项目的名称、背景;本文档用到的专门术语和参考资料。
- 2 .可行性研究前提。说明开发项目的功能、性能和基本要求;达到的目标;各种限制条件;可行性研究方法和决定可行性的主要因素。
- 3 .对现有系统的分析。说明现有系统的处理流程和数据流程;工作负荷;各项费用支出;所需各类专业技术人员和数量;所需各种设备;现有系统存在什么问题。
- 4 .所建议系统的技术可行性分析。所建议系统的简要说明;处理流程和数据流程;与现有系统比较的优越性;采用所建议系统对用户的影响;对各种设备、现有软件、开发环境、运行环境的影响;对经费支出的影响;对技术可行性的评价。
- 5 .所建议系统的经济可行性分析。说明所建议系统的各种支出,各种效益;收益投资比;投资回收周期。
- 6 .社会因素可行性分析。说明法律因素,对合同责任、侵犯专利权、侵犯版权等问题的分析;说明用户使用可行性,是否满足用户行政管理、工作制度、人员素质的要求。
- 7 .其它可供选择方案。逐一说明其它可供选择的方案,并说明未被推荐的理由。
- 8 结论意见。说明项目是否能开发;还需什么条件才能开发;对项目目标有何变动等。

### (四)成本 - 效益分析

#### 1 .货币的时间价值

经过成本估算后,得到项目开发时需要的费用,该费用就是项目的投资。项目开发后,应取得相应的效益,有多少效益才合算?这就要考虑货币的时间价值。通常用利率表

示货币的时间价值。

设年利率为  $i$ , 现存入  $p$  元,  $n$  年后可得钱数为  $F$  元, 若不计复利则

$$F = P \times (1 + n \times i)$$

$F$  就是  $P$  元在  $n$  年后的价值。反之, 若  $n$  年能收入  $F$  元, 那么这些钱现在的价值是:

$$P = F / (1 + n \times i)$$

## 2. 投资回收期

投资回收期就是使累计的经济效益等于最初的投资费用所需的时间。投资回收期越短, 就越快获得利润, 则该项目就越值得开发。

## 3. 纯收入

纯收入就是在整个生存周期之内的累计经济效益(折合成现在值)与投资之差。这相当于投资开发一个项目与把钱存入银行中进行比较, 看这两种方案的优劣。若纯收入为零, 则项目的预期效益和在银行存款一样, 但是开发一个项目要冒风险, 因此, 从经济观点看这个项目, 可能是不值得投资开发的。若纯收入小于零, 那么这个项目显然不值得投资开发。

## (五) 项目开发计划

项目开发计划是一个管理性的文档, 它的主要内容如下:

1. 项目概述: 说明项目的各项主要工作; 说明软件的功能、性能; 为完成项目应具备的条件; 用户及合同承包者承担的工作、完成期限及其它条件限制; 应交付的程序名称, 所使用的语言及存储形式; 应交付的文档。

2. 实施计划: 说明任务的划分, 各项任务的责任人; 说明项目开发进度, 按阶段应完成的任务, 用图表说明每项任务的开始时间和完成时间; 说明项目的预算, 各阶段的费用支出预算。

3. 人员组织及分工; 说明开发该项目所需人员的类型、组成结构、数量等。

4. 交付期限: 说明项目最后完工交付的日期。

# 反馈测试题解

## 一、名词解释

### 1. 软件项目开发计划

答: 软件项目开发计划是软件工程中的一种管理性文档。主要是对开发的软件项目的费用、时间、进度、人员组织、硬件设备的配置、软件开发环境和运行环境的配置等进行说明和规划, 是项目管理人员对项目进行管理的依据, 据此对项目的费用、进度、资源进行控制和管理。

### 2. 投资回收期

答: 就是使累计的经济效益等于最初的投资费用所需的时间。利用这一概念来衡量开发项目的价值。投资回收期越短, 获得利润就越快, 该项目的投资效益就高。而累计经济效益就是将来每年因运行开发的软件项目而获得的效益的现在价值的总和。要用货币时间价值公式来计算将来效益的现在的价值。

### 3. 货币时间价值

答:通常利用银行存款的利息来表示货币的时间价值。设年利率为  $i$ , 现存入  $p$  元,  $n$  年后得到本金和利息为  $F$ 。若不计复利, 则  $P$  元在  $n$  年后的价值为:  $F = P \times (1 + n \times i)$ 。反过来, 若  $n$  年后能收入的本金和利息为  $F$ , 则将来  $F$  元的现在价值(本金)  $P$  为:  $P = F / (1 + n \times i)$ 。可用这个公式来计算将来收入的现在价值。这是效益分析的最基本公式。

### 4. 纯收入

答:是指软件生存周期内, 累计的经济效益与投资(软件开发成本)之差, 这是项目的经济效益的另一指标。

## 二、填空题

1. \_\_\_\_\_的目的就是用最小的代价在尽可能短的时间内确定该软件项目是否能够开发, 是否值得去开发。

答: 软件可行性研究

2. 可行性研究的目的不是去开发一个软件项目, 而是研究这个软件项目是否 \_\_\_\_\_, \_\_\_\_\_。

答: 值得去开发 其中的问题能否解决

3. 技术可行性是对要开发项目的 \_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_进行分析, 确定在现有的资源条件下, 技术风险有多大, 项目是否能实现。

答: 功能 性能 限制条件

4. 典型的可行性研究有下列步骤: 确定项目规模和目标, \_\_\_\_\_, \_\_\_\_\_, 导出和评价各种方案, 推荐可行的方案, 编写可行性研究报告。

答: 研究正在运行的系统 建立新系统的高层逻辑模型

5. 项目开发计划的主要内容有: 项目概述、\_\_\_\_\_、\_\_\_\_\_、交付期限。

答: 实施计划 人员组织及分工

6. 一个可行性研究报告的主要内容如下: 引言; 可行性研究前提; 对现有系统的分析; \_\_\_\_\_; \_\_\_\_\_; \_\_\_\_\_; 其他可供选择方案; 结论意见。

答: 所建议系统的技术可行性分析 所建议系统的经济可行性分析 社会因素可行性分析

7. 成本——效益分析首先是估算将要开发的系统的 \_\_\_\_\_, 然后与可能取得的效益进行 \_\_\_\_\_。

答: 开发成本 比较和权衡

8. \_\_\_\_\_就是使累计的经济效益等于最初的投资费用所需的时间。项目的 \_\_\_\_\_是指在整个生存周期之内的累计经济效益(折合成现在值)与投资之差。

答: 投资回收期 纯收入

9. 效益分有形效益和无形效益两种。有形效益可以用 \_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_等指标进行度量; 无形效益主要从性质上、心理上进行衡量, 很难直接进行量的比较。

答: 货币的时间价值 投资回收期 纯收入

10. 可行性研究在进行简要需求分析和设计时, 要在高层次上以 \_\_\_\_\_ 进行。

答: 较抽象形式

11. 纯收入是软件生存周期内两项值之差, 这两项是 \_\_\_\_\_。

答: 经济效益与投资

12. 系统流程图用图形符号表示系统中各个元素, 表达了系统中各种元素之间的 \_\_\_\_\_。

答:信息流动

13. 软件工程有两种效益,它们是无形效益和\_\_\_\_\_。

答:有形效益

14. 可行性研究的目的是用最小的代价,在尽可能短的时间内,确定\_\_\_\_\_。

答:项目值得开发否

15. 可行性研究具体步骤的最后一步是\_\_\_\_\_。

答:编写可行性报告

16. 技术可行性一般要考虑的情况包括\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_。

答:开发的风险 资源的有效性 技术

17. 可行性研究可从\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_三个方面分析研究每种解决方法的可行性。

答:技术可行性 经济可行性 社会可行性

18. 经济可行性研究包括:成本 - \_\_\_\_\_分析、公司经营\_\_\_\_\_策略、开发所需的\_\_\_\_\_和\_\_\_\_\_、潜在的\_\_\_\_\_前景。

答:效益 长期 成本 资源 市场

19. 系统流程图符号的名称是\_\_\_\_\_,用于连接其它符号,指明数据流动方向。

答:数据流

20. 假定开发库房管理系统共需 5000 元,系统建成后估计每年能节约 2500 元,若年利率为 5%,项目生存周期为 5 年,则该项目的纯收入预计为\_\_\_\_\_千元。

答:6.361

21. 社会可行性研究包括\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_、用户组织的\_\_\_\_\_模块及\_\_\_\_\_,其他一些技术人员常常不了解的\_\_\_\_\_等。

答:合同 责任 侵权 管理 规范 陷阱

22. 系统的经济效益等于\_\_\_\_\_加上\_\_\_\_\_。

答:因使用新的系统而增加的收入 使用新的系统可以节省的运行费用

23. 系统的经济效益等于因使用新的系统而增加的收入加上使用新的系统可以节省的运行费用。运行费用包括\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_等。

答:操作人员人数 工作时间 消耗的物资

24. 效益分有形效益和无形效益两种。\_\_\_\_\_可以用货币的时间价值、投资回收期、纯收入等指标进行度量;\_\_\_\_\_主要从性质上、心理上进行衡量,很难直接进行量的比较。

答:有形效益 无形效益

25. \_\_\_\_\_分析首先是估算将要开发的系统的开发成本,然后与可能取得的效益进行比较和权衡。

答:成本—效益

26. 可行性研究实质上是要进行一次简化、压缩了的需求分析、设计过程,要在较高层次上以较抽象的方式进行\_\_\_\_\_和\_\_\_\_\_。

答:需求分析 设计过程

27. 成本—效益分析的目的是从\_\_\_\_\_评价开发一个新的软件项目是否可行。

答:经济角度

28. 系统流程图是描绘物理系统的传统工具,它用\_\_\_\_\_来表示系统中的各个元素。

答:图形符号

29. 可行性研究中描述系统高层物理模型的工具是\_\_\_\_\_。

答:系统流程图



30. 可行性研究实质上是进行一次简化、压缩了的\_\_\_\_\_。

答:需求分析和设计

31. 可行性研究的第一个具体步骤是\_\_\_\_\_。

答:确定项目的规模和目标

三、选择题

- 1 对每个合理的方案分析员都应该准备如下资料 ( )
- A .系统流程
  - B .组成系统的物理元素清单 ,成本——效益分析
  - C .实现这个系统的进度计划
  - D .以上全部

答:D

- 2 软件生存周期法中 ,用户的参与主要在 ( )
- A .软件定义期
  - B 软件开发期
  - C .软件维护期
  - D 整个软件生存周期过程中

答:A

- 3 软件定义期问题定义阶段涉及的人员有 ( )
- A .用户、使用部门负责人
  - B .软件开发人员、用户、使用部门负责人
  - C .系统分析员、软件开发人员
  - D 系统分析员、软件开发人员、用户与使用部门负责人

答:D

- 4 系统定义明确之后 ,应对系统的可行性进行研究。可行性研究应包括 ( )
- A .软件环境可行性、技术可行性、经济可行性、社会可行性
  - B .经济可行性、技术可行性、社会可行性
  - C .经济可行性、社会可行性、系统可行性
  - D 经济可行性、实用性、社会可行性

答:B

- 5 在遵循软件工程原则开发软件的过程中 ,计划阶段应该依次完成 ( )
- A .软件计划、需求分析、系统定义
  - B .系统定义、软件计划、需求分析
  - C .需求分析、概要设计、软件计划
  - D 软件计划、需求分析、概要设计

答:B

6. 技术可行性要解决 ( )
- A. 存在侵权否
  - B. 成本效益问题
  - C. 运行方式可行
  - D. 技术风险问题

答:D

7. 可行性分析中 ,系统流程图用于描述 ( )
- A. 当前运行系统
  - B. 当前逻辑模型
  - C. 目标系统
  - D. 新系统

答:A

8. 研究硬软件资源的有效性是进行( )研究的一方面。

- A. 技术可行性  
B. 经济可行性  
C. 社会可行性  
D. 操作可行性

答: A

9. 可行性研究要进行的需求分析和设计应是 ( )

- A. 详细的  
B. 全面的  
C. 简化、压缩的  
D. 彻底的

答：C

10. 表示人工操作的系统流程图的符号是 ( )

- A . |                      |                      B .                       C .                      D .

答:C

11. 表示文档的系统流程图的符号是 ( )

- A .                      B .                       C .                      D .

答:D

12. 假定开发库房管理系统共需 5000 元,系统建成后估计每年能节约 2500 元,若年利率为 5%,其投资回收期约等于( )年。

- A.1                      B.2                      C.3                      D.4

答: B

13. 表示连接的系统流程图的符号是 ( )

- A .                      B .                      C .                      D .

答: B

14. 表示输入/输出的系统流程图的符号是 ( )

- A .                      B .                      C .                      D .

答:A

15. 表示磁盘的系统流程图的符号是 ( )

- A . |                      B .                                           C .                      D .

答:A

16. 系统流程图用于可行性分析中的( )的描述。

- A. 当前运行系统  
B. 当前逻辑模型  
C. 目标系统  
D. 新系统

答:A

17. 系统流程图是描述( )的工具。

- A. 逻辑系统                  B. 程序系统                  C. 体系结构                  D. 物理系统

答:D

18. 研究开发资源的有效性是进行( )可行性研究的一方面。

- A. 技术                      B. 经济                      C. 社会                      D. 操作

答:A

19. 可行性研究要进行一次( )需求分析。

- |            |        |
|------------|--------|
| A. 详细的     | B. 全面的 |
| C. 简化的、压缩的 | D. 彻底的 |

答:C

20. 可行性研究的目的是( )

- |         |            |
|---------|------------|
| A. 开发项目 | B. 项目值得开发否 |
| C. 规划项目 | D. 维护项目    |

答:A

#### 四、简答题

1 研究项目的技术可行性一般要考虑哪些情况 ?

答:技术可行性一般要考虑的情况包括:

- (1)开发的风险:在给出的限制范围内,能否设计出系统并实现必须的功能和性能。
- (2)资源的有效性:可用于开发的人员是否存在问题。可用于建立系统的其他资源是否具备。
- (3)技术:相关技术的发展是否支持这个系统。
- (4)开发人员在评估技术可行性时,一旦估计错误,将会出现灾难性后果。

2. 项目开发计划主要内容有哪些 ?

答:经过可行性研究后,若一个项目是值得开发的,则接下来应制定项目开发计划。项目开发计划是一个管理性的文档,它的主要内容如下:

(1)项目概述

说明项目的各项主要工作;说明软件的功能、性能;为完成项目应具备的条件;用户及合同承包者承担的工作、完成期限及其他条件限制;应交付的程序名称,所使用的语言及存储形式;应交付的文档。

(2)实施计划

说明任务的划分,各项任务的责任人;说明项目开发进度,按阶段应完成的任务,用图表说明每项任务的开始时间和完成时间;说明项目的预算,各阶段的费用支出预算。

(3)人员组织及分工

说明开发该项目所需人员的类型、组成结构、数量等。

(4)交付期限

说明项目最后完工交付的日期。

3. 成本效益分析的目的是什么 ?

答:成本效益分析主要用于项目的经济可行性研究。它的目的是从经济角度评价开发一个新的软件项目是否可行。

4. 可行性研究的主要内容有哪些 ?

答:(1)必要性

为了减少大型工程项目的风险,必须进行可行性研究。同样,对于大型软件项目,由于周期长、投资大、使用资源多,开发之后是否能达到预期目的等原因,也必须进行可行性研究,以避免时间、人力、资源、经费的巨大浪费。

可行性研究的目的就是用最小的代价,在尽可能短的时间内,确定该软件项目是否能够开发,是否值得开发。

(2)任务

进行可行性研究时,首先要进行概要分析,初步确定项目的规模、目标、约束和限制条件。然后进行简要的、高层次的需求分析,决定系统的主要功能。在此基础上进行简要的、压缩的设计,提出几种解决方案。对每种方案进行可行性研究,可行性研究有三方面内容:

技术可行性:主要研究开发项目的技术风险有多大,可以从下列方面研究:现有的开发技术能否解决开发项目的技术难题?现有的开发人员的技术水平是否能够解决这些难题?现有的硬软件资源是否满足解决技术难题的需要?在现有的约束限制条件下,能否设计出满足功能和性能要求的系统来?技术可行性较为困难,因为这时项目的目标、功能、性能都是概要的、模糊的,在不准确的情况下较难解决。

经济可行性:主要研究系统开发以后能否得到应有的效益。可用成本效益分析法来研究这一问题。

社会可行性:主要研究两方面的问题,一是采用的技术是否存在责任和侵权的问题,二是开发的系统运行时与当前管理制度、人员素质、操作方式的矛盾能否解决。

#### 5. 可行性研究的具体步骤有哪些?

答:典型的可行性研究有下列步骤:

##### (1)确定项目规模和目标

分析员对有关人员进行调查访问,仔细阅读和分析有关材料,对项目的规模和目标进行定义和确认,清晰地描述项目的一切限制和约束,确保分析员正在解决的问题确实是要解决的问题。

##### (2)研究正在运行的系统

正在运行的系统可能是一个人工操作的系统,也可能是旧的计算机系统,要开发一个新的计算机系统来代替现有系统。因此,现有的系统是信息的重要来源,要研究它的基本功能,存在什么问题,运行现有系统需要多少费用,对新系统有什么新的功能要求,新系统运行时能否减少使用费用等等。

应该收集、研究、分析现有系统的文档资料,实地考察现有系统,在考察的基础上,访问有关人员,然后描绘现有系统的高层系统流程图,与有关人员一起审查该系统流程图是否正确。这个系统流程图反映了现有系统的基本功能和处理流程。

##### (3)建立新系统的高层逻辑模型

根据对现有系统的分析研究,逐渐明确了新系统的功能、处理流程以及所受的约束,然后使用建立逻辑模型的工具——数据流图和数据字典来描述数据在系统中的流动和处理情况。注意,现在还不是软件需求分析阶段,不是完整、详细地描述,只是概括地描述高层的数据处理和流动。

##### (4)导出和评价各种方案

分析员建立了新系统的高层逻辑模型之后,要从技术角度出发,提出实现高层逻辑模型的不同方案,即导出若干较高层次的物理解法。根据技术可行性、经济可行性、社会可行性对各种方案进行评估,去掉行不通的解法,就得到了可行的解法。

##### (5)推荐可行的方案

根据上述可行性研究的结果,应该决定该项目是否值得去开发。若值得开发,那么可行的解决方案是什么,并且说明该方案可行的原因和理由。该项目是否值得开发的主要因素是从经济上看是否合算,这就要求分析员对推荐的可行方案进行成本——效益分析。

##### (6)编写可行性研究报告

将上述可行性研究过程的结果写成相应的文档,即可行性研究报告,提请用户和使用部门仔细审查,从而决定该项目是否进行开发,是否接受可行的实现方案。

#### 6. 可行性研究报告的主要内容?

答:一个可行性研究报告的主要内容如下:

##### (1)引言

说明编写本文档的目的;项目的名称、背景;本文档用到的专门术语和参考资料。

## (2)可行性研究前提

说明开发项目的功能、性能和基本要求;达到的目标;各种限制条件;可行性研究方法和决定可行性的主要因素。

## (3)对现有系统的分析

说明现有系统的处理流程和数据流程;工作负荷;各项费用支出;所需各类专业技术人员和数量;所需各种设备;现有系统存在什么问题。

## (4)所建议系统的技术可行性分析

所建议系统的简要说明;处理流程和数据流程;与现有系统比较的优越性;采用所建议系统对用户的影响;对各种设备、现有软件、开发环境、运行环境的影响;对经费支出的影响;对技术可行性的评价。

## (5)所建议系统的经济可行性分析

说明所建议系统的各种支出,各种效益;收益投资比;投资回收周期。

## (6)社会因素可行性分析

说明法律因素,对合同责任、侵犯专利权、侵犯版权等问题的分析;说明用户使用可行性,是否满足用户行政管理、工作制度、人员素质的要求。

## (7)其他可供选择方案

逐一说明其他可供选择的方案,并说明未被推荐的理由。

## (8)结论意见

说明项目是否能开发;还需什么条件才能开发;对项目目标有何变动等。

## 7. 项目开发计划的主要内容是什么?

答:(1)引言

编写目的

项目背景

参考资料

术语

## (2)项目概述

功能

条件(含主要参加人员)

运行环境

产品

验收标准

## (3)实施计划

计划

工作任务分解

进度

预算

关键问题

开发约定(含标准、规则、规范、开发工具与技术、配置管理约定等)

## (4)支持条件

人员组织及分工

计算机系统支持

需要用户承担的工作


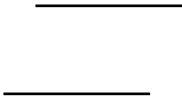









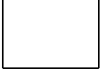
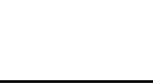
需由外单位提供的条件

(5)交付日期

(6)专题计划(如测试计划、质量保证计划、配置管理计划、人员培训计划等)

8. 系统流程图的符号有哪些？

答：系统流程图的符合

符 号	名称	说 明
	处理	能改变数据值或数据位置的加工或部件，例如，程序模块、处理机等都是处理。
	输入/ 输出	表示输入或输出 (或既输入又输出 )，是一个广义的不指明具体设备的符号。
	连接	指出转到图的另一部分或从图的另一部分转来，通常在同一页上。
	换页连接	指出转到另一页图上或由另一页图转来。
	数据流	用来连接其他符号，指明数据流动方向。
	文档	通常表示打印输出，也可表示用打印终端输入数据。
	联机存储	表示任何种类的联机存储，包括磁盘、软盘和海量存储器件等。
	磁盘	磁盘输入/ 输出，也可表示存储在磁盘上的文件或数据库。
	显示	CRT 终端或类似的显示部件，可用于输入或输出，也可既输入又输出。
	人工输入	人工输入数据的脱机处理，例如，填写表格。
	人工操作	人工完成的处理，例如，会计在工资支票上签名。
	辅助操作	使用设备进行的脱机操作。
	通信链路	通过远程通信线路或链路传送数据。

五、应用题

1 成本——效益分析可用哪些指标进行度量？

答:DNSG—效益分析的目的是从经济角度评价开发一个新的软件项目是否可行。成本—效益分析

首先是估算将要开发的系统的开发成本,然后与可能取得的效益进行比较和权衡。效益分有形效益和无形效益两种。有形效益可以用货币的时间价值、投资回收期、纯收入等指标进行度量;无形效益主要从性质上、心理上进行衡量,很难直接进行量的比较。系统的经济效益等于因使用新的系统而增加的收入加上使用新的系统可以节省的运行费用。运行费用包括操作人员人数、工作时间、消耗的物资等。

下面主要介绍有形效益的分析。

(1)货币的时间价值。成本估算的目的是对项目投资。经过成本估算后,得到项目开发时需要的费用,该费用就是项目的投资。项目开发后,应取得相应的效益,有多少效益才合算,这就要考虑货币的时间价值。通常用利率表示货币的时间价值。

设年利率为*i*,现存入*P*元,*n*年后可得钱数为*F*元,若不计复利,则:

$$F = P \times (1 + n \times i)$$

*F*就是*P*元在*n*年后的价值。反之,若*n*年能收入*F*元,那么这些钱现在的价值是:

$$P = F / (1 + n \times i)$$

(2)投资回收期。通常用投资回收期衡量一个开发项目的价值。投资回收期就是使累计的经济效益等于最初的投资费用所需的时间。投资回收期越短,就越快获得利润,则该项目就越值得开发。

(3)纯收入。衡量项目价值的另一个经济指标是项目的纯收入,也就是在整个生存周期之内的累计经济效益(折合成现在值)与投资之差。这相当于投资开发一个项目与把钱存入银行中进行比较,看这两种方案的优劣。若纯收入为零,则项目的预期效益和在银行存款一样,但是开发一个项目要冒风险,因此,从经济观点看,这个项目可能是不值得投资开发的。若纯收入小于零,那么这个项目显然不值得投资开发。

2. 假定开发库房管理系统共需 5000 元,系统建成后估计每年能节约 2500 元。若年利率为 5%,利用计算货币现在价值的公式,请计算建立库房管理系统后,5 年内每年预计节省的费用的现在价值。

答:已知  $F = 2500, i = 0.05, n = 1 \dots 5$

由公式:  $P = F / (1 + n \times i)$  可得表结果。

将来的收入折算成现在值

年	将来值(千元)	(1 + n × 0.05)	现在值(千元)	累计的现在值(千元)
1	2.5	1.05	2.381	2.381
2	2.5	1.1	2.273	5.104
3	2.5	1.15	2.174	7.278
4	2.5	1.2	2.083	9.361
5	2.5	1.25	2.0	11.361

## 第三章 软件需求分析

需求分析是软件生存期的一个重要阶段,本章总的要求是深刻理解需求分析阶段的概念及任务。熟练掌握面向数据流的分析方法——结构化分析方法。

深刻理解数据流图、数据字典的作用及应用。

掌握几种加工逻辑的描述方法。

熟练掌握的技能是对于某一个小型项目,能够使用数据流图、数据字典构造它的逻辑模型。

### 考试要求

#### 1. 需求分析的任务

需求分析的概念,要求达到识记层次。

需求分析的基本任务,要求达到领会层次。

#### 2. 结构化分析方法

结构化分析方法,需要达到识记层次。

结构化分析步骤,要求达到领会层次。

#### 3. 数据流图

数据流图,要求达到识记层次。

数据流图中的符号、画数据流图注意的事项,要求达到领会层次。

分层的数据流图,要求达到简单应用层次。

#### 4. 数据字典

数据字典的作用,要求达到识记层次。

数据字典中的条目,要求达到识记层次。

数据字典,要求达到简单应用层次。

#### 5. 加工逻辑的描述

结构化语言、判定表(树)的构成,要求达到领会层次。

结构化语言、判定表(树)的描述,要求达到简单应用层次。

#### 6. IDEF 方法

IDEF<sub>0</sub> 的图形表示,要求达到领会层次。

建立功能模型的基本方法,要求达到领会层次。

#### 7. 结构化分析方法小结

数据流图、数据字典,要求达到综合应用层次。



# 知识重点

## (一) 需求分析

需求分析是指,开发人员要准确理解用户的要求,进行细致的调查分析,将用户非形式的需求陈述转化为完整的需求定义,再由需求定义转换到相应的形式功能规约(需求规格说明)的过程。需求分析虽处于软件开发过程的开始阶段,但它对于整个软件开发过程以及软件产品质量是至关重要的。

近几年来已提出许多软件需求分析与说明的方法(如结构化分析方法和面向对象分析方法),每一种分析方法都有独特的观点和表示法,但都适用下面的基本原则。

1 .必须能够表达和理解问题的数据域和功能域。

2 .可以把一个复杂问题按功能进行分解并可逐层细化。在需求分析过程中,软件领域中的数据、功能、行为都可以划分。

3 .建模。建立模型可以帮助分析人员更好地理解软件系统的信息、功能、行为,这些模型也是软件设计的基础。

需求分析的基本任务是要准确地定义新系统的目标,为了满足用户需求,回答系统必须“做什么”的问题。本阶段要进行以下几方面的工作:

1 .分析人员和用户对问题识别,双方确定对问题的综合需求。这些需求包括:功能需求、性能需求、环境需求和用户界面需求。另外还有可靠性、安全性、保密性、可移植性、可维护性等方面的需求,这些需求一般通过双方交流、调查研究来获取,并达到共同的理解。

2 .分析与综合,导出软件的逻辑模型。分析人员对获取的需求,进行一致性的分析检查,在分析、综合中逐步细化软件功能,划分成各个子功能。这里也包括对数据域进行分解,并分配到各个子功能上,以确定系统的构成及主要成分,并用图文结合的形式,建立起新系统的逻辑模型。

3 .编写文档。这一阶段的文档有“需求规格说明书”,初步用户使用手册,确认测试计划。在需求分析阶段,对待开发的系统有了更进一步的了解。因此还要修改完善软件开发计划。

## (二) 结构化分析方法

结构化分析(Structure Analysis, 简称 SA),是面向数据流进行需求分析的方法。SA是一种建模活动,该方法使用简单易读符号,根据软件内部数据传递、变换的关系,自顶向下逐层分解,描绘出满足功能要求的软件模型。

结构化分析方法采取的是自顶向下逐层分解的分析策略,即面对一个复杂的问题,把一个复杂的问题划分成若干小问题,然后再分别解决,将问题的复杂性降低到人可以掌握的程度。分解可分层进行,先考虑问题最本质的方面,忽略细节,形成问题的高层概念,然后再逐层添加细节,即在分层过程中采用不同程度的“抽象”级别,最高层的问题最抽象,而低层的较为具体。

SA 方法利用图形等半形式化的描述方式表达需求, 简明易懂, 用它们形成需求说明书中的主要部分。这些描述工具是:

- (1) 数据流图;
- (2) 数据字典;
- (3) 描述加工逻辑的结构化语言、判定表、判定树。

其中, “数据流图”, 描述系统的分解, 即描述系统由哪几部分组成, 各部分之间有什么联系等等。“数据字典”字义了数据流图中每一个图形元素。结构化语言、判定表或判定树则详细描述数据流图中不能被再分解的每一个加工。

SA 分析步骤:

- 1. 了解当前系统的工作流程, 获得当前系统的物理模型。这一模型包含了许多具体因素, 反映现实世界的实际情况。
- 2. 抽象出当前系统的逻辑模型。
- 3. 建立目标系统的逻辑模型。
- 4. 作进一步补充和优化。说明目标系统的人机界面, 说明至今尚未详细考虑的细节, 如出错处理、输入输出格式、存储容量、响应时间等性能要求与限制。

### (三) 数据流图 (DFD)

数据流图, 简称 DFD, 是 SA 方法中用于表示系统逻辑模型的一种工具, 它以图形的方式描绘数据在系统中流动和处理的过程, 由于它只反映系统必须完成的逻辑功能, 所以它是一种功能模型。

数据流图由数据流、加工 (又称为数据处理)、数据存储 (又称为文件)、数据源点或终点四种基本成分组成。

- 1. 数据流。数据流是数据在系统内传播的路径, 因此由一组成成分固定的数据项组成。
- 2. 加工 (又称为数据处理)。对数据流进行某些操作或变换。
- 3. 数据存储 (又称为文件)。指暂时保存的数据, 它可以是数据库文件或任何形式的数据组织。
- 4. 数据源点或终点。是本软件系统外部环境中的实体 (包括人员、组织或其他软件系统), 统称外部实体。

为了表达较为复杂问题的数据处理过程, 用一张数据流图是不够的。要按照问题的层次结构进行逐步分解, 并以一套分层的数据流图反映这种结构关系。

### (四) 数据字典 (DD)

数据流图仅描述了系统的“分解”, 系统由哪几部分组成, 各部分之间的联系, 并没有对各个数据流、加工、数据存储进行详细说明。数据字典 (Data Dictionary, 简称 DD) 就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性的定义和详细的描述。它和数据流图共同构成了系统的逻辑模型, 是需求规格说明书的主要组成部分。

数据字典有以下四类条目: 数据流、数据项、数据存储、基本加工。数据项是组成数据

流和数据存储的最小元素。源点、终点不在系统之内,故一般不在字典中说明。

### (五)加工逻辑的描述

加工逻辑也称为“小说明”,描述加工逻辑一般用以下三种工具:结构化语言、判定表、判定树。

1.结构化语言。结构化语言是介于自然语言(英语或汉语)和形式语言之间的一种半形式语言。它是在自然语言基础上加了一些限定,使用有限的词汇和有限的语句来描述加工逻辑,它的结构可分成外层和内层两层:

(1)外层:用来描述控制结构,采用顺序、选择、重复三种基本结构。

(2)内层:一般是采用祈使语句的自然语言短语,使用数据字典中的名词和有限的自定义词,其动词含义要具体,尽量不用形容词和副词来修饰。还可使用一些简单的算术运算和逻辑运算符号。

2.判定表。在有些情况下,数据流图中的某个加工的一组动作依赖于多个逻辑条件的取值。这时,用自然语言或结构化语言都不易清楚地描述出来。而用判定表就能够清楚地表示复杂的条件组合与应做的动作之间的对应关系。

3.判定树。判定树是判定表的变形,一般情况下它比判定表更直观,且易于理解和使用。

这三种描述加工逻辑的工具各有优缺点,对于顺序执行和循环执行的动作,用结构化语言描述。对于存在多个条件复杂组合的判断问题,用判定表和判定树。判定树较判定表直观易读,判定表进行逻辑验证较严格,能把所有的可能性全部都考虑到。可将两种工具结合起来,先用判定表作底稿,在此基础上产生判定树。

### (六)IDEF 方法

IDEF 是 ICAM Definition 的缩写。IDFF 方法分为三部分:

IDEF<sub>0</sub>:用来描述系统的功能活动及其联系,建立系统的功能模型。

IDEF<sub>1</sub>:用来描述系统的信息及其联系,建立系统的信息模型。美国空军项目组对 IDEF<sub>1</sub> 进行了扩充与完善,于 1985 年正式推出了 IDEF<sub>1X</sub>。

IDEF<sub>2</sub>:用来进行系统模拟,建立系统的动态模型。

系统的 IDEF<sub>0</sub> 功能模型,反映系统“做什么”的功能。建立功能模型的基本步骤为:

1.确定建模的范围、观点及目的。

2.建立系统的内外关系图——A—0 图。该图用来抽象地描述所研究的问题及其边界或数据接口。

3.建立顶层图——A0 图。把 A—0 图分解为 3~6 个主要部分得到 A0 图,它清楚地表达了 A—0 图在同样信息范围内的细节。

4.建立低层次的图形。按照自顶向下的方法,从 A0 图开始逐层分解,建立一系列的活动图形,直到最低层为止。

# 反馈测试题解

## 一、名词解释

### 1. 数据存储(又称为文件)

答:指暂时保存的数据,它可以是数据库文件或任何形式的数据组织。流向数据存储的数据流可理解为写入文件或查询文件,从数据存储流出的数据可理解为从文件读数据或得到查询结果。

### 2. 数据源点或终点

答:是本软件系统外部环境中的实体(包括人员、组织或其他软件系统),统称外部实体。它们是为了帮助理解系统接口界面而引入的,一般只出现在数据流图的顶层图中。

### 3. 数据流

答:数据流是数据在系统内传播的路径,因此由一组成分固定的数据项组成。如订票单由旅客姓名、年龄、单位、身份证号、日期、目的地等数据项组成。由于数据流是流动中的数据,所以必须有流向,在加工之间,加工与源终点之间,加工与数据存储之间流动,除了与数据存储之间的数据流不用命名外,数据流应该用名词或名词短语命名。

### 4. 加工(又称为数据处理)

答:对数据流进行某些操作或变换。每个加工也要有名字,通常是动词短语,简明地描述完成什么加工。在分层的数据流图中,加工还应编号。

### 5. 需求分析

答:需求分析是指,开发人员要准确理解用户的要求,进行细致的调查分析,将用户非形式的需求陈述转化为完整的需求定义,再由需求定义转换到相应的形式功能规约(需求规格说明)的过程。需求分析虽处于软件开发过程的开始阶段,但它对于整个软件开发过程以及软件产品质量是至关重要的。

### 6. IDEF 方法

答:IDEF 方法也是结构化分析法中的另一种图形表示法,该方法用方框和箭头等简单图形符号和简洁的语言描述系统的活动、数据流、实现该活动所受到的约束、实现机制以及各活动、数据流之间的联系,全面地描述整个系统的各个侧面。该方法严格地采用自顶向下、逐层分解的原则,有序地逐步展开有关活动的细节。

### 7. 加工逻辑

答:加工逻辑也称为“小说明”,描述加工逻辑一般用以下三种工具:结构化语言、判定表、判定树。

### 8. 数据流条目

答:数据流条目给出了 DFD 中数据流的定义,通常列出该数据流的各组成数据项。

### 9. 数据字典

答:数据字典(Data Dictionary,简称 DD)就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。它和数据流图共同构成了系统的逻辑模型,是需求规格说明书的主要组成部分。

### 10. 数据流图(DFD)

答:数据流图,简称 DFD,是 SA 方法中用于表示系统逻辑模型的一种工具,它以图形的方式描绘数据在系统中流动和处理的过程,由于它只反映系统必须完成的逻辑功能,所以它是一种功能模型。

## 二、填空题

1 需求分析的基本任务是要准确地定义\_\_\_\_\_,为了满足用户需要,回答系统必须\_\_\_\_\_

的问题。

答:新系统的目标 “做什么”

2.在需求分析阶段,首先进行问题识别,即双方确定对问题的综合需求,这些需求包括:\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_。另外还有可靠性、安全性、保密性、可移植性、可维护性等方面的功能。

答:功能需求 性能需求 环境需求 用户界面需求

3.在需求分析阶段要进行以下几方面的工作:问题识别、\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_。

答:分析与综合 导出软件的逻辑模型 编写文档

4.需求分析阶段所要编写的文档有:\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_。

答:需求规格说明书 初步用户使用手册 确认测试计划

5.传统的 SA 方法主要用于\_\_\_\_\_的问题,主要工具 DFD 体现了系统\_\_\_\_\_的功能,但它仅是一个\_\_\_\_\_,没有反映处理的顺序,即\_\_\_\_\_

答:数据处理方面 “做什么” 静态模型 控制流程

6.形式化是软件自动化发展的基础。形式化方法是将需求规格说明用\_\_\_\_\_来描述。典型的有\_\_\_\_\_及\_\_\_\_\_。

答:形式规约语言 基于模型的 Z 语言 VDM 开发方法(维也纳开发方法)

7.数据字典中的加工逻辑主要描述该加工\_\_\_\_\_,即实现加工的策略,而不是实现加工的细节,它描述如何把输入数据流变换为输出数据流的\_\_\_\_\_。

答:“做什么” 加工规则

8.在数据流图中,\_\_\_\_\_是数据在系统内传播的路径,因此由一组\_\_\_\_\_组成。加工(又称为数据处理),是对数据流进行某些\_\_\_\_\_。

答:数据流 成分固定的数据项 操作或变换

9.数据流图有四种基本成分:\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_。

答:数据流 加工(又称为数据处理) 数据存储 数据的源点或终点

10.在 SA 方法的需求描述工具中,\_\_\_\_\_描述系统的分解,即描述系统由哪几部分组成,各部分之间有什么联系等等。\_\_\_\_\_定义了数据流图中每一个图形元素。结构化语言、判定表或判定树则详细描述数据流图中不能被再分解的\_\_\_\_\_。

答:数据流图 数据字典 每一个加工

11.SA 方法利用图形等半形式化的描述方式表达需求,简明易懂,用它们形成需求说明书中的主要部分。这些描述工具是:\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_。

答:数据流图 数据字典 描述加工逻辑的结构化语言、判定表、判定树

12.在进行可行性研究和软件计划以后,如果确认开发一个新的软件系统是必要的而且是可能的,那么就进入\_\_\_\_\_阶段。

答:需求分析

13.需求分析是指,开发人员要准确理解\_\_\_\_\_,进行细致的\_\_\_\_\_,将用户非形式的需求陈述转化为\_\_\_\_\_,再由\_\_\_\_\_转换到相应的形式功能规约(需求规格说明)的过程。

答:用户的要求 调查分析 完整的需求定义 需求定义

14.需求分析的困难主要体现在四个方面:问题的复杂性、\_\_\_\_\_、\_\_\_\_\_、需求易变性。

答:交流障碍 不完备性和不一致性

15.用于描述基本加工的小说明的三种描述工具是\_\_\_\_\_。

答:结构化语言、判定表、判定树

16.IDEF 方法分为三部分,IDEF<sub>0</sub>:用来描述系统的\_\_\_\_\_,建立系统的\_\_\_\_\_模型。IDEF<sub>1</sub>:用

来描述系统的\_\_\_\_\_,建立系统的\_\_\_\_\_模型。IDEF<sub>2</sub>:用来进行系统\_\_\_\_\_,建立系统的\_\_\_\_\_模型。

答:功能活动及其联系、功能、信息及其联系、信息、模拟、动态

17. IDEF<sub>0</sub>方法中,将系统功能称为\_\_\_\_\_,将表示系统功能的图形称为\_\_\_\_\_。在活动图形中,用\_\_\_\_\_和\_\_\_\_\_表示系统的各种活动及相互间的关系。在系统分解的某一层次,可能有多个活动,每个活动编号注在\_\_\_\_\_。

答:活动、活动图形、方框、箭头、方框的右下角

18. 数据字典有以下四类条目:\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_。\_\_\_\_\_是组成数据流和数据存储的最小元素。\_\_\_\_\_和\_\_\_\_\_不在系统之内,故一般不在字典中说明。

答:数据流、数据项、数据存储、基本加工、数据项、源点、终点

19. 在数据字典的定义式中:

+ 的含义为\_\_\_\_\_, $x = a + b$ 表示\_\_\_\_\_;

[...|...]的含义为\_\_\_\_\_, $X = [a|b]$ 表示\_\_\_\_\_;

{...}的含义为\_\_\_\_\_, $X = \{a\}$ 表示\_\_\_\_\_;

$m\{...\}n$  或  $\{...\}_m^n$  的含义为\_\_\_\_\_,表示\_\_\_\_\_;

(...)的含义为\_\_\_\_\_, $X = (a)$ 表示\_\_\_\_\_;

"..."的含义为\_\_\_\_\_, $X = (a)$ ,表示\_\_\_\_\_;

.. 的含义为\_\_\_\_\_, $X = 1..9$ ,表示\_\_\_\_\_;

答: 与、x 由 a 和 b 组成

或、x 由 a 或 b 组成

重复、x 由 0 个或多个 a 组成

重复、x 中最少出现 2 次 a,最多出现 5 次 a

可选、a 可在 x 中出现,也可不出现

基本数据元素、x 是取值为字符 a 的数据元素

连接符、x 可取 1 到 9 中任意一个值

20. 结构化分析的英文简称是\_\_\_\_\_,数据流图的英文简称是\_\_\_\_\_,数据字典的英文简称是\_\_\_\_\_。

答:SA、DFD、DD

21. 结构化语言的外层采用\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_三种基本结构描述控制结构。

答:顺序 选择 重复

22. 由于数据流是流动中的数据,所以必须有\_\_\_\_\_。除了与\_\_\_\_\_之间的数据流不用命名外,数据流应该用名词或名词短语命名。

答:流向 数据存储

23. 建立数据字典一般的两种形式是\_\_\_\_\_和\_\_\_\_\_。

答:手工建立 利用计算机辅助建立并维护

24. 数据存储条目的主要内容有\_\_\_\_\_等。

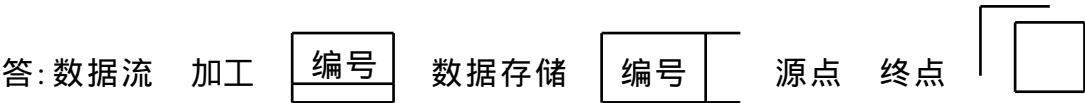
答:数据存储名称、别名、简述、组成、组织方式、查询要求

25. 近几年来已提出许多软件分析与说明的方法,每一种分析方法必须能够表达和理解问题的数据域和功能域。数据域包括\_\_\_\_\_,\_\_\_\_\_和\_\_\_\_\_,而功能域反映上述三方面的\_\_\_\_\_。

答:数据流 数据内容 数据结构 控制信息

26. 数据流图有四种基本图形符号:箭头表示\_\_\_\_\_;圆或椭圆表示\_\_\_\_\_,或可以用\_\_\_\_\_符号表示;双杠表示\_\_\_\_\_,还可以用\_\_\_\_\_符号表示;方框表示数据的\_\_\_\_\_或\_\_\_\_\_还可

以用\_\_\_\_\_符号表示。



27. 利用“数据字典的定义式中出现的符号”,试解释以下定义的数据流组成及数据项:

机票 = 姓名 + 日期 + 航班号 + 起点 + 终点 + 费用,表示\_\_\_\_\_。  
姓名 = {字母}<sup>18</sup><sub>2</sub>,表示\_\_\_\_\_。  
航班号 = “Y7100”..“Y8100”,表示\_\_\_\_\_。  
终点 = [上海|北京|西安],表示\_\_\_\_\_。

答: 机票由日期、航班号、起点、终点、费用组成

姓名中最少出现 2 次“字母”,最多出现 18 次“字母”  
航班号可以为“Y7100”到“Y8100”中任意一个值  
终点可以是上海或北京或西安

28. 数据项条目的主要内容有\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_及含义。

答:数据项名称    别名    简述    类型    长度    取值范围

29. 结构化语言是介于\_\_\_\_\_语言和\_\_\_\_\_语言之间的一种半形式语言。它是在\_\_\_\_\_语言基础上加了一些限定,使用有限的词汇和有限的语句来描述加工逻辑,它的结构可分外层和内层两层。

答:自然    形式    自然

30. 流向数据存储的数据流可理解为\_\_\_\_\_文件或\_\_\_\_\_文件,从数据存储流出的数据可理解为从文件\_\_\_\_\_数据或得到\_\_\_\_\_结果。

答:写入    查询    读    查询

31. 结构化分析(Structured Analysis,简称 SA),是面向\_\_\_\_\_进行需求分析的方法。它是 70 年代后期由 Yourdon、Constantine、DeMarco 等人提出和发展,并得到广泛的应用。

答:数据流

32. 三种描述加工逻辑的工具各有优缺点,对于顺序执行和循环执行的动作,用结构化语言描述。对于存在多个条件复杂组织的判断问题,用\_\_\_\_\_和\_\_\_\_\_。

答:判定表    判定树

33. 在有些情况下,数据流图中的某个加工的一组动作依赖于多个逻辑条件的取值。这时,用自然语言或结构化语言都不易清楚地描述出来。而用\_\_\_\_\_就能够清楚地表示复杂的条件组合与应做的动作之间的对应关系。

答:判定表

34. IDEF<sub>0</sub> 方法采用简单的图形符号和简洁的文字说明,描述系统在不同层次上的功能。该方法中,将系统功能称为\_\_\_\_\_,将表示系统功能的图形称为\_\_\_\_\_。

答:活动    活动图形

35. 经过需求分析,开发人员已经基本上理解了用户的要求,确定了目标系统的功能,定义了系统的数据,描述了处理这些数据的基本策略。将这些共同的理解进行整理,最后形成文档\_\_\_\_\_。

答:需求说明书

36. 判定树较判定表直观易读,判定表进行逻辑验证较严格,能把所有的可能性全部都考虑到。可将两种工具结合起来,先用\_\_\_\_\_作底稿,在此基础上产生\_\_\_\_\_。

答:判定表    判定树

37. \_\_\_\_\_是判定表的变形,一般情况下它比判定表更直观,且易于理解和使用。

答:判定树

38. 结构化分析的基本思想是采用\_\_\_\_\_的方法,能有效地控制系统开发的复杂性。

答:自顶向下逐步分解

39. 在 IDEF<sub>0</sub> 图中,连在方框上的箭头有四种类型,它们分别是\_\_\_\_\_。

答:输入、输出、控制与机制

40. 在画分层的 DFD 时,父图与子图的输入输出数据流要\_\_\_\_\_。

答：平衡

41. 在 IDEF<sub>0</sub> 图中,表示系统功能的图形称为\_\_\_\_\_图形。

答：活动

42. 在 IDEF<sub>0</sub> 是建立系统\_\_\_\_\_模型的有效方法。

答:功能

43. 在 IDEF<sub>0</sub> 方法中,被标志为 A - 0 的图称为系统的\_\_\_\_\_图。

答:内外关系图

44. 数据流图中有四种符号元素,它们是\_\_\_\_\_。

答:数据流、加工、数据存储、数据源点和终点

### 三、选择题

1 进行需求分析可使用多种工具,但( )是不适用的。

- A .数据流图                      B .判定表  
C .PAD 图                         D .数据词典

答:C

2. 数据存储和数据流都是( ), 仅仅所处的状态不同。

- A .分析结果  
B .事件  
C .动作  
D .数据

答:D

3 软件需求分析的任务不应包括 ( )

- A .问题分析  
B .信息域分析  
C .结构化程序设计  
D .确定逻辑模型

答:C

4. 在数据流图的基本图形符号中,加工是以数据结构或( )作为加工对象的。

- A .数据内容                      B .信息内容  
C .信息结构                      D .信息流

答:A

5. 需求分析阶段研究的对象是软件项目的 ( )

- A .用户要求                      B .合理要求
- C .模糊要求

答:A

6. 数据词典的任务是对于数据流图中出现的所有被命名的数据元素,在数据词典中作为一个词条加以定义,使得每一个图形元素的名字都有一个确切的 ( )

- A. 对象      B. 解释      C. 符号      D. 描述



答:D

7. 在结构化分析方法(SA),与数据流图配合使用的是 ( )

- A .网络图    B .实体联系图  
C .数据字典                                        D .程序流程图

答：C

8 通过( )可以完成数据流图的细化。

- A .结构分解                      B 功能分解  
C .数据分解                      D .系统分解

答: B

9.需求分析过程中,对算法的简单描述记录在( )中。

- A .层次图  
B .数据字典  
C .数据流图  
D .IPO 图

答:D

10.数据存储和数据流都是数据,仅仅所处的( )不同。

- A.属性      B.状态      C.阶段      D.位置

答: B

11. 在数据流图中,不能由计算机处理的成份是 ( )

- A 控制流                      B .结点  
C 数据流                     D .数据源/ 终点

答:D

12. 在数据流图中,有名字及方向的成份是 ( )

- A .控制流                          B .信息流  
C .数据流                         D .信号流

答：

13. 结构化分析方法(SA)最为常见的图形工具是 ( )

- A 程序流程图                      B 实体联系图  
C 数据流图                         D 结构图

答:C

14. 在结构化分析方法中,用以表达系统内数据的运动情况的工具有 ( )

- A .数据流图                      B .数据词典  
C .结构化英语                  D .判定树与判定表

答:A

15. 软件需求分析阶段的工作, 可以分成以下四个方面: 对问题的识别, 分析与综合, 制定规格说明 ( )

- A. 总结  
B. 实践性报告  
C. 需求分析评审  
D. 以上答案都不正确

答:C

16. 各种分析方法都有它们共同适用的 ( )

- A .说明方法                      B .描述方法  
C .准则                          D .基本原则

答:D

17. 结构化分析方法使用的描述工具“( )”描述系统由哪几部分组成,各部分之间有什么联系

等等。

- A. 数据流图
- B. 数据字典
- C. 判定表
- D. 判定树

答:A

18. 找出下面错误的说法 ( )

- A. 结构化语言外层中的顺序结构是一组祈使语句、选择语句、重复语句的顺序排列。
- B. 结构化语言外层中的选择结构使用 IF - THEN - ELSE - ENDIF 等关键词
- C. 结构化语言的内层可以采用祈使语句的自然语言短语
- D. 结构化语言外层中的重复结构使用 CASE - OF - ENDCASE 等关键词

答:D

19. 结构化分析方法使用的描述工具“ ( ) ”定义了数据流图中每一个图形元素。

- A. 数据流图
- B. 数据字典
- C. 判定表
- D. 判定树

答:B

20. 下列说法正确的是 ( )

- A. 对于顺序执行和循环执行的动作,用判定表和判定树
- B. 对于存在多个条件复杂组合的判断问题,用结构化语言描述
- C. 判定表较判定树直观易读,判定树进行逻辑验证较严格
- D. 可将判定表和判定树两者结合起来,先用判定表作底稿,在此基础上产生判定树

答:D

21. 需求规格说明书的内容不应包括对( )的描述。

- A. 主要功能
- B. 算法的详细过程
- C. 用户界面及运行环境
- D. 软件的性能

答:B

22. 需求规格说明书的作用不应包括 ( )

- A. 软件设计的依据
- B. 用户与开发人员对软件要做什么的共同理解
- C. 软件验收的依据
- D. 软件可行性研究的依据

答:D

23. 分层 DFD 是一种比较严格又易于理解的描述方式,它的顶层图描述了系统的 ( )

- A. 细节
- B. 输入与输出
- C. 软件的作者
- D. 绘制的时间

答:B

24. 数据字典中,一般不包括( )条目。

- A. 数据流
- B. 数据存储
- C. 加工
- D. 源点与终点

答:D

25. 需求分析最终结果是产生 ( )

- A. 项目开发计划
- B. 可行性分析报告
- C. 需求规格说明书
- D. 设计说明书

答:C

26. 需求分析中,开发人员要从用户那里解决的最重要的问题是 ( )

- A. 要让软件做什么
- B. 要给该软件提供哪些信息
- C. 要求软件工作效率怎样
- D. 要让该软件具有何种结构

答:A

27. IDEF<sub>0</sub> 图并不反映出系统 ( )

- A. 做什么                      B. 怎么做                      C. 谁来做                      D. 什么情况下做

答:B

28. 需求分析最终结果是产生 ( )

- A. 项目开发计划                      B. 可行性分析报告  
C. 需求规格说明书                      D. 设计说明书

答:C

29. SA 方法用 DFD 描述 ( )

- A. 系统的控制流程                      B. 系统的数据结构  
C. 系统的基本加工                      D. 系统的功能

答:D

30. 初步用户手册在 ( ) 阶段编写。

- A. 可行性研究                      B. 需求分析  
C. 软件概要设计                      D. 软件详细设计 答:B

31. SA 方法的分析步骤是首先调查了解当前系统的工作流程, 然后 ( )

- A. 获得当前系统的物理模型, 抽象出当前系统的逻辑模型, 建立目标系统的逻辑模型  
B. 获得当前系统的物理模型, 抽象出目标系统的逻辑模型, 建立目标系统的物理模型  
C. 获得当前系统的逻辑模型, 建立当前系统的物理模型, 抽象出目标系统的逻辑模型  
D. 获得当前系统的逻辑模型, 建立当前系统的物理模型, 建立目标系统的物理模型

答:A

32. 需求分析阶段不适用于描述加工逻辑的工具是 ( )

- A. 结构化语言                      B. 判定表                      C. 判定树                      D. 流程图

答:D

33. SA 方法的基本思想是

- A. 自底向上逐步抽象                      B. 自底向上逐步分解  
C. 自顶向下逐步分解                      D. 自顶向下逐步抽象

答:C

34. IDEF<sub>0</sub> 的图形表示中, 连在方框上的箭头有四种类型: 输入、输出、控制和机制。下列说法正确的是

- A. 输入指完成某项活动所需的数据, 用连在方框右边的箭头表示  
B. 输出指执行活动时产生的数据, 用连在方框左边的箭头表示  
C. 机制指所受到的约束条件, 用连在方框下边的箭头表示  
D. 控制活动指活动是由谁来完成的, 用连在方框上边的箭头表示  
E. 当无法区分输入和控制时可将输入看作控制。一个活动可无输入, 但必须至少有一个控制

答:D

35. 找出下面错误的说法 ( )

- A. 每个数据流必须用名词或名词短语命名  
B. 每个加工必须有名字, 通常是动词短语  
C. 每个数据存储必须用名词或名词短语命名  
D. 每个数据源点或终点必须有名字

答:A

36. 找出下面错误的说法 ( )

- A. 判定表能够把在什么条件下系统应做什么动作准确无误地表示出来

- B. 判定表能够描述循环的处理特性
- C. 结构化语言同样能够描述循环的处理特性
- D. 判定树是判定表的变形,一般情况下它比判定表更直观,且易于理解和使用

答: B

#### 四、简答题

##### 1. 什么是数据字典? 其作用是什么? 它有哪些条目?

答: 数据字典(Data Dictionary, 简称 DD)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性的定义和详细的描述。

数据流图仅描述了系统的“分解”, 系统由哪几部分组成, 各部分之间的联系, 并没有对各个数据流、加工、数据存储进行详细说明, 如数据流、数据存储的名字并不能反映其中的数据成分、数据项目内容和数据特性, 在加工中不能反映处理过程等等。分析人员仅靠“图”来完整地理解一个系统的逻辑功能是不可能的。数据字典就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性的定义和详细的描述。它和数据流图共同构成了系统的逻辑模型, 是需求规格说明书的主要组成部分。

数据字典是为分析人员查找数据流图中有关名字的详细定义而服务的, 因此也像普通字典一样, 要把所有条目按一定的次序排列起来, 以便查阅。数据字典有以下四类条目: 数据流、数据项、数据存储、基本加工。数据项是组成数据流和数据存储的最小元素。

数据流条目给出了 DFD 中数据流的定义, 通常列出该数据流的各组成数据项。数据存储条目是对数据存储的定义, 数据项条目是不可再分解的数据单位, 加工条目是用来说明 DFD 中基本加工的处理逻辑的, 数据字典中的加工逻辑主要描述该加工“做什么”, 即实现加工的策略, 而不是实现加工的细节, 它描述如何把输入数据流变换为输出数据流的加工规则。有几种常用的描述方法, 它们是结构化语言、判定表、判定树。

##### 2. 需求分析阶段的文档是什么?

答: 需求分析阶段的文档有:

(1)“需求规格说明书”, 把双方共同的理解与分析结果用规范的方式描述出来, 作为今后各项工作的基础。

(2) 初步用户使用手册, 着重反映被开发软件的用户功能界面和用户使用的具体要求, 用户手册能强制分析人员从用户使用的观点考虑软件。

(3) 确认测试计划, 作为今后确认和验收的依据。

##### 3. 建立数据字典的形式是什么?

答: 建立数据字典一般有两种形式:

(1) 手工建立: 数据字典的内容用卡片形式存放。按四类条目规范的格式印制卡片。在卡片上分别填写各类条目的内容。先按图号顺序排列, 同一图号的所有条目按数据流、数据项、数据存储和加工的顺序排列。同一图号中的同一类条目(如数据流卡片)可按名字的字典顺序存放, 加工一般按编号顺序存放。同一成分在父图和子图都出现时, 则只在父图上定义。建立索引目录。

(2) 利用计算机辅助建立并维护。编制一个“字典生成与管理程序”, 可以按规定的格式输入各类条目, 能对字典条目增、删、改, 能打印出各类查询报告和清单, 能进行完整性、一致性检查等等。利用已有的数据库开发工具, 针对数据字典建立一个数据库文件, 可将数据流、数据项、数据存储和加工分别以矩阵表的形式来描述各个表项的内容, 然后使用开发工具建成数据库文件, 便于修改、查询, 并可随时

打印出来。另外,有的 DBMS 本身包含一个数据字典子系统,建库时能自动生成数据字典。

计算机辅助开发数据字典比手工建立数据字典有更多的优点,能保证数据的一致性和完整性,使用也方便,但增加了技术难度与机器开销。

4 描述加工逻辑有哪些工具？

答:加工逻辑也称为“小说明”,描述加工逻辑一般用以下三种工具:结构化语言、判定表、判定树。

(1)结构化语言。结构化语言是介于自然语言(英语或汉语)和形式语言之间的一种半形式语言。它是在自然语言基础上加了一些限定,使用有限的词汇和有限的语句来描述加工逻辑,它的结构可分成外层和内层两层:

外层:用来描述控制结构,采用顺序、选择、重复三种基本结构。

( )顺序结构:是一组祈使语句、选择语句、重复语句的顺序排列。祈使语句指至少包含一个动词及一个名词,指出要执行的动作及接受动作的对象。

( )选择结构:一般用 IF—THEN—ELSE—ENDIF,CASE—OF—ENDCASE 等关键词。

( )重复结构:一般用 DO—WHILE—ENDDO、REPEAT—UNTIL 等关键词。

内层:一般是采用祈使语句的自然语言短语,使用数据字典中的名词和有限的自定义词,其动词含义要具体,尽量不用形容词和副词来修饰。还可使用一些简单的算术运算和逻辑运算符号。

(2)判定表。在有些情况下,数据流图中的某个加工的一组动作依赖于多个逻辑条件的取值。这时,用自然语言或结构化语言都不易清楚地描述出来。而用判定表就能够清楚地表示复杂的条件组合与应做的动作之间的对应关系。

判定表由四部分组成,如图所示。

条件定义	条件取值的组合
动作定义	在各种取值的组合下应执行的动作

判定表能够把在什么条件下系统应做什么动作准确无误地表示出来,但不能描述循环的处理特性,循环处理还需结构化语言。

(3)判定树。判定树是判定表的变形,一般情况下它比判定表更直观,且易于理解和使用。

这三种描述加工逻辑的工具各有优缺点,对于顺序执行和循环执行的动作,用结构化语言描述。对于存在多个条件复杂组合的判断问题,用判定表和判定树。判定树较判定表直观易读,判定表进行逻辑验证较严格,能把所有的可能性全部都考虑到。可将两种工具结合起来,先用判定表作底稿,在此基础上产生判定树。

5 简述 SA 方法的优缺点。

答:结构化分析方法的优点:结构化分析方法是软件需求分析中公认的、有成效的、技术成熟、使用广泛的一种方法,它较适合于开发数据处理类型软件的需求分析。该方法利用图形等半形式化工具表达需求,简明、易读,也易于使用,为后一阶段的设计、测试、评价提供了有利条件。

结构化分析方法的缺点:

(1)传统的 SA 方法主要用于数据处理方面的问题,主要工具 DFD 体现了系统“做什么”的功能,但它仅是一个静态模型,没有反映处理的顺序,即控制流程。因此,不适合描述实时控制系统。

(2)60 年代末出现的数据库技术,使许多大型数据处理系统中的数据都组织成数据库的形式,SA 方法使用 DFD 在分析与描述“数据要求”方面是有局限的,DFD 应与数据库技术中的实体联系图(ER 图)结合起来(如同 IDEF<sub>0</sub> 功能模型与 IDEF<sub>1</sub> 信息模型相结合一样)。ER 图能增加对数据存储的细节以及数据与数据之间、数据与处理过程之间关系的理解,还解决了在 DD 中所包含的数据内容表示问题,这样才能较完整地描述用户对系统的需求。

(3)对于一些频繁的人机交互的软件系统,如飞机订票、银行管理、文献检索等系统,用户最关心的是如何使用它,输入命令、操作方式、系统响应方式、输出格式等等,都是用户需求的重要方面,DFD 不适合描述人机界面系统的需求。SA 方法往往对这一部分用自然语言作补充。

(4)描述软件需求的精确性有待于提高。

6. 写出在数据字典的定义式中出现的符号。

答:

符号	含义
=	被定义为
+	与
[ ...   ... ]	或
...	重复
m{ ... } n 或 { ... } <sub>m</sub> <sup>n</sup>	重复
( ... )	可选
“ ... ”	基本数据元素
..	

7. 判定表由哪些部分组成？

答:判定表由四部分组成,用双线分割开四个区域,如图所示。

条件定义	条件取值的组合
动作定义	在各种取值的组合下应执行的动作

判定表结构

各部分的含义在图上标出。

8. 结构化语言的结构由哪些部分组成？

答:结构化语言是介于自然语言(英语或汉语)和形式语言之间的一种半形式语言。形式语言精确,但不易被理解,自然语言易理解,但它不精确,可能产生二义性。结构化语言取“长”补“短”,它是在自然语言基础上加了一些限定,使用有限的词汇和有限的语句来描述加工逻辑,它的结构可分成外层和内层两层:

(1)外层:用来描述控制结构,采用顺序、选择、重复三种基本结构。

顺序结构:是一组祈使语句、选择语句、重复语句的顺序排列。祈使语句指至少包含一个动词及一个名词,指出要执行的动作及接受动作的对象。

选择结构:一般用 IF—THEN—ELSE—ENDIF、CASE—OF—ENDCASE 等关键词。

重复结构:一般用 DO—WHILE—ENDDO、REPEAT—UNTIL 等关键词。

(2)内层:一般是采用祈使语句的自然语言短语,使用数据字典中的名词和有限的自定义词,其动词含义要具体,尽量不用形容词和副词来修饰。还可使用一些简单的算术运算和逻辑运算符号。

9. 描述数据流图还有哪些符号？

答:为了使数据流图便于在计算机上输入与输出,以下给出了描述数据流图的另一套基本符号:

————→:表示数据流,只能水平或垂直画。

编号
----

:表示加工。

编号

:表示数据存储。



:表示源点或终点。

#### 10. 数据流图有哪些基本图形符号？

答： :箭头,表示数据流；

:圆或椭圆,表示加工；

=:双杠,表示数据存储；

:方框,表示数据的源点或终点。

#### 11. 画数据流图的步骤有哪些？

答:(1)首先画系统的输入输出,即先画顶层数据流图。顶层流图只包含一个加工,用以表示被开发的系统,然后考虑该系统有哪些输入数据,这些输入数据从哪里来;有哪些输出数据,输出到哪里去。这样就定义了系统的输入、输出数据流。顶层图的作用在于表明被开发系统的范围以及它和周围环境的数据交换关系。顶层图只有一张。

(2)画系统内部,即画下层数据流图。一般将层号从0开始编号,采用自顶向下,由外向内的原则。画0层数据流图时,一般根据当前系统工作分组情况,并按新系统应有的外部功能,分解顶层流图的系统为若干子系统,决定每个子系统间的数据接口和活动关系。

##### (3)注意事项

命名。不论数据流、数据存储还是加工,合适的命名使人们易于理解其含义。数据流的名字代表整个数据流的内容,而不仅仅是它的某些成分,不使用缺乏具体含义的名字,如“数据”、“信息”等,加工名也应反映整个处理的功能,不使用“处理”、“操作”这些笼统的词。

画数据流而不是控制流。数据流图反映系统“做什么”,不反映“如何做”,因此箭头上的数据流名称只能是名词或名词短语,整个图中不反映加工的执行顺序。

一般不画物质流。数据流反映能用计算机处理的数据,并不是实物,因此对目标系统的数据流图一般不要画物质流,如机票预订系统中,人民币也在流动,但并未画出,因为交款是“人工”行为。

每个加工至少有一个输入数据流和一个输出数据流,反映出此加工数据的来源与加工的结果。

编号。如果一张数据流图中的某个加工分解成另一张数据流图时,则上层图为父图,直接下层图为子图。子图应编号,子图上的所有加工也应编号,子图的编号就是父图中相应加工的编号,加工的编号由子图号、小数点及局部号组成。

父图与子图的平衡。子图的输入输出数据流同父图相应加工的输入输出数据流必须一致,此即父图与子图的平衡。

局部数据存储。当某层数据流图中的数据存储不是父图中相应加工的外部接口,而只是本图中某些加工之间的数据接口,则称这些数据存储为局部数据存储。一个局部数据存储只要当它作为某些加工的数据接口或某个加工特定的输入或输出时,就把它画出来,这样有助于实现信息隐蔽。

提高数据流图的易理解性。

#### 12. 试述需求分析的概念及主要分析方法。

答:(1)需求分析

需求指用户对软件系统的需求,需求分析指通过对系统的调查、分析、综合,产生完整的需求说明的过程,即用易读、直观的格式,表达系统“做什么”的问题。需求分析的方法有多种,如结构化分析方法和面向对象分析方法,这些方法有共同适用的基本原则:

能够表达和理解问题的数据域和功能域。

可以把一个复杂问题按某种方式进行划分,并能逐步细化,从而使复杂的问题简化,软件的功能

域、数据域都可以划分。

建立模型。模型(及模式化表示)可以帮助分析人员更好地理解软件系统。逻辑模型是系统功能与数据信息的逻辑表示,不是实现的细节,它反映了系统本质的东西。需求分析的任务也就是该阶段要进行的工作,必须明确这些工作。

## (2) 结构化分析方法

结构化分析方法简称 SA 法,是需求分析中使用最多的方法之一,适用于数据处理类型软件的需求分析。这一方法除了简单、易于掌握之外,还能和设计阶段的结构化设计(SD)衔接,从而取得良好的设计结果。SA 方法的基本手段是“分解”和“抽象”,这是系统开发技术中控制复杂性的两种手段。它先将系统“抽象”成一个模型,此模型是有输入和输出并有系统名称的盒子,然后打开这个子,对它进行逐层分解,直到能被理解、可以实现为止。因此分析的策略是自顶向下、逐层加细,由抽象到具体的过程。由于分析中的主要依据是数据传递及数据变换所形成的数据流,所以结构化分析一般采用的方法是使用数据流图的分析方法,最终结果是产生需求规格说明书,该文档包括一套数据流图、对数据流图中的成分进行定义的一本数据字典及对加工逻辑的描述。

### 13. IDEF 方法分为哪三部分?

答: IDEF<sub>0</sub>: 用来描述系统的功能活动及其联系,建立系统的功能模型。

IDEF<sub>1</sub>: 用来描述系统的信息及其联系,建立系统的信息模型。美国空军项目组对 IDEF<sub>1</sub> 进行了扩充与完善,于 1985 年正式推出了 IDEF<sub>1x</sub>。

IDEF<sub>2</sub>: 用来进行系统模拟,建立系统的动态模型。

### 14. 在分解数据流图时易犯的错误有哪些?

答: (1) 将数据流画成控制流,加工(圆圈)与加工之间只能是数据流。

(2) 有的加工无输入或输出。每个加工至少有一个输入流和一个输出流,反映出加工有数据来源及加工结果。

(3) 父图与子图不平衡,子图反映的是父图中某个加工的分解,因此它的输入、输出数据流应与父图中相应加工的输入、输出流一致。

### 15. 结构化分析的描述工具有哪些?

答: (1) 数据流图

它是表示系统逻辑模型的工具,以图形的方式来表达系统的功能。它往往是分层的,顶层图表明系统的范围及它与周围环境的数据交换关系,下层图是上层图中数据处理的细化,分解层次的多少由系统的复杂程度来决定。教材中强调了画数据流图应注意的事项,这也是初画数据流图时容易出错的情况。这里需要说明的是:

数据流图是为分析人员理解系统功能而服务的,因此它描述的系统既可以是人工系统,又可以是计算机系统,还可以是二者混合的系统,也就是说数据流图既可以描述现行系统又可以描述目标系统的逻辑功能。

每一层的数据处理可以进一步分解以求得对问题的全面理解。

着重强调的是数据流而不是控制流。

## (2) 数据字典

数据字典是用来定义数据流图中各个成分的具体含义的,使这些成分得到确切的解释。注意各个条目中的内容模式。在建立一本数据字典时,应和数据流图中的层次概念相似,先按图号顺序排列,同一图号层按条目顺序排列。同一类条目中除数据处理外按字典顺序排列,数据处理按编号顺序排列。在数据流和数据存储的定义式中如有复杂的数据项,可采取逐级定义的方式,以提高易读性。

## (3) 加工逻辑

加工逻辑指对数据流图中的数据处理(即加工)进行逻辑上的说明,对加工逻辑可以选择三种工具



进行说明。一般情况下选择结构化语言。之所以称为结构化语言,是受结构化程序设计思想的启发而扩展出来的,结构化语言与自然语言不同之处是它只用了有限的词汇,它同程序设计语言不同的是没有严格的语法规则,只使用简单的祈使句、选择语句、循环语句及由它们组成的复合语句。当某个动作的执行不只依赖于一个条件、与多个条件有关时,用多层嵌套的选择语句使加工逻辑说明不能一目了然,此种情况下,可选择判定表或判定树。判定表的优点是能够把所有的条件合并一个不漏地表达出来,特别在条件很多,而且每一个条件的取值有若干个,相应的动作也很多的情况下,在分析这类问题时,判定表比判定树更有效,它能帮助分析员澄清问题,甚至能够发现用户可能遗漏的尚未提出的逻辑要求。判定树容易掌握,用图形表达易读,易于与用户讨论。因此对于条件组合不是太多、逻辑判定不是太复杂的情况,可选择判定树来描述,反之用判定表。

#### 16. 需求分析的难点有哪些?

答:在计算机发展的早期,所求解问题的规模较小,需求分析因此而被忽视。随着软件系统复杂性的提高及规模的扩大,需求分析在软件开发中的所处的地位愈加突出,从而也愈加困难,它的难点主要体现在以下几个方面:

- (1)问题的复杂性。因用户需求所涉及的因素繁多引起,如运行环境和系统功能等等。
- (2)交流障碍。需求分析涉及人员较多,如软件系统用户、问题领域专家、需求工程师和项目管理员等,这些人具备不同的背景知识,处于不同的角度,扮演不同角色,造成了相互之间交流的困难。
- (3)不完备性和不一致性。由于各种原因,用户对问题的陈述往往是不完备的,其各方面的需求还可能存在着矛盾,需求分析要消除其矛盾,形成完备及一致的定义。
- (4)需求易变性。用户需求的变动是一个极为普遍的问题,即使是部分变动,也往往会影响到需求分析的全部,导致不一致性和不完备性。

#### 17. 数据字典的实现有哪几种形式?

答:建立数据字典一般有两种形式:

##### (1)手工建立:数据字典的内容用卡片形式存放

按四类条目规范的格式印制卡片。

在卡片上分别填写各类条目的内容。

先按图号顺序排列,同一图号的所有条目按数据流、数据项、数据存储和加工的顺序排列。

同一图号中的同一类条目(如数据流卡片)可按名字的字典顺序存放,加工一般按编号顺序存放。

同一成分在父图和子图都出现时,则只在父图上定义。

建立索引目录。

##### (2)利用计算机辅助建立并维护

编制一个“字典生成与管理程序”,可以按规定的格式输入各类条目,能对字典条目增、删、改,能打印出各类查询报告和清单,能进行完整性、一致性检查等等。

利用已有的数据库开发工具,针对数据字典建立一个数据库文件,可将数据流、数据项、数据存储和加工分别以矩阵表的形式来描述各个表项的内容。另外,有的 DBMS 本身包含一个数据字典子系统,建库时能自动生成数据字典。

计算机辅助开发数据字典比手工建立数据字典有更多的优点,能保证数据的一致性和完整性,使用也方便,但增加了技术难度与机器开销。

#### 18. 需求分析的基本任务是什么?本阶段主要进行哪些工作?

答:需求分析的基本任务是要准确地定义新系统的目标,为了满足用户需要,回答系统必须“做什么”的问题。在可行性研究和软件计划阶段对这个问题的回答是概括的、粗略的。本阶段要进行以下几方面的工作:

##### (1)问题识别

双方确定对问题的综合需求,这些需求包括:

功能需求:所开发的软件必须具备什么样的功能,这是最重要的。

性能需求:待开发的软件的技术性能指标。如存储容量、运行时间等限制。

环境需求:软件运行时所需要的软、硬件(如机型,外设,操作系统,数据库管理系统等)的要求。

用户界面需求:人机交互方式、输入输出数据格式等等。

另外还有可靠性、安全性、保密性、可移植性、可维护性等方面的需求,这些需求一般通过双方交流、调查研究来获取,并达到共同的理解。

## (2)分析与综合,导出软件的逻辑模型

分析人员对获取的需求,进行一致性的分析检查,在分析、综合中逐步细化软件功能,划分成各个子功能。这里也包括对数据域进行分解,并分配到各个子功能上,以确定系统的构成及主要成份,并用图文结合的形式,建立起新系统的逻辑模型。

## (3)编写文档

编写“需求规格说明书”,把双方共同的理解与分析结果用规范的方式描述出来,作为今后各项工作的基础。

编写初步用户使用手册,着重反映被开发软件的用户功能界面和用户使用的具体要求,用户手册能强制分析人员从用户使用的观点考虑软件。

编写确认测试计划,作为今后确认和验收的依据。

修改完善软件开发计划。在需求分析阶段对待开发的系统有了更进一步的了解,所以能更准确地估计开发成本、进度及资源要求,因此对原计划要进行适当修正。

## 19. 软件需求分析与说明的方法有哪些基本原则?

答:(1)必须能够表达和理解问题的数据域和功能域。数据域包括数据流(即数据通过一个系统时的变化方式)、数据内容和数据结构,而功能域反映上述三方面的控制信息。

(2)可以把一个复杂问题按功能进行分解并可逐层细化。通常软件要处理的问题如果太大、太复杂就很难理解,划分成几部分,并确定各部分间的接口,就可完成整体功能。在需求分析过程中,软件领域中的数据、功能、行为都可以划分。

(3)建模。建立模型可以帮助分析人员更好地理解软件系统的信息、功能、行为,这些模型也是软件设计的基础。

## 20. SA 分析步骤是什么?

答:(1)了解当前系统的工作流程,获得当前系统的物理模型。当前系统(也称现行系统)指目前正在运行的系统,可能是需要改进的正在计算机上运行的软件系统,也可能是人工的处理系统。通过对当前系统的详细调查,了解当前系统的工作过程,同时收集资料、文件、数据、报表等,将看到的、听到的、收集到的信息和情况用图形描述出来。也就是用一个模型来反映自己对当前系统的理解,如画系统流程图。这一模型包含了许多具体因素,反映现实世界的实际情况。

(2)抽象出当前系统的逻辑模型。物理模型反映了系统“怎么做”的具体实现,去掉物理模型中非本质的因素(如物理因素),抽取出本质的因素。所谓本质的因素指系统固有的、不依赖运行环境变化而变化的因素,任何实现均这样做。非本质因素不是固有的,随环境不同而不同,随实现不同而不同。对物理模型进行分析,区别本质因素和非本质因素,去掉非本质因素,应形成当前系统的逻辑模型,反映了当前系统“做什么”的功能。

(3)建立目标系统的逻辑模型。目标系统指待开发的新系统。分析、比较目标系统与当前系统逻辑上的差别,即在当前系统的基础上决定变化的范围,把那些要改变的部分找出来,将变化的部分抽象为一个加工,这个加工的外部环境及输入输出已确定。然后对“变化的部分”重新分解,分析人员根据自己的经验,采用自顶向下逐步求精的分析策略,逐步确定变化部分的内部结构,从而建立目标系统的逻辑模型。

(4)作进一步补充和优化。为了完整地描述目标系统,还要作一些补充:说明目标系统的人机界面,它所处的应用环境及它与外界环境的相互联系,决定人机界面;说明至今尚未详细考虑的细节,如出错处理、输入输出格式、存储容量、响应时间等性能要求与限制。

21.SA 描述工具有哪些？

答:SA 方法利用图形等半形式化的描述方式表达需求,简明易懂,用它们形成需求说明书中的主要部分。这些描述工具是:

- (1)数据流图;
- (2)数据字典;
- (3)描述加工逻辑的结构化语言、判定表、判定树。

其中,“数据流图”描述系统的分解,即描述系统由哪几部分组成,各部分之间有什么联系等等。“数据字典”定义了数据流图中每一个图形元素。结构化语言、判定表或判定树则详细描述数据流图中不能被再分解的每一个加工。

22. 建立功能模型的基本方法是什么？

答:(1)确定建模的范围、观点及目的

在开始为系统建立模型时,首先要确定建模的立足点,包括范围、观点及目的。范围指所讨论的对象是什么,它的边界和外部接口是什么;观点指从什么角度去考虑所研究的问题;目的指确定所研究问题的意图及理由。

(2)建立系统的内外关系图——A - 0 图

IDEF<sub>0</sub> 方法建立的功能模型是一组有层次关系的图形,以字母 A 开头的编号来标志图形在层次中的位置。先建立系统的内外关系图,该图用来抽象地描述所研究的问题及其边界或数据接口。图中只有一个活动,活动名概括地描述系统的内容,用进入和离开的箭头表示系统与环境的数据接口,确定了系统边界。

(3)建立顶层图——A0 图

把 A - 0 图分解为 3~6 个主要部分得到 A0 图,它清楚地表达了 A - 0 图在同样信息范围内的细节,从结构上反映了模型的观点,是系统功能模型真正的顶层图。该图中各方框所表示活动的详细含义由低层次的图形说明。

(4)建立低层次的图形

按照自顶向下的方法,从 A0 图开始逐层分解,建立一系列的活动图形,直到最低层为止。

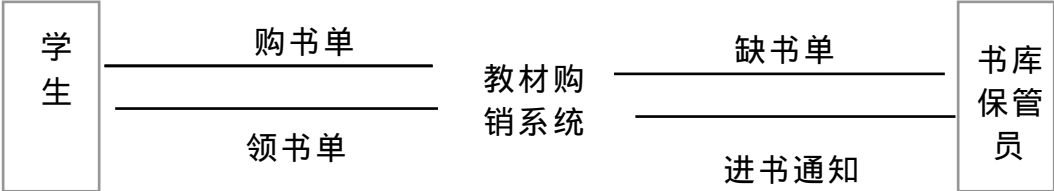
五、应用题

1 某学校计算机教材购销系统有以下功能：

学生买书,首先填写购书单,计算机根据各班学生用书表以及售书登记表审查有效性,若有效,计算机根据教材存量表进一步判断书库是否有书,若有书,计算机把领书单返回给学生,学生凭领书单到书库领书。对脱销的教材,系统用缺书单的形式通知书库,新书购进库后,也由书库将进书通知返回给系统。

请就以上系统功能画出分层的 DFD 图,并建立重要条目的数据字典。

答:

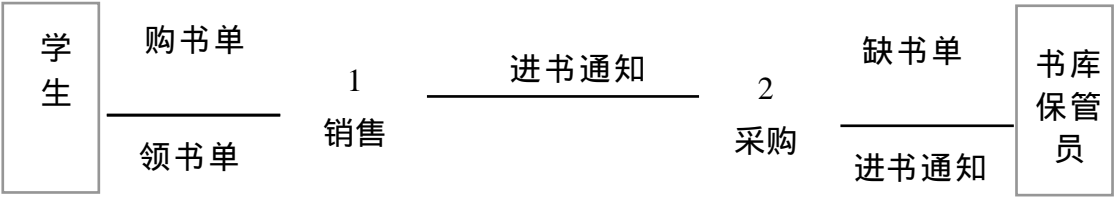


教材购销系统的顶层 DFD

---

F1 教材存量表

---



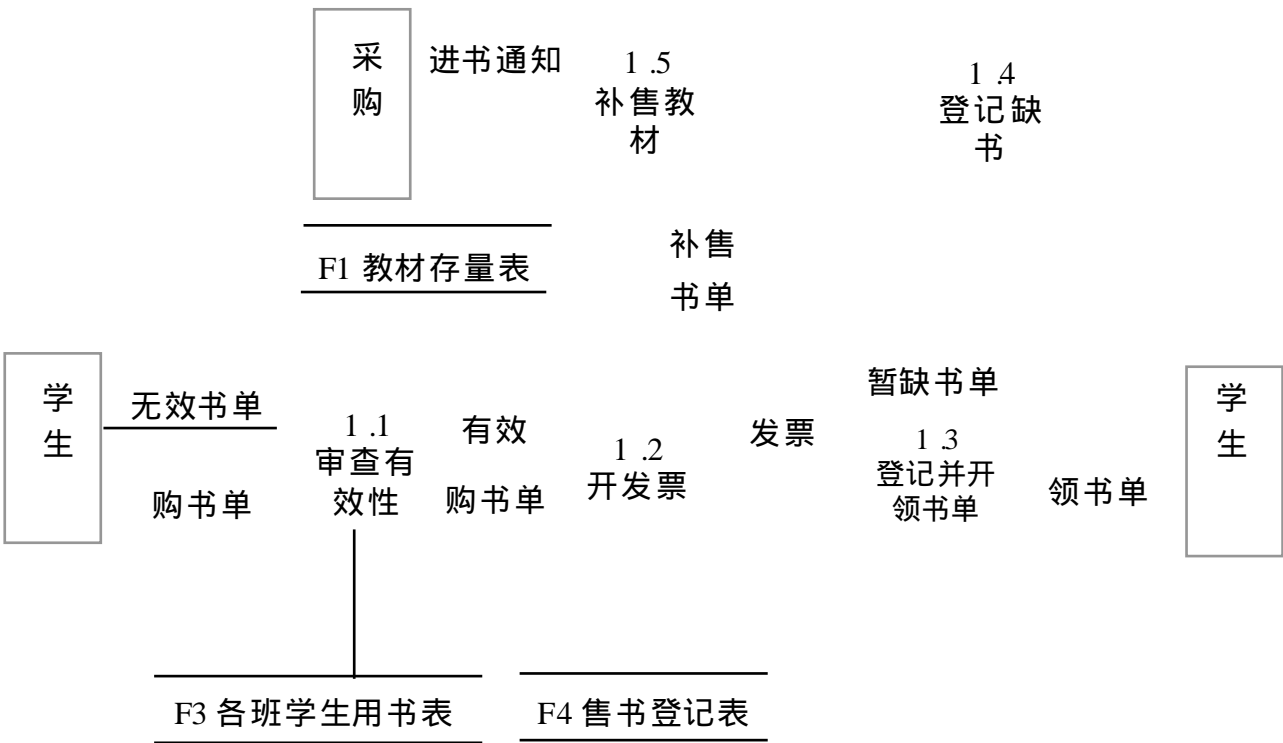
## F2 缺书登记表

## 第二层 DFD - 教材购销系统

---

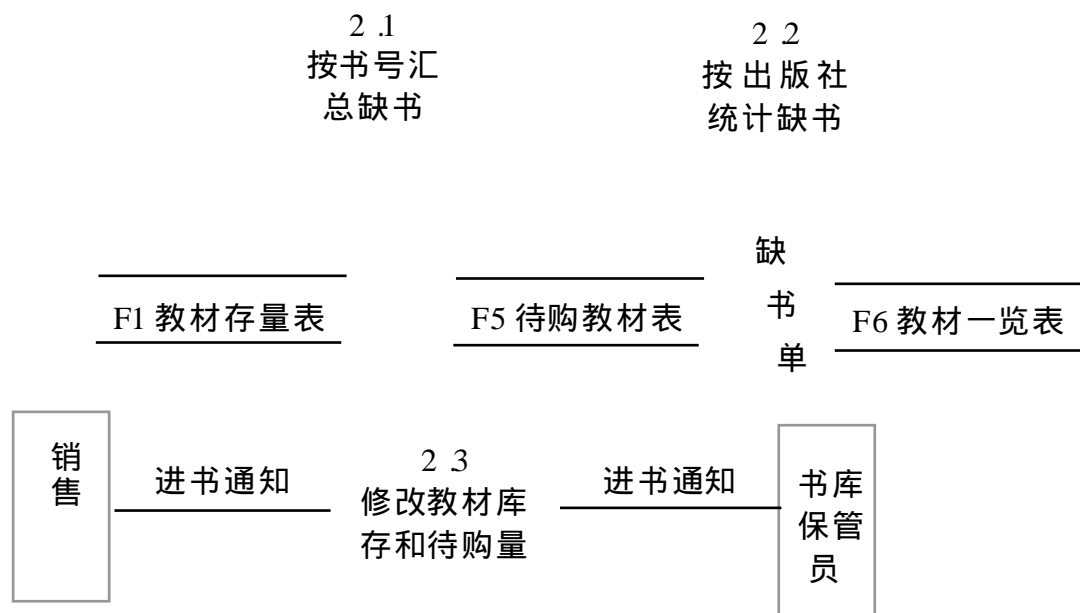
F2 缺书登记表

---



### 第三层 DFD - 销售子系统

F2 缺书登记表



第三层 DFD - 采购子系统

2 某厂对部分职工重新分配工作的政策是:年龄在 20 岁以下者,初中文化程度脱产学习,高中文化程度当电工;年龄在 20 岁至 40 岁之间者,中学文化程度男性当钳工,女性当车工,大学文化程度都当技术员;年龄在 40 岁以上者,中学文化程度当材料员,大学文化程度当技术员。请用结构化语言、判定表或判定树描述上述问题的加工逻辑。

答: (1) 结构化语言。

```
IF 年龄 < 20
    IHEN IF 文化程度 = 初中
        THEN 脱产学习
        ELSE 工作为电工
    ENDIF
ELSE IF 年龄 >= 40
    THEN IF (文化程度 = 大学)
        THEN 工作为技术员
        ELSE IF 性别 = 男性
            THEN 工作为钳工
            ELSE 工作为钳工
        ENDIF
    ENDIF
ELSE IF (文化程度 = 大学)
    THEN 工作为技术员
    ELSE 工作为材料员
ENDIF
ENDIF
```

ENDIF

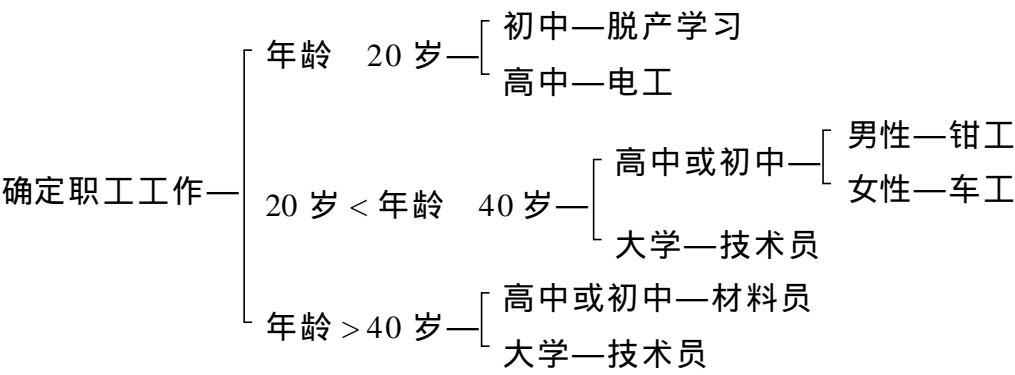
(2)判定表:先画出条件取值表如下表:

条件名	取 值	符 号	取值数
年 龄	年龄 20 20 < 年龄 40 年龄 > 40	C Y L	$m_1 = 3$
文化程度	初中 高中 大学	J S U	$m_2 = 3$
性 别	男 女	M F	$m_3 = 2$

判定表如下:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
年 龄	C	C	C	C	C	C	Y	Y	Y	Y	Y	Y	L	L	L	L	L	L
文化程度	J	J	S	S	U	U	J	J	S	S	U	U	J	J	S	S	U	U
性 别	M	F	M	F	M	F	M	F	M	F	M	F	M	F	M	F	M	F
脱产学习																		
电 工																		
钳 工																		
车 工																		
技术员																		
材料员																		

判定树如下:



3. 假设某航空公司规定,乘客可以免费托驼行李的重量不超过 30 公斤。当行李的重量超过 30 公斤时,对一般舱的国内乘客超重部分每公斤收费 4 元,对头等舱的国内乘客超重部分每公斤收费 6 元。对国外乘客超重部分每公斤收费比国内乘客多一倍,对残疾乘客超重部分每公斤收费比正常乘客少一

半。试画出相应判定表。

答: 设乘客托运行李的重量为  $W$ , 则判定表如下表所示。

乘客条件 收费标准 (元)	$W \leq 30$	$W > 30$							
		国内乘客				国外乘客			
		一般舱		头等舱		一般舱		头等舱	
		残疾	普通	残疾	普通	残疾	普通	残疾	普通
免费									
$2 * (w - 30)$									
$3 * (w - 30)$									
$4 * (w - 30)$									
$6 * (w - 30)$									
$8 * (w - 30)$									
$12 * (w - 30)$									

## 第四章 软件概要设计

本章总的要求是:必须深刻理解软件设计的基本任务,软件设计的基本原理。熟练掌握结构化设计方法。

深刻理解模块化、抽象、信息隐蔽、模块独立性等概念,明确度量模块独立性的标准——耦合性与内聚性。

要求熟练掌握的技能是:能划分数据流的类型,将其转换成软件结构图,并能根据优化准则将其优化。

### 考试要求

#### 1. 软件概要设计的基本任务

软件设计、概要设计、详细设计,要求达到识记层次。

软件设计的基本任务,要求达到领会层次。

#### 2. 软件设计的基本原理

模块化、抽象、信息隐蔽、模块独立性、内聚性、耦合性,要求达到领会层次。

内聚性、耦合性的划分,要求达到领会层次。

#### 3. 软件结构优化准则

软件结构、模块的影响范围、模块的控制范围,要求达到领会层次。

软件结构设计的优化准则,要求达到领会层次。

#### 4. 面向数据流的设计方法

变换流、事务流,要求达到识记层次。

将变换流或事务流类型的数据流图,转换成软件结构,要求达到简单应用层次。

将一个复杂型数据流图转换成软件结构图并优化,要求达到综合应用层次。

#### 5. 基于 IDEF<sub>0</sub> 图的设计方法

IDEF<sub>0</sub> 图的设计方法,要求达到领会层次。

#### 6. 表示软件结构的另一种图形工具——HIPO 图

表示软件结构的另一种图形工具——HIPO 图,要求达到领会层次。

### 知识重点

#### (一) 软件概要设计的基本任务

1. 设计软件系统结构(简称软件结构)。为了实现目标管理,首先进行结构设计,具体为:

(1) 采用某种设计方法,将一个复杂的系统按功能划分为模块。



- (2)确定每个模块的功能。
- (3)确定模块之间的调用关系。
- (4)确定模块之间的接口,即模块之间传递的信息。
- (5)评价模块结构的质量。

## 2 .数据结构及数据库设计

(1)数据结构的设计。采用逐步细化的方法,对需求分析阶段获得的数据字典中的数据的结构特性等加以细化。

(2)数据库的设计。数据库的设计指数据存储文件的设计,主要进行概念设计、逻辑设计、物理设计三方面设计。

## 3 .编写概要设计文档。文档主要有:

- (1)概要设计说明书。
- (2)数据库设计说明书。
- (3)进一步补充需求分析阶段编写的用户手册。
- (4)修订测试计划,对测试策略、方法、步骤提出明确要求。

## 4 .评审

对设计部分是否完整地实现了需求中规定的功能、性能等要求,设计方案的可行性,关键的处理内外部接口定义正确性、有效性,各部分之间的一致性等等都一一进行评审。

# (二)软件设计的基本原理

## 1 .模块化

模块化是指解决一个复杂问题时自顶向下逐层把软件系统划分成若干模块的过程。每个模块完成一个特定的子功能,所有的模块按某种方法组装起来,成为一个整体,完成整个系统所要求的功能。

## 2 .抽象

抽象是认识复杂现象过程中使用的思维工具,即抽出事物本质的共同的特性而暂不考虑它的细节,不考虑其他因素。软件工程过程中的每一步都可以看作是对软件解决方法的抽象层次的一次细化。在进行软件设计时,抽象与逐步求精、模块化密切相关,帮助我们定义软件结构中模块的实体,由抽象到具体地分析和构造出软件的层次结构,提高软件的可理解性。

## 3 .信息隐蔽

通过抽象,可以确定组成软件的过程实体。通过信息隐蔽,可以定义和实施对模块的过程细节和局部数据结构的存取限制。信息隐蔽指在设计和确定模块时,使得一个模块内包含的信息(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。“隐蔽”的意思是,有效的模块化通过定义一组相互独立的模块来实现,这些独立的模块彼此之间仅仅交换那些为了完成系统功能所必需的信息,而将那些自身的实现细节与数据“隐藏”起来。信息隐藏为软件系统的修改、测试及以后的维护都带来好处。

## 4 .模块独立性

- (1)耦合性。也称块间联系。指软件系统结构中各模块间相互联系紧密程度的一种

度量。模块之间联系越紧密,其耦合性就越强,模块的独立性则越差。模块间耦合高低取决于模块间接口的复杂性、调用的方式及传递的信息。模块的耦合性有以下六种类型:无直接耦合、数据耦合、标记耦合、控制耦合、公共耦合、内容耦合,它们的耦合程度由低到高。

(2)内聚性。又称块内联系。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。若一个模块内各元素(语句之间、程度段之间)联系的越紧密,则它的内聚性就越高。内聚性有以下六种类型:偶然内聚、逻辑内聚、时间内聚、通信内聚、顺序内聚、功能内聚,它们的内聚程度由低到高。

### (三)软件结构图

#### 1 .软件结构图概念

软件结构图是软件系统的模块层次结构,反映了整个系统的功能实现,即将来程序的控制层次体系。

软件结构往往用树状或网状结构的图形来表示。结构图 (Structure Chart, 简称 SC) 的主要内容有:模块、模块的控制关系、模块间的信息传递。

#### 2 .结构图的形态特性

(1)深度:指结构图控制的层次,也是模块的层数。

(2)宽度:指一层中最大的模块个数。

(3)扇出:指一个模块直接下属模块的个数。

(4)扇入:指一个模块直接上属模块的个数。

### (四)软件结构设计优化准则

软件概要设计的主要任务就是软件结构的设计,为了提高设计的质量,必须根据软件设计的原理改进软件设计,提出以下软件结构的设计优化准则:

1 .划分模块时,尽量做到高内聚、低耦合,保持模块相对独立性,并以此原则优化初始的软件结构。

2 .一个模块的作用范围应在其控制范围之内,且判定所在的模块应与受其影响的模块在层次上尽量靠近。

3 .软件结构的深度、宽度、扇入、扇出应适当。软件结构从形态上,总的考虑是顶层扇出数较高一些,中间层扇出数较低一些,底层扇入数较高一些。

4 .模块的大小要适中。

5 .模块的接口要简单、清晰、含义明确,便于理解,易于实现、测试与维护。

### (五)面向数据流的设计方法

面向数据流的设计是以需求分析阶段产生的数据流图为基础,按一定的步骤映射成软件结构,因此又称结构化设计 (Structured Design, 简称 SD)。该方法与结构化分析 (SA) 衔接,构成了完整的结构化分析与设计技术,是目前使用最广泛的软件设计方法之一。

#### 1 .数据流的类型

数据流图(DFD)一般可分为变换型和事务型两类。

(1)变换型数据流图。变换型的 DFD 是输入、变换(或称处理)和输出三部分组成。

变换型数据处理的工作过程一般分为三步:取得数据、变换数据和给出数据,这三步体现了变换型 DFD 的基本思想。变换是系统的主加工,变换输入端的数据流为系统的逻辑输入,输出端为逻辑输出。而直接从外部设备输入数据称为物理输入,反之称为物理输出。外部的输入数据一般要经过输入正确性和合理性检查、编辑、格式转换等预处理,这部分工作都由逻辑输入部分完成,它将外部形式的数据变成内部形式,送给主加工。同理,逻辑输出部分把主加工产生的数据的内部形式转换成外部形式后物理输出。因此变换型的 DFD 是一个顺序结构。

(2)事务型的数据流图。若某个加工将它的输入流分离成许多发散的数据流,形成许多加工路径,并根据输入的值选择其中一个路径来执行,这种特征的 DFD 称为事务型的数据流图,这个加工称为事务处理中心。

一个大型的软件系统的 DFD,经常既具有变换型的特征,又具有事务型的特征,如事务型 DFD 中的某个加工路径可能是变换型。

## 2 .SD 方法设计过程

(1)精化 DFD。仔细地研究分析 DFD 并参照数据字典,认真理解其中的有关元素,检查有无遗漏或不合理之处,进行必要的修改。

(2)确定 DFD 类型。如果是变换型,确定变换中心和逻辑输入、逻辑输出的界线,映射为变换结构的顶层和第一层;如果是事务型,确定事务中心和加工路径,映射为事务结构的顶层和第一层。

(3)分解上层模块,设计中下层模块结构。

(4)根据优化准则对软件结构求精。

(5)描述模块功能、接口及全局数据结构。

(6)复查,如果有错,转向(2)修改完善,否则进入详细设计。

## 3 .变换分析设计方法

当 DFD 具有较明显的变换特征时,则按照以下步骤设计:

(1)确定 DFD 中的变换中心、逻辑输入和逻辑输出。

(2)设计软件结构的顶层和第一层——变换结构。

(3)设计中、下层模块。对第一层的输入、变换、输出模块自顶向下逐层分解。

输入模块下属模块的设计。输入模块的功能是向它的调用模块提供数据,所以必须要有数据来源。这样输入模块应由两部分组成:接收数据、转换成调用模块所需的信息。因此,每个输入模块可以设计成两个下属模块:一个接收,一个转换,用类似的方法一直分解下去,直到物理输入端。

输出模块下属模块的设计。输出模块的功能是将它的调用模块产生的结果送出。因此也应由两部分组成:将数据转换成下属模块所需的形式,发送数据。这样每个输出模块可以设计成两个下属模块:一个转换,一个发送,一直到物理输出端。

变换模块下属模块的设计。根据 DFD 中变换中心的组成情况,按照模块独立性的原则来组织其结构,一般对 DFD 中每个基本加工建立一个功能模块。

设计的优化。

#### 4.事务分析设计方法

(1)确定 DFD 中的事务中心和加工路径,当 DFD 中的某个加工具有明显地将一个输入数据流分解成多个发散的输出数据流时,该加工就是事务中心。从事务中心辐射出去的数据流为各个加工路径。

(2)设计软件结构的顶层和第一层——事务结构,首先设计一个顶层模块,它是一个全控模块,因此,事务型软件结构应包括两个部分:一个接收分支和一个发送分支。

接收分支:负责接收数据,它的设计与变换型 DFD 的输入部分设计方法相同。

发送分支:通常包含一个调度模块,它控制管理所有的下层的事务处理模块。当事务类型不多时,调度模块可与主模块合并。

(3)事务结构中。下层模块的设计、优化等工作同变换结构。

#### 5.综合型数据流图映射成软件结构的设计

当一个系统的 DFD 中既有变换流,又有事务流时,这就是一个综合的数据流图,其软件结构设计方法如下:

(1)确定 DFD 整体上的类型。变换型具有顺序处理的特点,而事务型具有平行分别处理的特点,只要从 DFD 整体、主要功能处理分析其特点,就可区分出该 DFD 整体类型。

(2)标出局部的 DFD 范围,确定其类型。

(3)按整体和局部的 DFD 特征,设计出软件结构。

#### (六)基于 IDEF<sub>0</sub> 图的设计方法

基于 IDEF<sub>0</sub> 图的设计也是结构化设计技术之一,它以系统的功能模型和信息结构为基础设计系统的软件结构。由于 IDEF<sub>0</sub> 图按照自顶向下逐层对系统进行分解,并且对系统的每一功能的输入、输出、约束机制都进行了全面的描述,因此,在系统概要设计时,一般按照 IDEF<sub>0</sub> 图的分解层次,逐层将其转换成软件结构图。

#### (七)表示软件结构的另一种图形工具——HIPO 图

HIPO 图是表示软件系统结构的另一种工具。它既可以描述软件总的模块层次结构——H 图(层次图),又可以描述每个模块输入输出数据、处理功能及模块调用的详细情况 HIPO 图。HIPO 图是以模块分解的层次性以及模块内部输入、处理、输出三大基本部分为基础建立的。

## 反馈测试题解

### 一、名词解释

#### 1 抽象

答:抽象是认识复杂现象过程中使用的思维工具,即抽出事物本质的共同的特性而暂不考虑它的细节,不考虑其他因素。

## 2 模块

答:在程序中是数据说明、可执行语句等程序对象的集合,或者是单独命名和编址的元素,在软件的体系结构中,模块是可组合、分解和更换的单元。

## 3 模块化

答:模块化是指解决一个复杂问题时自顶向下逐层把软件系统划分成若干模块的过程。每个模块完成一个特定的子功能,所有的模块按某种方法组装起来,成为一个整体,完成整个系统所要求的功能。

## 4 数据耦合

答:数据耦合指两个模块之间有调用关系,传递的是简单的数据值,相当于高级语言中的值传递。

## 5 耦合性

答:耦合性也称块间联系。指软件系统结构中各模块间相互联系紧密程度的一种度量。

## 6 控制耦合

答:控制耦合指一个模块调用另一个模块时,传递的是控制变量(如开关、标志等),被调模块通过该控制变量的值有选择地执行块内某一功能。

## 7 无直接耦合

答:无直接耦合指两个模块之间没有直接的关系,它们分别从属于不同模块的控制与调用,它们之间不传递任何信息。

## 8 内聚性

答:内聚性又称块内联系。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。

## 9 模块独立性

答: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。

## 10 标记耦合

答:标记耦合指两个模块之间传递的是数据结构,如高级语言中的数组名、记录名、文件名等这些名字即为标记,其实传递的是这个数据结构的地址。

## 11. 软件结构

答:指系统的体系结构,即整个系统的有层次的模块控制结构。教材中介绍了两种表示软件结构的工具——结构图(SC)和 HIPO 图中的 H 图,要熟悉结构图中的符号。

## 12. 公共耦合(Common Coupling)

答:指通过一个公共数据环境相互作用的那些模块间的耦合。公共数据环境可以是全程变量或数据结构、共享的通信区、内存的公共覆盖区及任何存储介质上的文件、物理设备等(也有将共享外部设备分类为外部耦合)。

## 13. 内容耦合(Content Coupling)

答:这是最高程度的耦合,也是最差的耦合。当一个模块直接使用另一个模块的内部数据,或通过非正常入口而转入另一个模块内部,这种模块之间的耦合为内容耦合,这种情况往往出现在汇编程序设计中。

## 14. 通信内聚(Communicational Cohesion)

答:指模块内所有处理元素都在同一个数据结构上操作(有时称之为信息内聚),或者指各处理使用相同的输入数据或者产生相同的输出数据。

## 15. 时间内聚(Temporal Cohesion)

答:指需要同时执行的动作组合在一起形成的模块为时间内聚模块。

## 16. 逻辑内聚(Logical Cohesion)

答:指模块内执行几个逻辑上相似的功能,通过参数确定该模块完成哪一个功能。

#### 17. 顺序内聚(Sequential Cohesion)

答:指一个模块中各个处理元素都密切相关于同一功能且必须顺序执行,前一功能元素的输出就是下一功能元素的输入。

#### 18. 偶然内聚(Coincidental Cohesion)

答:指一个模块内的各处理元素之间没有任何联系。

## 二、填空题

1 进入了设计阶段,要把软件“做什么”的\_\_\_\_\_变换为“怎么做”的\_\_\_\_\_,即着手实现软件的需求,并将设计的结果反映在\_\_\_\_\_文档中。

答:逻辑模型 物理模型 设计规格说明书

2 在软件需求分析阶段,已经搞清楚了软件\_\_\_\_\_的问题,并把这些需求通过\_\_\_\_\_描述了出来,这也是目标系统的\_\_\_\_\_。

答:做什么 规格说明书 逻辑模型

3 软件设计是一个把\_\_\_\_\_转换为\_\_\_\_\_的过程,包括\_\_\_\_\_和\_\_\_\_\_。

答:软件需求 软件表示 概要设计 详细设计

4 设计软件结构,具体为: 采用某种设计方法,将一个复杂的系统按功能划分成\_\_\_\_\_。确定每个模块的\_\_\_\_\_。 确定模块之间的\_\_\_\_\_。 确定模块之间的\_\_\_\_\_,即模块之间传递的信息。 评价模块结构的质量。

答:模块 功能 调用关系 接口

5 概要设计文档主要有:\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_。

答:概要设计说明书 数据库设计说明书 进一步补充的用户手册 修订的测试计划

6 在一个模块中,\_\_\_\_\_反映模块的外部特性,\_\_\_\_\_反映它的内部特性。

答:功能、状态与接口 逻辑

7 概要设计评审是对设计部分是否完整地实现了需求中规定的\_\_\_\_\_、\_\_\_\_\_等要求,设计方案的\_\_\_\_\_,关键的处理及内外部接口定义\_\_\_\_\_,各部分之间的\_\_\_\_\_等等都一一进行评审。

答:功能 性能 可行性 正确性 有效性 一致性

8 \_\_\_\_\_是指解决一个复杂问题时自顶向下逐层把软件系统划分成若干模块的过程。每个模块完成一个特定的\_\_\_\_\_,所有的模块按某种方法\_\_\_\_\_起来,成为一个整体,完成整个系统所要求的功能。

答:模块化 子功能 组装

9 通过\_\_\_\_\_,可以确定组成软件的过程实体。通过\_\_\_\_\_,可以定义和实施对模块的过程细节和局部数据结构的存取限制。

答:抽象 信息隐蔽

10 开发一个大而复杂的软件系统,将它进行适当的分解,不但可降低其复杂性,还可减少\_\_\_\_\_,从而降低\_\_\_\_\_,提高\_\_\_\_\_,这就是\_\_\_\_\_的依据。

答:开发工作量 开发成本 软件生产率 模块化

11 信息隐蔽指在设计和确定模块时,使得一个模块内包含的信息(过程或数据),对于\_\_\_\_\_的其他模块来说,是不能\_\_\_\_\_的。

答:不需要这些信息 访问

12.抽象是认识复杂现象过程中使用的思维工具,即抽出事物\_\_\_\_\_特性而暂不考虑它的\_\_\_\_\_,不考虑其他因素。

答:本质的共同的 细节

13.控制耦合指一个模块调用另一个模块时,传递的是(如开关、标志等),被调模块通过\_\_\_\_\_有选择地执行块内某一功能。因此被调模块内应具有多个功能,哪个功能起作用受其\_\_\_\_\_控制。

答:控制变量 该控制变量的值 调用模块

14.公共耦合指通过一个\_\_\_\_\_相互作用的那些模块间的耦合。公共耦合的复杂程度随\_\_\_\_\_增加而增加。

答:公共数据环境 耦合模块的个数

15.\_\_\_\_\_是最高程度的耦合。这种耦合出现在当一个模块直接使用另一个模块的\_\_\_\_\_,或通过\_\_\_\_\_转入另一个模块内部时。

答:内容耦合 内部数据 非正常入口

16.顺序内聚指一个模块中各个处理元素都密切相关于\_\_\_\_\_且必须\_\_\_\_\_,前一功能元素的\_\_\_\_\_就是下一功能元素的\_\_\_\_\_。

答:同一功能 顺序执行 输出 输入

17.通信内聚指模块内所有处理元素都在\_\_\_\_\_上操作,有时称之为\_\_\_\_\_,或者指各处理使用相同的\_\_\_\_\_或者产生相同的\_\_\_\_\_。

答:同一个数据结构 信息内聚 输入数据 输出数据

18.功能内聚是内聚程度最\_\_\_\_\_的内聚,指模块内所有元素共同完成\_\_\_\_\_,缺一不可。功能内聚的模块与其他模块的耦合是\_\_\_\_\_的。

答:强 一个功能 弱

19.若某个加工将它的输入流分离成许多发散的数据流,形成许多加工路径,并根据输入的值选择其中一个路径来执行,这种特征的 DFD 称为\_\_\_\_\_的数据流图,这个加工称为\_\_\_\_\_。

答:事务型 事务处理中心

20.数据库的设计指数据存储文件的设计,主要进行的设计方面有:\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_。

答:概念设计 逻辑设计 物理设计

21.结构化设计简称\_\_\_\_\_。数据流图一般可分为\_\_\_\_\_型和\_\_\_\_\_型两类。\_\_\_\_\_型的 DFD 是一个顺序结构。

答:SD 变换 事务 变换

22.通过信息隐蔽,可以定义和实施对模块的过程细节和局部数据结构的\_\_\_\_\_。

答:存取限制

23.软件概要设计阶段的基本任务主要指\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_等四个方面。

答:设计软件系统结构 数据结构及数据库设计 编写概要设计文档 评审

24.将软件系统划分模块时,要尽量做到\_\_\_\_\_,是高模块的\_\_\_\_\_。

答:高内聚低耦合 独立性

25.软件结构从形态上总的考虑是:顶层扇出数较\_\_\_\_\_一些,中间层扇出数较\_\_\_\_\_一些,底层扇入数较\_\_\_\_\_一些。

答:高 低 高

26.模块的基本属性有:\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_。

答:接口 功能 逻辑 状态

27.在结构图中,模块用\_\_\_\_\_表示,并用名字标识该模块。两个模块间用\_\_\_\_\_或\_\_\_\_\_连接表示它们的控制关系,调用模块和被调用模块的关系称为\_\_\_\_\_与\_\_\_\_\_的关系或者

“\_\_\_\_\_”与“\_\_\_\_\_”的关系。模块间还经常用带注释的短箭头表示模块调用过程中来回传递的信息。有时箭头尾部带\_\_\_\_\_的表示传递的是数据,带\_\_\_\_\_的表示传递的是控制信息。

答:方框 单向箭头 直线 上属 下属 统率 从属 空心圆 实心圆

28. 软件结构往往用\_\_\_\_\_状或\_\_\_\_\_状结构的图形来表示。软件工程中,一般采用 70 年代中期美国 Yourdon 等人提出的称为\_\_\_\_\_,简称\_\_\_\_\_的工具来表示软件结构。

答:树 网 结构图 SC

29. 一个模块内各元素联系得越紧密,则它的内聚性就越\_\_\_\_\_。按由低到高的顺序,模块的内聚类型有:\_\_\_\_\_内聚、\_\_\_\_\_内聚、\_\_\_\_\_内聚、\_\_\_\_\_内聚、\_\_\_\_\_内聚、\_\_\_\_\_内聚。

答:高 偶然 逻辑 时间 通信 顺序 功能

30. 两个模块间用\_\_\_\_\_(或直线)连接表示它们的控制关系。

答:单向箭头

31. 数据库的“概念设计”与“逻辑设计”分别对应于系统开发中的“\_\_\_\_\_”与“\_\_\_\_\_”,而数据库的“物理设计”与模块的“\_\_\_\_\_”相对应。

答:需求分析 概要设计 详细设计

32. 面向数据流的设计是以需求分析阶段产生的数据流图为基础,按一定的步骤映射成软件结构,因此又称\_\_\_\_\_(Structured Design,简称 SD)。

答:结构化设计

33. \_\_\_\_\_是软件系统的模块层次结构,反映了整个系统的功能实现,即将来程序的控制层次体系。

答:软件结构图

34. 从以上内容看,软件结构的设计是以\_\_\_\_\_为基础的,在需求分析阶段,已经把系统分解成层次结构。设计阶段,以需求分析的结果为依据,从实现的角度进一步划分为模块,并组成模块的层次结构。

答:模块

35. 要把数据流图(DFD)转换成软件结构,首先必须研究 DFD 的类型。各种软件系统,不论 DFD 如何庞大与复杂,一般可分为\_\_\_\_\_和\_\_\_\_\_两类。

答:变换型 事务型

36. 软件结构的设计是\_\_\_\_\_关键的一步,直接影响到下一阶段详细设计与编码的工作。

答:概要设计

37. 模块间还经常用带注释的短箭头表示模块调用过程中来回传递的信息。有时箭头尾部带空心圆的表示传递的是\_\_\_\_\_,带实心圆的表示传递的是\_\_\_\_\_。

答:数据 控制信息

38. 模块用\_\_\_\_\_表示,并用名字标识该模块,名字应体现该模块的功能。

答:方框

39. 为了防止软件概要设计的错误传播到开发的后续阶段,在概要设计文档完成以后,要进行\_\_\_\_\_。

答:软件评审

40. 两个模块通过全程变量相互作用,这种耦合方式称为\_\_\_\_\_。

答:公共耦合

41. 一个模块的作用范围指\_\_\_\_\_的集合。

答:受该模块内一个判定影响的所有模块

42. 将与同一张年报表有关的所有程序段组成一个模块,该模块的内聚性为\_\_\_\_\_。



答:通信内聚

43. 一个模块的控制范围指\_\_\_\_\_的集合。

答:模块本身以及其所有下属模块

44. 按国际 GB8576 - 88 的“ 计算机软件产品开发文件编制指南 ”规定, 软件设计文档可分为“ \_\_\_\_\_ ”、“ \_\_\_\_\_ ”、“ \_\_\_\_\_ ”。

答:概要设计说明书 详细设计说明书 数据库设计说明书

45. 对于软件的独立性的衡量,根据模块的外部特征和内部特征,提出了两个定性的度量标准,即:\_\_\_\_\_和\_\_\_\_\_。

答:耦合性 内聚性

46. 如果只有两个模块之间有公共数据环境,这种公共耦合有两种情况:一是一个模块只是给公共数据环境送数据,另一个模块只是从公共环境中取数据,这是\_\_\_\_\_耦合。二是两个模块都既往公共数据环境中送数据,又从里面取数据,这是\_\_\_\_\_耦合。

答:比较松散的公共 紧密的数据

47. 按由低到高的顺序,模块的耦合类型有:\_\_\_\_\_耦合、\_\_\_\_\_耦合、\_\_\_\_\_耦合、\_\_\_\_\_耦合、\_\_\_\_\_耦合、\_\_\_\_\_耦合。

答:无直接 数据 标记 控制 公共 内容

48. 模块之间联系越紧密,其耦合性就越\_\_\_\_\_,模块的独立性就越\_\_\_\_\_。

答:强 差

49. 数据库技术是一项专门的技术,数据库的“ 概念设计 ”与“ 逻辑设计 ”分别对应于系统开发中的“ \_\_\_\_\_ ”与“ \_\_\_\_\_ ”,而数据库的“ 物理设计 ”与模块的“ \_\_\_\_\_ ”相对应。答:需求分析 概要设计 详细设计

### 三、选择题

1. 首先将系统中的关键部分设计出来,再让系统其余部分的设计去适应它们,这称为 ( )

- A .模块化设计
- B 逐步求精
- C .由底向上设计
- D .自顶向下设计

答:C

2. 模块( ),则说明模块的独立性越强。

- A .耦合越强
- B 扇入数越高
- C .耦合越弱
- D 扇入数越低

答:C

3. ( )数据处理问题的的工作过程大致分为三步,即取得数据、变换数据和给出数据。

- A .变换型
- B 事务型
- C .结构化
- D 非结构化

答:A

4. 块间的信息可以作“ 控制信息 ”用,也可以作为( )使用。

- A .控制流
- B 数据结构
- C .控制结构
- D 数据

答:D

6. 结构化设计方法(SD)与结构化分析方法(SA)一样,遵循( )模型,采用逐步求精技术,SD 方法通常与 SA 相联,即依据数据流图设计程序的结构。

- A .实体模型
- B 原型
- C .抽象思维
- D 生命期

答:C

7 .( )把已确定的软件需求转换成特定形式的设计表示,使其得以实现。

- A .系统设计
- B .详细设计
- C .逻辑设计
- D 软件设计

答:D

8 结构化设计的方法中使用的图形工具是 ( )

- A .软件结构图
- B 数据流程图
- C .程序流程图
- D 实体联系图

答:A

9 程序内部的各个部分之间存在的联系,用结构图表达时,最关心的是模块的( )和耦合性。

- A .一致性
- B .作用域
- C .嵌套限制
- D .内聚性

答:D

10 .程序内部的各个部分之间存在的联系,用结构图表达时,( )是在模块之间的联系。

- A .内聚性
- B .耦合性
- C 独立性
- D .有效性

答:B

11 .在多层次的结构图中,其模块的层次数称为结构图的 ( )

- A 深度
- B .跨度
- C 控制域
- D .粒度

答:A

12 .下列几种耦合中,( )的耦合性最强。

- A .公共耦合
- B .数据耦合
- C 控制耦合
- D .内容耦合

答:D

13 .一个模块把一个数值量作为参数传送给另一模块。这两个模块之间的耦合是 ( )

- A 逻辑耦合
- B .数据耦合
- C 控制耦合
- D .内容耦合

答:B

14 .一个模块直接引用另一模块中的数据,这两个模块之间的耦合是 ( )

- A .公共耦合
- B .数据耦合
- C 控制耦合
- D .内容耦合

答:D

15 .( )应该考虑对模块相联和资源共享问题进行描述和制约。

- A 系统设计
- B .详细设计
- C 接口控制
- D .结构化编辑工具

答:C

16 .一个模块把开关量作为参数传送给另一模块,这两个模块之间的耦合是 ( )

- A 外部耦合
- B .数据耦合
- C 控制耦合
- D .内容耦合

答:C

17. ( )复审应该把重点放在系统的总体结构、模块划分、内外接口等方面。
- A .详细设计
  - B .系统设计
  - C .正式
  - D .非正式

答:B

18. ( )是程序中的一个能逻辑地分开的部分,也就是离散的程序单位。
- A .模块
  - B .复合语句
  - C .循环结构
  - D .数据块

答:A

19. 模块( )定义为受该模块内一个判断影响的所有模块集合。
- A .控制域
  - B .作用域
  - C .宽度
  - D .接口

答:B

20. 在进行软件结构设计时应该遵循的最主要的原理是( )原理。
- A .抽象
  - B .模块化
  - C .模块独立
  - D .信息隐藏

答:C

21. ( )是数据说明,可执行语句等程序对象的集合,它是单独命名的而且可通过名字访问。
- A .模块化
  - B 抽象
  - C .精化
  - D .模块

答:D

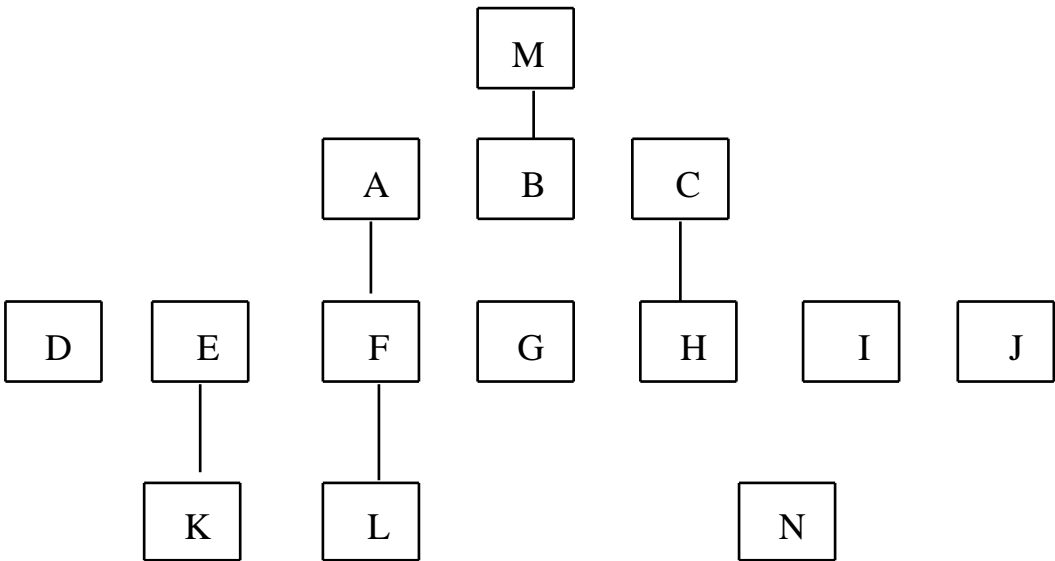
22. 软件结构使用的图形工具,一般采用( )图。
- A.DFD
  - B.PAD
  - C.SC
  - D.ER

答:C

23. 属于软件设计的基本原理是 ( )
- A. 数据流分析设计
  - B. 变换流分析设计
  - C. 事务流分析设计
  - D. 模块化

答:D

24. 对于下图中,下列说法正确的是 ( )
- A. 模块 F 从属于 A,也从属于 C
  - B. 模块 F 从属于 L,也从属于 K
  - C. 模块 N 从属于 A、B、C
  - D. 模块 A 从属于 B,也从属于 C



25. 对于上图中,该结构图的深度为 ( )
- A. 3
  - B. 4
  - C. 5
  - D. 6

答:B

26. 对于上图中,该结构图的宽度为 ( )

- A. 1                      B. 3                      C. 4                      D. 7

答:D

27. 在上图中,模块 A 统率模块 ( )

- A. M、B、C                      B. D、E、F  
C. F、L、K                      D. D、E、F、K、L

答:B

28. 对于上图中,下列说法正确的是 ( )

- A. 模块 A 的扇出为 5                      B. 模块 A 的扇出为 1  
C. 模块 C 的扇出为 4                      D. 模块 C 的扇出为 5

答:C

29. 对于上图中,下列说法正确的是 ( )

- A. 模块 K 的扇入为 4                      B. 模块 K 的扇入为 6  
C. 模块 K 的扇入为 5                      D. 模块 K 的扇入为 3

答:D

30. 在软件结构设计后处理中,下列说法错误的是 ( )

- A. 为模块写的处理说明及接口说明可采用 IPO 图  
B. 数据结构的描述可用 Warnier 图或 Jackson 图  
C. 给出设计约束或限制。如数据的边界值,数据类型、格式、内存容量及时间的限制  
D. 在概要设计评审中,应着重评审软件需求是否得到满足及软件结构的质量等  
E. 设计的优化工作主要放在软件结构设计后处理阶段

答:E

31. 通过抽象,可以 ( )

- A. 确定组成软件的过程实体  
B. 定义和实施对模块的过程细节存取限制  
C. 定义和实施对局部数据结构的存取限制

答:A

32. 标记耦合指 ( )

- A. 两个模块之间没有直接的关系,它们之间不传递任何信息  
B. 两个模块之间有调用关系,传递的是简单的数据值  
C. 两个模块之间传递是数据结构  
D. 一个模块调用另一个模块时,传递的是控制变量

答:C

33. 通信内聚指 ( )

- A. 把需要同时执行的动作组合在一起形成的模块为时间内聚模块  
B. 指各处理使用相同的输入数据或者产生相同的输出数据  
C. 指一个模块中各个处理元素都密切相关于同一功能且必须顺序执行  
D. 这是最强的内聚,指模块内所有元素共同完成一个功能,缺一不可

答:B

34. 内容耦合指 ( )

- A. 两个模块之间传递是数据结构

- B. 一个模块调用另一个模块时,传递的是控制变量
- C. 通过一个公共数据环境相互作用的那些模块间的耦合
- D. 一个模块直接使用另一个模块的内部数据,或通过非正常入口而转入另一个模块内部

答:D

35. 下列说法错误的是 ( )

- A. 变换型的 DFD 是由输入、变换(或称处理)和输出三部分组成
- B. 变换型数据处理的工作过程一般分为三步:处理数据、变换数据和输出数据
- C. 变换输入端的数据流为系统的逻辑输入,它将外部形式的数据变成内部形式,送给主加工
- D. 变换输出端为逻辑输出,它把主加工产生的数据的内部形式转换成外部形式后物理输出。
- E. 在变换型的 DFD 中,直接从外部设备输入数据称为物理输入,反之称为物理输出。

答:B

36. 在基于 IDEF<sub>0</sub> 图的设计方法中,下列说法错误的是 ( )

- A. IDEF<sub>0</sub> 图以系统的功能模型和信息结构为基础设计系统的软件结构
- B. IDEF<sub>0</sub> 图按照自顶向下逐层对系统进行分解
- C. IDEF<sub>0</sub> 图对系统每一功能的输入、输出、约束、机制都进行了全面的描述
- D. 在系统概要设计时,一般按照 DFD 图的分解层次,逐层将其转换成软件结构图

答:D

37. 在分层数据流图映射成软件结构的设计中,下列说法错误的是 ( )

- A. 分层的数据流图映射成软件结构图也应该是分层的
- B. 软件结构图的物理输入与输出部分放在主图中较为合适
- C. 分层 DFD 的映射方法:主图是变换型,子图是事务型;或主图是事务型,子图是变换型
- D. 变换型通常用于高层数据流图的转换,而事务型通常用于较低层数据流图的转

答:D

38. 在软件结构设计完成后,对于下列说法,正确的是 ( )

- A. 非单一功能模块的扇入数大比较好,说明本模块重用率高
- B. 单一功能的模块扇入高时应重新分解,以消除控制耦合的情况
- C. 一个模块的扇出太多,说明该模块过分复杂,缺少中间层
- D. 一个模块的扇入太多,说明该模块过分复杂,缺少中间层

答:C

39. 偶然内聚指 ( )

- A. 一个模块内的各处理元素之间没有任何联系
- B. 指模块内执行几个逻辑上相似的功能,通过参数确定该模块完成哪一个功能
- C. 把需要同时执行的动作组合在一起形成的模块为时间内聚模块
- D. 指模块内所有处理元素都在同一个数据结构上操作

答:A

40. 下列说法完全正确的是 ( )

- A. HIPO 图可以描述软件总的模块层次结构——IPO 图
- B. HIPO 图可以描述每个模块输入/输出数据、处理功能及模块调用的详细情况——H 图
- C. HIPO 图以模块分解的层次性以及模块内部输入、处理、输出三大基本部分为基础建立的
- D. H 图说明了模块间的信息传递及模块内部的处理

答:C

41. 软件概要设计结束后得到 ( )  
A. 初始化的软件结构图 B. 优化的软件结构图  
C. 模块详细的算法 D. 程序编码  
答:B
42. 划分模块时,一个模块的 ( )  
A. 作用范围应在其控制范围之内  
B. 控制范围应在其作用范围之内  
C. 作用范围与控制范围互不包含  
D. 作用范围与控制范围不受任何限制  
答:A
43. 结构化设计方法在软件开发中,用于 ( )  
A. 测试用例设计 B. 概要设计  
C. 程序设计 D. 详细设计  
答:B
44. 软件结构使用的图形工具,一般采用( )图。  
A.DFD B.PAD C.SC D.ER  
答:C
45. 为了提高模块的独立性,模块内部最好是 ( )  
A. 逻辑内聚 B. 时间内聚 C. 功能内聚 D. 通信内聚  
答:C
46. 在软件概要设计中,不使用的图形工具是( )图。  
A.SC B.IPO C.IDEF D.PAD  
答:D
47. 概要设计与详细设计衔接的图形工具是 ( )  
A. 数据流图 B. 结构图 C. 程序流程图 D. PAD 图  
答:B
48. 软件结构图中,模块框之间若有直线连接,表示它们之间存在着( )关系。  
A. 调用 B. 组成 C. 链接 D. 顺序执行  
答:A
49. 结构化设计是一种面向( )设计方法。  
A. 数据流 B. 数据结构 C. 数据库 D. 程序  
答:A
50. 为了提高模块的独立性,模块之间最好是 ( )  
A. 公共耦合 B. 控制耦合 C. 内容耦合 D. 数据耦合  
答:D
51. 设计软件结构一般不确定 ( )  
A. 模块的功能 B. 模块的接口  
C. 模块内的局部数据 D. 模块间的调用关系  
答:C
52. 软件概要设计结束后得到 ( )  
A. 初始化的软件结构图 B. 优化后的软件结构图  
C. 模块详细的算法 D. 程序编码

答:B

53. 结构化设计方法是一种面向( )的设计方法。

- A. 数据流                      B. 数据结构                      C. 数据库                      D. 程序

答:A

54. 为了提高模块的独立性,模块内部最好是 ( )

- A. 公共耦合                      B. 控制耦合                      C. 内容耦合                      D. 数据耦合

答:D

55. 变换流的 DFD 由三部分组成,不属于其中一部分的是 ( )

- A. 事务中心                      B. 变换中心                      C. 输入流                      D. 输出流

56. 软件结构图中,模块框之间若有直线连接,表示它们之间存在着( )关系。

- A. 调用                      B. 组成                      C. 链接                      D. 顺序执行

答:A

57. 软件设计阶段一般又可分为 ( )

- A. 逻辑设计与功能设计                      B. 概要设计与详细设计  
C. 概念设计与物理设计                      D. 模型设计与程序设计

答:B

58. 结构图中,不是其主要成分的是 ( )

- A. 模块                      B. 模块间传递的数据  
C. 模块内部数据                      D. 模块的控制关系

答:C

59. 好的软件结构应该是 ( )

- A. 高耦合、高内聚                      B. 低耦合、高内聚  
C. 高耦合、低内聚                      D. 低耦合、低内聚

答:B

60. 结构化设计方法在软件开发中,用于 ( )

- A. 测试用例设计                      B. 软件概要设计  
C. 程序设计                      D. 软件详细设计

答:B

61. 划分模块时,一个模块的 ( )

- A. 作用范围应在其控制范围之内                      B. 控制范围应在其作用范围之内  
C. 作用范围与控制范围互不包含                      D. 作用范围与控制范围不受任何限制

答:A

## 四、简答题

1 衡量模块独立的两个标准是什么?它们各表示什么含义?

答:衡量模块的独立性的标准是两个定性的度量标准:耦合性和内聚性。

(1)耦合性。也称块间联系。指软件系统结构中各模块间相互联系紧密程度的一种度量。模块之间联系越紧密,其耦合性就越强,模块的独立性则越差。模块间耦合高低取决于模块间接口的复杂性、调用的方式及传递的信息。

(2)内聚性。又称块内联系。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。若一个模块内各元素(语句之间、程序段之间)联系得越紧密,则它的内聚性就高。

耦合性与内聚性是模块独立性的两个定性标准,将软件系统划分模块时,尽量做到高内聚低耦合,提高模块的独立性,为设计高质量的软件结构奠定基础。

## 2 影响公共耦合的复杂程度的因素是什么?公共耦合会引起什么问题?

答:公共耦合的复杂程度随耦合模块的个数增加而增加。如果只有两个模块之间有公共数据环境,这种公共耦合有两种情况:

(1)一个模块只是给公共数据环境送数据,另一个模块只是从公共环境中取数据,这只是数据耦合的一种形式,是比较松散的公共耦合。

(2)两个模块都既往公共数据环境中送数据,又从里面取数据,这是紧密的数据耦合。

如果在模块之间共享的数据很多,且通过参数的传递很不方便时,才使用公共耦合。因为公共耦合会引起以下问题。

- 耦合的复杂程度随模块的个数增加而增加,无法控制各个模块对公共数据的存取,若某个模块有错,可通过公共区将错误延伸到其他模块,影响到软件的可靠性。

- 使软件的可维护性变差。若某一模块修改了公共区的数据,会影响到与此有关的所有模块。

- 降低了软件的可理解性。因为各个模块使用公共区的数据,使用方式往往是隐含的,某些数据被哪些模块共享,不易很快搞清。

## 3 模块的内聚性由哪几种?各表示什么含义?

答:内聚性有六种类型:偶然内聚、逻辑内聚、时间内聚、通信内聚、顺序内聚、功能内聚,它们的内聚性由低到高。

(1)偶然内聚。指一个模块内的各处理元素之间没有任何联系。这是最差的内聚情况。

(2)逻辑内聚。指模块内执行几个逻辑上相似的功能,通过参数确定该模块完成哪一个功能。如产生各种类型错误的信息输出放在一个模块,或从不同设备上的输入放在一个模块,这是一个单入口多功能模块。这种模块内聚程度有所提高,各部分之间在功能上有相互关系,但不易修改,当某个调用模块要修改此模块公用代码时,而另一些调用模块又不要求修改。另外,调用时需要进行控制参数的传递,造成模块间的控制耦合,调用此模块时,不用的部分也占据了主存,降低了系统效率。

(3)时间内聚。把需要同时执行的动作组合在一起形成的模块为时间内聚模块。如初始化一组变量,同时打开若干文件,同时关闭文件等等,都与特定时间有关。时间内聚比逻辑内聚程度高一些,因为时间内聚模块中的各部分都要在同一时间内完成。但是由于这样的模块往往与其他模块联系得比较紧密,如初始模块对许多模块的运行有影响,因此和其他模块耦合的程度较高。

(4)通信内聚。指模块内所有处理元素都在同一个数据结构上操作(有时称之为信息内聚),或者指各处理使用相同的输入数据或者产生相同的输出数据。

通信内聚的模块各部分都紧密相关于同一数据(或者数据结构),所以内聚性要高于前几种类型。同时,可把某一数据结构、文件、设备等操作都放在一个模块内,可达到信息隐藏。

(5)顺序内聚。指一个模块中各个处理元素都密切相关于同一功能且必须顺序执行,前一功能元素的输出就是下一功能元素的输入。

(6)功能内聚。这是最强的内聚,指模块内所有元素共同完成一个功能,缺一不可。因此模块不能再分割。功能内聚的模块易理解、易修改,因为它的功能是明确的、单一的,因此与其他模块的耦合是弱的。功能内聚的模块有利于实现软件的重用,从而提高软件开发的效率。

## 4 什么是软件结构?结构图的主要内容是什么?

答:软件结构是软件系统的模块层次结构,反映了整个系统的功能实现,即将来程序的控制层次体系。对于一个“问题”,可用不同的软件结构来解决,不同的设计方法和不同的划分和组织,得出不同的软件结构。

软结构往往用树状或网状结构的图形来表示。软件工程中,一般采用结构图(Structure Chart,简称



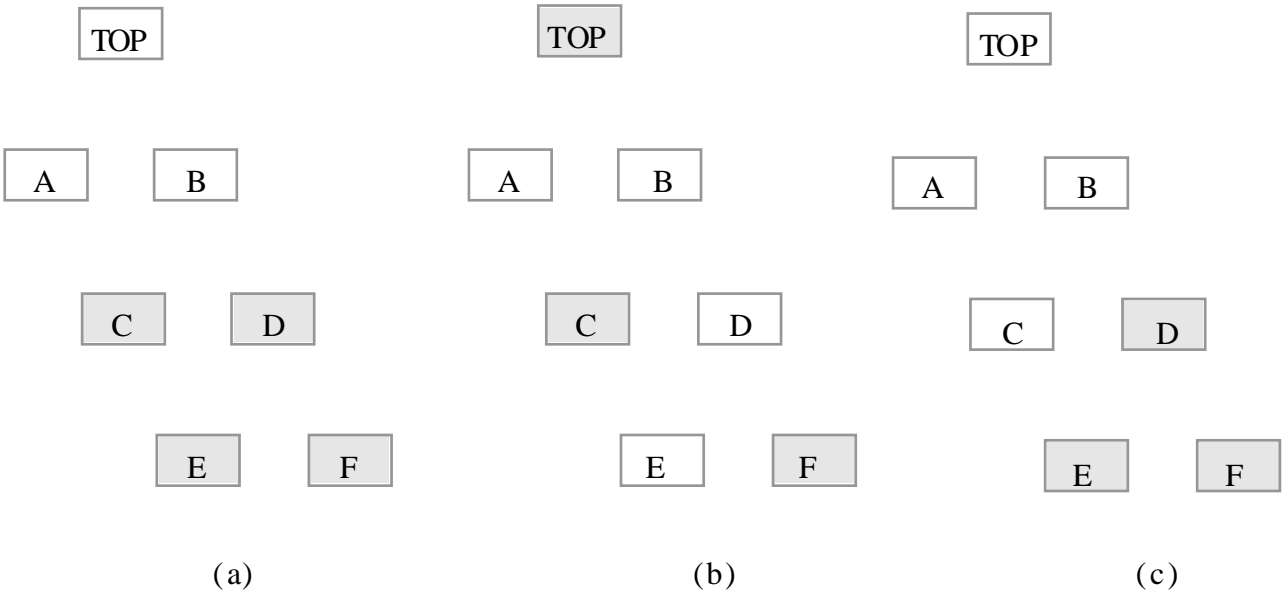
SC)的工具来表示软件结构。结构图的主要内容有:模块、模块的控制关系、模块间的信息传递和表示模块有选择地调用或循环调用的两个附加符号。

5 什么是模块的作用范围？什么是模块的控制范围？它们之间应该建立什么关系？

答:一个模块的作用范围指受该模块内一个判定影响的所有模块的集合。一个模块的控制范围指模块本身以及其所有下属模块(直接或间接从属于它的模块)的集合。

一个模块的作用范围应在其控制范围之内,且判定所在的模块应与受其影响的模块在层次上尽量靠近。

如图所示(符号 表示模块内有判定功能,阴影表示模块的作用范围),在下图(a)中,模块 D的作用范围是 C、D、E 和 F,控制范围是 D、E、F,作用范围不在控制范围以内,这种结构最差。在下图(b),模块 TOP 的作用范围在控制范围之内,但是判定所在模块与受判定影响的模块位置太远。最理想的结构图是下图(c),模块 D的作用范围在其控制范围之内,且判定所在的模块与受其影响的模块在层次上靠近,消除了额外的数据传递。

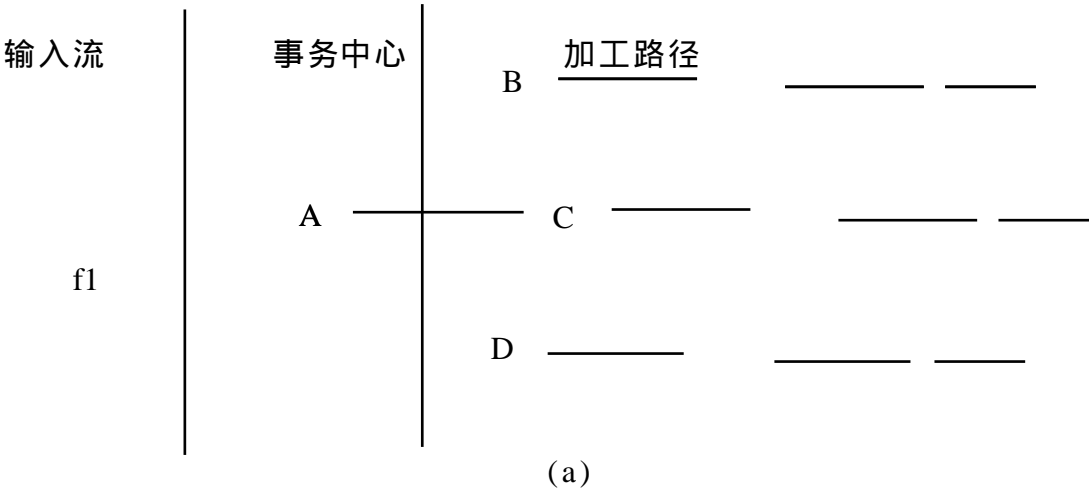


模块的作用范围与控制范围

6 试述“ 事务分析 ”的设计步骤。试将事务型 DFD 数流图转换成软件结构图。

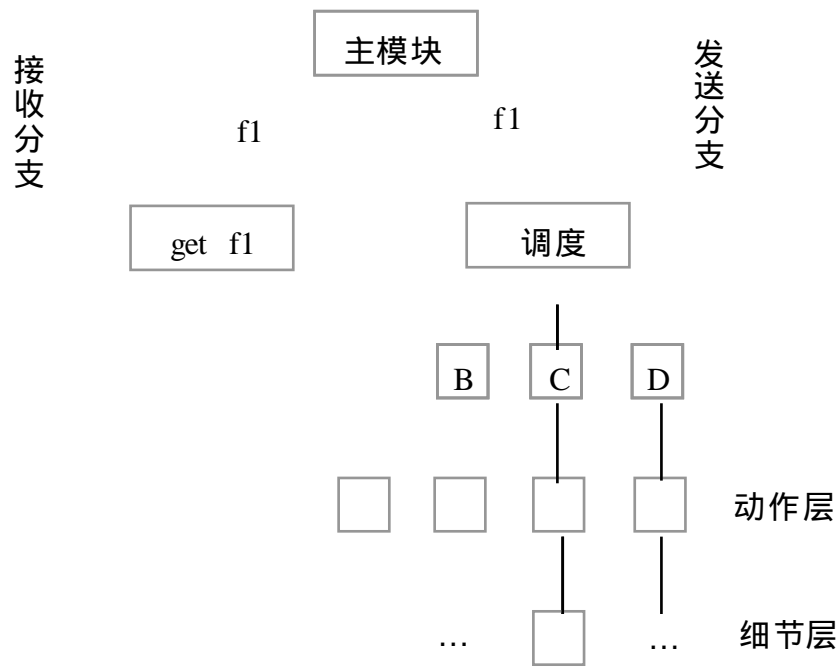
答:事务分析设计步骤:

对于具有事务型特征的 DFD,则采用事务分析的设计方法,结合下图(a)、(b),说明该方法的设计步骤:



事务分析设计举例

- (1)确定 DFD 中的事务中心和加工路径,当 DFD 中的某个加工具有明显地将一个输入数据流分解成多个发散的输出数据流时,该加工就是事务中心。从事务中心辐射出去的数据流为各个加工路径。
- (2)设计软件结构的顶层和第一层—事务结构,首先设计一个顶层模块,它是一个全控模块,有两个功能:一是接收数据,二是根据事务类型调度相应的处理模块,因此,事务型软件结构应包括两个部分:一个接收分支和一个发送分支。
- 接收分支:负责接收数据,它的设计与变换型 DFD 的输入部分设计方法相同。
- 发送分支:能常包含一个调度模块,它控制管理所有的下层的事务处理模块。当事务类型不多时,调度模块可与主模块合并。
- (3)事务结构中、下层模块的设计、优化等工作同变换结构。



(b)  
事务分析设计举例

- 7 叙述由 IDEF<sub>0</sub> 图导出初始软件结构图的方法。
- 答:对于某一层的 IDEF<sub>0</sub> 图按以下方法转换:
- (1)找出该层 IDEF<sub>0</sub> 图的父图,搞清父、子图之间的输入、输出、控制关系。
  - (2)以父图的活动为上层模块,子图中的活动为下层模块,画出系统的单层结构图。
  - (3)根据 IDEF<sub>0</sub> 图各个活动的输入、输出数据,控制信息及数据库的结构,数据项定义等,确定模块的接口。
  - (4)综合所有层次的结构图,得到系统初始的软件结构图。
  - (5)根据软件结构的优化准则进行精化。

在由 IDEF<sub>0</sub> 图导出初始软件结构图的过程中,往往将一个活动方框对应于一个处理模块,应反复地理解全部 IDEF<sub>0</sub> 图的内容和含义,对最初形成的模块结构进行必要的调整、修改、分解或合并,最终的软件结构与基于 DFD 图设计的软件结构不会有太大的差别。

8. 为了降低模块间的耦合度,可采取哪些措施?
- 答:(1)在耦合方式上降低模块间接口的复杂性。模块间接口的复杂性包括模块的接口方式、接口信息的结构和数量。接口方式不采用直接引用(内容耦合),而采用调用方式(如过程语句调用方式)。接口信息通过参数传递且传递信息的结构尽量简单,不用复杂参数结构(如过程、指针等类型参数),参数的个数也不宜太多,如果很多,可考虑模块的功能是否庞大复杂。

(2)在传递信息类型上尽量使用数据耦合,避免控制耦合,慎用或有控制地使用公共耦合。这只是原则,耦合类型的选择要根据实际情况综合地考虑。

#### 9. 什么是面向数据流的设计方法?有哪些策略?

答:该方法也称结构化设计方法(SD),它与结构化分析(SA)相衔接,它按一定的设计策略将数据流图转换成软件的模块层次结构。有两种设计策略:

(1)事务型分析设计:一个大的复杂的系统分解成较小的,相对简单的子系统,这些子系统彼此之间相对独立一些,而高层数据流图的数据处理往往反映这些子系统的功能,有平行分别处理的特点,因此,高层数据流图的转换通常可作为事务型处理,把一个加工逻辑看成是一类特定的事务,把它们分别映射成一个模块,最高层模块为系统模块,通过对输入初始命令的判断决定调用哪个模块。这种事务型分析设计的策略也用于较低层数据流图向软件结构图的转换。

(2)变换型分析设计:变换型数据流图具有主要的处理功能及实现这项处理功能所需要的输入数据流和经过处理后产生的输出数据流。确定了第三部分,高层模块就可分解出三个从属于它的新模块,分别执行输入、变换、输出功能。变换分析设计一般用于对较低层数据流图向软件结构图的转换。

软件结构图设计好后,还需要为每个模块提供必要的说明,如功能说明、接口说明等,IPO图是常采用的图形工具,软件结构图和各个模块的IPO图结合在一起才能较完整地描绘软件系统在总体上对需求功能的实现。

#### 10. 如何设计软件系统结构(简称软件结构)?

答:为了实现目标系统,最终必须设计出组成这个系统的所有程序和数据库(文件),对于程序,则首先进行结构设计,具体为:

- (1)采用某种设计方法,将一个复杂的系统按功能划分成模块。
- (2)确定每个模块的功能。
- (3)确定模块之间的调用关系。
- (4)确定模块之间的接口,即模块之间传递的信息。
- (5)评价模块结构的质量。

#### 11. 概要设计文档主要有哪些?

答:文档主要有:

- (1)概要设计说明书。
- (2)数据库设计说明书,主要给出所使用的DBMS简介、数据库的概念模型、逻辑设计、结果。
- (3)用户手册,对需求分析阶段编写的用户手册进行补充。
- (4)修订测试计划,对测试策略、方法、步骤提出明确要求。

#### 12. 试述软件设计的基本原理。

答:要设计好的软件结构,必须要有一定的标准准则,根据几十年来的经验,总结出了以下设计原理。

(1)模块与模块化。解决一个复杂的问题,最有效的方法是把问题分解成几个部分,然后再分别考虑。软件系统中,用什么来实现这些分解的“部分”呢?就是用模块,模块是可以组合、可以分解或者更换的元素。模块有功能,有接口,有运行环境,这是它的外部特征,这些特征外部模块可以“看到”。模块也有外部模块不需要知道的内部特性,即内部的数据及处理过程。把复杂的问题划分成模块的过程称为模块化。

(2)抽象。抽象是划分模块过程中的思维原则,设计人员不可能一下就把系统划分成很详细的模块,要自顶向下,逐步求精。开始分解的时候考虑主要的方面。如:主要的功能、主要的数据、主要的行为,这就是抽象,然后再逐步加细。由抽象到具体设计的结果,产生的软件结构必然是层次结构,这种结构具有可理解性。

(3)信息隐蔽。应用“抽象”确定软件模块,应用信息隐蔽定义对模块的过程细节及局部数据的存取限制来提高模块的独立性。信息隐蔽指模块将内部处理细节和内部数据用某种手段“隐蔽”起来,即外部模块不能随便访问它们,模块之间仅仅交换为完成系统功能所必需的信息。信息隐蔽为软件系统的修改、测试及以后的维护带来好处。

(4)模块的独立性。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,与其他模块联系最少,则模块的独立性就强。衡量模块独立性有两个定性标准:

耦合性:指软件结构中模块之间相互的依赖程度。耦合性越高的模块,其独立性越差。影响模块间耦合程度最主要的因素是模块间信息传递的复杂性,教材中列举了6种耦合类型,读者应结合程序设计经验理解它们。

内聚性:内聚性是度量一个模块功能强度的一个相对指标,表现在模块内部各组成部分执行功能组合在一起的相关程度。各成分(数据、语句、段等)之间都紧密相关于同一功能,模块的内聚性就高,显然独立性则强。教材中列举了6种内聚类型,内聚性高的模块(如功能内聚)有利于实现软件重用。

### 13. 数据库的设计主要进行哪些方面设计?

答:(1)概念设计。在数据分析的基础上,采用自底向上的方法从用户角度进行视图设计,一是用ER模型来表示数据模型,这是一个概念模型。ER模型既是设计数据库的基础,也是设计数据结构的基础。IDEF<sub>1x</sub>技术也支持概念模式,用IDEF<sub>1x</sub>方法建立系统的信息模型,使其模型具有一致性、可扩展性和可变性等特性,同样可作为数据库设计的主要依据。

(2)逻辑设计。ER模型或IDEF<sub>1x</sub>模型是独立于数据库管理系统(DBMS)的,要结合具体的DBMS特征来建立数据库的逻辑结构,对于关系型的DBMS来说将概念结构转换为数据模式、子模式并进行规范,要给出数据结构的定义,即定义所含的数据项、类型、长度及它们之间的层次或相互关系的表格等等。

(3)物理设计。对于不同的DBMS,物理环境不同,提供的存储结构与存取方法各不相同。物理设计就是设计数据模式的一些物理细节,如数据项存储要求、存取方式、索引的建立等等。

### 14. 软件结构设计优化准则是什么?

答:软件概要设计的主要任务就是软件结构的设计,为了提高设计的质量,必须根据软件设计的原理改进软件设计。提出以下软件结构的设计优化准则:

(1)划分模块时,尽量做到高内聚,低耦合,保持模块相对独立性,并以此原则优化初始的软件结构。

如果若干模块之间耦合强度过高,每个模块内功能不复杂,可将它们合并,以减少信息的传递和公共区的引用。

若有多个相关模块,应对它们的功能进行分析,消去重复功能。

(2)一个模块的作用范围应在其控制范围之内,且判定所在的模块应与受其影响的模块在层次上尽量靠近。

(3)软件结构的深度、宽度、扇入、扇出应适当。

深度是软件结构设计完成后观察到的情况,能粗略地反映系统的规模和复杂程度,宽度也能反映系统的复杂情况。宽度与模块的扇出有关,一个模块的扇出太多,说明本模块过分复杂,缺少中间层。单一功能模块的扇入数大比较好,说明本模块为上层几个模块共享的公用模块,重用率高。但是不能把彼此无关的功能凑在一起形成一个通用的超级模块,虽然它扇入高,但低内聚。因此非单一功能的模块扇入高时应重新分解,以消除控制耦合的情况。软件结构从形态上,总的考虑是顶层扇出数较高一些,中层扇出数较低一些,底层扇入数较高一些。

(4)模块的大小要适中。

在考虑模块的独立性同时,为了增加可理解性,模块的大小最好在50~150条语句左右。

(5)模块的接口要简单、清晰、含义明确,便于实现、测试与维护。

15. 概要设计说明书包括哪些内容？

答: (1) 引言

目的

项目背景

参考资料

术语

(2) 总体设计

软件描述

运行环境

软件结构

外部接口

(3) 模块设计

功能

内部接口

(4) 数据结构设计

逻辑结构设计

物理结构设计

(5) 运行设计

(6) 系统出错处理设计

故障信息

补救措施

系统维护设计

(7) 安全保密设计

16. 画结构图应注意的事项是什么？

答: (1) 同一名字的模块在结构图中仅出现一次。

(2) 调用关系只能从上到下。

(3) 不严格表示模块的调用次序, 习惯上从左到右。有时为了减少连线的交叉, 适当地调整同一层模块左右位置, 以保持结构图的清晰性。

17. 面向数据流设计方法的过程是什么？

答: (1) 精化 DFD。指把 DFD 转换成软件结构图前, 设计人员要仔细地研究分析 DFD 并参照数据字典, 认真理解其中的有关元素, 检查有无遗漏或不合理之处, 进行必要的修改。

(2) 确定 DFD 类型, 如果是变换型, 确定变换中心和逻辑输入、逻辑输出的界线, 映射为变换结构的顶层和第一层; 如果是事务型, 确定事务中心和加工路径, 映射为事务结构的顶层和第一层。

(3) 分解上层模块, 设计中下层模块结构。

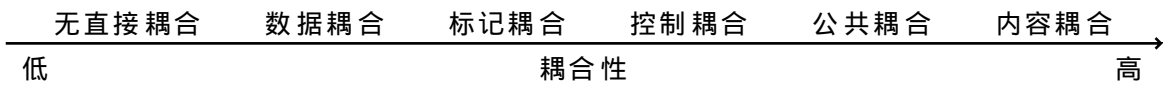
(4) 根据优化准则对软件结构求精。

(5) 描述模块功能、接口及全局数据结构。

(6) 复查, 如果有错, 转向(2)修改完善, 否则进入详细设计。

18. 什么是耦合性? 模块的耦合性有哪些内容?

答: 也称块间联系。指软件系统结构中各模块间相互联系紧密程度的一种度量。模块之间联系越紧密, 其耦合性就越强, 模块的独立性则越差。模块间耦合高低取决于模块间接口的复杂性、调用的方式及传递的信息。模块的耦合性有以下几种类型:



(1)无直接耦合(No Direct Coupling)。

指两个模块之间没有直接的关系,它们分别从属于不同模块的控制与调用,它们之间不传递任何信息。因此模块间耦合性最弱,模块独立性最高。

(2)数据耦合(Data Coupling)。

指两个模块之间有调用关系,传递的是简单的数据值,相当于高级语言中的值传递。这种耦合程度较低,模块的独立性较高。

(3)标记耦合(Stamp Coupling)。

指两个模块之间传递的是数据结构,如高级语言中的数组名、记录名、文件名等这些名字即为标记,其实传递的是这个数据结构的地址。两个模块必须清楚这些数据结构,并按要求对其进行操作,这样降低了可理解性。可采用“信息隐蔽”的方法,把该数据结构以及在其上的操作全部集中在一个模块,就可消除这种耦合,但有时因为还有其他功能的缘故,标记耦合是不可避免的。

(4)控制耦合(Control Coupling)。

指一个模块调用另一个模块时,传递的是控制变量(如开关、标志等),被调模块通过该控制变量的值有选择地执行块内某一功能。因此被调模块内应具有多个功能,哪个功能起作用受其调用模块控制。

19. 结构图的形态特征是什么?

答: (1)深度:指结构图控制的层次,也是模块的层数,结构图的深度为 5。

(2)宽度:指一层中最大的模块个数,宽度为 8。

(3)扇出:指一个模块直接下属模块的个数,模块 M 的扇出为 3。

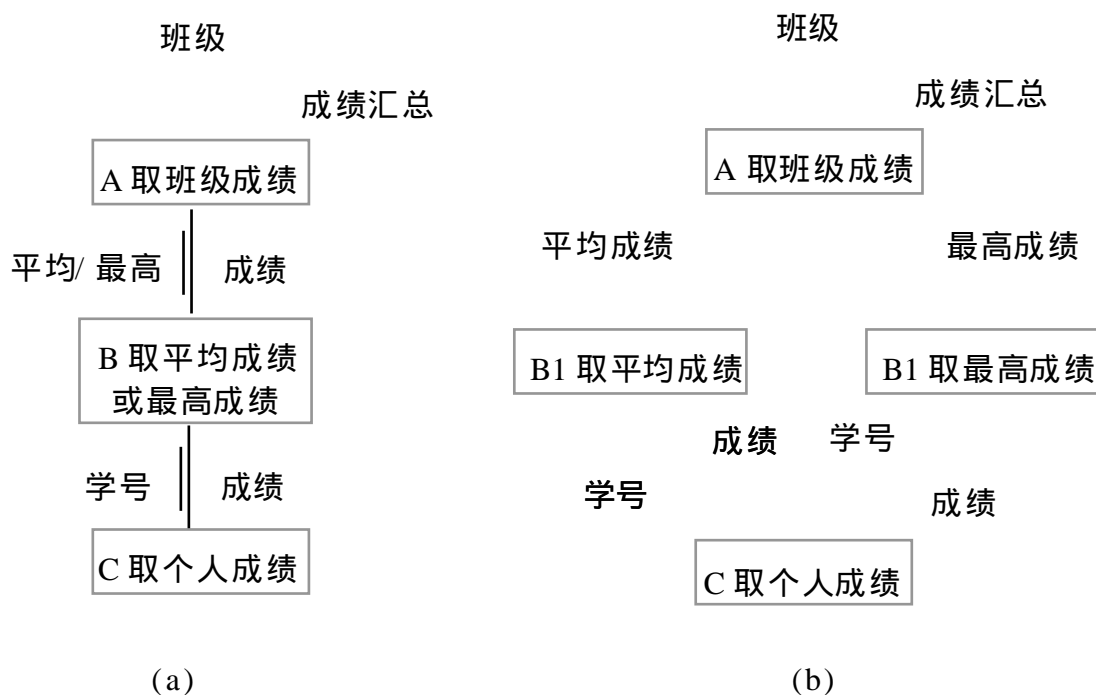
(4)扇入:指一个模块直接上属模块的个数,模块 T 的扇入为 4。

20. 什么是模块独立性?

答: 为了降低软件系统的复杂性,提高可理解性、可维护性,必须把系统划分成为多个模块,但模块不能任意划分,应尽量保持其独立性。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。模块独立性概念是模块化、抽象、信息隐蔽这些软件工程基本原理的直接产物。只有符合和遵守这些原则才能得到高度独立的模块。良好的模块独立性能使开发的软件具有较高的质量。因为模块独立性强,则信息隐藏性能好,并完成独立的功能,且它的可理解、可维护性、可测试性好,必然导致软件的可靠性。另外,接口简单、功能独立的模块易开发,且可并行工作,有效地提高软件的生产率。如何衡量软件的独立性呢?根据模块的外部特征和内部特征,提出了两个定性的度量标准耦合性和内聚性。

五、应用题

1. 下图是某系学籍管理的一部分,(a)、(b)分别是同一模块 A 的两个不同设计方案,你认为哪一个设计方案较好?请陈述理由。

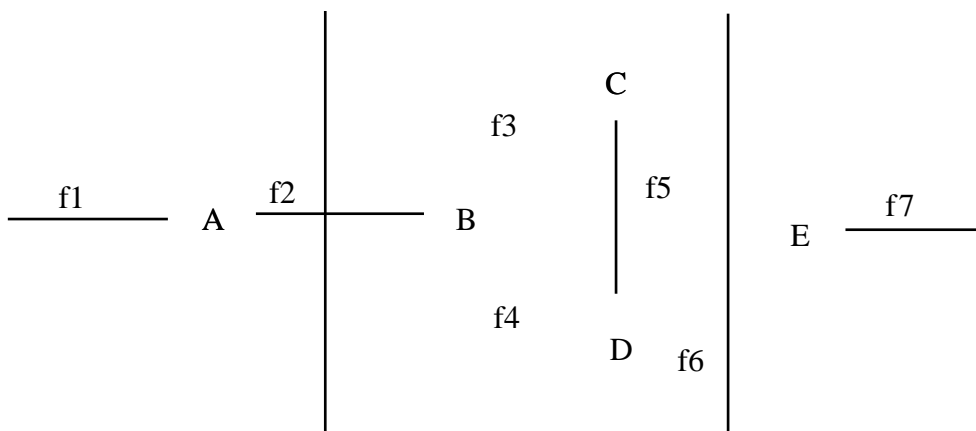


答: (b)图的设计方案好。

因为(a)图中模块 B 的功能是取平均成绩或最高成绩,包含两个功能,内聚弱,与模块 A 的耦合是控制耦合,耦合程度较高,模块的相对独立性差。而在(b)图中,模块 B 分解为两个功能相对独立的模块 B1 和 B2,模块 B1 和 B2 的内聚程度高,模块 B1 和 B2 与 A 之间的耦合是数据耦合,耦合程度较低,因此,模块的独立性好。所以(b)图的设计方案好。

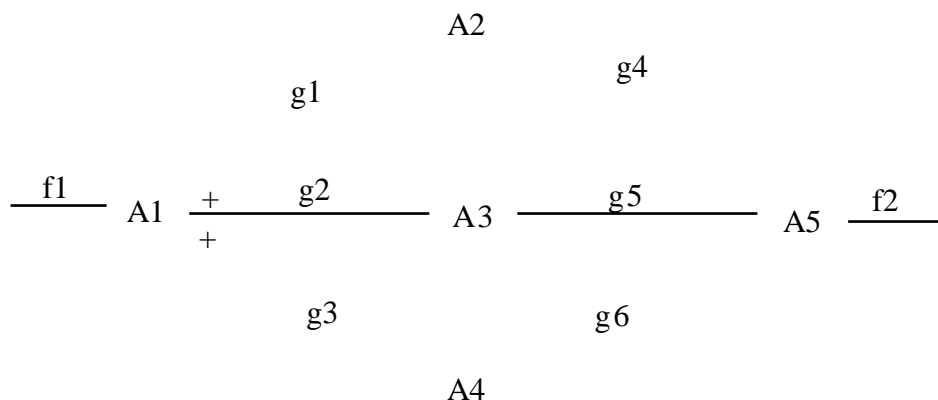
2. 请将下图的 DFD 转换为软件结构图(注:图中的 表示“或者”)

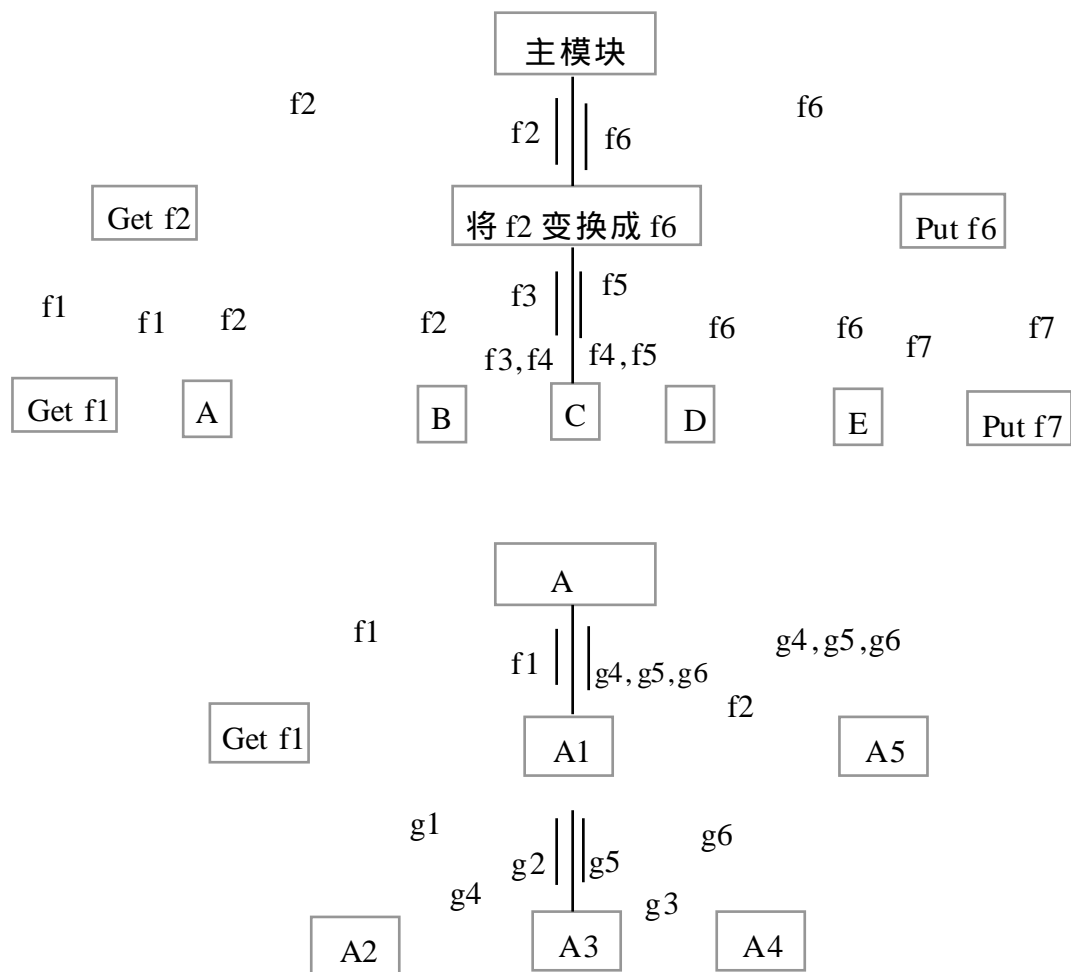
主图:



子图 A:

答:





3. 某厂对部分职工重新分配工作的政策是:年龄在 20 岁以下者,初中文化程度脱产学习,高中文化程度当电工;年龄在 20 岁至 40 岁之间者,中学文化程度男性当钳工,女性当车工,大学文化程度都当技术员。年龄在 40 岁以上者,中学文化程度当材料员,大学文化程度当技术员。请用结构化语言、判定表和判定树描述上述问题的加工逻辑。

答: (1)用结构化语言描述上述问题的加工逻辑:

```

IF 年龄 < 20
    THEN IF 文化程度 = 初中 THEN 脱产学习 ENDIF
    IF 文化程度 = 高中 THEN 电工 ENDIF
ENDIF
IF 20 < 年龄 < 40
    THEN IF 文化程度 = 中学
        THEN IF 性别 = 男
            THEN 钳工
        ELSE 车工
        ENDIF
    ENDIF
    IF 文化程度 = 大学 THEN 技术员 ENDIF
ENDIF
IF 年龄 > 40
    THEN IF 文化程度 = 中学 THEN 材料员 ENDIF
    IF 文化程度 = 大学 THEN 技术员 ENDIF
ENDIF

```



(2)用判定表描述上述问题的加工逻辑：

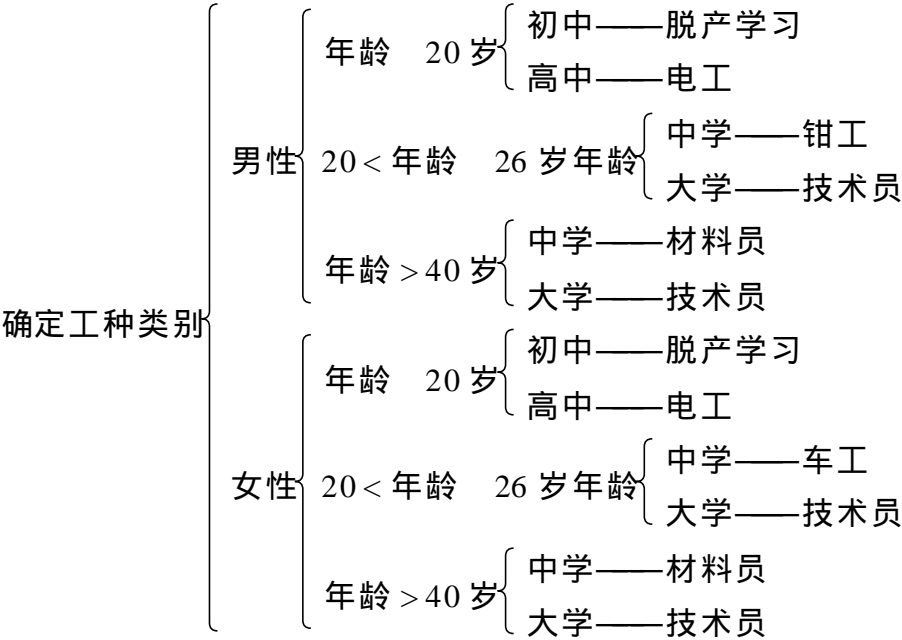
条件选取表

条件名	取值	符号	取值数 m
年龄	年龄 20 20 < 年龄 40 年龄 > 40	C Y L	m <sub>1</sub> = 3
性别	男 女	M F	m <sub>2</sub> = 2
文化程度	初中 高中 大学	Z G D	m <sub>3</sub> = 3

判定表(注释：“ ”表示选取的运作，“ ?”表示不能确定的动作)

	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	G	H	I
年龄	C	C	C	C	C	C	Y	Y	Y	Y	Y	Y	L	L	L	L	L	L
性别	F	F	F	M	M	M	F	F	F	M	M	M	F	F	F	M	M	M
文化程度	Z	G	D	Z	G	D	Z	G	D	Z	G	D	Z	G	D	Z	G	D
脱产学习																		
电工																		
钳工																		
车工																		
技术员			?			?												
材料员																		

(3)用判定树描述上述问题的加工逻辑：



4. 某数据流图中有一个“确定保险类别”的加工,指的是申请汽车驾驶保险时,要根据申请者的情况确定不同的保险类别。加工逻辑为:如果申请者的年龄在 21 岁以下,要额外收费;如果申请者是 21 岁以上并是 26 岁以下的女性,适用于 A 类保险;如果申请者是 26 岁以下的已婚男性,或者是 26 岁以上的男性,适用于 B 类保险;如果申请者是 21 岁以下的女性或是 26 岁以下的单身男性,适用于 C 类保险。除此之外在其它申请者都适用于 A 类保险。请用结构化语言、判定表和判定树描述上述问题的加工逻辑。

答: (1) 结构化语言

```
IF 性别 = 男性
    THEN IF 年龄 ≤ 21
        THEN IF 婚姻 = 未婚
            THEN C 类保险且额外收费
            ELSE B 类保险且额外收费
        ENDIF
    ENDIF
    IF 21 < 年龄 ≤ 26
        THEN IF 婚姻 = 未婚
            THEN C 类保险
            ELSE B 类保险
        ENDIF
    ENDIF
    IF 年龄 > 26
    ELSE IF 年龄 ≤ 21
        THEN C 类保险且额外收费
        ELSE A 类保险
    ENDIF
ENDIF
```

(2) 判定表

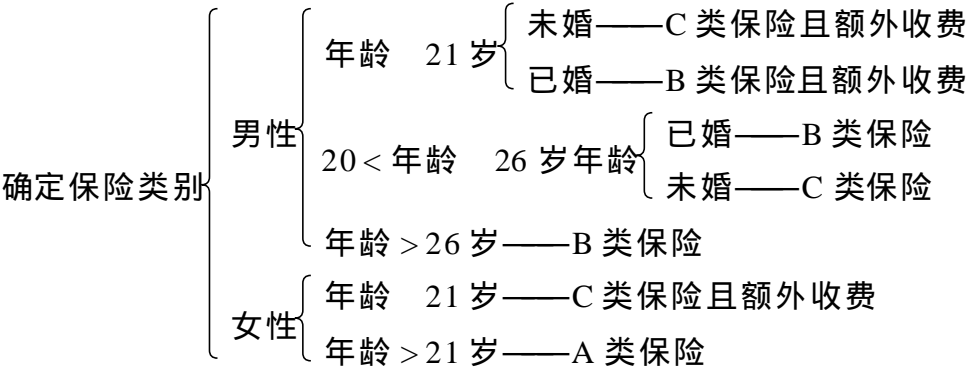
条件选取表

条件名	取值	符号	取值数 m
年龄	年龄 21	C	$m_1 = 3$
	21 < 年龄 26	Y	
	年龄 > 26	L	
性别	男	M	$m_2 = 2$
	女	F	
婚姻	未婚	S	$m_3 = 2$
	已婚	E	

判定表(注释:“ ”表示选取的运作)

	1	2	3	4	5	6	7	8	9	10	11	12
年龄	C	C	C	C	Y	Y	Y	Y	L	L	L	L
性别	F	F	M	M	F	F	M	M	F	F	M	M
婚姻	S	E	S	E	S	E	S	E	S	E	S	E
A 类保险												
B 类保险												
C 类保险												
额外收费												

(3)判定树



5. 房产管理系统旨在用计算机对房产进行管理,包括住房的分配、调整和计算房租等。用户可以查询住房情况和房租金额,还可以对房产进行一些统计,给出统计表格,以便掌握全面的住房情况。

房管部门首先把住户要求(按照统一的格式由住户填写)输入进来,系统检查要求的合法性,如不合法,系统拒绝接受。如是合法要求,根据要求类型处理。假定住户要求分三类:分房要求、调房要求、退房要求。三种类型的要求分别进行不同的处理。分房要求根据分房单,先核准住户够不够分房资格,这要根据住户的情况,从住房标准文件中读出住房标准,进行核准,如不够标准,则不予分房,如够标准,则输出核准后的分房单,然后再根据分房单进行住房分配。分配住房要从房产文件中读出相应的空房信息,如房号、面积、单位面积房租等,并登记相应的住房信息,如户言姓名、部门、住房分数、家庭人口等,再写回房产文件中去,同时写入住房文件中去,输出分配后的住房单。同时进行房租的计算,计算好的房租写入到房租文件中去。

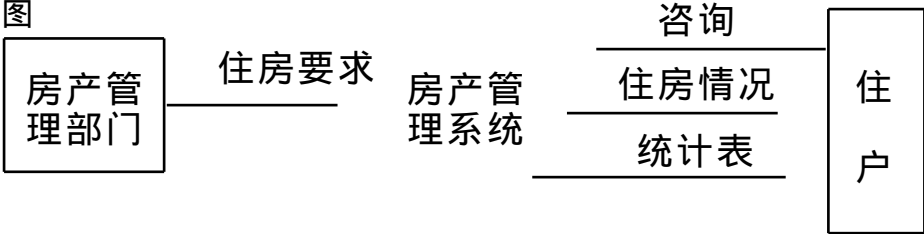
调房处理和退房处理与分房处理类似。

咨询要求分查询住户情况、查询房租和查询全局住房情况(统计)三种。查询住房情况可根据住户名从住房文件读出该住房的住房情况并打印出来。查询房租可根据住户名从房租文件读出该住户的房租信息并打印出来。统计要求做一些统计处理后打印出统计表。

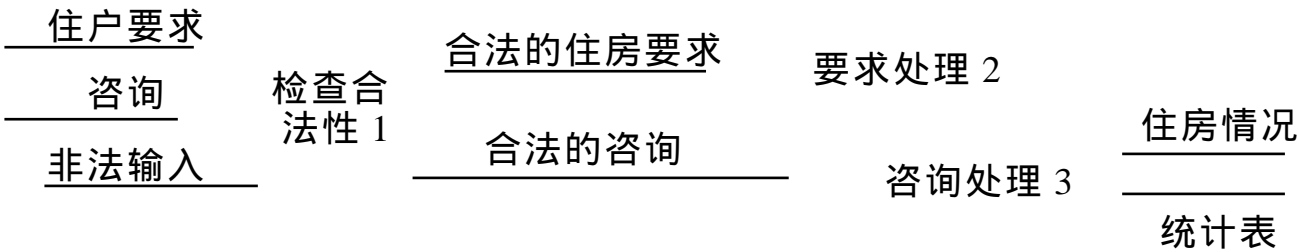
请采用 SA 方法画出该系统的分层 DFD, 并建立相应的数据字典。

答:

顶层图



0 层图



1 层图图 2

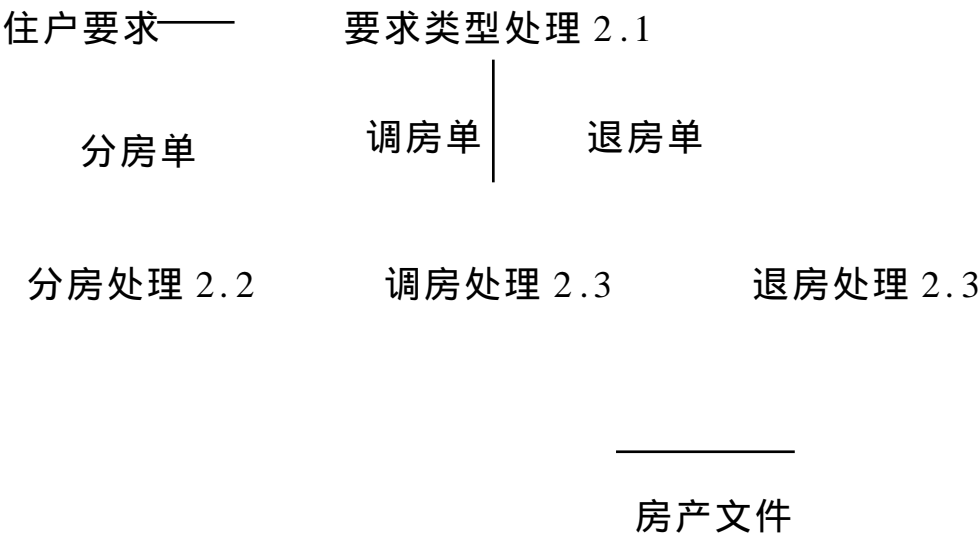
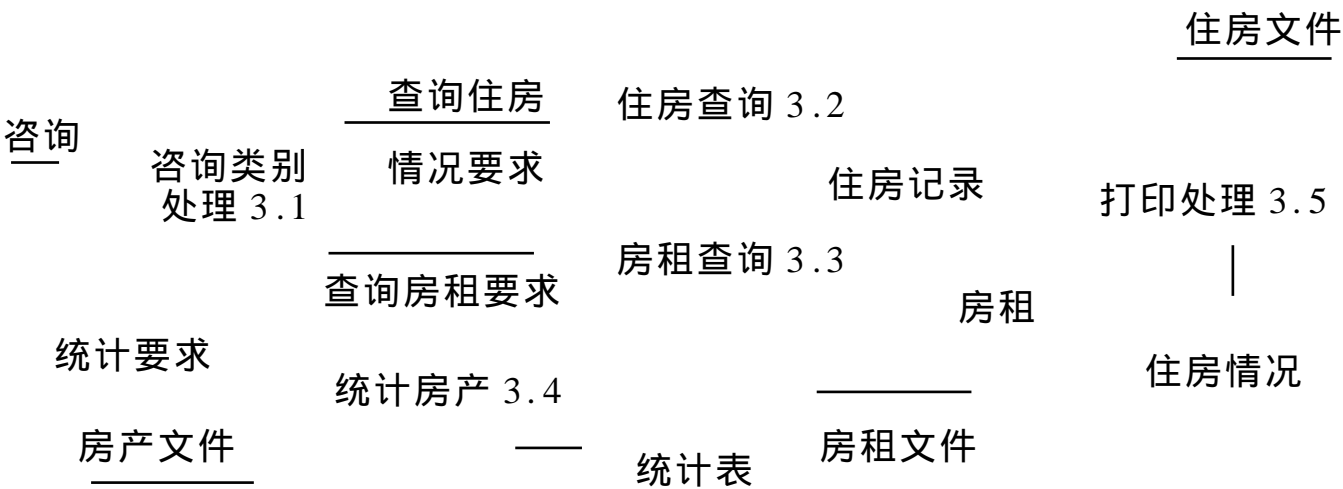


图 3



房产管理系统的分层数据流图

(1)数据流条目

住户要求 = 户主 + [分房要求|调房要求|退户要求]

分房要求 = 部门 + 职称 + 家庭人口 + 住房分数 + 要求住房面积

调房要求 = 部门 + 职称 + 家庭人口 + 住房分数 + 原住房面积 + 原房号 + 要求调房面积

退房要求 = 部门 + 房号

住房情况 = 户主 + 部门 + 职称 + 家庭人口 + 住房分数 + 住房面积 + 房租 + 房号

咨询要求 = 户主 + [住房情况咨询|户租咨询|统计要求]

统计表 = { 住户面积 + 已分住房数 + 空房数 }

分房单 = 户主 + 部门 + 职称 + 住房分数 + 要求住房面积

2 层图图 2 .2

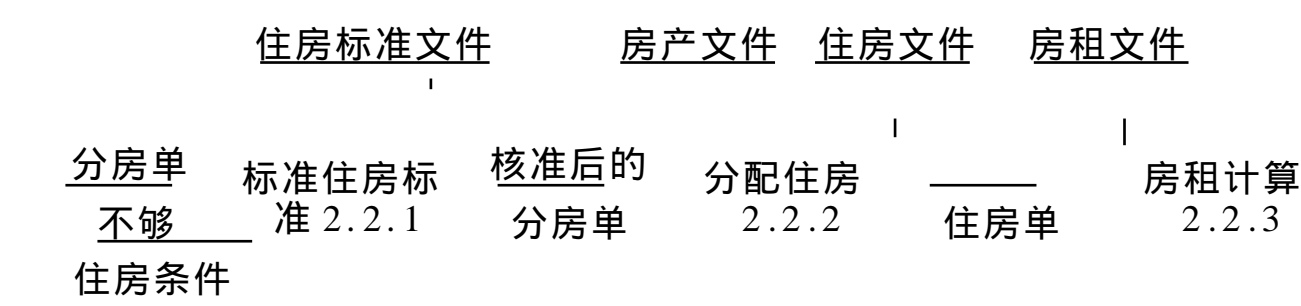


图 2 3

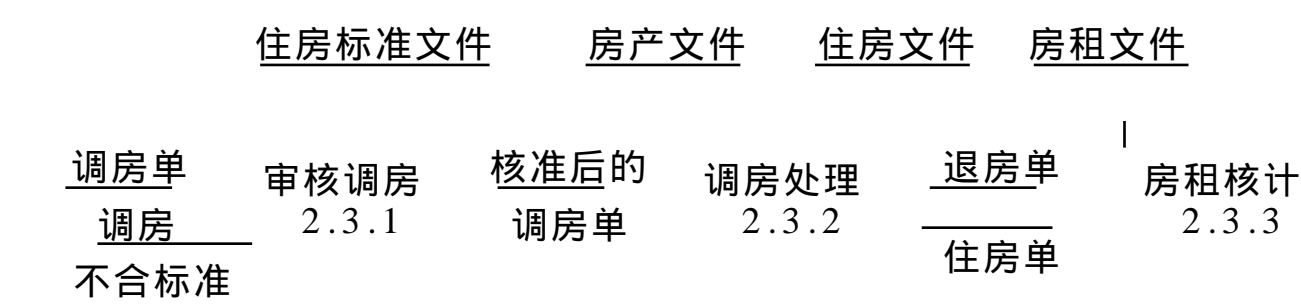
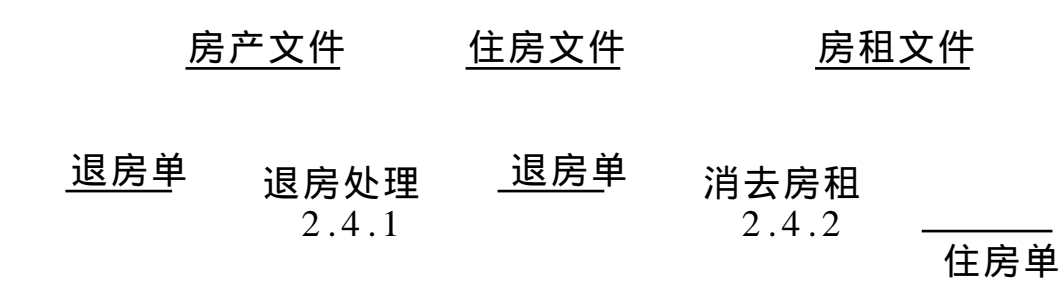


图 2 4



调房单 = 房主 + 部门 + 职称 + 住房分数 + 原住房面积 + 原住房 + 要求调房面积

退房单 = 户主 + 房号

序号 = 楼号 + 房间号

(2)数据存储条目

文件名: 住房标准文件

组成: 住房面积 + 最低住房分数

组织: 按住户面积大小递增排列

文件名: 房产文件

组成: 房号 + 住房面积 + 分配标志 + 每平方米房租

组织: 按序号递增排列

文件名: 住房文件  
组成: 户主 + 部门 + 职称 + 家庭人员 + 住房分数 + 房号 + 住房面积  
组织: 按户主名拼音字母顺序排列

文件名: 房租文件  
组成: 住房情况  
组织: 按户主名拼音字母顺序排列

(3)加工条目  
加工编号: 1  
加工名: 检查合法性  
加工逻辑: 检查输入要求的合法性  
有关信息: 当有要求输入时执行此加工

加工编号: 2.1  
加工名: 要求类型处理  
加工逻辑: 根据住户要求选择  
    case 1: 要求分房, 输出分房单  
    case 2: 要求调房, 输出调房单  
    case 3: 要求退房, 输出退房单  
有关信息: 当有合法住户要求输入时执行此加工

加工编号: 3.1  
加工名: 咨询类型处理  
加工逻辑: 根据咨询要求选择:  
    case 1: 查询住房  
    case 2: 查询房租  
    case 3: 统计要求  
有关信息: 当有咨询要求时执行此加工

加工编号: 3.2  
加工名: 住房咨询  
加工逻辑: 根据查询要求的住户名从住房文件读出住房记录  
有关信息: 有住房查询要求的执行此加工

加工编号: 3.3  
加工名: 房租查询  
加工逻辑: 根据查询要求的户主名从房租文件读出房租记录  
有关信息: 有房租查询要求时执行此加工

加工编号: 3.4  
加工名: 统计房产  
加工逻辑: 读房产文件, 按面积分类, 统计已分和未分配的住房数。输出统计表

有关信息:有统计要求时执行此加工

加工编号:3.5

加工名:打印处理

加工逻辑:将住房记录或房租记录变换成住房情况。打印住房情况

有关信息:收到住房记录或房租记录时执行此加工

加工编号:2.2.1

加工名:核准住房条件

加工逻辑:根据分房要求的住房面积从住房标准文件读出住房标准

IF 住房分数 最低住房分数

THEN 输出审核后的分房单

ELSE 取消分房资格

有关信息:当有分房单输入时执行此加工

加工编号:2.2.2

加工名:分配住房

加工逻辑:从房产文件读出一个合适的住房记录

把分房单和房产文件的房产记录的有关信息拼成住房文件的记录,输出分房单并把这个记录写入住房文件中

在房产文件的记录中填写分配标志并把这个记录写入房产文件中

有关信息:收到核准后的分房单时执行此加工

加工编号:2.2.3

加工名:房租计算

加工逻辑:根据分房单计算房租并写入房租文件

有关信息:收到住房单时执行此加工

加工编号:2.3.1

加工名:审查调房

加工逻辑:根据调房要求的住房面积从住房标准文件中读出住房标准

IF 住房分数 最低住房分数

THEN 输出审核后的调房单

ELSE 不予调房

有关信息:当有调房单输入时执行此加工

加工编号:2.3.2

加工名:调房处理

加工逻辑:从住房文件中读出原住房的住房记录,输出一个退房单

把这个记录从住房文件中删除

从房产文件中把这个房号的房产记录读出,修改分配标志

写回原文件中

把这个房号的房租记录从房租文件中删除  
以下部分同加工“ 分配住房 ”

有关信息：收到审核后的调房单时执行此加工

加工编号：2.33

加工名：房租核算

加工逻辑：根据退房单的户主名找到原房租记录，把它从文件中删除  
根据分房单计算房租写入房租文件

有关信息：收到分房单和退房单时执行此加工

加工编号：2.4.1

加工名：退房处理

加工逻辑：从住房文件中读出原住房的住房记录，输出退房单  
把这个记录从住房文件中删除  
从房产文件中把这个房号的房产记录读出  
修改分配标志写回原文件中去

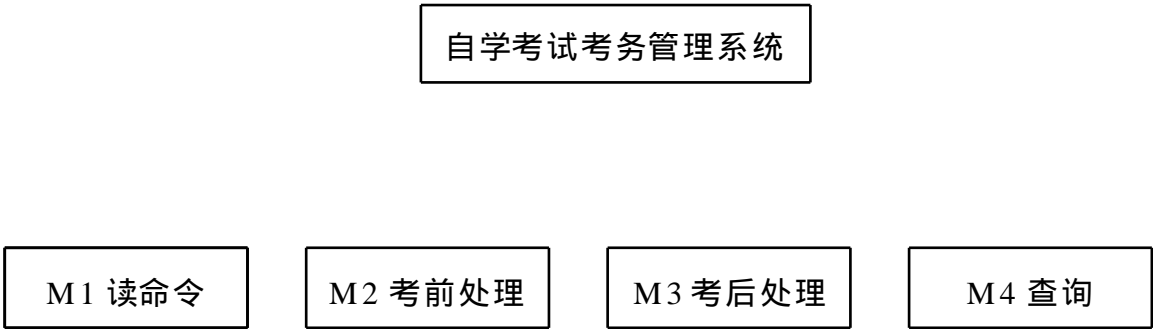
有关信息：收到退房单时执行此加工

加工编号：2.4.2

加工名：消去房租

加工逻辑：根据退房单的户主名找到原房租记录，把它从文件中删除  
有关信息：收到退房单时执行此加工

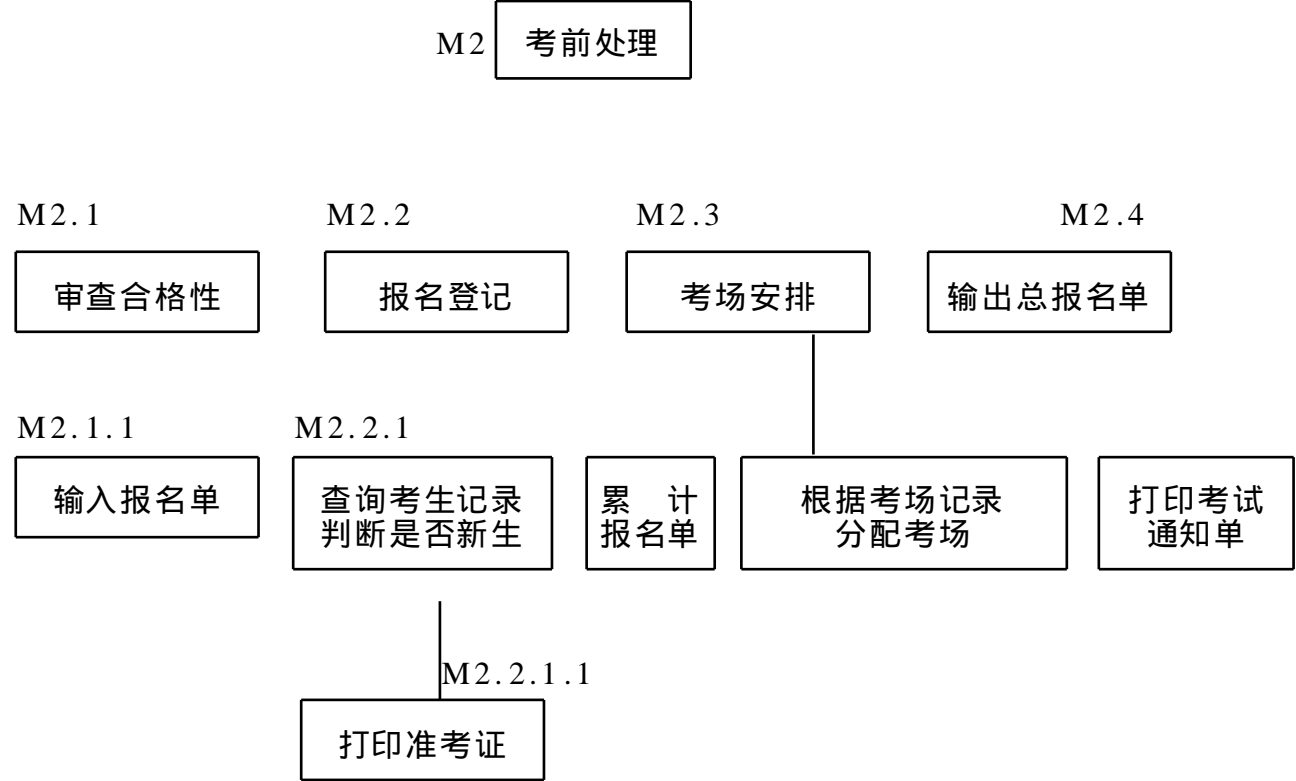
6. 将第 3 章 3.2 节例题 2“ 成人自学考试考务管理系统 ”的 DFD 设计成软件结构图  
答：主图：



成人自学考试考务管理系统主图

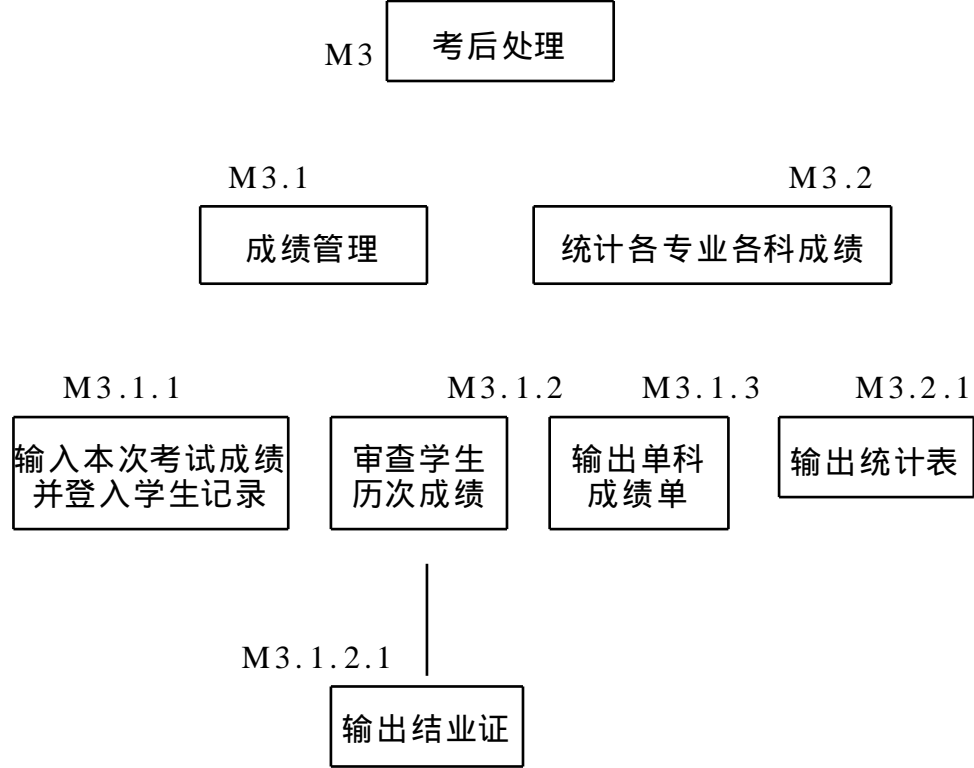


子图 M2(考前处理):



子图 M2

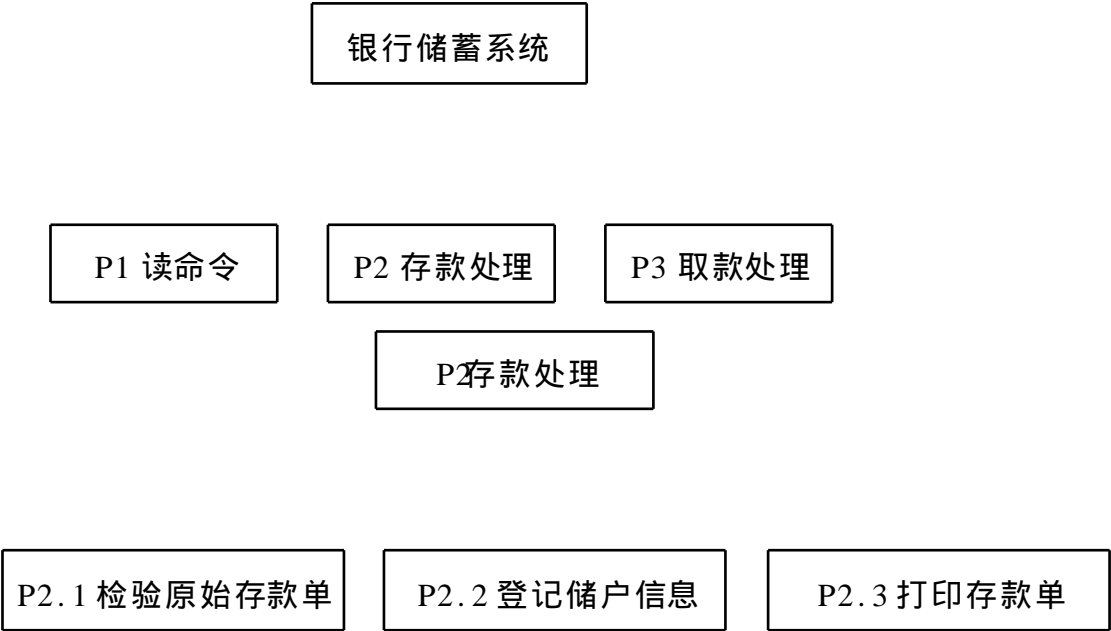
子图 M3(考后处理):



子图 M3

查询模块的成功比较简单,可以将准考证编号作为索引关键字查阅各科成绩,这里不再赘述,读者可自行设计。

7. 将第 3 章 3.2 节例题 1“ 银行储蓄系统 ”的 DFD 设计成软件结构图。  
答：



子图 P2

取款处理的结构图读者可以自己设计。

## 第五章 软件详细设计

本章总的要求是:能熟练地使用详细设计描述方法来设计模块中的算法及程序的逻辑结构。

理解 Jackson 方法的概念,学会使用 Jackson 方法设计输入输出数据结构和程序结构。

### 考试要求

1. 详细设计的基本任务,要求达到领会层次。
2. 结构化程序设计方法  
结构化程序设计的基本要点,要求达到识记层次。
3. 详细设计描述  
流程图、PAD 图及过程设计语言,要求达到简单应用层次。
4. Jackson 方法  
Jackson 方法设计小型题目,要求达到简单应用层次。

### 知识重点

#### (一)详细设计的基本任务

详细设计是软件设计的第二阶段,主要确定每个模块具体执行过程,故也称“过程设计”。其基本任务有:

- (1)为每个模块进行详细的算法设计。用某种图形、表格、语言等工具将每个模块处理过程的详细算法描述出来。
- (2)为模块内的数据结构进行设计。
- (3)对数据库进行物理设计,即确定数据库的物理结构。
- (4)其它设计。根据软件系统类型,还可能要进行代码设计、输入/输出格式设计、人机对话设计。
- (5)编写详细设计说明书。
- (6)评审。

#### (二)结构化程序设计方法

详细设计并不是具体地编写程序,而是细化成很容易地从中产生程序的图纸。详细设计的结果基本决定了最终程序的质量。处理过程设计中采用的典型方法是结构化程序设计(简称 SP)方法,其基本要点有:

1 .采用自顶向下、逐步求精的程序设计方法

2 .使用“ 顺序、选择、重复 ”三种基本控制结构构造程序

对一个模块处理过程细化时,开始是模糊的,可以用下面三种方式进行分解:

(1)用顺序方式对过程分解,确定各部分的执行顺序。

(2)用选择方式对过程分解,确定某个部分的执行条件。

(3)用循环方式对过程分解,确定某个部分进行重复的开始和结束的条件。

(4)对处理过程仍然模糊的部分反复使用以上分解方法,最终可将所有细节确定下来。

3 .主程序员组的组织形式

指开发程序的人员组织形式应采用由一个主程序员(负责全部技术活动)、一个后备程序员(协调、支持主程序员)和一个程序管理员(负责事务性工作,如收集、记录数据,文档资料管理等)三人为核心,再加上一些专家、其它技术人员组成小组。这种组织形式使设计责任集中在少数人身上。有利于提高软件质量,并且能有效地提高软件生产率。

### (三) 程序流程图

程序流程图又称为程序框图,是一种描述程序逻辑结构的工具,其优点是直观清晰、易于使用,但是它有严重缺点:

(1)可以随心所欲地画控制流程线的流向,容易造成非结构化的程序结构。

(2)流程图不易反映逐步求精的过程,往往反映的是最后的结果。

(3)不易表示数据结构。

为了克服流程图的最大缺陷,要求流程图都应由三种基本控制结构顺序组合和完整嵌套而成,不能有相互交叉的情况,这样的流程图是结构化的流程图。

### (四) PAD 图

PAD 图是指问题分析图,是一种算法描述工具,它是一种由左往右展开的二维树型结构。

PAD 图的控制流程为自上向下、从左到右地执行。其优点是:

(1)清晰地反映了程序的层次结构。

(2)支持逐步求精的设计方法,左边层次中的内容可以抽象,然后由左到右逐步细化。

(3)易读易写,使用方便。

(4)支持结构化的程序设计原理。

(5)可自动生成程序。

### (五) 过程设计语言

过程设计语言(简称 PDL),也称程序描述语言,又称伪码。它是一种用于描述模块算法设计和处理细节的语言。它分内外两层:外层语法应符合一般程序设计语言常用的语法规则,而内层语法则用一些简单的句子、短语和通用的数学符号,来描述程序应执行的功能。PDL 具有严格的关键字外层语法,用于定义控制结构、数据结构和模块接口,而

它表示实际操作和条件的内层语法又是灵活自由的,使用自然语言的词汇。

PDL 也是结构化语言,它与需求分析中描述加工逻辑的“结构化语言”统属于伪码,但它们的作用不同。“结构化语言”无严格的外层语法,内层自然语言描述较抽象、较概括。而 PDL 外层语言更严格一些,更趋于形式化,内层自然语言描述实际操作更具体更详细一些。它一般具有以下特点:

- (1)所有关键字都有固定语法,以便提供结构化控制结构、数据说明和模块的特征。
- (2)描述处理过程的说明性语言没有严格的语法。
- (3)具有数据说明机制,包括简单的与复杂的数据说明。

(4)具有模块定义和调用机制,因此开发人员应根据系统编程所用的语种,说明 PDL 表示的有关程序结构。

## (六)JSP 设计步骤

JSP 方法一般通过以下五个步骤来完成设计:

(1)分析并确定输入数据和输出数据的逻辑结构,并用 Jackson 结构图表示这些数据结构。

(2)找出输入数据结构和输出数据结构中有对应关系的数据单元。“对应关系”指这些数据单元在数据内容上、数量上和顺序上有直接的因果关系,对于重复的数据单元,重复的次序和次数都相同才有对应关系。

- (3)按一定的规则由输入、输出的数据结构导出程序结构。
- (4)列出基本操作与条件,并把它们分配到程序结构图的适当位置。
- (5)用伪码写出程序。

# 反馈测试题解

## 一、名词解释

### 1. 过程设计语言

答:过程设计语言(Process Design Language,简称 PDL),也称程序描述语言(Program Description Language),又称为伪码。它是一种用于描述模块算法设计和处理细节的语言。伪码的结构一般分为内外两层,外层语法应符合一般程序设计语言常用的语法规则,而内层语法则用一些简单的句子、短语和通用的数学符号,来描述程序应执行的功能。PDL 具有严格的关键字外层语法,用于定义控制结构、数据结构和模块接口,而它表示实际操作和条件的内层语法又是灵活自由的,使用自然语言的词汇。

### 2. PAD 图

答:PAD 图指问题分析图(Problem Analysis Diagram),是日本日立公司于 1979 年提出的一算法描述工具,它是一种由左往右展开的二维树型结构。

PAD 图的控制流程为自上而下、从左到右地执行。

## 二、填空题

- 1.详细设计是软件设计的第二阶段,主要确定每个模块\_\_\_\_\_,故也称\_\_\_\_\_。

答:具体执行过程 过程设计

2.详细设计的基本任务是为每个模块进行详细的\_\_\_\_\_ ;为模块内的\_\_\_\_\_ 进行设计 ;对\_\_\_\_\_ 进行物理设计 ;其他设计 ;编写详细设计说明书和\_\_\_\_\_ 。

答:算法设计 数据结构 数据库 评审

3.处理过程设计中采用的典型方法是\_\_\_\_\_ (简称\_\_\_\_\_ )方法。

答:结构化程度设计 SP

4.结构化程度设计方法的基本要点是: 采用\_\_\_\_\_ 、\_\_\_\_\_ 的程序设计方法; 使用\_\_\_\_\_ 构造程序; \_\_\_\_\_ 。

答:自顶向下 逐步求精 三种基本控制结构 主程序员组的组织形式

5.详细设计的目标不仅是逻辑上正确地实现\_\_\_\_\_ ,还应使设计出的处理过程\_\_\_\_\_ 。\_\_\_\_\_ 是实现该目标的关键技术之一,它指导人们用良好的思想方法开发易于\_\_\_\_\_ 、易于\_\_\_\_\_ 的程序。

答:每个模块的功能 清晰易读 结构化程度设计 理解验证

6.程序流程图又称为\_\_\_\_\_ ,应由\_\_\_\_\_ 顺序组合和完整嵌套而成,不能有\_\_\_\_\_ 的情况,这样的流程图是\_\_\_\_\_ 的流程图。

答:程序框图 三种基本控制结构 相互交叉 结构化

7.在算法描述工具中,PAD图可自动生成程序。由机器自动通过\_\_\_\_\_ 生成相应的源代码,大大提高了\_\_\_\_\_ 。

答:走树的办法 软件的生产率

8.伪码的结构一般分为内外两层,外层语法应符合一般\_\_\_\_\_ 常用的语法规则,而内层语法则用一些简单的句子、短语和通用的数学符号,来描述程序\_\_\_\_\_ 。

答:程序设计语言 应执行的功能

9.过程设计语言(简称\_\_\_\_\_ ),也称\_\_\_\_\_ ,又称为伪码。它是一种用于描述\_\_\_\_\_ 的语言。

答:PDL 程序描述语言 模块算法设计和处理细节

10.70年代中期出现了“面向数据结构”的设计方法,其中有代表性的是\_\_\_\_\_ 和\_\_\_\_\_ 。

答:Jackson方法 Warnier方法

11.JSP方法定义了一组以数据结构为指导的\_\_\_\_\_ ,它根据\_\_\_\_\_ 、\_\_\_\_\_ 的数据结构,按一定的规则映射成\_\_\_\_\_ ,即\_\_\_\_\_ ,而不是软件的体系结构,因此该方法适用于\_\_\_\_\_ 。

答:映射过程 输入 输出 软件的过程描述 程序结构 详细设计阶段

12.Jackson指出,无论数据结构还是程序结构,都限于\_\_\_\_\_ 、\_\_\_\_\_ 和\_\_\_\_\_ 三种基本结构及它们的组合。

答:顺序结构 选择结构 重复结构

13.Jackson结构图能对结构进行\_\_\_\_\_ 分解,因此可以表示\_\_\_\_\_ 。

答:自顶向下 层次结构

14.在JSP设计中,需要找出输入数据结构和输出数据结构中有对应关系的数据单元。“对应关系”指这些数据单元在\_\_\_\_\_ 上、\_\_\_\_\_ 上和\_\_\_\_\_ 上有直接的\_\_\_\_\_ ,对于重复的数据单元,重复的\_\_\_\_\_ 和\_\_\_\_\_ 都相同才有对应关系。

答:数据内容 数量 顺序 因果关系 次序 次数

15.在详细设计阶段,为了提高数据的输入、储存、检索等操作的效率并节约存储空间,对某些数据项的值要进行\_\_\_\_\_ 设计。

答:代码

16. 在详细设计阶段,一种历史最悠久、使用最广泛的描述程序逻辑结构的工具是\_\_\_\_\_。

答:程序流程图

17. 结构化程序设计方法简称\_\_\_\_\_。PAD 图指\_\_\_\_\_图。过程设计语言简称\_\_\_\_\_,也称\_\_\_\_\_语言,又称\_\_\_\_\_。

答:SP 问题分析 PDL 程序描述 伪码

18. 过程设计语言与需求分析中描述加工逻辑的“结构化语言”统属于\_\_\_\_\_码。

答:伪

19. 过程设计语言的出口结构有:\_\_\_\_\_结构、\_\_\_\_\_结构两种。

答:ESCAPE CYCLE

20. 过程设计语言的顺序结构采用\_\_\_\_\_描述。

答:自然语言

21. 过程设计语言的选择结构有:\_\_\_\_\_结构、\_\_\_\_\_结构、\_\_\_\_\_结构三种。

答:IF - ELSE IF - OR IF - ELSE CASE

22. Jackson 方法可用\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_三种基本结构来表示。Jackson 方法中的伪码也称\_\_\_\_\_,与 Jackson 所示的\_\_\_\_\_图完全对应。

答:顺序 选择 重复 图解逻辑 程序结构

23. 详细描述处理过程常用三种工具是:\_\_\_\_\_,\_\_\_\_\_和\_\_\_\_\_。

答:图形 表格 语言

24. 为了克服流程图的最大缺陷,要求流程图都应由三种基本控制结构\_\_\_\_\_组合和\_\_\_\_\_嵌套而成,不能有相互\_\_\_\_\_的情况,这样的流程图是结构化的流程图。

答:顺序 完整 交叉

25. 过程设计语言的重复结构有:\_\_\_\_\_结构、\_\_\_\_\_结构、\_\_\_\_\_结构三种。

答:FOR WHILE UNTIL

26. 过程设计语言分\_\_\_\_\_两层,\_\_\_\_\_语法应符合一般程序设计语言常用的语法规则,而\_\_\_\_\_语法则用一些简单的句子、短语和通用的数学符号,来描述程序应执行的功能。

答:内外 外层 内层

27. PAD 图的控制流程为\_\_\_\_\_,\_\_\_\_\_地执行。

答:自上向下、从左到右

28. 任何程序都可由\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_三种基本控制结构构造。这三种基本结构的共同点是\_\_\_\_\_。

答:顺序 选择 重复 单入口 单出口

29. Jackson 方法是面向\_\_\_\_\_的设计方法。早期的 Jackson 方法用于开发规模较小的数据处理系统的设计,简称为\_\_\_\_\_。80 年代后期, Jackson 在\_\_\_\_\_的基础上扩展成了一种系统的开发方法,简称\_\_\_\_\_。

答:数据结构 JSP JSP JSD

30. 三种基本控制结构的共同特点是\_\_\_\_\_。

答:单入口、单出口

31. 结构化程序设计技术指导人们用良好的思想方法开发\_\_\_\_\_的程序。

答:易理解、易验证

32. 在详细设计阶段,经常采用的工具有\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_等。

答:程序流程图 PDL PAD 图

33. 详细设计的目标不仅是逻辑上正确地实现每个模块的功能,还应使设计上的处理过程\_\_\_\_\_。

答:清晰易读

34. 结构化程序设计方法的要点是使用\_\_\_\_\_结构,自顶向下,逐步求精地构造算法或程序。

答:三种基本控制

35. 在 JSP 方法中解决结构冲突的具体办法是\_\_\_\_\_。

答:中间数据结构或中间文件

36. PAD 图清晰地反映了程序的层次结构,图中的竖线为程序的\_\_\_\_\_。

答:层次线

37. 结构化程序设计方法提倡的工作方式为\_\_\_\_\_的组织形式。

答:主程序员组

38. PDL 具有严格的关键字外语法,用于定义\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_。

答:控制结构 数据结构 模块接口

39. 为了产生结构化的流程图,要求应由三种基本控制结构\_\_\_\_\_和\_\_\_\_\_而成。

答:顺序组织 完整嵌套

40. Jackson 图不仅可表示程序结构,还可表示\_\_\_\_\_。

答:数据结构

41. 用 JSP 方法导出程序结构后,还要列出\_\_\_\_\_,并把它们分配到程序结构图的适当位置上去。

答:操作与条件

42. 在详细设计阶段,除了对模块内的算法进行设计,还应对模块内的\_\_\_\_\_进行设计。

答:数据结构

43. 详细描述处理过程常用的三种工具是图形、语言和\_\_\_\_\_。

答:表格

### 三、选择题

1 程序控制一般分为( )、分支、循环三种基本结构。

- A .分块
- B 顺序
- C .循环
- D .分支

答:B

2 在描述软件的结构和过程,提出的设计表达工具不正确的是 ( )

- A .图形表达工具:流程图、NS 图等
- B .文字表达工具:伪代码、PDL 等
- C .表格表达工具:判定表等
- D .系统设计表达工具:用于表达软件过程

答:D

3 数据元素组成数据的方式有如下( )基本类型。

- A .顺序
- B 选择
- C .重复
- D .以上全是

答:D

4 .一个程序如果把它作为一个整体,它也是只有一个入口、一个出口的单个顺序结构,这是一种

( )



- A .结构程序
- B 组合的过程
- C .自顶向下设计
- D .分解过程

答:B

5 指出 PDL 是下列哪种语言 ( )

- A .高级程序设计语言
- B .伪码
- C .中级程序设计语言
- D .低级程序设计语言

答:B

6 详细设计规格说明通常是使用如下手段 ( )

- A .IPO 图与层次图
- B .HIPO
- C .IPO 或 PDL
- D .HIPO 或 PDL

答:D

7 软件详细设计主要采用的方法是: ( )

- A .结构程序设计
- B .模型设计
- C .结构化设计
- D .流程图设计

答:C

8 Jackson 方法根据( )来导出程序结构。

- A .数据结构
- B 数据间的控制结构
- C .数据流图
- D IPO 图

答:A

9 流程图中的顺序结构中各个方框是对程序的( )进行分块,使之表达得更清晰。

- A .物理意义
- B 流程
- C .函数
- D 逻辑意义

答:D

10 .在下述情况下,从供选择的答案中,选出合适的( )描述工具。当算法中需要用一个模块去计算多种条件的复杂组合,并根据这些条件完成适当的功能。

- A 程序流程图
- B .NS 图
- C .PAD 图或 PDL
- D .判定表

答:D

11 模块之间的层次关系一般可用不同的层次名来描述。写法一般有两种:( )和并列。

- A .NS 图
- B .嵌套
- C .PAD 图
- D .循环

答:B

12 .在软件开发过程中,以下说法正确的是 ( )

- A 程序流通图是逐步求精的好工具
- B .NS 图不可能任意转移控制,符合结构化原则
- C .判定表是一种通用的设计工具
- D .程序流程图和 NS 图都不易表达模块的层次结构

答:B

13 Jackson 方法是一种面向( )的方法。

- A 对象
- B .数据结构
- C 数据流
- D .控制流

答:B

14 .模块的内部过程描述就是模块内部的( ),它的表达形式就是详细设计语言。

- A .模块化设计
- B .算法设计
- C .程序设计
- D .详细设计

答:B

15 .以下说法正确的是 ( )

- A .所有改变循环条件的成分都在循环体外
- B .在直到型循环中,循环体至少要执行一次
- C .在当型循环中,循环体至少要执行一次
- D .基本程序结构不允许嵌套

答:B

16 .( )是一种结构设计语言,它陈述系统模块是什么和它们如何结合在一起实现系统的功能,它表达的是软件系统结构设计的信息。

- A .PDL
- B .C 语言
- C .C + +
- D .模块化互连语言

答:D

17 .程序控制的三种基本结构中,( )结构可提供多条路径选择。

- A .反序
- B .顺序
- C .循环
- D .分支

答:D

18 .面向数据结构的设计方法(Jackson 方法)是进行( )的形式化的方法。

- A .系统设计
- B .详细设计
- C .软件设计
- D .编码

答:B

19 .程序控制的三种基本结构中,( )结构可提供程序重复控制。

- A .遍历
- B .排序
- C .循环
- D .分支

答:C

20 Jackson 方法主要适用于规模适中的( )系统的开发。

- A .数据处理
- B .文字处理
- C .实时控制
- D .科学计算

答:A

21 .下列叙述正确的是 ( )

- A .NS 图可以用于系统设计
- B .PDL 语言可以用于运行
- C .PAD 图表达的软件过程成树型结构
- D .结构化程序设计强调效率第一

答:C

22 .工程上常用的表达工具有 ( )

- A .图形工具
- B .表格工具
- C .语言工具
- D .以上全是

答:D

23 .以下说法正确的是 ( )

- A. 程序流程图是一种算法描述工具
- B. PAD 图是一种描述程序逻辑结构的工具
- C. 过程设计语言是一种用于描述模块算法设计和处理细节的语言
- D. PAD 图是一种由左往右展开的二维树型结构

答:B

24. 对于过程设计语言,下面说法错误的是 ( )

- A. PDL 的总体结构与一般程序完全相同
- B. PDL 的外语法同相应程序语言一致
- C. PDL 的内语法使用自然语言,虽不能转换成源程序,但可作为注释嵌入在源程序中
- D. PDL 提供的机制比图形全面,可自动生成程序代码,提高软件生产率

答:C

25. 对于详细设计,下面说法错误的是 ( )

- A. 详细设计是具体地编写程序
- B. 详细设计是细化成很容易地从中产生程序的图纸
- C. 详细设计的结果基本决定了最终程序的质量
- D. 详细设计中采用的典型方法是结构化程序设计方法

答:A

26. 以下说法错误的是 ( )

- A. PAD 图支持逐步求精的设计方法
- B. 程序流程图往往反映的是最后的结果
- C. 程序流程图容易造成非结构化的程序结构
- D. PAD 图支持结构化的程序设计原理
- E. 程序流程图清晰反映了逐步求精的过程

答:E

27. 程序的三种基本控制结构是 ( )

- |               |             |
|---------------|-------------|
| A. 过程、子程序和分程序 | B. 顺序、选择和重复 |
| C. 递归、堆栈和队列   | D. 调用、返回和转移 |

答:B

28. 对一个模块处理过程的分解,以下正确的说法是 ( )

- A. 用循环方式对过程分解,确定各部分的执行顺序
- B. 用选择方式对过程分解,确定某个部分的执行条件
- C. 用顺序方式对过程分解,确定某个部分进行重复的开始和结束的条件
- D. 对处理过程仍然模糊的部分反复使用循环方式对过程进行分解

答:B

29. 详细设计与概要设计衔接的图形工具是 ( )

- |          |         |          |          |
|----------|---------|----------|----------|
| A. DFD 图 | B. SC 图 | C. PAD 图 | D. 程序流程图 |
|----------|---------|----------|----------|

答:B

30. 结构化程序设计的一种基本方法是 ( )

- |        |        |        |          |
|--------|--------|--------|----------|
| A. 筛选法 | B. 递归法 | C. 迭代法 | D. 逐步求精法 |
|--------|--------|--------|----------|

答:D

31. JSP 方法是一种面向( )的设计方法。

- |       |        |         |         |
|-------|--------|---------|---------|
| A. 对象 | B. 数据流 | C. 数据结构 | D. 控制结构 |
|-------|--------|---------|---------|

答:C

32.( )工具在软件详细设计过程中不采用。

- A. 判定表                      B. IPO 图                      C. PDL                      D. DFD 图

答:D

33. 详细设计的任务是确定每个模块的( ),即模块的 ( )

- A. 外部特性                      B. 内部特性  
C. 算法和使用的数据                      D. 功能和输入输出数据

答:B

34. JSP 方法主要用于规模适中的( )系统的开发。

- A. 数据处理                      B. 实时处理                      C. 文字处理                      D. 科学计算

答:A

35. 结构化程序设计主要强调的是 ( )

- A. 程序的效率                      B. 程序执行速度                      C. 程序易读性                      D. 程序的规模

答:C

36. 结构经程序设计主要强调的是 ( )

- A. 程序的效率                      B. 程序的执行速度  
C. 程序的易读性                      D. 程序的规模

答:C

37. 在软件详细设计过程中不采用的描述工具是 ( )

- A. 判定表                      B. IPO 图                      C. PAD 图                      D. DFD 图

答:D

38. Jackson 方法实现从( )导出 ( )

- A. 数据结构                      B. 数据流图                      C. 程序结构                      D. 软件模块层次结构

答:C

39. 详细设计的任务是确定每个模块的 ( )

- A. 算法                      B. 功能                      C. 调用关系                      D. 输入输出数据

答:A

40. 程序的三种基本控制结构的共同特点是 ( )

- A. 不能嵌套使用                      B. 只能用来写简单程序  
C. 已经用硬件实现                      D. 只有一个入口和一个出口

答:D

41. JSP 方法是一种面向( )的设计方法。

- A. 对象                      B. 数据流                      C. 数据结构                      D. 控制结构

答:C

42. 在详细设计阶段,一种二维树型结构并可自动生成程序代码的描述工具是 ( )

- A. PAD                      B. PDL                      C. IPO                      D. 判定树

答:A

43. PDL 是软件开发过程中用于( )阶段的描述工具。

- A. 需求分析                      B. 概要设计                      C. 详细设计                      D. 编程

答:C

44. JSP 方法根据输入输出的数据结构按一定的规则映射成软件的 ( )

- A. 体系结构                      B. 数据结构                      C. 程序结构                      D. 顺序结构

答:C

45. 结构化程序设计的一种基本方法是 ( )

- A. 筛选法                      B. 递归法                      C. 迭代法                      D. 逐步求精法

答:D

46. Jackson 图上下层之间的关系是 ( )

- A. 调用关系                      B. 组成关系                      C. 继承关系                      D. 嵌套关系

答:B

47. 在详细设计阶段,可自动生成程序代码并可作为注释出现在源程序中的描述工具是 ( )

- A. PAD                              B. PDL                              C. IPO                              D. 流程图

答:B

48. 对于 PDL 与需求分析中描述加工逻辑的“结构化语言”的区别,以下说法错误的是 ( )

- A. PDL 不是结构化语言  
B. 需求分析中描述加工逻辑的“结构化语言”无严格的外语法。  
C. PDL 外层语言更严格一些,更趋于形式化。  
D. 需求分析中描述加工逻辑的“结构化语言”内层自然语言描述较抽象、较概括

答:A

## 四、简答题

1 简述 Jackson 方法的设计步骤。

答:Jackson 方法(JSP)设计步骤:

- (1)分析并确定输入数据和输出数据的逻辑结构,并用 Jackson 结构图表示这些数据结构。
- (2)找出输入数据结构和输出数据结构中有对应关系的数据单元。“对应关系”指这些数据单元在数据内容上、数量上和顺序上有直接的因果关系,对于重复的数据单元,重复的次序和次数都相同才有对应关系。
- (3)按一定的规则由输入、输出的数据结构导出程序结构。
- (4)列出基本操作与条件,并把它们分配到程序结构图的适当位置。
- (5)用伪码写出程序。

2 程序流程图的特点是什么?

答:流程图的优点是清晰、易于使用,是开发者普遍采用的工具,但是它有严重缺点:

- (1)可以随心所欲地画控制流程线的流向,容易造成非结构化的程序结构。编码时势必不加限制地使用 GOTO 语句,导致基本控制块多入口多出口,这样会使软件质量受到影响,与软件设计的原则相违背。
- (2)流程图不易反映逐步求精的过程,往往反映的是最后的结果。
- (3)不易表示数据结构。

为了克服流程图的最大缺陷,要求流程图都应由三种基本控制结构顺序组合和完整嵌套而成,不能有相互交叉的情况,这样的流程图是结构化的流程图。

3. PDL 有哪些优点?

答:PDL 的总体结构与一般程序完全相同。外语法同相应程序语言一致,内语法使用自然语言,易编写,易理解,也很容易转换成源程序。除此以外,还有以下优点:

- (1)提供的机制比图形全面,为保证详细设计与编码的质量创造了有利条件。
- (2)可作为注释嵌入在源程序中一起作为程序的文档,并可同高级程序设计语言一样进行编辑、修

改,有利于软件的维护。

(3)可自动生成程序代码,提高软件生产率。目前已有 PDL 多种版本(如 PDL/ pascal、PDL/ C、PDL/ Ada 等),为自动生成相应代码提供了便利条件。

4. 详细设计的基本任务是什么 ?

答: (1)为每个模块进行详细的算法设计。用某种图形、表格、语言等工具将每个模块处理过程的详细算法描述出来。

(2)为模块内的数据结构进行设计。对于需求分析、概要设计确定的概念性的数据类型进行确切的定义。

(3)对数据库进行物理设计,即确定数据库的物理结构。物理结构主要指数据库的存储记录格式、存储记录安排和存储方法,这些都依赖于具体所使用的数据库系统。

(4)其他设计:根据软件系统的类型,还可能要进行以下设计:

代码设计。为了提高数据的输入、分类、存储、检索等操作,节约内存空间,对数据库中的某些数据项的值要进行代码设计。

输入/ 输出模式设计。

人机对话设计。对于一个实时系统,用户与计算机频繁对话,因此要进行对话方式、内容、格式的具体设计。

(5)编写详细设计说明书(主要内容见本书附录)。

(6)评审。对处理过程的算法和数据库的物理结构都要评审。

5. 试述详细设计的主要方法与描述工具 ?

答: (1)结构化程序设计方法

该方法的要点是使用三种基本控制结构自顶向下,逐步求精地构造算法或程序。该方法在设计处理过程时,要注意以下几点:

首先考虑程序完成的主要功能的步骤,细节问题(如出错处理、例外情况等)放在求精的步骤中考虑。

要考虑判断处理和重复处理的问题,特别是判断的条件、重复开始和终止的条件,把这些条件要写在算法中,可借助判定表把判定的条件及动作整理出来,不至于有漏掉及冗余的情况。

要考虑数据对程序的影响,外部数据一般在数据字典中有定义,而模块内部定义的数据,不会在数据字典中出现,应列出表来,有利于算法的理解。

三种基本控制结构只能按顺序出现或完整嵌套,不能出现相互交叉的情况,否则算法就是非结构化的,不易于理解与修改。

6. 结构化流程图、PAD 图和 PDL 语言 ?

答:注意它们各自的特点,目前描述算法采用最多的是 PDL,即伪码。伪码的外语法不一定是教材中规定的格式,一般根据系统选定的编程语言结构规定统一的外语法,如类 × × 语言。伪码书写算法方便,同时还有模块定义、数据定义等格式,便于在编程阶段向程序过渡。

7. 用 PDL 表示的程序结构一般有哪几种 ?

答: (1)顺序结构:采用自然语言描述。

(2)选择结构:

IF - ELSE 结构

```
IF 条件          IF 条件
    处理 S1      或    处理 S
ELSE
    处理 S2          ENDIF
```

```
ENDIF
IF - ORIF - ELSE 结构
IF 条件 1
    处理 S1
ORIF 条件 2
    处理 S2
...
ELSE 处理 Sn
ENDIF
```

```
CASE 结构
CASE OF
CASE(1)
    处理 S1
CASE(2) 处理 S2
...
ELSE 处理 Sn
ENDCASE
```

(3)重复结构:

```
FOR 结构
FOR i= 1 TO n
    循环体
ENDFOR
```

```
WHILE 结构
WHILE 条件
    循环体
ENDWHILE
```

```
UNTIL 结构
REPEAT
    循环体
UNTIL 条件
```

(4)出口结构:

```
ESCAPE 结构(退出本层结构)
WHILE 条件
    处理 S1
ESCAPE L IF 条件
    处理 S2
ENDWHILE
```

L:.....

```
CYCLE 结构(循环内部进入循环的下一次)
L: WHILE 条件
    处理 S1
CYCLE L IF 条件
```

处理 S2

ENDWHILE

(5) 模块定义与调用：

模块定义

PROCEDURE 模块名(参数)

...

RETURN

END

模块调用

CALL 模块名(参数)

(6) 数据定义：

DECLARE 属性 变量名.....

属性有：字符、整型、实型、双精度、指针、数组、结构等类型。

(7) 输入/ 输出：

GET(输入变量表)

PUT(输出变量表)

8. 详细设计说明书有哪些内容？

答：(1) 引言

编写目的

项目背景

参考资料

术语

(2) 软件结构

(3) 模块设计说明

各模块结构

算法

数据结构

程序逻辑

存储分配与数组分配

单元说明

9. 结构化程序设计方法的基本要点是什么？

答：(1) 采用自顶向下、概要设计中,都采用了自顶向下、逐层细化的方法。使用“抽象”这个手段,上层对问题抽象、对模块抽象、对数据抽象,下层则进一步分解,进入另一个抽象层次。在详细设计中,虽然处于“具体”设计阶段,但在设计某个模块内部处理过程中,仍可以逐步求精,降低处理细节的复杂程序。

(2) 使用三种基本控制结构构造程序

任何程序都可由顺序、选择、重复三种基本控制结构构造。这三种基本结构的共同点是单入口、单出口。不但能有效的限制作用 GOTO 语句,还创立了一种新的程序设计思想、方法和风格,同时为自顶向下、逐步求精的设计方法提供了具体的实施手段。如对一个模块处理过程细化时,开始是模糊的,可以用下面三种方式对模糊过程进行分解：

用顺序方式对过程分解,确定各部分的执行顺序。

用选择方式对过程分解,确定某个部分的执行条件。



用循环方式对过程分解,确定某个部分进行重复的开始和结束的条件。

对处理过程仍然模糊的部分反复使用以上分解方法,最终可将所有细节确定下来。

### (3)主程序员组的组织形式

指开发程序的人员组织方式应采用由一个主程序员(负责全部技术活动)、一个后备程序员(协调、支持主程序员)和一个程序管理员(负责事务性工作,如收集、记录数据,文档资料管理等)三人为核心,再加上一些专家(如通信专家、数据库专家)、其他技术人员组成小组。这种组织形式突出了主程序员的领导,设计责任集中在少数人身上,有利于提高软件质量,并且能有效地提高软件生产率。这种组织形式最先由 IBM 公司实施,随后其他软件公司也纷纷采用主程序员制的工作方式。

因此,结构化程序设计方法是综合应用这些手段来构造高质量程序的思想方法。

### 10. 流程图有哪些种类?

答:流程图的优点是直观清晰、易于使用,是开发者普遍采用的工具,但是它有严重缺点:

(1)可以随心所欲地画控制流程线的流向,容易造成非结构化的程序结构。编码时势必加限制地使用 GOTO 语句,导致基本控制块多入口多出口,这样会使软件质量受到影响,与软件设计的原则相违背。

(2)流程图不易反映逐步求精的过程,往往反映的是最后的结果。

(3)不易表示数据结构。

为了克服流程图的最大缺陷,要求流程图都应由三种基本控制结构顺序组合和完整嵌套成,不能有相互交叉的情况,这样的流程图是结构化的流程图。

### 11. PDL 一般具有哪些特点?

答:(1)所有关键字都有固定语法,以便提供结构化控制结构、数据说明和模块的特征。

(2)描述处理过程的说明性语言没有严格的语法。

(3)具有数据说明机制,包括简单的与复杂的数据说明。

(4)具有模块定义和调用机制,因此开发人员应根据系统编程所用的语种,说明 PDL 表示的有关程序结构。

### 12. 什么是 Jackson 方法?主导思想是什么?

答:以数据结构为基础导出程序结构的这一设计过程称为 JSP 方法,是一种面向数据结构的开发方法。JSP 方法的主导思想是被解问题的程序结构往往与数据结构相对应,当问题的数据结构具有选择性质时,程序一般用选择结构来表示;若数据结构具有重复性质时,须用循环程序来处理;对分层的数据结构总是用分层的程序处理。对于数据处理系统,程序的功能是将输入数据变换成输出数据且程序结构对数据结构有一定的依赖性,因此 Jackson 设计了以输入数据、输出数据结构为基础映射成程序结构的规则。但在许多情况下,输入数据和输出数据之间没有结构上的对应关系,JSP 方法把这种情况称为结构冲突。解决冲突的方法是引入中间数据结构,即在输入数据和输出数据结构之间以中间数据结构发生关联,然后建立多个程序结构。对于解决规模不大的、输入输出数据结构清晰且结构不互相冲突的问题,使用 JSP 方法较为简便。该方法可与 SD 方法结合起来,用 SD 方法设计总的软件结构,用 JSP 方法设计某些模块。

### 13. 试述 JSP 设计步骤?

答:JSP 方法一般通过以下五个步骤来完成设计:

(1)分析并确定输入数据和输出数据的逻辑结构,并用 Jackson 结构图表示这些数据结构。

(2)找出输入数据结构和输出数据结构中对应关系的数据单元。

(3)按一定的规则由输入、输出的数据结构导出程序结构。

(4)列出基本操作与条件,并把它们分配到程序结构图的适当位置。

(5)用伪码写出程序。

14. Jackson 结构图有哪些特点？

- 答: (1) 能对结构进行自顶向下分解, 因此可以表示层次结构。  
(2) 结构易读, 形象直观。  
(3) 既能表示数据结构也能表示程序结构, 且表示的是组成关系。

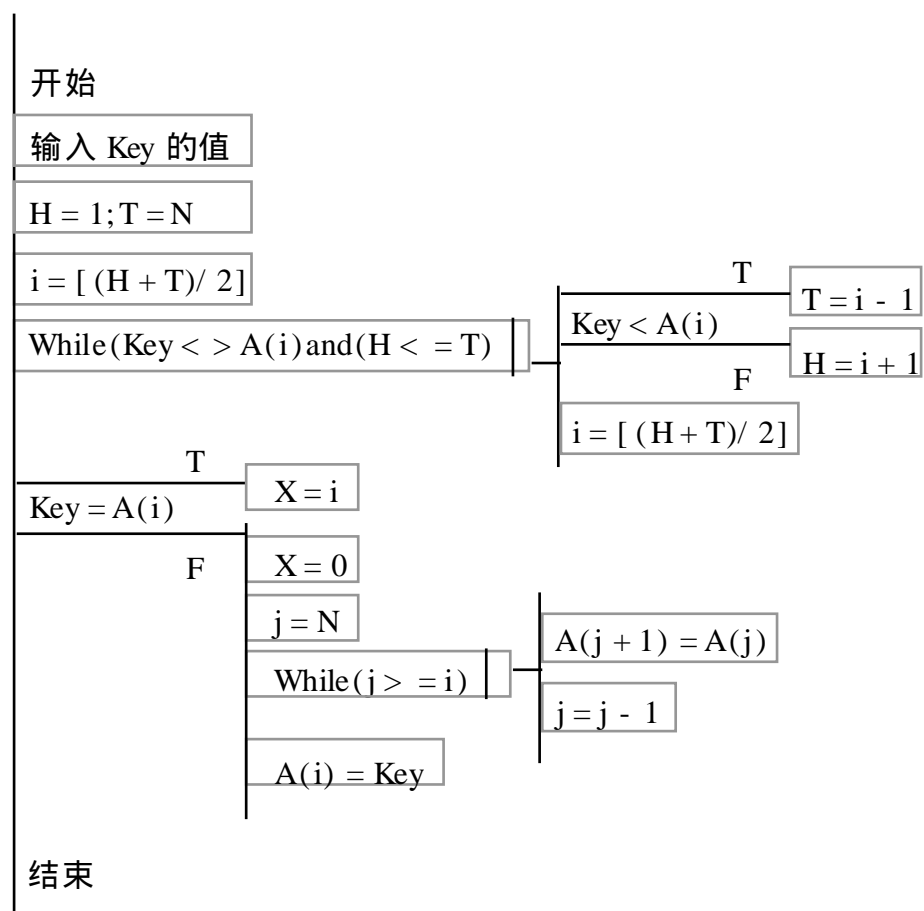
五、应用题

1 用 PAD 图描述下面问题的控制结构。

有一个表 A(1)、A(2)、...A(N), 按递增顺序排列。给定一个 Key 值, 在表中用折半法查找。若找到, 将表位置 i 送入 x, 否则将零送到 x, 同时将 Key 值插入表中。

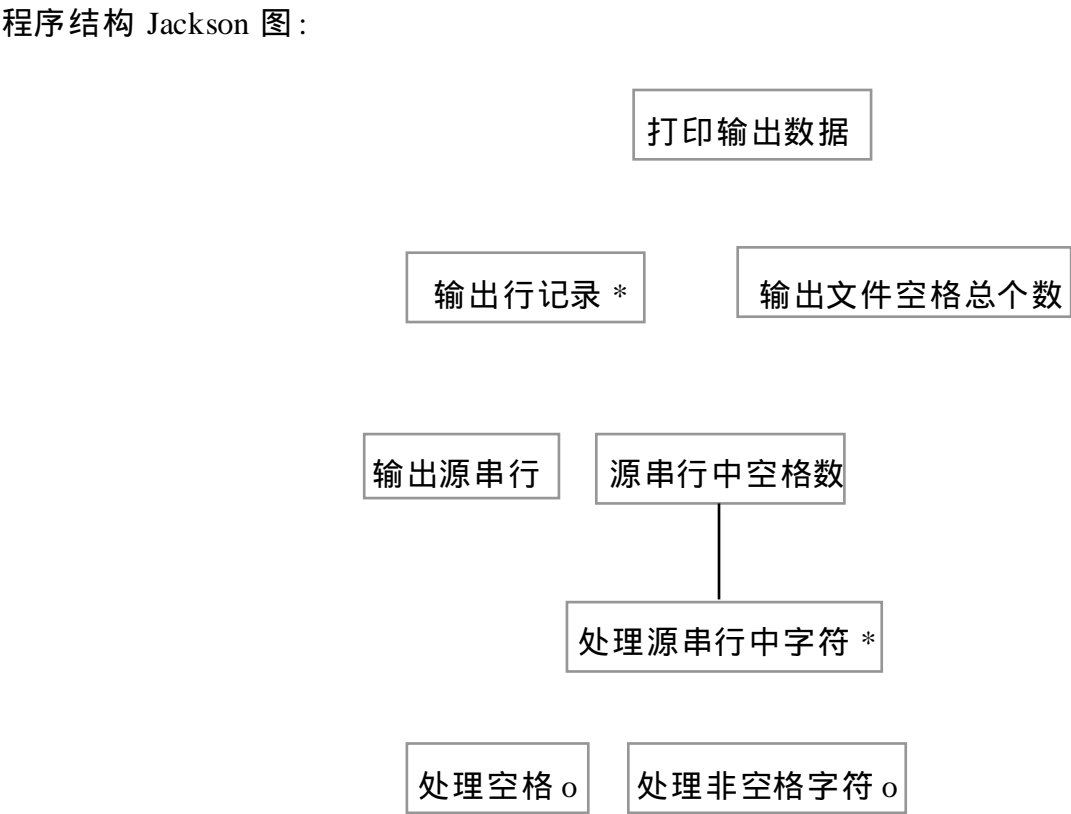
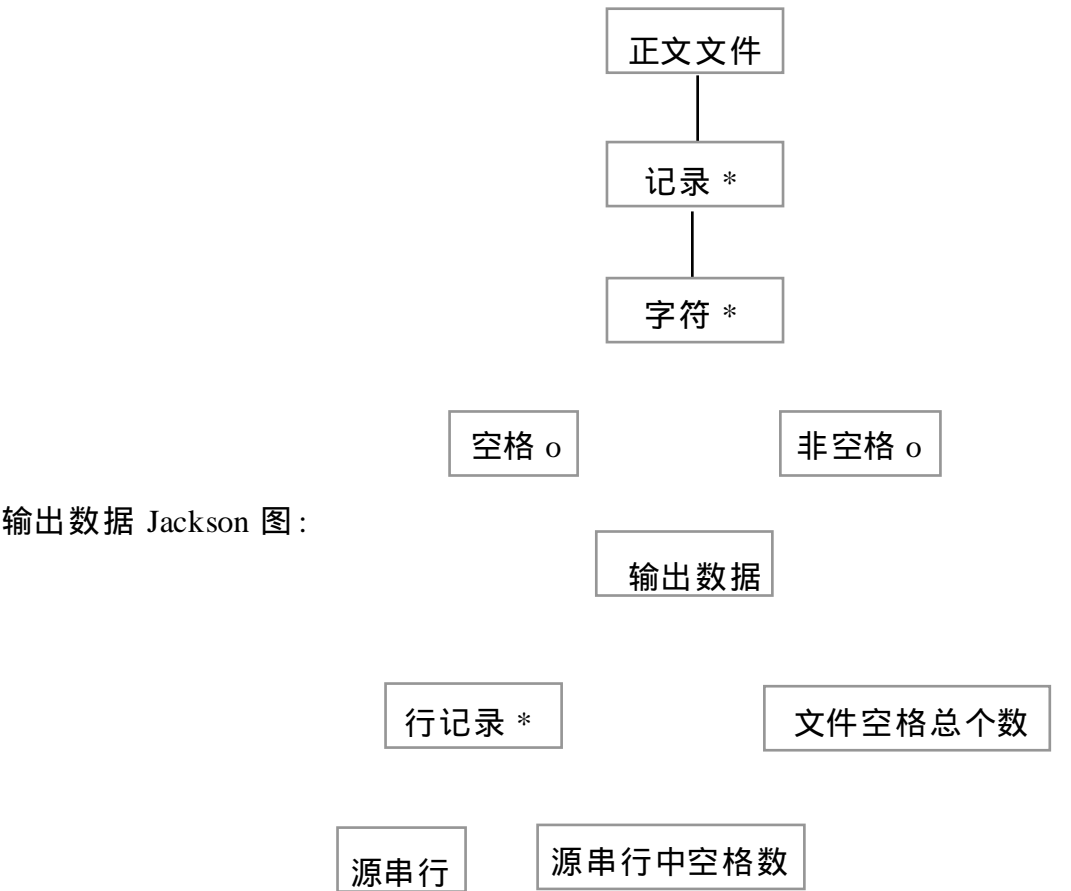
算法:  
置初值  $H = 1$  (表头),  $T = N$  (表尾)。  
置  $i = [(H + T) / 2]$  (取整)。  
若  $Key = A(i)$ , 则找到, i 送到 x; 若  $Key > A(i)$ , 则 Key 在表的后半部分, i + 1 送入 H。  
若  $Key < A(i)$ , 则 Key 在表的前半部分, i - 1 送 T, 重复第 2 步查找直到  $H > T$  为止。  
查不到时, 将 A(i), ..., A(N) 移到 A(i + 1), ..., A(N + 1), Key 值送入 A(i) 中。

答: PAD 图描述如下:



2 .一个正文文件由若干记录组成, 每个记录是一个字符串, 要求统计每个记录中空格字符的个数及文件中空格字符的总个数。要求输出数据格式是每复制一行字符串之后, 另起一行打印出这个字符串中的空格数, 最后打印出文件空格的总个数, 用 Jackson 方法设计该程序结构。

答: 输入数据 Jackson 图:

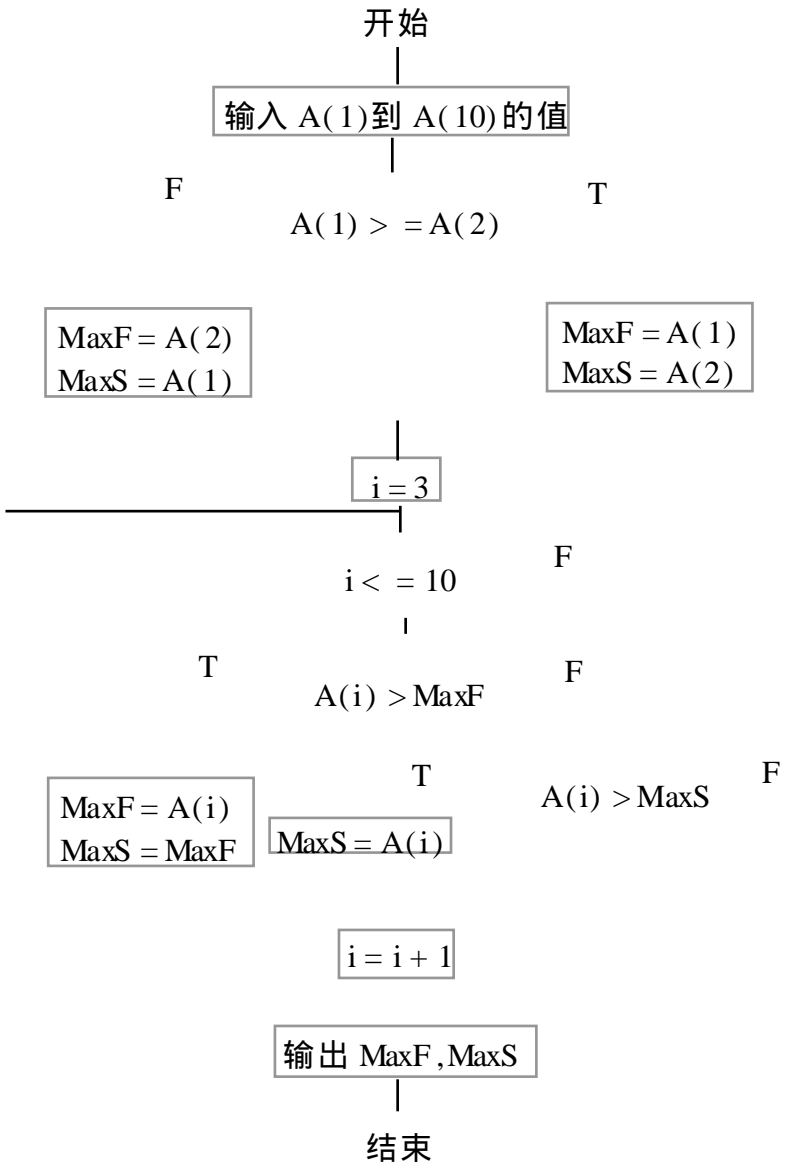


3 .请使用流程图、PAD 图和 PDL 语言描述下列程序的算法。

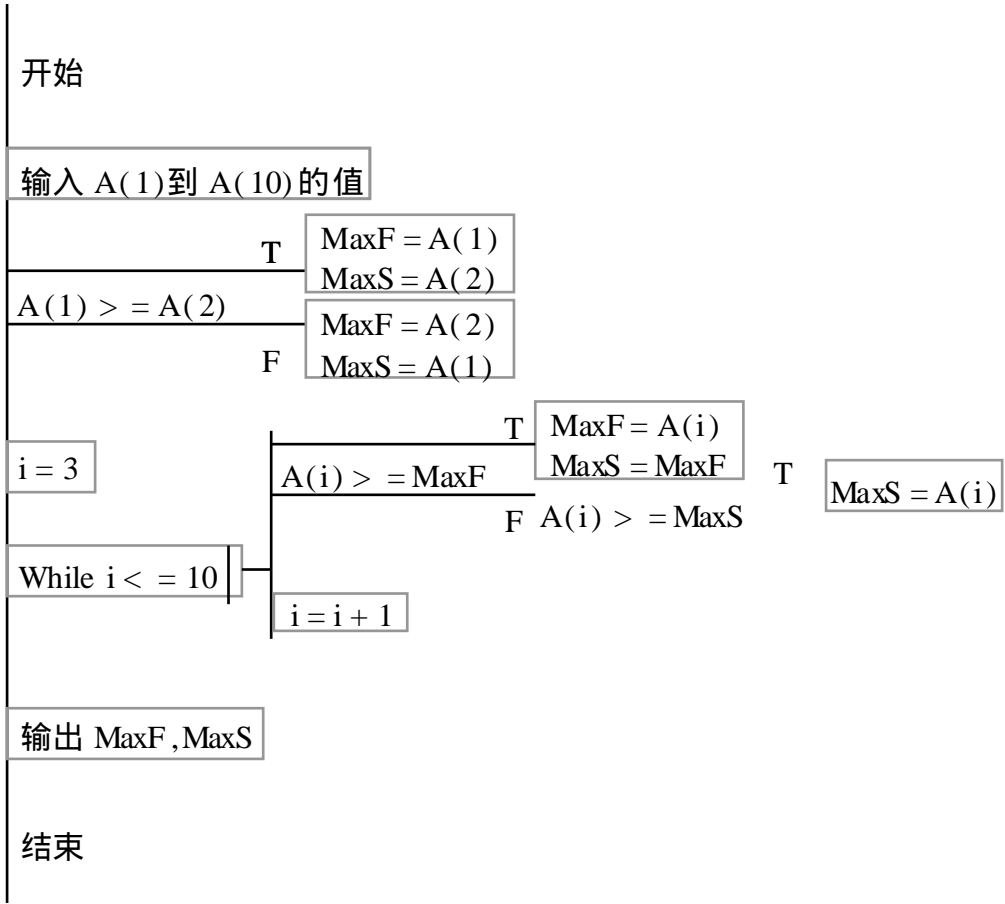
(1)在数据 A(1) ~ A(10)中求最大数和次大数。

(2)输入三个正整数作为边长,判断该三条边构成的三角形是等边、等腰或一般三角形。

答: ( )流程图描述:



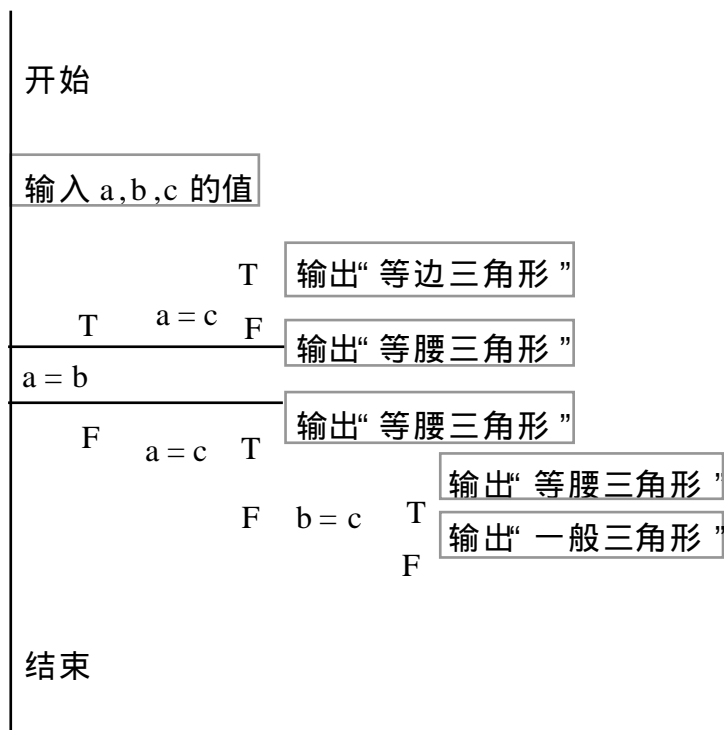
( ) PAD 图描述：



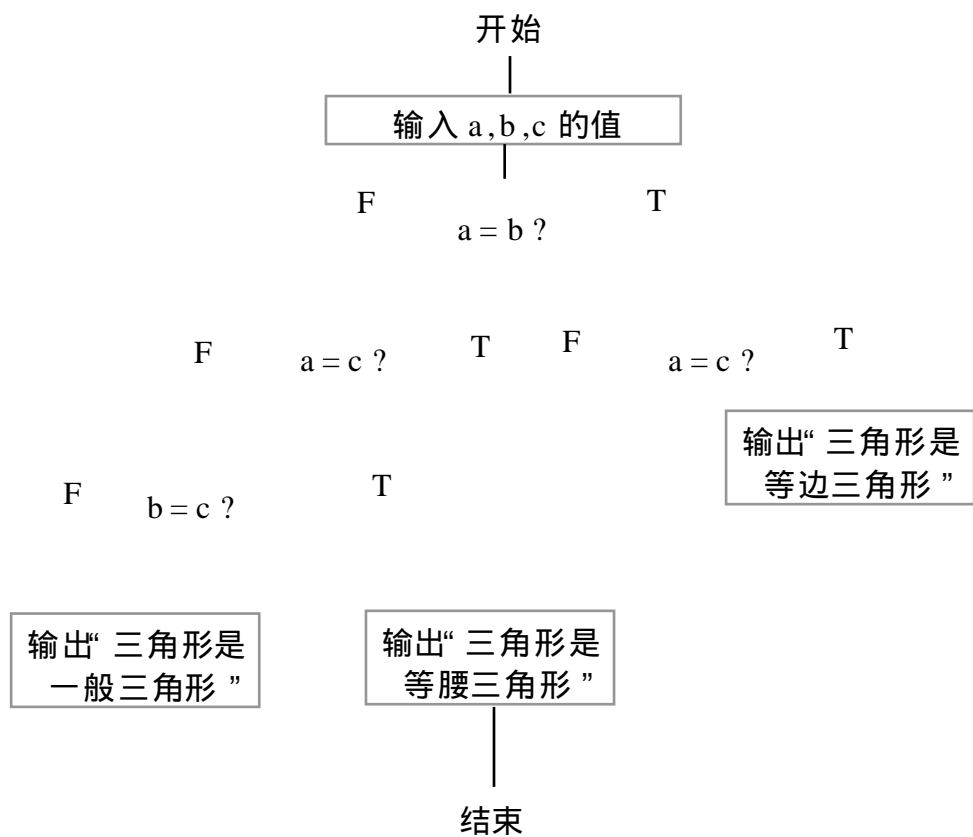
( )PDL 描述：

```
MAXF = 0
MAXS = 0
IF  A ( 1 ) > = A( 2 )
    MAXF = A( 1 )
    MAXS = A( 2 )
ELSE
    MAXF = A( 2 )
    MAXS = A( 1 )
ENDIF
FOR i = 3 TO 10
    IF A( i ) > MAXF
        MAXF = A( i )
        MAXS = MAXF
    ORIF A( i ) > MAXS
        MAXS = A( i )
    ENDIF
ENDFOR
PUT( MAXF , MAXS )
```

( )PAD 图描述：



( )流程图描述：



( ) PDL 描述：

```

GET(a,b,c)
IF (a = b)
    IF (a = c)
        PUT(" 三角形是等边三角形 ")
    ELSE
        PUT(" 三角形是等腰三角形 ")
    ENDIF
ELSE
    IF (a = c)
        PUT(" 三角形是等腰三角形 ")
    ELSE
        IF (b = c)
            PUT(" 三角形是等腰三角形 ")
        ELSE
            PUT(" 三角形是一般三角形 ")
        ENDIF
    ENDIF
ENDIF
  
```

## 第六章 软件编码

掌握几种常用的程序设计语言的特点,领会程序设计中应注意的问题,注重培养良好的编程风格。

### 考试要求

#### 1. 程序设计语言的特点及选择

几种常用的程序设计语言的特点,要求达到识记层次。

#### 2. 程序设计风格,要求达到领会层次。

### 知识重点

#### (一) 程序设计语言的特性

##### 1. 心理特性

(1)歧义性 (2)简洁性 (3)局部性和顺序性 (4)传统性

##### 2. 工程特性

(1)可移植性。它指程序从一个计算机环境移植到另一个计算机环境的容易程度。

(2)开发工具的可利用性。有效的开发工具可以缩短编码时间,改进源代码的质量。

(3)软件的可重用性。

(4)可维护性。

##### 3. 技术特性

语言的技术特性对软件工程各阶段有一定的影响,特别是确定了软件需求之后,程序设计语言的特性就很重要了,要根据项目的特性选择相应特性的语言。

#### (二) 程序设计语言的选择

程序设计语言的选择考虑的因素通常有:

##### 1. 项目的应用领域

这是选择语言的关键因素,项目应用领域一般为以下几种类型:

(1)科学与工程计算。

(2)数据处理与数据库应用。

(3)实时处理。

(4)系统软件。

编写操作系统、编译系统等系统软件时,可选用汇编语言、C语言、Pascal语言和Ada语言。

(5)人工智能。

如果要完成人工智能领域内的系统,应选择 Prolog、Lisp 语言。

## 2.软件开发的方法

有时编程语言的选择依赖于开发的方法,如果要用快速原型模型来开发,要求能快速实现原型,宜采用 4GL。如果是面向对象方法,宜采用面向对象的语言编程。面向对象的语言主要有:C++、Java

## 3.软件执行的环境

## 4.算法和数据结构的复杂性

## 5.软件开发人员的知识

## (三)程序设计风格

程序设计风格是指一个人编制程序时表现出来的特点、习惯、逻辑思路等。因此与编程风格有关的因素有:

### 1.源程序文档化

(1)标识符应按意取名。

(2)程序应加注释。

注释分序言性注释和功能性注释。

序言性注释应置于每个模块的起始部分,主要内容有:

说明每个模块的用途、功能。

说明模块的接口:调用形式、参数描述及从属模块的清单。

数据描述:重要数据的名称、用途、限制、约束及其它信息。

开发历史:设计者、审阅者姓名及日期,修改说明及日期。

功能性注释嵌入在源程序内部,说明程序段或语句的功能以及数据的状态。注意以下几点:

注释用来说明程序段,而不是每一行程序都要加注释。

使用空行或缩格或括号,以便很容易区分注释和程序。

修改程序也应修改注释。

### 2.数据说明

为了使数据定义更易于理解和维护,有以下指导原则:

(1)数据说明顺序应规范,例如按以下顺序:常量说明、类型说明、全程量说明、局部量说明。

(2)一个语句说明多个变量时,各变量名按字典序排列。

(3)对于复杂的数据结构,要加注释、说明在程序实现时的特点。

### 3.语句构造

语句构造的原则是:简单直接,不能为了追求效率而使代码复杂化。

### 4.输入和输出

在编写输入和输出程序时考虑以下原则:

(1)输入操作步骤和输入格式尽量简单。



- (2)应检查输入数据的合法性、有效性,报告必要的输入状态信息及错误信息。
- (3)输入一批数据时,使用数据或文件结束标志,而不要用计数来控制。
- (4)交互式输入时,提供可用的选择和边界值。
- (5)当程序设计语言有严格的格式要求时,应保持输入格式的一致性。
- (6)输出数据表格化、图形化。

## 5.效率

效率指处理机时间和存储空间的使用,对效率的追求明确以下几点:

- (1)效率是一个性能要求,目标在需求分析给出。
- (2)追求效率建立在不损害程序可读性或可靠性基础之上,要先使程序正确,再提高程序效率,先使程序清晰,再提高程序效率。
- (3)提高程序效率的根本途径在于选择良好的设计方法、良好的数据结构与算法,而不是靠编程时对程序语句做调整。

# 反馈测试题解

## 一、名词解释

### 1 程序设计风格

答:程序设计风格指一个人编制程序时所表现出来的特点、习惯、逻辑思路等。

### 2 程序可移植性

答:指程序从一个计算机环境移植到另一个计算机环境的容易程度。

### 3. 心理特性

答:程序设计语言经常要求程序员改变处理问题的方法,使这种处理方法适合于语言的语法规定。而程序是人设计的,人的因素在设计程序时是至关重要的。语言的心理特性指影响程序员心理的语言性能,许多这类特性是作为程序设计的结果而出现的,虽不能用定量的方法来度量,但可以认识到在语言中的表现形式:

- (1)歧义性。
- (2)简洁性。
- (3)局部性和顺序性。
- (4)传统性。

### 4. 工程特性

答:从软件工程的观点,程序设计语言特性着重考虑软件开发项目的需要,因此对程序编码有如下要求:

- (1)可移植性。它指程序从一个计算机环境移植到另一个计算机环境的容易程度。
- (2)开发工具的可利用性。有效的软件开发工具可以缩短编码时间,改进源代码的质量。
- (3)软件的可重用性。编程语言能否提供可重用的软件成分,如模块子程序可通过源代码剪贴、包含、继承等方式实现软件重用。可重用软件在组装时,从接口到算法都可能调整,需考虑额外代价。
- (4)可维护性。源程序的可维护性对复杂的软件开发项目尤其重要。易于把详细设计翻译为源程序,易于修改需要变化的源程序,这些都需要首先读懂源程序。

## 二、填空题

1 近年来,推出了许多面向对象的语言,如\_\_\_\_\_、\_\_\_\_\_等。

答:C++ Java

2 程序设计风格指一个人编制程序时所表现出来的\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_等。

答:特点 习惯 思路

3 语句构造的原则是\_\_\_\_\_,不能为了追求效率而使代码\_\_\_\_\_。

答:简单直接 复杂化

4 效率是一个\_\_\_\_\_要求,目标在\_\_\_\_\_给出。

答:性能 需求分析

5 追求效率建立在不损害\_\_\_\_\_或\_\_\_\_\_基础之上。

答:程序可读性 可靠性

6 提高程序效率的根本途径在于选择良好的\_\_\_\_\_,良好的\_\_\_\_\_,而不是靠编程时对程序语句做调整。

答:设计方法 数据结构与算法

7 Lisp 是一种\_\_\_\_\_语言,Prolog 是一种\_\_\_\_\_语言。

答:函数型 逻辑型

8 如果要完成知识库系统、专家系统、决策支持系统、推理工程、语言识别、模式识别、机器人视觉、自然语言处理等人工智能领域内的系统,应选择\_\_\_\_\_,\_\_\_\_\_。答:Prolog Lisp 语言

9 编写操作系统、编译系统等系统软件时,可选用\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_。

答:汇编语言 C 语言 Pascal 语言 Ada 语言

10 汇编语言是面向\_\_\_\_\_的,可以完成\_\_\_\_\_语言无法满足要求的特殊功能,如与外部设备之间的一些接口操作。

答:机器 高级

11 实时处理软件一般对性能的要求很高,可选用的语言有:\_\_\_\_\_、\_\_\_\_\_。

答:汇编语言 Ada 语言

12 为开发一个特定项目选择程序设计语言时,必须从\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_几方面考虑。

答:技术特性 工程特性 心理特性

13 \_\_\_\_\_指程序从一个计算机环境移植到另一个计算机环境的容易程度。

答:可移植性

14 从软件工程的观点,语言的工程特性指\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_。

答:可移植性 开发工具的可利用性 软件的可重用性 可维护性

15 程序设计语言的\_\_\_\_\_指语言的联想性。在编码过程中,由语句组合成模块,由模块组装成系统结构,并在组装过程中实现模块的高内聚,低耦合,使\_\_\_\_\_得到加强。

答:局部性 局部性

16 程序设计语言的简洁性是指人们必须记住的\_\_\_\_\_的数量。人们要掌握一种语言,需要记住的成分数量越多,简洁性越\_\_\_\_\_。

答:语言成分 差

17 编码是将\_\_\_\_\_阶段得到的\_\_\_\_\_的描述转换为基于某种计算机语言的程序,即源程

序代码。

答:详细设计 处理过程

18. 语言的心理特性指\_\_\_\_\_的语言性能,许多这类特性是作为\_\_\_\_\_的结果而出现的。

答:影响程序员心理 程序设计

19. 与编码风格有关的因素有\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_和\_\_\_\_\_。

答:源程序文档化 数据说明 语句构造 输入和输出 效率

20. 程序设计语言的工程特性主要表现在\_\_\_\_、\_\_\_\_、\_\_\_\_和\_\_\_\_\_。

答:可移植性 开发工具的可利用性 软件的可重用性 可维护性

21. 程序设计语言的心理特性在语言中的表现形式为\_\_\_\_、\_\_\_\_、\_\_\_\_和\_\_\_\_\_。

答:歧义性 简洁性 局部性和顺序性 传统性

22. 程序设计语言的特性主要有\_\_\_\_、\_\_\_\_和\_\_\_\_\_。

答:心理特性 工程特性 技术特性

23. 项目的应用领域一般有\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_和\_\_\_\_\_几种类型。

答:科学与工程计算 数据处理与数据库应用 实时处理 系统软件 人工智能

24. 通常考虑选用语言的因素有\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_和\_\_\_\_\_。

答:项目的应用领域 软件开发的方法 软件执行环境 算法与数据结构的复杂性 软件开发人员的知识

25. 源程序中加注释是帮助理解程序的重要手段,注释分为\_\_\_\_\_两类。

答:序言性注释和功能性注释

26. 为了提高程序的易读性,同时减少错误,提高软件开发效率,编码时注意养成良好的\_\_\_\_\_。

答:程序设计风格

27. 软件需求分析之后,软件的设计、编码、测试与语言的特性有很大的关系,这个特性主要是语言的\_\_\_\_\_特性。

答:技术

28. 语言的心理特性在语言中的表现形式有:\_\_\_\_\_性、\_\_\_\_\_性、\_\_\_\_\_性和\_\_\_\_\_性、\_\_\_\_\_性。

答:歧义 简洁 局部 顺序 传统

29. 程序语言的共同特性包括:\_\_\_\_\_特性、\_\_\_\_\_特性和\_\_\_\_\_特性。

答:心理 工程 技术

30. 从软件工程的观点,程序设计语言的特性应着重考虑软件开发项目的需要,因此对程序编码的要求包括:可\_\_\_\_\_性、开发工具的可\_\_\_\_\_性、软件的可\_\_\_\_\_性、可\_\_\_\_\_性。

答:移植 利用 重用 维护

### 三、选择题

1. 在编制程序时,应采纳的原则之一是 ( )

A .不限制 goto 语句的使用

B .减少或取消注解行

C .程序越短越好

D .程序结构应有助于读者理解

答:D

2. 一个程序如果把它作为一个整体,它也是只有一个入口、一个出口的单个顺序结构,这是一种 ( )

A .结构程序

B 组合的过程

C .自顶向下设计

D .分解过程

答:B

3 程序控制一般分为( )、分支、循环三种基本结构。

- A .分块                      B .顺序                      C .循环                      D .分支

答:B

4 将非结构化程序转换为结构程序的过程中,下列哪种方法不适用于具有循环结构的程序 ( )

- A .重复编码法                      B .状态变量法  
C .布尔标记法                      D .以上全部

答:A

5. 源程序文档化要求在每个模块之前加序言性注释。该注释内容不应有 ( )

- A. 模块的功能                      B. 语句的功能                      C. 模块的接口                      D. 开发历史

答:B

6. 提高程序效率的根本途径并非在于 ( )

- A. 选择良好的设计方法                      B. 选择良好的数据结构  
C. 选择良好的算法                      D. 对程序语句作调整

答:D

7. 与选择编程语言无关的因素是 ( )

- A. 软件开发的方法                      B. 软件执行的环境  
C. 程序设计风格                      D. 软件开发人员的知识

答:C

8. 在结构化程序设计思想提出之前,在程序设计中曾强调程序的效率,现在人们更重视程序的 ( )

- A. 技巧性                      B. 保密性                      C. 一致性                      D. 可理解性

答:D

9. 不适合作为数据处理的语言是 ( )

- A. PROLOG                      B. C                      C. 4GL                      D. SQL

答:A

10. 为了提高易读性,源程序内部应加功能性注释,用于说明 ( )

- A. 模块总的功能                      B. 程序段或语句的功能  
C. 模块参数的用途                      D. 数据的用途

答:B

11. 项目的应用领域是选择编程语言关键的因素之一,不适合作为项目应用领域的类型是 ( )

- A. 系统软件                      B. 数据处理与数据库应用  
C. 实时处理                      D. UNIX 操作系统

答:D

12. 以下说法错误的是 ( )

- A. 适用于实时处理的语言有:汇编语言、Ada 语言  
B. 编写系统软件时,可选用汇编语言、C 语言、Pascal 语言和 Ada 语言  
C. 如果要完成人工智能领域内的系统,应选择 Prolog、Lisp、C 语言和 Ada 语言  
D. 适用于数据处理与数据库应用的语言有:Cobol、SQL、4GL 语言

答:C

13. 以下说法正确的是 ( )

- A. FORTRAN 语言具有汇编语言的某些特性,使程序运行效率高

- B. Pascal 语言是世界上第一个被正式推广应用的计算机语言
- C. C 语言是第一个体现结构化编程思想的语言
- D. PL/ 1 能够适用于多种不同的应用领域,因太庞大,难以推广使用

答:D

14. 以下说法正确的是 ( )

- A. FORTRAN、Cobol 是第三代语言
- B. Pascal、C 是第二代语言
- C. 4GL 是第四代语言
- D. FORTRAN、C 是第三代语言

答:C

15. 程序设计语言的心理特性在语言中表现不应包括 ( )

- A. 歧义性
- B. 简洁性
- C. 保密性
- D. 传统性

答:C

16. 程序设计语言的工程特性其中之一表现在 ( )

- A. 软件的可重用性
- B. 数据结构的描述性
- C. 抽象类型的描述性
- D. 数据库的易操作性

答:A

17. 程序设计语言的技术特性不应包括 ( )

- A. 数据结构的描述性
- B. 抽象类型的描述性
- C. 数据库的易操作性
- D. 软件的可移植性

答:D

18. 在结构化程序思想提出之前,在程序设计中曾强调程序的( ),现在人们更重视程序的 ( )

- A. 效率
- B. 安全性
- C. 一致性
- D. 可理解性

答:A D

## 四、简答题

1 在项目开发时,选择程序设计语言通常考虑哪些因素?

答:为开发一个特定项目选择程序设计语言时,必须从技术特性、工程特性和心理特性几方面考虑。在选择语言时,首先从问题入手,确定它的要求是什么,以及这些要求的相对重要性。由于一种语言不可能同时满足它的各种需求,所以要对各种要求进行权衡,比较各种可用语言的适用程度,最后选择认为是最适用的语言。通常,考虑选用语言的因素有:

(1)项目的应用领域。这是选择语言的关键因素,项目应用领域一般为以下几种类型: 科学工程计算。需要大量的标准库函数,以便处理复杂的数值计算,可供选用的语言有:FORTRAN 语言、Pascal 语言、C 语言、PL/ 1。 数据处理与数据库应用。可供选用的语言有: Cobol 语言、SQL、第 4 代语言(4GL)。 实时处理。实时处理软件一般对性能的要求很高,可选用的语言有:汇编语言、Ada 语言。系统软件。编写操作系统、编译系统等系统软件时,可选用汇编语言、C 语言、Pascal 语言和 Ada 语言。人工智能。如果要完成知识库系统、专家系统、决策支持系统、推理工程、语言识别、模式识别、机器人视觉、自然语言处理等人工智能领域内的系统,应选择 Prolog、Lisp 语言。

(2)软件开发的方法。有时编程语言的选择依赖于开发的方法,如果要用快速原型模型来开发,要求能快速实现原型,因此宜采用 4GL。如果是面向对象方法,宜采用面向对象的语言编程。近年来,推出了许多面向对象的语言,主要有 C++ 和 Java 等。

(3)软件执行的环境。良好的编程环境不但有效提高软件生产率,同时能减少错误,有效提高软件

质量。

(4)算法和数据结构的复杂性。科学计算、实时处理和人工智能领域中的问题算法较复杂,而数据处理、数据库应用、系统软件领域内的问题,数据结构比较复杂,因此选择语言时可考虑是否有完成复杂算法的能力,或者有构造复杂数据结构的能力。

(5)软件开发人员的知识。有时编程语言的选择与软件开发人员的知识水平及心理因素有关,新的语言虽然有吸引力,但软件开发人员若熟悉某种语言,而且有类似项目的开发经验,往往愿选择原有的语言。开发人员应仔细地分析软件项目的类型,敢于学习新知识,掌握新技术。

2.为了具有良好的设计风格,应注意哪些方面的问题?

答:为了具有良好的设计风格,应注意下几个方面的问题:

(1)源程序的文档化。标识符应按意取名。程序应加注释。注释说明了程序的功能,特别在维护阶段,对理解程序提供了明确指导。注释分序言性注释和功能性注释。

序言性注释应置于每个模块的起始部分,主要内容有:

- ( )说明每个模块的用途、功能。
- ( )说明模块的接口:调用形式、多数描述及从属模块的清单。
- ( )数据描述:重要数据的名称、用途、限制、约束及其他信息。
- ( )开发历史:设计者、审阅者姓名及日期、修改说明及日期。

功能性注释嵌入在源程序内部,说明程序段或语句的功能以及数据的状态。注意以下几点:

- ( )注释用来说明程序段,而不是每一行程序都要加注释。
- ( )使用空行或缩格或括号,以便很容易区分注释和程序。
- ( )修改程序也应修改注释。

(2)数据说明。为了使数据定义更易于理解和维护,有以下指导原则:数据说明顺序应规范,使数据的属性更易于查找,从而有利于测试、纠错与维护。一个语句说明多个变量时,各变量名按字典顺序排列。对于复杂的数据结构,要加注释,说明在程序实现时的特点。

(3)语句构造。语句构造的原则是简单直接,不能为了追求效率而使代码复杂化。为了便于阅读和理解,不要一行多个语句。不同层次的语句采用缩进形式,使程序的逻辑结构和功能特征更加清晰。要避免复杂的判定条件,避免多重的循环嵌套。表达式中使用括号以提高运算次序的清晰度等等。

(4)输入和输出。在编写输入和输出程序时考虑以下原则:输入操作步骤和输入格式尽量简单。应检查输入数据的合法性、有效性,报告必要的输入状态信息及错误信息。输入一批数据时,使用数据或文件结束标志,而不要用计数来控制。交互式输入时,提供可用的选择和边界值。当程序设计语言有严格的格式要求时,应保持输入格式的一致性。输出数据表格化、图形化。

(5)效率。效率指处理机时间和存储空间的使用,对效率的追求应明确以下几点:效率是一个性能要求,目标在需求分析给出。追求效率建立在不损害程序可读性或可靠性基础之上,要先使程序正确,再提高程序效率,先使程序清晰,再提高程序效率。提高程序效率的根本途径在于选择良好的设计方法、良好的数据结构与算法,而不是靠编程时对程序语句做调整。

总之,在编码阶段,要善于积累编程经验,培养和学习良好的编程风格,使编出的程序清晰易懂,易于测试与维护,从而提高软件的质量。

3.举例说明各种程序设计语言的特点及适用范围。

答:FORTRAN语言是世界上第一个被正式推广应用的计算机语言,产生于1954年,经过FORTRAN0到FORTRANIV,又相继扩展为FORTRAN77、FORTRAN90,通过几个版本不断的更新,它不仅面向科学计算,数据处理能力也极强。

Pascal语言产生于60年代末,具有很强的数据和过程结构化的能力,它是第一个体现结构化编程思想的语言,由于它语言简明,数据类型丰富,程序结构严谨,许多算法都用类Pascal来概括。用Pascal语

言写程序,也有助于培养良好的编程风格。

C 语言产生于 70 年代初,最初用于描述 UNIX 操作系统及其上层软件,后来发展成具有很强功能的语言,支持复杂的数据结构,可大量运用指针,具有丰富灵活的操作运算符及数据处理操作符。此外还具有汇编语言的某些特性,使程序运行效率高。

Cobol 语言产生于 50 年代末,是广泛用于商业数据处理的语言,它具有极强的数据定义能力,程序说明与硬件环境说明分开,数据描述与算法描述分开,结构严谨层次分明,说明采用类英语的语法结构,可读性强。

SQL 最初为 IBM 公司开发的数据库查询语言,目前不同的软件开发公司有了不同的扩充版本,如 80 年代后期我国引入 Informix SQL、Microsoft SQL 可以方便地对数据库进行存取管理。

Ada 语言是美国国防部出资开发的,主要用于适时、并发和嵌入系统的语言,Ada 语言是在 Pascal 基础上开发出来的,但其功能更强、更复杂。它提供了一组丰富的实时特性,包括多任务处理、中断处理、任务间同步与通信等等,它还提供了许多程序包供程序员选择。通过修订,已成为安全、高效、灵活的面向对象的编程语言。

Lisp 是一种函数型语言,产生于 60 年代初,它特别适用于组合问题中的符号运算和表处理,因此用于定理证明、树的搜索和其他问题的求解。近年来 Lisp 广泛应用于专家系统的开发,对于定义知识库系统中的事实、规则和相应的推理相对要容易一些。

Prolog 是一种逻辑型语言,产生于 70 年代初,它提供了支持知识表示的特性,每一个程序由一组表示事实、规则和推理的子句组成,比较接近自然语言,符合人的思维方式。

以上讨论的语言,一般适用于相应的应用领域,但要根据具体情况灵活掌握。有的语言功能强,适用的范围较广,但比较庞大。

#### 4 第 4 代语言(4GL)有哪些主要特征?

答:第 4 代语言(4GL)的主要特征是:

- (1)友好的用户界面。操作简单,使非计算机专业人员也能方便地使用它。
- (2)兼有过程性和非过程性双重特性。非过程性指将语言的抽象层次又提高到一个新的高度,只需告诉计算机“做什么”,而不必描述“怎么做”,“怎么做”的工作由与语言系统运用它的专门领域的知识来填充过程细节。
- (3)高效的程序代码。能缩短开发周期,并减少维护的代价。
- (4)完备的数据库。在 4GL 中实现数据库功能,不再把 DBMS(数据库管理系统)看成是语言以外的成分。
- (5)应用程序生成器。提供一些常用的程序来完成文件维护、屏幕管理、报表生成、查询等任务,从而有效地提高了软件生产率。

#### 5. 试述语言的三个特性及选择因素?

答:语言的心理特性指影响程序设计员心理的语言性能;语言的工程特性指根据开发软件项目的需要,总体上考虑的软件特性;而语言的技术特性指具体实现软件系统所提供的语言特性。一种语言不可能同时满足项目的各种要求,因此要根据项目的应用领域、软件的开发方法、软件的执行环境、算法及数据结构的复杂性、软件开发人员的知识等诸因素选择合适的编程语言。

#### 6. 在编写输入和输出程序时考虑哪些原则?

答:(1)输入操作步骤和输入格式尽量简单。

(2)应检查输入数据的合法性、有效性,报告必要的输入状态信息及错误信息。

(3)交互式输入时,提供可用的选择和边界值。

(4)当程序设计语言有严格的格式要求时,应保持输入格式的一致性。

(5)输出数据表格化、图形化。

输入、输出风格还受其他因素的影响,如输入、输出设备,用户经验及通信环境等。

#### 7. 对效率的追求应明确哪几点?

答:(1)效率是一个性能要求,目标在需求分析给出。

(2)追求效率建立在不损害程序可读性或可靠性基础之上,要先使程序正确,再提高程序效率,先使程序清晰,再提高程序效率。

(3)提高程序效率的根本途径在于选择良好的设计方法、良好的数据结构与算法,而不是靠编程时对程序语句做调整。

总之,在编码阶段,要善于积累编程经验,培养和学习良好的编程风格,使编出的程序清晰易懂,易于测试与维护,从而提高软件的质量。

#### 8. 语句构造的原则是什么?

答:语句构造的原则是:简单直接,不能为了追求效率而使代码复杂化。为了便于阅读和理解,不要一行多个语句。不同层次的语句采用缩进形式,使程序的逻辑结构和功能特征更加清晰。要避免复杂的判定条件,避免多重的循环嵌套。表达式中使用括号以提高运算次序的清晰度等等。

#### 9. 数据说明有哪些指导原则?

答:为了使数据定义更易于理解和维护,有以下指导原则:

(1)数据说明顺序应规范,使数据的属性更易于查找,从而有利于测试、纠错与维护。例如按以下顺序:常量说明、类型说明、全程量说明、局部量说明。

(2)一个语句说明多个变量时,各变量名按字典序排列。

(3)对于复杂的数据结构,要加注释,说明在程序实现时的特点。

#### 10. 什么是注释?有哪些内容?

答:程序应加注释。注释是程序员与日后读者之间通信的重要工具,用自然语言或伪码描述。它说明了程序的功能,特别在维护阶段,对理解程序提供了明确指导。注释分序言性注释和功能性注释。

序言性注释应置于每个模块的起始部分,主要内容有:

(1)说明每个模块的用途、功能。

(2)说明模块的接口:调用形式、参数描述及从属模块的清单。

(3)数据描述:重要数据的名称、用途、限制、约束及其他信息。

(4)开发历史:设计者、审阅者姓名及日期,修改说明及日期。

功能性注释嵌入在源程序内部,说明程序段或语句的功能以及数据的状态。注意以下几点:

(1)注释用来说明程序段,而不是每一行程序都要加注释。

(2)使用空行或缩格或括号,以便很容易区分注释和程序。

(3)修改程序也应修改注释。

#### 11. 程序设计风格是什么?

答:程序设计风格一般指人们编程的习惯特点。培养良好的设计风格可以使程序清晰易读,减少错误,不但能提高软件开发效率,还为以后的软件维护奠定了良好的基础。主要从以下方面注意编程风格:

(1)程序加注释:注释是编写者与读者之间通信的手段之一,应该在每一个模块的开头加序言性注释,在程序内部加功能性注释。

(2)变量、数据按意取名,增加可理解性;复杂的数据结构也应加注释。

(3)语句简单直接,内层要缩进,使用空格、空行以提高程序的清晰度。

(4)输入输出提示化、表格化、图形化。



## 第七章 软件测试

软件测试是保证软件可靠性的主要手段之一。本章总的要求是:掌握测试阶段的任务、测试方法及测试步骤。

设计测试方案是关键技术问题,其目标是选用最少的、高效的测试数据以发现尽可能多的错误。

深刻理解白盒、黑盒测试技术。深刻理解测试过程中单元测试、集成测试、确认测试的任务及采用的方法。

掌握调试程序的方法。

要求熟练掌握的技能是:能针对某一问题采用白盒法或黑盒法进行测试用例的设计。

### 考试要求

#### 1. 软件测试的目的及原则

软件测试的目的,要求达到识记层次。

#### 2. 测试方法

白盒法、黑盒法,要求达到领会层次。

#### 3. 测试用例的设计

逻辑覆盖、等价类划分、边界值分析、错误推测、因果图,要求达到识记层次。

逻辑覆盖中各种覆盖之间的区别,要求达到领会层次。

用白盒法、黑盒法设计测试用例,要求达到简单应用层次。

#### 4. 测试过程

单元测试、集成测试、确认测试、渐增式、非渐增式,要求达到识记层次。

单元测试的内容及方法、集成测试的方法、渐增式及非渐增式测试的区别、自顶向下及自底向上结合模块的步骤、三种测试与软件开发各阶段之间的关系,要求达到领会层次。

#### 5. 调试

测试,要求达到识记层次。

归纳法、演绎法,要求达到领会层次。

### 知识重点

#### (一) 软件测试的目的和原则

软件测试是为了发现错误而执行程序的过程。一个好的测试用例能够发现至今尚未发现的错误。一个成功的测试是发现了至今尚未发现的错误的测试。

因此,测试阶段的基本任务应该是根据软件开发各阶段的文档资料和程序的内部结构,精心设计一组“高产”的测试用例,利用这些实例执行程序,找出软件中潜在的各种错误和缺陷。

在软件测试中,应注意以下指导原则:

- 1.测试用例应由输入数据和预期的输出数据两部分组成。
- 2.测试用例不仅选用合理的输入数据,还要选择不合理的输入数据。
- 3.除了检查程序是否做了它应该做的事,还应该检查程序是否做了它不应该做的事。
- 4.应制定测试计划并严格执行,排除随意性。
- 5.长期保留测试用例。
- 6.对发现错误较多的程序段,应进行更深入的测试。
- 7.程序员避免测试自己的程序。

## (二)测试方法

软件测试方法一般分为两大类:动态测试方法与静态测试方法,而动态测试方法中又根据测试用例的设计方法不同,分为黑盒测试与白盒测试两类。

### 1.静态测试

静态测试指被测试程序不在机器上运行,而是采用人工检测和计算辅助静态分析的手段对程序进行检测。

### 2.动态测试

动态测试指通过运行程序发现错误。对软件产品进行动态测试时,一般有两种方法,分别称为黑盒测试法和白盒测试法。

## (三)白盒技术

由于白盒测试是结构测试,所以被测对象基本上是源程序,以程序的内部逻辑为基础设计测试用例。

### 1.逻辑覆盖

追求程序内部的逻辑覆盖程度,测试时,要设计使覆盖程度较高的或覆盖最有代表性的路径的测试用例。常用的覆盖技术有以下几种。

(1)语句覆盖。语句覆盖是指设计足够的测试用例,使被测程序中每个语句至少执行一次。

(2)判定覆盖。判定覆盖是指设计足够的测试用例,使得被测程序中每个判定表达式至少获得一次“真”和“假”值,从而使程序的每一个分支至少都通过一次。

(3)条件覆盖。条件覆盖指设计足够的测试用例,使得判定表达式中每个条件的各种可能的值至少出现一次。

(4)判定/条件覆盖。该覆盖标准指设计足够的测试用例,使得判定表达式中的每个条件的所有可能取值至少出现一次,并使每个判定表达式所有可能的结果也至少出现一次。

(5)条件组合覆盖。条件组合覆盖是比较强的覆盖标准,它是指设计足够的测试用

例,使得每个判定表达式中条件的各种可能的值的组合都至少出现一次。

(6)路径覆盖。路径覆盖是指设计足够的测试用例,覆盖被测程序中所有可能的路径。

## 2.循环覆盖

循环也是程序的主要结构,要覆盖含有循环结构的所有路径是不可能的,但可通过限制循环次数来测试。

## 3.基本路径测试

基本路径测试是在程序控制流程图的基础上,通过分析控制构造的环路复杂性,导出基本路径集合,从而设计测试用例,保证这些路径至少通过一次。

基本路径测试的步骤为:

(1)以详细设计或源程序为基础,导出控制流程图的拓扑结构——示程序图。

(2)计算程序图  $G$  的环路复杂性  $V(G)$ 。

(3)确定只包含独立路径的基本路径集。环路复杂性可导出程序基本路径集中的独立路径条数。独立路径是指包括一组以前没有处理的语句或条件的一条路径。从程序图来看,一条独立路径是至少包含有一条在其他独立路径中未有过的边的路径。

(4)设计测试用例,确保基本路径集合中每条路径的执行。

## (四)黑盒技术

用黑盒技术设计测试用例的方法有四种。

### 1.等价类划分

等价类划分是将输入数据域按有效的或无效的(也称合理的或不合理的)划分成若干个等价类,测试每个等价类的代表值就等于对该类其他值的测试。

用等价类划分的方法设计测试用例的步骤为:

(1)划分等价类。

(2)确定测试用例。根据已划分的等价类,按以下步骤设计测试用例: 为每一个等价类编号。 设计一个测试用例,使其尽可能多地覆盖尚未被覆盖过的合理等价类。重复这步,直到所有合理等价类被测试用例覆盖。 设计一个测试用例,使其只覆盖一个不合理等价类。重复这一步,直到所有不合理等价类被覆盖。

### 2.边界值分析

实践经验表明,程序往往在处理边界情况时发生错误。边界情况指输入等价类和输出等价类边界上的情况,因此检查边界情况的测试用例是比较高效的,可以查出更多的错误。

### 3.错误推测

在测试程序时,人们可能根据经验或直觉推测程序中可能存在的各种错误,从而有针对性地编写检查这些错误的测试用例,这就是错误推测法。

### 4.因果图

因果图能有效地检测输入条件的各种组合可能会引起的错误。因果图的基本原理是通过画因果图,把用自然语言描述的功能说明转换为判定表,最后为判定表的每一列设计

一个测试用例。

## 5.综合策略

前面介绍的软件测试方法,各有所长。每种方法都能设计出一组有用例子,用这组例子容易发现某种类型的错误,但可能不易发现另一种类型的错误。因此在实际测试中,联合使用各种测试方法,形成综合策略,通常先用黑盒法设计基本的测试用例,再用白盒法补充一些必要的测试用例。

## (五)软件测试过程

### 1.软件测试的步骤

软件测试一般要经过以下四步测试:单元测试、集成测试、确认测试和系统测试。

### 2.单元测试

单元测试主要针对模块的五个基本特征进行测试:模块接口;局部数据结构;重要的执行路径;错误处理;边界条件。

由于被测试的模块往往不是独立的程序,它处于整个软件结构的某一层位置上,被其他模块调用或调用其他模块,其本身不能进行单独运行,因此在单元测试时,需要为被测试模块设计驱动模块和桩模块。

### 3.集成测试

集成测试是指在单元测试的基础上,将所有模块按照设计要求组装成一个完整的系统进行的测试,故也称组装测试或联合测试。

集成测试的方法主要有两种:非渐增式测试和渐增式测试。

(1)非渐增式测试:首先对每个模块分别进行单元测试,然后再把所有的模块按设计要求组装在一起进行测试。

(2)渐增式测试:逐个把未经过测试的模块组装到已经过测试的模块上去,进行集成测试。每加入一个新模块进行一次集成测试,重复此过程直至程序组装完毕。渐增式测试有两种不同的组装模块的方法:自顶向下结合和自底向上结合。

### 4.确认测试

确认测试又称有效性测试。它的任务是检查软件的功能与性能是否与需求规格说明书中确定的指标相符合。因而需求规格说明是确认测试的基础。

确认测试阶段有两项工作:进行确认测试与软件配置审查。

## (六)调试

### 1.调试的目的

软件测试的目的是尽可能地发现程序中的错误,而调试则是在进行了成功的测试之后才开始的工作。调试的目的是确定错误的原因和位置,并改正错误,因此调试也称为纠错。

### 2.调试技术

(1)简单的调试方法。在程序中插入打印语句或运行部分程序。

(2)归纳法调试。归纳法调试从测试结果发现的线索(错误迹象、征兆)入手,分析它

们之间的联系,导出错误原因的假设,然后再证明或否定这个假设。

(3)演绎法调试。演绎法调试是列出所有可能的错误原因的假设,然后利用测试数据排除不适当的假设,最后再测试数据验证余下的假设确实是出错的原因。

(4)回溯法调试。该方法从程序产生错误的地方出发,人工沿程序的逻辑路径反向搜索,直到找到错误的原因为止。

## 反馈测试题解

### 一、名词解释

#### 1 驱动模块

答:驱动模块是用来模拟被测模块的上级调用模块的模块,功能要比真正的上级模块简单得多,它只完成接受测试数据,以上级模块调用被测模块的格式驱动被测模块,接收被测模块的测试结果并输出。

#### 2 桩模块

答:桩模块用来代替被测试模块所调用的模块。它的作用是返回被测模块所需的信息。

#### 3 单元测试

答:单元测试指对源程序中每一个程序单元进行测试,检查各个模块是否正确实现规定的功能,从而发现模块在编码中或算法中的错误。

#### 4 确认测试

答:又称有效性测试。是为了检查软件的功能与性能是否与需求规格说明书中确定的指标相符合所进行的测试。

#### 5 调试

答:调试是为了确定错误的原因和位置,并改正错误所进行的工作,因此调试也称为纠错。

#### 6 计算机辅助静态分析

答:利用静态分析工具对被测试程序进行特性分析,从程序中提取一些信息,以便检查程序逻辑的各种缺陷和可疑的程序构造。如用错的局部量和全程量、不匹配参数、不适当的循环嵌套和分支嵌套、潜在的死循环、不会执行到的代码等等。

#### 7 语句覆盖

答:指设计足够的测试用例,使被测程序中每个语句至少执行一次。语句覆盖是比较弱的覆盖标准。

#### 8 判定覆盖

答:指设计足够的测试用例,使得被测程序中每个判定表达式至少获得一次“真”值和“假”值,从而使程序的每一个分支至少都通过一次,因此判定覆盖也称分支覆盖。

#### 9 条件覆盖

答:指设计足够的测试用例,使得判定表达式中每个条件的各种可能的值至少出现一次。满足条件覆盖并不一定满足判定覆盖。

#### 10 判定/条件覆盖

答:指设计足够的测试用例,使得判定表达式中的每个条件的所有可能取值至少出现一次,并使每个判定表达式所有可能的结果也至少出现一次。

#### 11 条件组合覆盖

答:指设计足够的测试用例,使得每个判定表达式中条件的可种可能的值的组合都至少出现一次:条件组合覆盖是比较强的覆盖标准。

#### 12. 路径覆盖

答:指设计足够的测试用例,覆盖被测程序中所有可能的路径。

#### 13. 基本路径测试

答:基本路径测试是在程序控制流程图的基础上,通过分析控制构造的环路复杂性,导出基本路径集合,从而设计测试用例,保证这些路径至少通过一次。

#### 14. 集成模块

答:集成模块是指在单元测试的基础上,将所有模块按照设计要求组装成一个完整的系统进行的测试,故也称组装测试或联合测试。

#### 15. 非渐增式测试

答:非渐增式测试首先对每个模块分别进行单元测试,然后再把所有的模块按设计要求组装在一起进行测试。

#### 16. 渐增式测试

答:渐增式测试逐个把未经过测试的模块组装到已经过测试的模块上去,进行集成测试。每加入一个新模块进行一次集成测试,重复此过程直到程序组装完毕。

#### 17. 白盒法

答:该方法把测试对象看作一个打开的盒子,测试人员须了解程序的内部结构和处理过程,以检查处理过程的细节为基础,对程序中尽可能多的逻辑路径进行测试,检验内部控制结构和数据结构是否有错,实际的运行状态与预期的状态是否一致。

#### 18. 黑盒法

答:该方法把被测试对象看成一个黑盒子,测试人员完全不考虑程序的内部结构和处理过程,只在软件的接口处进行测试,依据需求规格说明书,检查程序是否满足功能要求。因此,黑盒测试又称为功能测试或数据驱动测试。

#### 19. 人工检测

答:人工检测是不依靠计算机而是靠人工审查程序或评审软件。人工审查程序偏重于编码质量的检验,对软件审查除了审查编码还要对各阶段的软件产品进行检验。人工检测可以发现计算机不易发现的错误,据统计,能有效地发现 30% ~ 70% 的逻辑设计和错误,可以减少系统测试的总工作量。

#### 20. 静态测试

答:静态测试指被测试程序不在机器上运行,而是采用人工检测和计算机辅助静态分析的手段对程序进行检测。

#### 21. 动态测试

答:动态测试指通过运行程序发现错误。一般意义上的测试大多是指动态测试。

## 二、填空题

1 判定覆盖指设计足够的测试用例,使得被测程序中每个\_\_\_\_\_至少获得一次\_\_\_\_\_和\_\_\_\_\_值,从而使程序的每一个\_\_\_\_\_至少都通过一次,因此判定覆盖也称\_\_\_\_\_。

答:判定表达式 “真” “假” 分支 分支覆盖

2 语句覆盖是指设计足够的\_\_\_\_\_,使被测程序中\_\_\_\_\_至少执行一次。语句覆盖是比较\_\_\_\_\_的覆盖标准。

答:测试用例 每个语句 弱

3 静态测试指\_\_\_\_\_不在机器上运行,而是采用\_\_\_\_\_和\_\_\_\_\_的手段对程序进行检测。

答:被测试程序 人工检测 计算机辅助静态分析

4 黑盒测试依据\_\_\_\_\_,检查程序是否满足\_\_\_\_\_.因此,黑盒测试又称为\_\_\_\_\_或\_\_\_\_\_。

答:需求规格说明书 功能要求 功能测试 数据驱动测试

5 黑盒法把被测试对象看成一个\_\_\_\_\_,测试人员完全不考虑程序的\_\_\_\_\_和\_\_\_\_\_。只在软件的\_\_\_\_\_处进行测试。

答:黑盒子 内部结构 处理过程 接口

6 动态测试指通过\_\_\_\_\_发现错误。对软件产品进行动态测试时,使用\_\_\_\_\_法和\_\_\_\_\_法。

答:运行程序 黑盒测试 白盒测试

7 黑盒测试是功能测试,因此设计测试用例时,需要研究\_\_\_\_\_和\_\_\_\_\_中有关程序功能或输入、输出之间的关系等信息,从而与测试后的结果进行分析比较。

答:需求规格说明 概要设计说明

8 独立路径是指包括一组以前没有处理的\_\_\_\_\_的一条路径。从程序图来看,一条独立路径是至少包含有一条\_\_\_\_\_的边的路径。

答:语句或条件 在其他独立路径中未有过

9 基本路径测试是在\_\_\_\_\_的基础上,通过分析控制构造的\_\_\_\_\_,导出\_\_\_\_\_集合,从而设计测试用例,保证这些路径至少通过一次。

答:程序控制流程图 环路复杂性 基本路径

10 语句覆盖发现错误能力最\_\_\_\_\_。判定覆盖包含了\_\_\_\_\_,但它可能会使一些\_\_\_\_\_得不到测试。

答:弱 语句覆盖 条件

11 判定/条件覆盖标准指设计足够的测试用例,使得判定表达式中的\_\_\_\_\_至少出现一次,并使每个判定表达式\_\_\_\_\_也至少出现一次。

答:每个条件的所有可能取值 所有可能的结果

12 条件覆盖指设计足够的测试用例,使得\_\_\_\_\_中每个条件的\_\_\_\_\_的值至少出现一次。

答:判定表达式 各种可能

13 因果图能有效地检测输入条件的\_\_\_\_\_可能会引起的错误。因果图的基本原理是通过\_\_\_\_\_,把用自然语言描述的功能说明转换为\_\_\_\_\_,最后为\_\_\_\_\_设计一个测试用例。

答:各种组合 画因果图 判定表 判定表的每一列

14 在测试程序时,人们可能根据\_\_\_\_\_或\_\_\_\_\_推测程序中可能存在的各种错误,从而有针对性地编写检查这些错误的测试用例,这就是错误推测法。

答:经验 直觉

15 使用边界值分析方法设计测试用例时一般与\_\_\_\_\_结合起来。但它不是从一个等价类中任选一个例子作为代表,而是将测试边界情况作为重点目标,选取\_\_\_\_\_,\_\_\_\_\_或\_\_\_\_\_边界值的测试数据。

答:等价类划分 正好等于 刚刚大于 刚刚小于

16 根据已划分的等价类,按以下步骤设计测试用例: 为每一个等价类编号。 设计一个测试用例,使其\_\_\_\_\_覆盖尚未被覆盖过的合理等价类。重复这步,直到所有合理等价类被测试用例覆

盖。 设计一个测试用例,使其\_\_\_\_\_。重复这一步,直到所有不合理等价类被覆盖。

答:尽可能多地 只覆盖一个不合理等价类

17.用等价类划分的方法设计测试用例的步骤为:\_\_\_\_\_、\_\_\_\_\_。

答:划分等价类 确定测试用例

18.确认测试一般是在模拟环境下运用\_\_\_\_\_方法,由\_\_\_\_\_和\_\_\_\_\_参加的测试。

答:黑盒测试 专门测试人员 用户

19.驱动模块的作用是用来模拟被测模块的\_\_\_\_\_,它只完成\_\_\_\_\_,以上级模块调用被测模块的格式\_\_\_\_\_被测模块,接收被测模块的\_\_\_\_\_并输出。

答:上级调用模块 接受测试数据 驱动 测试结果

20.集成测试是指在\_\_\_\_\_的基础上,将所有模块按照设计要求\_\_\_\_\_成一个完整的系统进行的测试,故也称\_\_\_\_\_或\_\_\_\_\_。

答:单元测试 组装 组装测试 联合测试

21.桩模块用来代替被测试模块\_\_\_\_\_。它的作用是\_\_\_\_\_的信息。

答:所调用的模块 返回被测模块所需

22.各模块经过单元测试后,将各模块组装起来进行\_\_\_\_\_,以检查与设计相关的\_\_\_\_\_的有关问题。

答:集成测试 软件体系结构

23.单元测试指对源程序中每一个\_\_\_\_\_进行测试,检查各个模块是否正确实现\_\_\_\_\_,从而发现模块在\_\_\_\_\_的错误。该阶段涉及\_\_\_\_\_的文档。

答:程序单元 规定的功能 编码中或算法中 编码和详细设计

24.软件产品在交付使用之前要经过哪些测试呢?一般要经过以下四步测试:\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_和\_\_\_\_\_。

答:单元测试 集成测试 确认测试 系统测试

25.归纳法调试的具体步骤如下:\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_。

答:收集有关数据 组织数据 提出假设 证明假设

26.归纳法调试从测试结果发现的\_\_\_\_\_入手,分析\_\_\_\_\_,导出\_\_\_\_\_,然后再\_\_\_\_\_。

答:线索(错误迹象、征兆) 它们之间的联系 错误原因的假设 证明或否定这个假设

27.软件测试的目的是\_\_\_\_\_,调试的目的是\_\_\_\_\_,并\_\_\_\_\_,因此调试也称为\_\_\_\_\_。

答:尽可能多地发现程序中的错误 确定错误的原因和位置 改正错误 纠错

28.软件配置审查的任务是检查软件的\_\_\_\_\_的\_\_\_\_\_,如发现遗漏和错误,应补充和改正。

答:所有文档资料 完整性 正确性

29.渐增式测试有以下两种不同的组装模块的方法:\_\_\_\_\_,\_\_\_\_\_。

答:自顶向下结合 自底向上结合

30.自顶向下结合的渐增式测试法,在组合模块时有两种组合策略:\_\_\_\_\_和\_\_\_\_\_。

答:深度优先策略 宽度优先策略

31.确认测试又称\_\_\_\_\_。它的任务是检查软件的\_\_\_\_\_与\_\_\_\_\_是否与\_\_\_\_\_中确定的指标相符合。

答:有效性测试 功能 性能 需求规格说明书

32.用黑盒技术设计测试用例的方法有\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_和\_\_\_\_\_。



答:等价类划分 边界值分析 错误推测 因果图

33. 在设计测试用例时,追求程序逻辑覆盖程度的几种常用覆盖技术为\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_和\_\_\_\_。

答:语句覆盖 判定覆盖 条件覆盖 判定/条件覆盖 条件组织覆盖 路径覆盖

34. 软件测试是为了\_\_\_\_而执行程序的过程。

答:发现错误

35. 要覆盖含有循环结构的所有路径是不可能的,一般通过限制\_\_\_\_来测试。

答:循环次数

36. 被测试程序不在机器上运行,而是采用人工检测和计算机辅助分析检测的手段称为\_\_\_\_测试,运行被测程序的方法称为\_\_\_\_测试。

答:静态 动态

37. 用等价类划分法设计测试用例时,如果被测程序的某个输入条件规定了取值范围,则可确定一个合理的等价类和\_\_\_\_。

答:两个不合理的等价类

38. 动态测试中,主要测试软件功能的方法称为\_\_\_\_法。

答:黑盒法

39. 选择测试用例,使得被测程序中每个判定的每个分支至少执行一次,这种逻辑覆盖标准称为\_\_\_\_。

答:判定覆盖

40. 在单元测试中,测试一个模块时,需要设计\_\_\_\_。

答:驱动模块和桩模块

41. 运行被测程序的方法称为\_\_\_\_测试。

答:动态测试

42. 凭经验或直觉推测程序中可能存在的错误而设计测试用例的方法是\_\_\_\_。

答:错误推测法

43. 在基本路径测试中,将程序流程图转换成程序图时,若判断中的逻辑表达式是复合条件,应分解为一系列只有\_\_\_\_条件的嵌套判断。

答:单个

44. 动态测试方法中根据测试用例的设计方法不同,分为\_\_\_\_测试与\_\_\_\_测试两类。

答:黑盒 白盒

45. 软件测试时需要的三类信息是\_\_\_\_、\_\_\_\_、\_\_\_\_。

答:软件配置 测试配置 测试工具

46. 白盒测试是\_\_\_\_测试,黑盒测试是\_\_\_\_测试。

答:结构 功能

47. 集成测试的方法主要\_\_\_\_测试和\_\_\_\_测试两种。

答:非渐增式 渐增式

48. 软件测试方法一般分为两大类:\_\_\_\_测试方法与\_\_\_\_测试方法。

答:动态 静态

49. 在基本路径测试中,程序图的环路复杂性为此平面图中区域的个数。区域个数为\_\_\_\_区域数加上\_\_\_\_的区域数 1。也可按另一种方法计算,即  $V(G) =$  \_\_\_\_\_。

答:边和结点圈定的封闭 图形外 判定结点数 + 1

50. 在白盒测试法中,按发现错误能力由弱到强的顺序,常用的逻辑覆盖技术有\_\_\_\_覆盖、

\_\_\_\_\_覆盖、\_\_\_\_\_覆盖、\_\_\_\_\_覆盖、\_\_\_\_\_覆盖、\_\_\_\_\_覆盖。

答: 语句 判定 条件 判定/条件 条件组合 路径

51. 统计资料表明,测试的工作量约占用整个项目开发工作量的\_\_\_\_\_ % 左右,对于关系到人的生命安全的软件,测试的工作量还要\_\_\_\_\_。

答: 40 成倍增加

52. 动态测试方法有两种:一是测试产品的\_\_\_\_\_,二是测试产品\_\_\_\_\_。对软件产品进行动态测试时,也用这两种方法,分别称为\_\_\_\_\_测试法和\_\_\_\_\_测试法。

答: 功能 内部结构及处理过程 黑盒 白盒

53. 采用黑盒技术设计测试用例的方法一般有\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_等四种。

答: 等价类划分 边界值分析 错误推测 因果图

### 三、选择题

1 软件的集成测试最好由( )承担,以提高集成测试的效果。

- A .该软件的设计人员
- B .该软件开发组的负责人
- C .该软件编程人员
- D .不属于该软件开发组的软件设计人员

答: D

2 组装测试计划是在( )阶段制定的。

- A .可行性研究和计划
- B .需求分析
- C .概要设计
- D .详细设计

答: C

3 因果图方法是根据( )之间的因果关系来设计测试用例的。

- A .输入与输出
- B .设计与实现
- C .条件与结果
- D .主程序与子程序

答: A

4 .为了提高测试的效率,应该 ( )

- A .随机地选取测试数据
- B .取一切可能的输入数据作为测试数据
- C .在完成编码以后制定软件的测试计划
- D .选择发现错误可能性大的数据作为测试数据

答: D

5 .在结构测试用例设计中,有语句覆盖、条件覆盖、判定覆盖(即分支覆盖)、路径覆盖等,其中( )是最强的覆盖准则。

- A .语句覆盖
- B .条件覆盖
- C .判定覆盖
- D .路径覆盖

答: D

6 .使用白盒测试方法时,确定测试数据应根据( )和指定的覆盖标准。

- A .程序的内部逻辑
- B .程序的复杂结构
- C .使用说明书
- D .程序的功能

答: A

7 .在模块测试的过程中,采用自底向上的测试比自顶向下的测试 ( )

- A .好
- B .差
- C .一样
- D .未知

答: A

8. 在程序测试中,目前要为成功的测试设计数据,产生这些测试用例主要依赖于 ( )

- A .黑盒方法  
B .测试人员的经验  
C .白盒测试  
D .猜错

答: B

9. 从下列叙述中选出能够与软件开发需求分析、设计、编码相对应的软件测试 ( )

- A. 组装测试、确认测试、单元测试  
B. 单元测试、组装测试、确认测试  
C. 单元测试、确认测试、组装测试  
D. 确认测试、组装测试、单元测试

答:D

10. 增量式测试可以对模块 ( )

- A.减少工作量      B.组合测试      C.验收      D.评估

答: B

11. 在进行软件测试时,首先应当进行单元测试,然后再进行( ),最后再进行有效性测试。

- A .组合测试  
B 集成测试  
C .有效性测试  
D .确认测试

答: B

12. 软件的开发与维护划分为 8 个阶段, 其中单元测试是在( )阶段完成的。

- A. 概要设计      B. 详细设计      C. 编码      D. 测试

答:C

13.一般说来与设计测试数据无关的文档是 ( )

- A .需求规格说明书                      B .设计说明书  
C .源程序                                  D 项目开发计划

答:D

14. 在进行软件测试时,首先应当进行( ),然后再进行组装测试,最后再进行有效性测试。

- A.单元测试  
B.集成测试  
C.确认测试  
D.组合测试

答:A

15.“高产”的测试是指 ( )

- A .用适量的测试用例,说明测试程序正确无误
- B .用适量的测试用例,说明测试程序符合相应的要求
- C .用适量的测试用例,发现被测试程序尽可能多的错误
- D .用适量的测试用例,纠正被测试程序尽可能多的错误

答:C

16 软件测试是软件质量保证的主要手段之一,测试的费用已超过( )的 30% 以上。因此提高测试的有效性非常重要。

- A .软件开发费用  
B 软件维护费用  
C .软件开发和软件维护费用  
D 软件研制费用

答:A

17. 在进行软件测试时,首先应当进行单元测试,然后再进行组装测试,最后再进行 ( )

- A .黑盒测试  
B 集成测试  
C .有效性测试  
D 组合测试

答:C

18. 调试的目的是为了 ( )

- A .证明软件符合设计要求
- B 发现软件中的错误和缺陷
- C .改善软件的功能和性能
- D 发掘软件的潜在能力

答:B

19 .确认测试计划是在( )阶段制定的。

- A .可行性研究和计划
- B .需求分析
- C .概要设计
- D .详细设计

答:B

20 .在测试中,下列说法错误的是 ( )

- A .测试是为了发现程序中的错误而执行程序的过程
- B .测试是为了表明程序是正确的
- C .好的测试方案是极可能发现迄今为止尚未发现的错误的测试方案
- D 成功的测试是发现了至今为止尚未发现的错误的方案

答:B

21 .( )是指查明程序错误时可能采用的工具和手段。

- A .纠错技术
- B 测试纠错
- C .跟踪法
- D 动态测试

答:A

22 .( )就是简化模拟较低层次模块功能的虚拟子程序。

- A .过程
- B 函数
- C .仿真
- D 桩

答:D

23 .从已经发现故障的存在到找到准确的故障位置并确定故障的性质,这一过程称为 ( )

- A .错误检测
- B 故障排除
- C .调试
- D 测试

答:C

24 .在程序设计过程中,要为程序调试做好准备,主要体现在以下几个方面 ( )

- A .采用模块化、结构化的设计方法设计程序
- B .编写程序时要为调试提供足够的灵活性
- C .根据程序调试的需要,选择并安排适当的中间结果输出和必要的断点
- D .以上全是

答:D

25 .软件测试的目的是 ( )

- A .评价软件的质量
- B 发现软件的错误
- C .找出软件的所有错误
- D .证明软件是正确的

答:B

26 .在黑盒测试中,着重检查输入条件的组合是 ( )

- A .等价类划分法
- B .边界值分析法
- C . 错误推测法
- D . 因果图法

答:D

27 .软件测试过程中的集成测试主要是为了发现( )阶段的错误。

- A .需求分析
- B .概要设计
- C .详细设计
- D .编码

答:B

28 .不属于白盒测试的技术是 ( )

- A .语句覆盖
- B .判定覆盖
- C .条件覆盖
- D .边界值分析

答:D

29. 集成测试时,能较早发现高层模块接口错误的测试方法为 ( )
- A. 自顶向下渐增式测试                      B. 自底向上渐增式测试
- C. 非渐增式测试                              D. 系统测试

答:A

30. 确认测试以( )文档作为测试的基础。
- A. 需求规格说明书                      B. 设计说明书
- C. 源程序                                  D. 开发计划

答:A

31. 软件测试中,白盒法是通过分析程序的( )来设计测试用例的。
- A. 应用范围                      B. 内部逻辑                      C. 功能                      D. 输入数据

答:B

32. 黑盒法是根据程序的( )来设计测试用例的。
- A. 应用范围                      B. 内部逻辑                      C. 功能                      D. 输入数据

答:C

33. 单元测试主要针对模块的几个基本特征进行测试,该阶段不能完成的测试是 ( )
- A. 系统功能                      B. 局部数据结构
- C. 重要的执行路径                      D. 错误处理

答:A

34. 测试的关键问题是 ( )
- A. 如何组织软件评审                      B. 如何选择测试用例
- C. 如何验证程序的正确性                      D. 如何采用综合策略

答:B

35. 下列几种逻辑覆盖标准中,查错能力最强的是 ( )
- A. 语句覆盖                      B. 判定覆盖                      C. 条件覆盖                      D. 条件组合覆盖

答:D

36. 以下说法错误的是 ( )
- A. 判定覆盖包含了语句覆盖,但它可能会使一些条件得不到测试
- B. 条件覆盖它的检错能力较判定覆盖强,但有时达不到判定覆盖的要求
- C. 判定/条件覆盖包含了判定覆盖和条件覆盖的要求,实际上不一定达到条件覆盖的标准
- D. 凡满足条件组合覆盖标准的测试用例,也必然满足其它所有覆盖种类的覆盖标准
- E. 路径覆盖可能使测试用例达不到条件组合覆盖的要求

答:D

37. 以下说法错误的是 ( )
- A. 穷举地输入测试数据进行黑盒测试是不可能的
- B. 白盒法也不可能进行穷举测试
- C. 黑盒法和白盒法都不能使测试达到彻底
- D. 人工检测不能发现计算机不易发现的错误

答:D

38. 以下说法错误的是 ( )
- A. 单元测试指对源程序中每一个程序单元进行测试
- B. 集成测试各模块组装起来,检查各个模块是否正确实现规定的功能

- C. 确认测试主要检查已实现的软件是否满足需求规格说明书中确定的各种需求
- D. 系统测试指把已确认的软件与其它系统元素结合在一起进行测试

答:B

39. 对于软件测试时需要的三类信息,以下完全正确的解释是 ( )

- A. 软件配置:指需求规格说明书、设计说明书、测试用例等
- B. 测试配置:指测试方案、测试驱动程序、源程序等
- C. 测试工具:指计算机辅助测试的有关工具

答:C

40. 以下说法错误的是 ( )

- A. 语句覆盖是比较弱的覆盖标准
- B. 对于多分支的判定,判定覆盖要使每一个判定表达式获得每一种可能的值来测试
- C. 语句覆盖较判定覆盖严格,但该测试仍不充分
- D. 条件组合覆盖是比较强的覆盖标准

答:C

41. 以下说法正确的是 ( )

- A. 单元测试涉及编码和详细设计的文档
- B. 集成测试涉及其它系统元素
- C. 确认测试涉及编码和需求规格说明书
- D. 系统测试涉及概要设计信息

答:A

42. 以下说法正确的是 ( )

- A. 语句覆盖使每个判定的每个分支至少执行一次
- B. 判定覆盖使每条语句至少执行一次
- C. 条件覆盖使每个判定的每个条件应取到各种可能的值
- D. 条件组合覆盖使程序中每一条可能的路径至少执行一次
- E. 路径覆盖使每个判定中各条件的每一种组合至少出现一次

答:C

43. 以下说法错误的是 ( )

- A. 自顶向上测试的优点是随着上移,驱动模块逐步减少,测试开销小一些
- B. 自顶向上测试的优点是更容易设计测试用例
- C. 自顶向下测试的优点是较早地发现高层模块接口、控制等方面的问题
- D. 自顶向下测试的优点是使低层模块的错误能较早发现

答:D

44. 以下说法错误的是 ( )

- A. 满足条件覆盖并不一定满足判定覆盖
- B. 判定/条件覆盖同时满足判定覆盖和条件覆盖
- C. 满足条件组合覆盖的测试一定满足“判定覆盖”、“条件覆盖”和“判定/条件覆盖”
- D. 满足路径覆盖也一定满足条件组合覆盖

答:D

45. 以下说法错误的是 ( )

- A. 自顶向上的缺点是系统整体功能最后才能看到
- B. 自顶向上的缺点是上层模块错误发现的晚,影响范围大

- C. 自顶向下的缺点是把许多测试推迟到用实际模块代替桩模块之后
- D. 自顶向下的缺点是设计较多的桩模块,测试开销大
- D. 自顶向下的缺点是早期不能并行工作,不能充分利用人力

答:C

#### 四、简答题

##### 1.什么是集成测试?为什么要进行集成测试?

答:集成测试是指在单元测试的基础上,将所有模块按照设计要求组装成一个完整的系统进行的测试,故也称组装测试或联合测试。

实践证明,单个模块能正常工作,组装后不见得仍能正常工作,这是因为:

(1)单元测试使用的驱动模块和桩模块,与它们所代替的模块并不完全等效,因此单元测试有不彻底、不严格的情况。

(2)各个模块组装起来,穿越模块接口的数据可能会丢失。

(3)一个模块的功能可能会对另一个模块的功能产生不利的影响。

(4)各个模块的功能组合起来可能达不到预期要求的主功能。

(5)单个模块可以接受的误差,组装起来可能累积和放大到不能接受的程度。

(6)全局数据可能会出现问題。

因此必须要进行集成测试,用于发现模块组装中可能出现的问题,最终构成一个符合要求的软件系统。

##### 2.渐增式测试中组装模块的方法有哪些?各有什么优点和缺点?

答:渐增式测试有以下两种不同的组装模块的方法:

(1)自顶向下结合。该方法不需要编写驱动模块,只需要编写桩模块,其步骤是从顶层模块开始,沿被测程序的软件结构图的控制路径逐步向下测试,从而把各个模块都结合进来,这里又有两种组合策略: 深度优先策略:先从软件结构中选择一条主控路径,把该路径上的模块一个个结合进来进行测试,以便完成一个特定的子功能,接着再结合其他需要优先考虑的路径。主控路径一般选择系统的关键路径或输入、输出路径。 宽度优先策略:逐层结合直接下属的所有模块。

自顶向下测试的优点是: 能较早地发现高层模块接口、控制等方面的问题。 初期的程序概貌可让人们较早地看到程序的主功能,增强开发人员的信心。

自顶向下测试的缺点是: 桩模块不可能提供完整的信息,因此把许多测试推迟到用实际模块代替桩模块之后。 设计较多的桩模块,测试开销大。 早期不能并行工作,不能充分利用人力。

(2)自底向上结合。该方法仅需编写驱动模块,不需编写桩模块。其步骤为: 把低层模块组合成实现一个个特定子功能的族。 为每一个族编写一个驱动模块,以协调测试用例的输入和测试结果的输出。 对模块族进行测试。 按软件结构图依次向上扩展,用实际模块替换驱动模块,形成一个个更大的族。 重复 至 步,直至软件系统全部测试完毕。

自底向上测试的优点是: 随着上移,驱动模块逐步减少,测试开销小一些。 比较容易设计测试用例。 早期可以并行工作。 低层模块的错误能较早发现。

自底向上测试的缺点是: 系统整体功能最后才能看到。 上层模块错误发现的晚,上层模块的问题是全局性的问题,影响范围大。

由于自顶向下渐增式测试和自底向上渐增式测试的方法各有利弊,实际应用时,应根据软件的特点、任务的进度安排选择合适的方法。一般是将这两种测试方法结合起来,低层模块使用自底向上结合的方法组装成子系统,然后由主模块开始自顶向下对各子系统进行集成测试。

### 3 非渐增式测试与渐增式测试有什么区别？

答:集成测试的方法主要有两种:非渐增式测试和渐增式测试。

(1)非渐增式测试。首先对每个模块分别进行单元测试,然后再把所有的模块按设计要求组装在一起进行测试。

(2)渐增式测试。逐个把未经过测试的模块组装到已经过测试的模块上去,进行集成测试。每加入一个新模块进行一次集成测试,重复此过程直至程序组装完毕。

渐增式与非渐增式测试的方法有以下区别:

(1)非渐增式方法把单元测试和集成测试分成两个不同的阶段,前一阶段完成模块的单元测试,后一阶段完成集成测试。而渐增式测试往往把单元测试与集成测试合在一起,同时完成。

(2)非渐增式需要更多的工作量,因为每个模块都需要驱动模块和桩模块,而渐增式利用已测试过的模块作为驱动模块或桩模块,因此工作量较少。

(3)渐增式可以较早地发现接口之间的错误,非渐增式最后组装时才发现。

(4)渐增式有利于排错,发现错误往往和最近加进来的模块有关,而非渐增式发现接口错误推迟到最后,很难判断是哪一部分接口出错。

(5)渐增式比较彻底,已测试的模块和新的模块再测试。

(6)渐增式占用的时间较多,但非渐增式需更多的驱动模块。桩模块也占用一些时间。

(7)非渐增式开始可并行测试所有模块,能充分利用人力,对测试大型软件很有意义。

### 4 用等价类划分的方法设计测试用例的步骤是什么？

答:用等价类划分的方法设计测试用例的步骤为:

(1)划分等价类。从程序的功能说明(如需求规格说明书)找出一个个输入条件(通常是一句话或一个短语),然后为每一个输入条件划分成为两个或多个等价类,将其列表。

(2)确定测试用例。根据已划分的等价类,按以下步骤设计测试用例: 为每一个等价类编号。设计一个测试用例,使其尽可能多地覆盖尚未被覆盖过的合理等价类。重复这步,直到所有合理等价类被测试用例覆盖。 设计一个测试用例,使其只覆盖一个不合理等价类。重复这一步,直到所有不合理等价类被覆盖。之所以这样做,是因为某些程序中对某一输入错误的检查往往会屏蔽对其他输入错误的检查。因此必须针对每一个不合理等价类,分别设计测试用例。

### 5 使用边界值分析方法设计测试用例的设计原则有哪些？

答:使用边界值分析方法设计测试用例的设计原则有:

(1)如果输入条件规定了值的范围,可以选择正好等于边界值的数据作为合理的测试用例,同时还要选择刚好越过边界值的数据作为不合理的测试用例。

(2)如果输入条件指出了输入数据的个数,则按最大个数、最小个数、比最小个数少 1、比最大个数多 1 等情况分别设计测试用例。

(3)对每个输出条件分别按照以上原则(1)或(2)确定输出值的边界情况。由于输出值的边界不与输入值的边界相对应,所以要检查输出值的边界不一定可能,要产生超出输出值之外的结果也不一定能做到,但必要时还需试一试。

(4)如果程序的规格说明给出的输入或输出域是个有序集合(如顺序文件、线性表、链表等),则应选取集合的第一个元素和最后一个元素作为测试用例。

### 6 什么是黑盒测试法?什么是白盒测试法?

答:黑盒测试是把被测试对象看成一个黑盒子,测试人员完全不考虑程序的内部结构和处理过程。只在软件的接口处进行测试,依据需求规格说明书,检查程序是否满足功能要求。因此,黑盒测试又称为功能测试或数据驱动测试。

通过黑盒测试主要发现以下错误:(1)是否有不正确或遗漏了的功能。(2)在接口上,能否正确地接



受输入数据,能否产生正确的输出信息。(3)访问外部信息是否有错。(4)性能上是否满足要求等等。

白盒测试是把测试对象看作一个打开的盒子,测试人员须了解程序的内部结构和处理过程,以检查处理过程的细节为基础,对程序中尽可能多的逻辑路径进行测试,检验内部控制结构和数据结构是否有错,实际的运行状态与预期的状态是否一致。

#### 7.简要说明如何划分等价类?

答:划分等价类是一个比较复杂的问题,以下提供几条经验:

(1)如果某个输入条件规定了取值范围或值的个数,则可确定一个合理的等价类(输入值或数在此范围内)和两个不合理等价类(输入值或个数小于这个范围的最小值或大于这个范围的最大值)。

(2)如果规定了输入数据的一组值,而且程序对不同的输入值做不同的处理,则每个允许输入值是一个合理等价类,此外还有一个不合理等价类(任何一个不允许的输入值)。

(3)如果规定了输入数据必须遵循的规则,可确定一个合理等价类(符合规则)和若干个不合理等价类(从各种不同角度违反规则)。

(4)如果已划分的等价类中各元素在程序中的处理方式不同,则应将此等价类进一步划分为更小的等价类。

以上这些划分输入数据等价类的经验也同样适于输出数据,这些数据也只是测试时可能遇到的情况的很小部分。为了能正确划分等价类,一定要正确分析被测程序的功能。

#### 8.采用黑盒技术设计测试用例有哪几种方法?这些方法各有什么特点?

答:黑盒测试是功能测试,因此设计测试用例时,需要研究需求规格说明和概要设计说明中有关程序功能或输入、输出之间的关系等信息,从而与测试后的结果进行分析比较。用黑盒技术设计测试用例的方法一般有以下四种,但没有一种方法能提供一组完整的测试用例,以检查程序的全部功能,在实际测试中应该把各种方法结合起来使用。

(1)等价类划分。为了保证从输入数据中选择一个适当的子集,使其发现更多的错误。等价类划分将输入数据域按有效的或无效的(也称合理的或不合理的)划分成若干个等价类,测试每个等价类的代表值就等于对该类其他值的测试。也就是说,如果从某个等价类中任选一个测试用例未发现程序错误,该类中其他测试用例也不会发现程序的错误。这样就把漫无边际的随机测试改变为有针对性的等价类测试,用少量有代表性的例子代替大量测试目的相同的例子能有效地提高测试效率。

等价类划分法比随机选择测试用例要好得多,但这个方法的缺点是没有注意选择某些高效的、能够发现更多错误的测试用例。

(2)边界值分析。实践经验表明,程序往往在处理边界情况时发生错误。边界情况指输入等价类和输出等价类边界上的情况,因此检查边界情况的测试用例是比较高效的,可以查出更多的错误。

使用边界值分析方法设计测试用例时一般与等价类划分结合起来。但它不是从一个等价类中任选一个例子作为代表,而是将测试边界情况作为重点目标,选取正好等于、刚刚大于或刚刚小于边界值的测试数据。

(3)错误推测。在测试程序时,人们可能根据经验或直觉推测程序中可能存在的各种错误,从而有针对性地编写检查这些错误的测试用例,这就是错误推测法。

错误推测法没有确定的步骤,凭经验进行。它的基本思想是列出程序中可能发生错误的情况,根据这些情况选择测试用例。如输入、输出数据为零是容易发生错误的情况,又如,输入表格为空或输入表格只有一行是容易出错的情况等等。

(4)因果图。等价类划分和边界值分析方法都只是孤立地考虑各个输入数据的测试功能,而没有考虑多个输入数据的组合引起的错误。因果图能有效地检测输入条件的各种组合可能会引起的错误。因果图的基本原理是通过画因果图,把用自然语言描述的功能说明转换为判定表,最后为判定表的每一列设计一个测试用例。

(5)综合策略。前面介绍的软件测试方法,各有所长。每种方法都能设计出一组有用例子,用这组例子容易发现某种类型的错误,但可能不易发现另一种类型的错误。因此在实际测试中,联合使用各种测试方法,形成综合策略,通常先用黑盒法设计基本的测试用例,再用白盒法补充一些必要的测试用例。

具体做法是: 在任何情况下都应使用边界值分析法,用这种方法设计的用例暴露程序错误能力强。设计用例时,应该既包括输入数据的边界情况又包括输出数据的边界情况。 必要时用等价类划分方法补充一些测试用例。 再用错误推测法补充测试用例。 检查上述测试用例的逻辑覆盖程度,如未满足所要求的覆盖标准,再增加例子。 如果规格说明中含有输入条件的组合情况,则一开始就可使用因果图法。

#### 9.什么是静态测试?什么是动态测试?

答:软件测试方法一般分为两大类:动态测试方法与静态测试方法,而动态测试方法中又根据测试用例的设计方法不同,分为黑盒测试与白盒测试两类。

(1)静态测试。静态测试指被测试程序不在机器上运行,而是采用人工检测和计算机辅助静态分析的手段对程序进行检测。

人工检测。人工检测是不依靠计算机而是靠人工审查程序或评审软件。人工审查程序偏重于编码质量的检验,而软件审查除了审查编码还要对各阶段的软件产品进行检验。人工检测可以发现计算机不易发现的错误,据统计,能有效地发现 30% ~ 70% 的逻辑设计和编码错误,可以减少系统测试的总工作量。

计算机辅助静态分析。利用静态分析工具对被测试程序进行特性分析,从程序中提取一些信息,以便检查程序逻辑的各种缺陷和可疑的程序构造。如用错的局部量和全程量、不匹配参数、不适当的循环嵌套和分支嵌套、潜在的死循环、不会执行到的代码等等。还可能是提供一些间接涉及程序欠缺的信息、各种类型语句出现的次数、变量和常量的引用表、标识符的使用方式、过程的调用层次、违背编码规则等等。静态分析中还可以用符号代替数值求得程序结果,以便对程序进行运算规律的检验。

(2)动态测试。动态测试指通过运行程序发现错误。一般意义上的测试大多是指动态测试。为使测试发现更多的错误,需要运用一些有效的方法。测试任何产品,一般有两种方法:一是测试产品的功能,二是测试产品内部结构及处理过程。对软件产品进行动态测试时,也用这两种方法,分别称为黑盒测试法和白盒测试法。

#### 10.测试分析报告有哪些内容?

答:(1)引言

编写目的

项目背景

参考资料

术语

#### (2)测试结果

测试名称

结果

问题

覆盖率

#### (3)软件功能结果

模块名称

功能

问题

#### (4)分析摘要

能力  
缺陷影响  
软件错误  
测试结论

#### 11. 测试计划有哪些内容？

答: (1) 引言

编写目的  
项目背景  
参考资料(任务书或合同书等)  
术语

#### (2) 测试计划

软件说明  
测试内容  
测试准备

#### (3) 测试内容说明

测试名称  
测试进度  
测试软件  
测试环境  
专用工具  
测试人员  
测试资料  
测试用例

#### (4) 评价

#### (5) 附录(列出测试用例清单)

#### 12. 软件测试的目的是什么？

答: 软件测试是为了发现软件中的错误而执行软件的过程。它的目标是尽可能多地发现软件中存在的错误, 将测试结果作为纠错的依据。

#### 13. 软件测试的步骤有哪些？

答: 软件测试的工作是沿着软件开发过程的反方向进行的, 先从每个模块的源程序出发, 进行单元测试, 然后按照概要设计说明书的要求, 将各个模块组装起来进行集成测试, 随后按照需求规格说明书的要求, 对软件进行确认测试, 最后将软件与系统中其他元素(硬件、其他软件、数据库、人工等)协调起来, 进行系统测试。这里重点讲一下单元测试与集成测试。

(1) 单元测试: 重点发现模块接口及内部逻辑结构与数据结构等方面的错误。多采用白盒法, 也可与黑盒法相结合。要测试一个模块需设计驱动模块(用来模拟被测模块的上层模块)和桩模块(模拟被测模块要调用的下层模块)。

(2) 集成测试: 用于发现模块组装过程中的错误。设计用例采用黑盒法, 组装模块的方法有两种:

非渐增式测试: 把所有经过单元测试的模块连接起来进行测试;

渐增式测试: 是逐步组装模块, 组装一部分测试一部分, 组装的具体方式又分为自底向上结合和自顶向下结合两种办法。

教材中给出了渐增式与非渐增式的比较, 相对而言, 渐增式更优越一些。

#### 14. 软件测试过程中的信息有哪些？

答:软件测试时需要三类信息:

- (1)软件配置:指需求规格说明书、设计说明书、源程序等。
- (2)测试配置:指测试方案、测试用例、测试驱动程序等。
- (3)测试工具:指计算机辅助测试的有关工具。

15. 什么是测试用例?动态测试有哪些方法?

答:测试用例的设计方法

所谓“测试用例”是指为寻找程序中的错误而精心设计的一组测试数据,每个测试用例一般是一个二元组(输入数据、预期结果)。动态测试主要有两种方法:白盒法和黑盒法。白盒法是从被测程序的内部逻辑结构入手来设计测试用例;黑盒法着重测试被测试程序的功能,而不关心内部如何实现其功能的结构,是从用户观点出发的测试。但无论白盒法还是黑盒法都只能选择一些有代表性的测试用例进行有限的测试。

(1)白盒法测试:有六种用于判定存在的逻辑覆盖标准,即程序内部路径的覆盖程度。教材中给出了各种覆盖特点及强弱的比较。对于有循环存在的覆盖用限制循环次数的办法来测试。但是对于一个实际问题的程序测试中,其路径是一个庞大的数字,基本路径测试告诉了最少路径计算办法,在分析程序控制流程图环路复杂性的基础上,导出基本路径集合,从而设计测试用例,保证这些基本路径至少执行一次,以最少的用例发现尽量多的错误。

(2)黑盒法测试:具体方法有等价类划分等四种,各种方法的特点及设计测试用例的步骤需掌握。采用黑盒法的综合策略是先用等价类划分法(包括界值分析法,即取边值上的数)设计出测试用例,然后用错误推测法补充。如果被测试程序含有多个条件的逻辑组合,则开始就用因果图法。

16. 通过黑盒测试主要发现哪些错误?

答:(1)是否有不正确或遗漏了的功能。

(2)在接口上,能否正确地接受输入数据,能否产生正确的输出信息。

(3)访问外部信息是否有错。

(4)性能上是否满足要求等等。

17. 使用边界值分析方法设计测试用例的设计原则是什么?

答:使用边界值分析方法设计测试用例时一般与等价类划分结合起来。但它不是从一个等价类中任选一个例子作为代表,而是将测试边界情况作为重点目标,选取正式等于、刚刚大于或刚刚小于边界值的测试数据。下面提供的一些设计原则供参考:

(1)如果输入条件规定了值的范围,可以选择正好等于边界值的数据作为合理的测试用例,同时还要选择刚好越过边界值的数据作为不合理的测试用例。

(2)如果输入条件指出了输入数据的个数,则按最大个数、最小个数、比最小个数少1、比最大个数多1等情况分别设计测试用例。

(3)对每个输出条件分别按照以上原则(1)或(2)确定输出值的边界情况。

(4)如果程序的规格说明给出的输入或输出域是个有序集合(如顺序文件、线性表、链表等),则应选取集合的第一个元素和最后一个元素作为测试用例。

18. 单元测试的内容是什么?

答:单元测试主要针对模块的以下五个基本特征进行测试:

(1)模块接口:主要检查数据能否正确地通过模块。检查的主要内容是参数的个数、属性及对应关系是否一致。当模块通过文件进行输入/输出时,要检查文件的具体描述包括文件的定义、记录的描述、文件的处理方式等是否正确。

(2)局部数据结构:局部数据结构主要检查以下几方面的错误:

说明不正确或不一致;初始化或缺少值错误;变量名未定义或拼写错误;数据类型不匹配;上溢、下

溢或地址错等等。

除了检查局部数据外,还应注意全局数据与模块的相互影响。

(3)重要的执行路径:重要模块要进行基本路径测试,仔细地选择测试路径是单元测试一项基本任务。注意选择测试用例能发现不正确的计算、错误的比较或不适当的控制流而成的错误。

(4)错误处理:主要测试程序对错误处理的能力,应检查是否存在以下问题:不能正确处理外部输入错误或内部处理引起的错误;对发生的错误不能正确描述或描述内容难以理解;在错误处理之前,系统已进行干预等等。

(5)边界条件:程序最容易在边界上出错,如输入/输出数据的等价类边界,选择条件和循环条件的边界,复杂数据结构(如表)的边界等等都应进行测试。

19. 软件测试的原则是什么?

答:在软件测试中,应注意以下指导原则:

(1)测试用例应由输入数据和预期的输出数据两部分组成。这样便于对照检查,做到“有的放矢”。

(2)测试用例不仅选用合理的输入数据,还要选择不合理的输入数据,这样能更多地发现错误,提高程序的可靠性。对于不合理的输入数据,程序应拒绝接受,并给出相应提示。

(3)除了检查程序是否做了它应该做的事,还应该检查程序是否做了它不应该做的事。程序正确打印出用户所需信息的同时还打印出用户并不需要的多余信息。

(4)应制定测试计划并严格执行,排除随意性。

(5)长期保留测试用例。测试用例的设计耗费很大的工作量,必须作为文档保存。因为以后程序可能有新的错误,需要进行回归测试。同时,为以后的维护提供方便。

(6)对发现错误较多的程序段,应进行更深入的测试。有统计数字表明,一段程序中已发现的错误数越多,其中存在的错误概率也越大。因为发现错误数多的程序段,其质量较同时在修改错误过程中又容易引入新的错误。

(7)程序员避免测试自己的程序。测试是一种“挑剔性”的行为,心理状态是测试自己的障碍。另外,对需求规格说明的理解而引入的错误则更难发现。因此应由别的人或别的机构来测试程序员编写的程序会更客观、更有效。

20. G. J. Myers 对软件测试的目的提出了哪些观点?

答:(1)软件测试是为了发现错误而执行程序的过程。

(2)一个好的测试用例能够发现至今尚未发现的错误。

(3)一个成功的测试是发现了至今尚未发现的错误的测试。

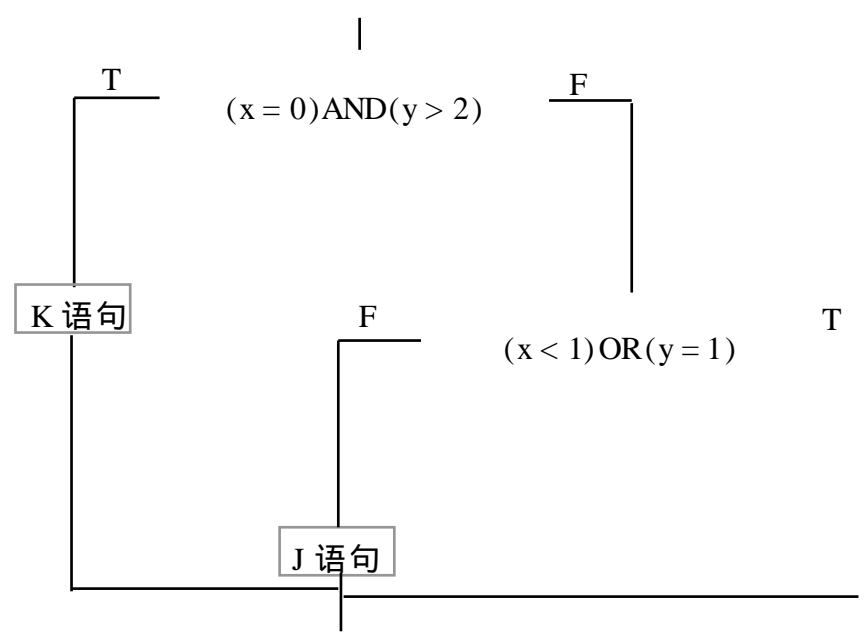
因此,测试阶段的基本任务应该是根据软件开发各阶段的文档资料和程序的内部结构,精心设计一组“高产”的测试用例,利用这些实例执行程序,找出软件中潜在的各种错误和缺陷。

## 五、应用题

1 将正确答案的编号填入题目空白处:

在白盒测试用例设计中,有语句覆盖、条件覆盖、判定覆盖、路径覆盖等,其中,(A)是最强的覆盖准则。为了对下图所示的程序进行覆盖测试,必须适当地选取测试数据。若  $x, y$  是两个变量,可供选择的测试数据组共有 、 、 、 四组(如表中给出),则实现语句覆盖至少应采用的测试数据组是(B);实现条件覆盖至少应采用的测试数据组是(C);实现路径覆盖至少应采用的测试数据组是(D)或(E)。

	x	y
测试数据组	0	3
测试数据组	1	2
测试数据组	- 1	2
测试数据组	3	1



[供选择的答案]:

- A:            语句覆盖        条件覆盖        判定覆盖        路径覆盖
- B ~ E        和 组            和 组            和 组            和 组
- 、 和                           、 和 组
- 、 和 组                       、 和 组

答:A .    B .    C .    D .    E .

2. 设计下表所示测试用例,可以达到路径覆盖标准。

测试用例表					
组	测试数据			覆盖路径	期望输出结果
	a	b	c		
1	5	5	5	1 - 2 - 3 - 7 - 6	等边三角形
2	6	5	5	1 - 2 - 3 - 4 - 8 - 6	等腰三角形
	5	5	6		
	5	6	5		
3	3	4	5	1 - 2 - 3 - 4 - 5 - 6	一般三角形
	4	5	3		
	5	3	4		
4	1	2	3	1 - 2 - 10 - 6	不能构成三角形
	2	3	1		
	3	1	2		
5	1	- 2	3	1 - 9 - 6	不能构成三角形
	2	3	0		
	0	0	0		

答: 由于 AND 或 OR 运算可能使某些条件抑制其他条件的测试,有些错误查不出来,因此应与条件组合覆盖结合起来。这里在某些路径上用增加例子的办法克服以上缺陷。

3. 某程序的功能是输入代表三角形三条边长的三个整数,判断它们能否组成三角形,若能则输出

等边、等腰或任意三角形的类型标记。请分别用黑盒法与白盒法对该程序设计测试用例。

答: (1)用等价类划分法设计测试用例:

建立等价类表,如下表所示。

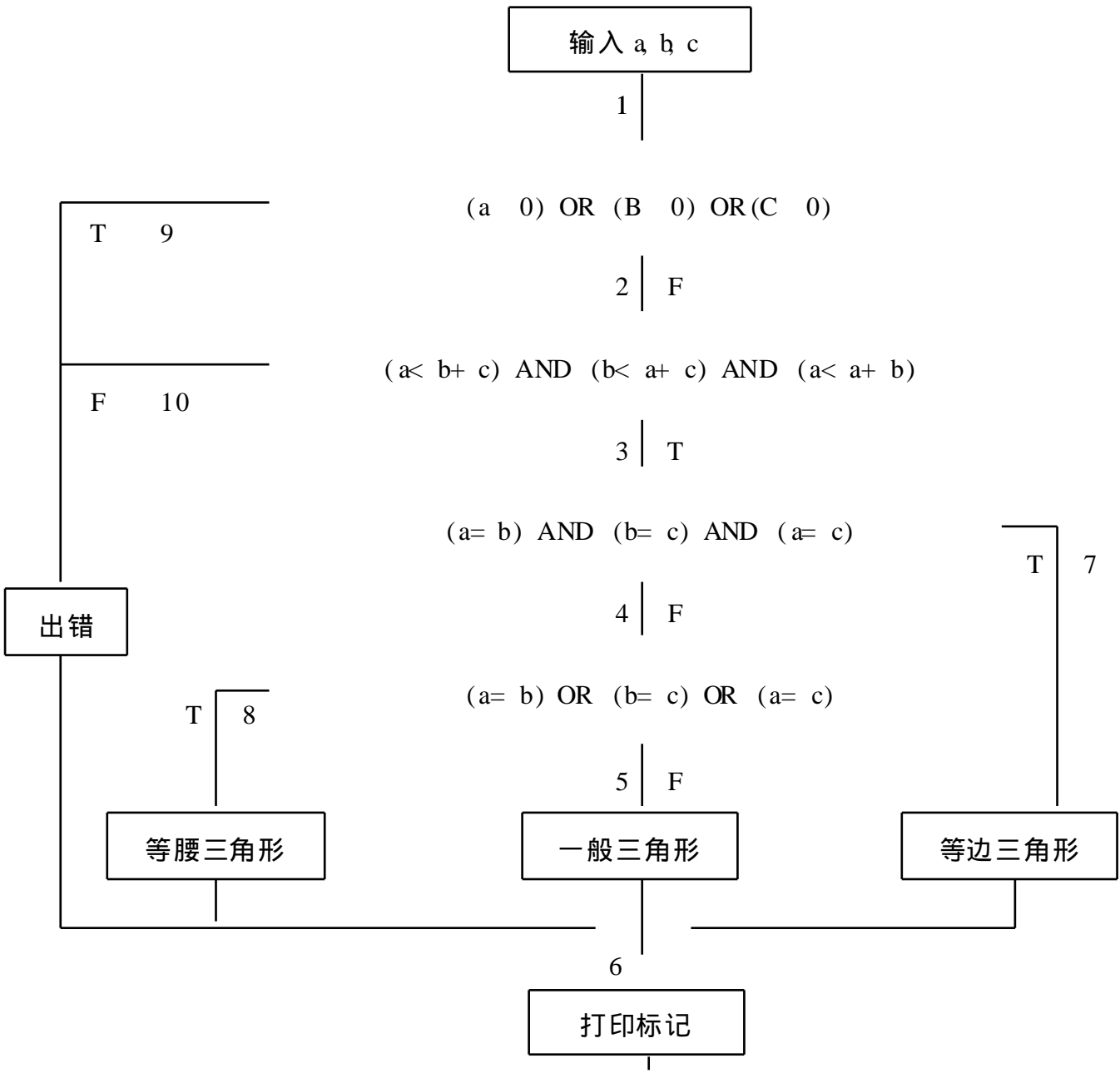
等价类表		
输入条件	合理等价类	不合理等价类
a,b,c 三个数 能否构成三角形	(1)a = b = c	(8)a + b < c
	(2)b = c 且 b + c > a	(9)a + c < b
	(3)a = b 且 a + b > c	(10)b + c < a
	(4)a = c 且 a + c > b	
	(5)a < b < c 且 a + b > c	
	(6)a < b < c 且 a + c > b	
	(7)a < b < c 且 b + c > a	
三个正数	(11)三个正整数	(12)含有零
		(13)含有负整数
		(14)含有实数
		(15)含有字符

确定测试用例,如下表所示。

(16)两个整数

测试用例表		
测试数据	覆盖范围	期望结果
a      b      c		
5      5      5	(1)	等边三角形
6      5      5	(2)	等腰三角形
5      5      6	(3)	等腰三角形
5      6      5	(4)	等腰三角形
3      4      5	(5)	一般三角形
3      5      4	(6)	一般三角形
5      3      4	(7)	一般三角形
1      2      3	(8)	不能构成三角形
1      3      1	(9))	不能构成三角形
6      2      3	(10)	不能构成三角形
5      6      7	(11)	三个正整数能构成一般三角形
0      3      5	(12)	含有零不能构成三角形
3      - 5      6	(13)	含有负数不能构成三角形
3      5      6.4	(14)	含有实数无效输入
a      3      5	(15)	含有字符无效输入
3      5	(16)	两个整数遗漏数据

(2)用逻辑覆盖法设计测试用例,该程序的流程图如下图所示。



判断能否组成三角形的流程图



## 第八章 软件维护

软件维护是软件生存周期中的最后一个阶段,也是时间最长、费用最多、困难最大的阶段。

本章总的要求是:要掌握软件维护的内容、特点、方法、技术、文档。

了解软件维护的各种困难、软件维护的特点、软件维护的文档。

理解软件维护的内容、维护任务的实施、维护的副作用。

深刻理解如何提高软件可维护性。

### 考试要求

#### 1. 维护的内容

校正性维护、适应性维护、完善性维护、预防性维护,要求达到领会层次。

#### 2. 维护的特点

结构化维护与非结构化维护、维护的困难性、软件维护的费用,要求达到识记层次。

#### 3. 维护任务的实施

维护的组织、维护的流程、维护的技术及维护的副作用,要求达到识记层次。

#### 4. 软件可维护性

可维护性的定义,要求达到领会层次。

可维护性的度量,要求达到领会层次。

提高可维护性的方法,要求达到领会层次。

### 知识重点

#### (一) 软件维护的内容

##### 1. 校正性维护

为了识别和纠正错误,修改软件性能上的缺陷,应进行确定和修改错误的过程,这个过程就称为校正性维护。

##### 2. 适应性维护

随着计算机的飞速发展,计算机硬件、软件及数据环境在不断发生变化,为了使应用软件适应这种变化而修改软件的过程称为适应性维护。

##### 3. 完善性维护

在软件运行时期中,用户往往会对软件提出新的功能要求与性能要求。这种增加软件功能、增强软件性能、提高软件运行效率而进行的维护活动称为完善性维护。

#### 4 .预防性维护

为了提高软件的可维护性和可靠性而对软件进行的修改称为预防性维护。

### (二) 维护任务的实施

#### 1 .维护的流程

软件维护的流程如下：

(1)制定维护申请报告。维护申请报告是一种由用户产生的文档,它用作计划维护任务的基础。

(2)审查申请报告并批准。由维护机构来评审维护请求,评审工作很重要,通过评审回答要不要维护,从而可以避免盲目的维护。

(3)实施维护并做详细记录。

(4)复审。在维护任务完成后,要对维护任务进行复审。

#### 2 .维护技术

有两类维护技术,它们是面向维护的技术和维护支援技术。面向维护的技术是在软件开发阶段用来减少错误、提高软件可维护性的技术。维护支援技术是在软件维护阶段用来提高维护作业的效率和质量的技术。

#### 3 .维护的副作用

因修改软件而造成的错误或其他不希望出现的情况称为维护的副作用。维护的副作用有编码副作用、数据副作用、文档副作用三种。

### (三) 软件可维护性

#### 1 .软件可维护性定义

软件可维护性是指软件能够被理解、校正、适应及增强功能的容易程度。

软件的可维护性是软件开发阶段的关键目标。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。

#### 2 .可维护性的度量

目前有若干对软件可维护性进行综合度量的方法,但还没有一种方法能够使用计算机对软件的可维护性进行综合性的定量评价。质量检查表、质量测试、质量标准是度量一个可维护的软件的七种特性时常采用的方法。

#### 3 .提高可维护性的方法

可从五个方面提高软件的可维护性:建立明确的软件质量目标;利用先进的软件开发技术和工具;建立明确的质量保证工作;选择可维护的程序设计语言;改进程序文档。

## 反馈测试题解

### 一、名词解释

#### 1 维护

答:在软件运行/维护阶段对软件产品所进行的修改就是维护。

## 2. 预防性维护

答:为了提高软件的可维护性和可靠性而对软件进行的修改称为预防性维护。

## 3. 完善性维护

答:在软件运行时期中,用户往往会对软件提出新的功能要求与性能要求。这种增加软件功能、增强软件性能、提高软件运行效率而进行的维护活动称为完善性维护。

## 4. 适应性维护

答:为了使软件系统适应计算机运行环境的不断变化而修改软件的过程称为适应性维护。因为软件维护阶段是相当长的,可能有十几年、几十年,而计算机的发展比其他任何行业的发展都要快,计算机硬件不断更新换代,操作系统、数据库系统也在不断更新换代,为了使软件能适应这些变化,就需要对其进行修改。

## 5. 校正性维护

答:识别和纠正隐含在软件中的错误的过程称为校正性维护。因为任何一个软件系统都不可能百分之百的没有任何隐含的错误,这些隐含错误带到运行阶段后,在某个特定使用环境中会暴露出来,所以存在着校正性维护。

## 6. 非结构化维护

答:因为只有源程序,而文档很少或没有文档,维护活动只能从阅读、理解、分析源程序开始。由于没有需求说明文档和设计文档,只有通过阅读源程序来了解系统功能、软件结构、数据结构、系统接口、设计约束等。这样做,第一是非常困难,第二是难于搞清楚这些问题,第三是常常误解这些问题。要想搞清楚,要花费大量的人力、物力,最终对源程序修改的后果是难以估量的,因为没有测试文档,不可能进行回归测试,很难保证程序的正确性。这就是软件工程时代以前进行维护的情况。

## 7. 结构化维护

答:用软件工程思想开发的软件具有各个阶段的文档,这对于理解和掌握软件功能、性能、系统结构、数据结构、系统接口和设计约束有很大作用。进行维护活动时,首先从评价需求说明开始,搞清楚功能、性能上的改变,然后对设计说明文档进行评价,对设计说明文档进行修改和覆查;根据设计的修改,再进行程序的变动;其后根据测试文档中的测试用例进行回归测试;最后,把修改后的软件再次交付使用。这对于减少精力、减少花费、提高软件维护效率有很大的作用。

## 8. 软件可维护性

答:软件能够被理解、校正、适应及增强功能的容易程度。

# 二、填空题

1 为了使应用软件适应计算机硬件、软件及数据环境所发生的变化而修改软件的过程称为\_\_\_\_\_。

答:适应性维护

2 为增加软件功能、增强软件性能、提高软件运行效率而进行的维护活动称为\_\_\_\_\_。

答:完善性维护

3 为了提高软件的可维护性和可靠性而对软件进行的修改为\_\_\_\_\_。

答:预防性维护

4 软件维护的流程为:(1)\_\_\_\_\_;(2)\_\_\_\_\_;(3)\_\_\_\_\_;(4)\_\_\_\_\_。

答:制定维护申请报告 审查申请报告并批准 进行维护并做详细记录 复审

5 \_\_\_\_\_是一种由用户产生的文档,它用作计划维护任务的基础。

答:维护申请报告

6.面向维护的技术是在软件开发阶段用来减少错误、提高\_\_\_\_\_的技术。面向维护的技术涉及软件开发的\_\_\_\_\_阶段。维护支援技术是在软件维护阶段用来提高\_\_\_\_\_的技术。

答:软件可维护性 所有 维护作业的效率和质量

7. 必须在软件交付之前对整个\_\_\_\_\_进行\_\_\_\_\_,以减少文档副作用。

答:软件配置 评审

8. 软件可维护性是指软件能够被\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_的容易程序。

答:理解 校正 适应及增强功能

9. 软件的\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_是衡量软件质量的几个主要特性。

答:可维护性 可使用性 可靠性

10. 软件可维护性可用下面七个质量特性来衡量,即\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_和\_\_\_\_\_。对于不同类型的维护,这七种特性的侧重点也不相同。

答:可理解性 可测试性 可修改性 可靠性 可移植性 可使用性 效率

11. 提高软件可维护性的方法有\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_。

答:建立明确的软件质量目标 利用先进的软件开发技术和工具 建立明确的质量保证工作 选择可维护的程序设计语言 改进程序文档

12. 用于软件维护工作的活动可分为两种:\_\_\_\_\_活动包括分析评价、修改设计和编写程序代码等。\_\_\_\_\_活动包括理解程序代码功能、解释数据结构、接口特点和设计约束。

答:生产性 非生产性

13. 长期维护小组由\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_等成员组成。

答:组长 副组长 维护负责人 维护程序员

14. 有两类维护技术,它们是\_\_\_\_\_技术和\_\_\_\_\_技术。

答:面向维护的 维护支援

15. 维护的副作用有\_\_\_\_\_副作用、\_\_\_\_\_副作用、\_\_\_\_\_副作用三种。

答:编码 数据 文档

16. 不管维护类型如何,大体上要开展的技术工作包括:\_\_\_\_\_。

答:修改软件设计、必要的代码修改、单元测试、集成测试、确认测试以及复审

17. 临时维护小组采用“\_\_\_\_\_”或“\_\_\_\_\_”等方法来提高维护工作的效率。

答:同事复审 同行复审

18. 软件维护的内容有\_\_\_\_\_性维护、\_\_\_\_\_性维护、\_\_\_\_\_性维护和\_\_\_\_\_性维护四种。

答:校正 适应 完善 预防

19. 为了支持应用软件系统,通常需要的文档有:\_\_\_\_\_文档、\_\_\_\_\_文档、\_\_\_\_\_文档、\_\_\_\_\_文档、\_\_\_\_\_文档。

答:用户 操作 数据 程序 历史

20. 为了保证可维护性,以下四类检查是非常有用的,这四类检查是:\_\_\_\_\_。

答:在检查点进行检查、验收检查、周期性的维护检查、对软件包的检查

21. 度量一个可维护的软件的七种特性时采用的方法有:\_\_\_\_\_。

答:质量检查表 质量测试 质量标准

22. 有两类维护技术:在开发阶段使用来减少错误、提高软件可维护性的\_\_\_\_\_技术,在维护阶段用来提高维护的效率和质量的\_\_\_\_\_技术。

答:面向维护的 维护支援

23. 在软件交付使用后,由于在软件开发过程中产生的\_\_\_\_\_没有完全彻底在\_\_\_\_\_阶段发现,必然有一部分隐含错误带到\_\_\_\_\_阶段。

答:错误 测试 维护

24. 维护阶段是软件生存周期中时期\_\_\_\_\_的阶段,花费精力和费用\_\_\_\_\_的阶段。

答:最长 最多

25. 采用手工方法开发软件只有程序而无文档,维护困难,这是一种\_\_\_\_\_维护;采用\_\_\_\_\_方法开发软件,各阶段均有文档,容易维护,这是一种\_\_\_\_\_维护。

答:非结构化 软件工程 结构化

26. 所有软件维护申请报告要按规定方式提出,该报告也称\_\_\_\_\_报告。

答:软件问题

27. 软件维护工作的活动分为\_\_\_\_\_活动和\_\_\_\_\_活动。

答:生产性 非生产性

28. 软件维护费用增加的主要原因是维护的\_\_\_\_\_非常低。

答:生产率

29. 在软件维护中,因修改软件而导致出现的错误或其他情况称为\_\_\_\_\_。

答:维护的副作用

30. 采用软件工程方法开发软件,各阶段均有文档,容易维护,这种维护是\_\_\_\_\_。

答:结构化维护

31. 为提高可维护性,要使用的先进的、强有力的、实用的软件开发方法是\_\_\_\_\_。

答:面向对象方法

32. 为了识别和纠正在运行中产生的错误而进行的维护称为\_\_\_\_\_维护。

答:校正性

### 三、选择题

1. 在整个软件维护阶段所花费的全部工作中,( )所占比例最大。

- |          |         |
|----------|---------|
| A .校正性维护 | B 适应性维护 |
| C .完善性维护 | D 预防性维护 |

答:C

2. 软件工程对维护工作的主要目标是提高( ),降低维护的代价。

- |            |          |
|------------|----------|
| A .软件的生产率  | B 软件的可靠性 |
| C .软件的可维护性 | D 维护的效率  |

答:C

3. 在软件生存周期中,工作量所占比例最大的阶段是( )阶段。

- |         |       |
|---------|-------|
| A .需求分析 | B .设计 |
| C .测试   | D 维护  |

答:D

4. 一个软件产品开发完成投入使用后,常常由于各种原因需要对它做适当的变更,通常把软件交付使用后所做的变更称为 ( )

- |          |        |
|----------|--------|
| A .维护    | B .设计  |
| C .软件再工程 | D 逆向工程 |

答:A

5.下面的叙述中,与可维护性关系最密切的是 ( )

- A.软件从一个计算机系统和环境转移到另一个计算机系统和环境的容易程度
- B.尽管有不合法的输入,软件仍能继续正常工作的能力
- C.软件能够被理解、校正、适应及增强功能的容易程度
- D.在规定的条件下和规定的一段时间内,实现所指定的功能的能力

答:C

6.在软件维护的实施过程中,为了正确、有效地修改,需要经历以下三个步骤:分析和理解程序、修改程序和重新验证程序。( )是决定维护成败和质量好坏的关键。

- A.分析和理解程序
- B.重新验证程序
- C.修改程序
- D.验收程序

答:C

7.软件的可维护性是软件开发阶段的关键目标。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、( )、可使用性和效率。

- A.完备性
- B.安全性
- C.可移植性
- D.灵活性

答:C

8.在软件维护的实施过程中,为了正确、有效地修改,需要经历以下三个步骤:分析和理解程序、修改程序和重新验证程序。重新验证程序包括( )确认、计算机确认和维护后的验收。

- A.动态
- B.静态
- C.人工
- D.自动

答:B

9.软件工程针对维护工作的主要目标是提高软件的可维护性,降低 ( )

- A.维护的效率
- B.维护的工作量
- C.文档
- D.维护的代价

答:D

10.在软件维护的实施过程中,为了正确、有效地修改,需要经历以下三个步骤:分析和理解程序、修改程序和 ( )

- A.建立目标程序
- B.重新验证程序
- C.验收程序
- D.测试程序

答:B

11.软件的可维护性、可使用性、( )是衡量软件质量的几个主要特性。

- A.可靠性
- B.可复用性
- C.可理解性
- D.可修改性

答:A

12.软件生存周期的( )的工作和软件可维护性有密切的关系。

- A.编码阶段
- B.设计阶段
- C.测试阶段
- D.每个阶段

答:D

13.不管维护类型如何,大体上要开展相同的技术工作。这些工作包括修改软件设计、( )、单元测试、集成测试、确认测试以及复审。

- A.分析
- B.测试
- C.检验
- D.代码修改

答:D

14.软件维护工作过程中,第一步是先确认 ( )

- A 维护环境
- B 维护类型
- C 维护要求
- D 维护者

答:B

15 人们称在软件运行/ 维护阶段对软件产品所进行的修改就是维护。( )是由于开发时测试的不彻底、不完全造成的。

- A 校正性维护
- B 适应性维护
- C 完善性维护
- D 预防性维护

答:A

16 在四种类型的维护中,( )维护是针对用户对软件提出的功能和性能要求的。

- A 校正性
- B 适应性
- C 完善性
- D 预防性

答:C

17 维护由引起的原因不同可分为几类,( )是由于外部环境或数据库的环境的变化造成的。

- A 校正性维护
- B 适应性维护
- C 完善性维护
- D 预防性维护

答:B

18 面向维护的技术涉及软件开发的所有阶段。以下说法错误的是 ( )

- A. 在需求分析阶段,采用灵活的数据结构,使程序相对独立于数据的物理结构
- B. 在设计阶段,划分模块时充分考虑将来改动或扩充的可能性
- C. 在编码阶段,对用户的需求进行严格的分析定义,使之没有矛盾和易于理解
- D. 在测试阶段,可以不保存或考虑测试用例和测试数据,但应尽可能多的发现错误

答:B

19 以下不属于软件维护的困难表现的是 ( )

- A. 读懂别人的程序是困难的
- B. 文档的不一致性
- C. 源程序及相关文档的错误或丢失
- D. 软件开发和软件维护在人员和时间上的差异
- E. 软件维护不是一项吸引人的工作

答:C

20 下面说法错误的是 ( )

- A. 非结构化维护对于理解和掌握软件功能、性能、系统结构等有很大作用
- B. 结构化的维护容易进行维护工作
- C. 软件维护费用增加的主要原因是软件维护的生产率非常低
- D. 软件维护的困难性是由于软件需求分析和开发方法的缺陷

答:A

21 一个维护申请提出后,经评审需要维护,则按下列过程实施维护。以下说法错误的是 ( )

- A. 由用户和维护机构协商评审维护请求
- B. 对校正性维护从评价错误的严格性开始
- C. 对适应性和完善性维护如同另一个开发工作,建立每个请求的优先权,安排所要求的工作
- D. 不管维护类型如何,大体上要开展相同的技术工作,仅是侧重点不一样
- E. 在发生重大的软件问题时,就会出现“救火”维护

答:A

22. 维护中,因误删除一个标识符而引起的错误是( )副作用。  
A. 文档                      B. 数据                      C. 编码                      D. 设计

答:C

23. 可维护性的特性中相互促进的是 ( )  
A. 可理解性和可测试性                      B. 效率和可移植性  
C. 效率和可修改性                      D. 效率和结构好

答:A

24. 可维护的特性中,相互矛盾的是 ( )  
A. 可修改性和可理解性                      B. 可测试性和可理解性  
C. 效率和可修改性                      D. 可理解性和可读性

答:C

25. 生产性维护活动是 ( )  
A. 修改设计                      B. 理解设计  
C. 解释数据结构                      D. 理解功能

答:A

26. 软件维护费用高的主要原因是 ( )  
A. 生产率高                      B. 生产率低  
C. 人员多                      D. 人员少

答:B

27. 维护阶段的文档是 ( )  
A. 软件需求说明                      B. 操作手册  
C. 软件问题报告                      D. 测试分析报告

答:C

28. 产生软件维护的副作用,是指 ( )  
A. 开发时的错误                      B. 隐含的错误  
C. 因修改软件而造成的错误                      D. 运行时误操作

答:C

29. 在生存周期中,时间长、费用高、困难大的阶段是 ( )  
A. 需求分析                      B. 编码  
C. 测试                      D. 维护

答:D

30. 为适应软硬件环境变化而修改软件的过程是 ( )  
A. 校正性维护                      B. 适应性维护  
C. 完善性维护                      D. 预防性维护

答:B

31. 软件维护的困难主要原因是 ( )  
A. 费用低                      B. 人员少  
C. 开发方法的缺陷                      D. 维护难

答:C

32. 维护中,因修改全局或公用数据而引起的错误是 ( )  
A. 文档副作用                      B. 数据副作用  
C. 编码副作用                      D. 设计副作用



答:B

33. 生产性维护活动是 ( )

- A. 修改设计
- B. 理解设计
- C. 解释数据结构
- D. 理解功能

答:A

34. 为增加软件功能和性能而进行的软件修改维护过程是 ( )

- A. 校正性维护
- B. 适应性维护
- C. 完善性维护
- D. 预防性维护

答:C

35. 维护中用来指出修改的工作量、工作性质、优先权、修改的事后性质的文档是 ( )

- A. 软件需求说明
- B. 软件修改报告
- C. 软件问题报告
- D. 测试分析报告

答:B

36. 以下说法错误的是 ( )

- A. 目前对软件可维护性的度量方法是综合度量法
- B. 目前仅有一种方法能够使用计算机对软件的可维护性进行综合性的定量评价
- C. 质量检查表是用于测试程序中某些质量特性是否存在的一个总是清单
- D. 质量测试与质量标准则用于定量分析和评价程序的质量

答:B

37. 软件维护活动总的工作量由  $M = P + K * \exp(C - D)$ 表示,对于该式的错误说法是 ( )

- A. 若 C 越大,D 越小,那么维护工作量将成指数增加
- B. C 增加表示从而使得软件为非结构化设计
- C. D 表示维护人员不是原来的开发人员
- D. K 表示生产性活动工作量

答:D

38. 下面说法错误的是 ( )

- A. 维护申请报告由申请维护的用户填写,软件维护组织内部还要制定一份软件修改报告
- B. 软件修改报告指出的问题之一是:为满足软件问题报告实际要求的工作量
- C. 软件修改报告指出的另外三个问题是:要求修改的性质、优先权和关于修改的事后数据
- D. 维护申请报告也称软件问题报告,用作计划维护任务的基础
- E. 提出维护申请报告之后,由用户和软件维护组来评审维护请求

答:E

## 四、简答题

1. 为了保证软件的可维护性,需要做哪些质量保证检查?

答:为了保证可维护性,需要做以下四类质量保证检查:

(1)在检查点进行检查。检查点是指软件开发的每一个阶段的终点。在检查点进行检查的目标是证实已开发的软件是满足设计要求的。在不同的检查点检查的内容是不同的。例如,在设计阶段检查的重点是可理解性、可修改性和可测试性,可理解性检查的重点是检查设计的复杂性。

(2)验收检查。验收检查是一个特殊的检查点的检查,它是把软件从开发转移到维护的最后一次检查。它对减少维护费用,提高软件质量是非常重要的。验收检查实际上是我们已讲过的验收测试的一

部分,只不过验收检查是从维护角度提出验收条件或标准。

(3)周期性的维护检查。上述两种软件检查适用于新开发的软件。对已运行的软件应进行周期性的维护检查。为了改正在开发阶段未发现的错误,使软件适应新的计算机环境并满足变化的用户需求,对正在使用的软件进行改变是不可避免的。改变程序可能引入新错误并破坏原来程序概念的完整性。为了保证软件质量应该对正在使用的软件进行周期性维护检查。实际上周期性维护检查是开发阶段对检查点进行检查的继续,采用的检查方法和检查内容与检查点的检查都是相同的。把多次维护检查结果同以前进行的验收检查结果以及检查点检查结果做比较,对检查结果的任何改变都要进行分析,找出原因。

(4)对软件包的检查。上述检查方法适用于组织内部开发和维护的软件或专为少数几个用户设计的软件,很难应用于享有多个用户的通用软件包。因为软件包属于卖方的资产,用户很难获得软件包的源代码和完整的文档。对于软件包的维护通常采用下述方法。使用单位的维护程序员在分析研究卖方提供的用户手册、操作手册、培训教程、新版本策略指导、计算机环境和验收测试的基础上,深入了解本单位的希望和要求,编制软件包检验程序。软件包检验程序是一个测试程序,它检查软件包程序所执行的功能是否与用户的要求和条件相一致。为了建立这个程序,维护程序员可以利用卖方提供的验收测试实例或重新设计新的测试实例,根据测试结果检查和验证软件的参数或控制机构,从而完成软件包的维护。

## 2. 什么是软件可维护性?可维护性度量的特性是什么?

答:软件可维护性是指软件能够被理解、校正、适应及增强功能的容易程度。

软件的可维护性是软件开发阶段的关键目标。影响软件可维护性的因素较多,设计、编码及测试中的疏忽和低劣的软件配置、缺少文档等都对软件的可维护性产生不良影响。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不相同。这些质量特性能常体现在软件产品的许多方面。为使每一个质量特性都达到预定的要求,需要在软件开发的各个阶段采取相应的措施加以保证,即这些质量要求要渗透到各开发阶段的各个步骤中。因此,软件的可维护性是产品投入运行以前各阶段针对上述各质量特性要求进行开发的最终结果。

目前有若干对软件可维护性进行综合度量的方法,但要可对可维护性作出定量度量还是困难的。还没有一种方法能够使用计算机对软件的可维护性进行综合性的定量评价。下面是度量一个可维护的软件的七种特性时常采用的方法,即质量检查表、质量测试、质量标准。

## 3. 影响软件维护代价的因素有哪些?

答:软件维护费用不断上升,这只是软件维护有形的代价。另外还有无形的代价,即要占用更多的资源。由于大量软件的维护活动要使用较多的硬件、软件、软件工程师等资源,这样一来,投入新的软件开发的资源就因不足而受到影响。由于维护时的改动,在软件中引入了潜在的故障,从而降低了软件的质量。

## 4. 什么是非结构化维护?非结构化维护的特点是什么?

答:软件的开发过程对软件的维护有较大的影响。若不采用软件工程的方法开发软件,则软件只有程序而无文档,维护工作非常困难,这是一种非结构化的维护。

因为只有源程序,而文档很少或没有文档,维护活动只能从阅读、理解、分析源程序开始。由于没有需求说明文档和设计文档,只有通过阅读源程序来了解系统功能、软件结构、数据结构、系统接口、设计约束等。这样做,第一是非常困难,第二是难于搞清楚这些问题,第三是常常误解这些问题。要想搞清楚,要花费大量的人力、物力,最终对源程序修改的后果是难以估量的,因为没有测试文档,不可能进行回归测试,很难保证程序的正确性。

## 5. 什么是结构化维护?结构化维护的特点是什么?

答:若采用软件工程的方法开发软件,则各阶段都有相应的文档,容易进行维护工作,这是一种结构化的维护。

用软件工程思想开发的软件具有各个阶段的文档,这对于理解和掌握软件功能、性能、系统结构、数据结构、系统接口和设计约束有很大作用。进行维护活动时,首先从评价需求说明开始,搞清楚功能、性能上的改变,然后对设计说明文档进行评价,对设计说明文档进行修改和复查;根据设计的修改,再进行程序的变动;其后根据测试文档中的测试用例进行回归测试;最后,把修改后的软件再次交付使用。这对于减少精力、减少花费、提高软件维护效率有很大的作用。

#### 6 好的文档的作用和意义是什么?

答:程序文档是对程序功能、程序各组成部分之间的关系、程序设计策略、程序实现过程的历史数据等的说明和补充。程序文档对提高程序的可阅读性有重要作用。为了维护程序,人们必须阅读和理解程序文档。好的文档有以下几方面的作用:(1)好的文档能提高程序的可阅读性,但坏的文档比没有文档更坏;(2)好的文档意味着简明性,风格的一致性,容易修改;(3)程序编码中应该有必要的注释以提高程序的可理解性;(4)程序越长、越复杂,则它对文档的需求也越迫切。

#### 7. 维护的实施有哪些?

答:(1)维护的组织

与软件开发一样,在进行软件维护时,也要成立相应的组织,即维护小组,有负责人员、管理人员、维护人员等,各司其职,完成各个维护任务。

##### (2)维护的流程

一个维护活动要经过下列过程:

制定维护申请报告。该报告也称软件问题报告,这是软件维护阶段的文档之一,用来说明软件错误出现的情况、环境、现象、原因等。对维护申请报告进行审查批准后,可实施维护。

进行维护工作。首先要确定维护的类型。对不同的维护任务,要作不同的处理,但维护的技术工作大致相同,即修改要求、设计、程序,然后进行测试。

编制软件修改报告。它是维护阶段的第二个文档,用来记录软件问题的修改情况。

最后,进行复审。即完成了哪些工作,用了哪些资源,工作障碍是什么等。

#### 8. 维护的副作用有哪些?

答:修改软件是一件危险的事情,因为每一次修改可能导致潜在的错误。

维护的副作用是指因修改软件而造成的错误或其他不希望出现的情况。

维护的副作用有编码副作用、数据副作用、文档副作用等。

#### 9. 维护的副作用有哪些?

答:维护的目的是为了延长软件的寿命并让其创造更多的价值,经过一段时间的维护,软件中的错误减少了,功能增强了。但修改软件是危险的,每修改一次,潜伏的错误就可能增加一分。这种因修改软件而造成的错误或其他不希望出现的情况称为维护的副作用。维护的副作用有编码副作用、数据副作用、文档副作用三种。

##### (1)编码副作用

在使用程序设计语言修改源代码时可能引入错误。

##### (2)数据副作用

在修改数据结构时,有可能造成软件设计与数据结构不匹配,因而导致软件错误。数据副作用是修改软件信息结构导致的结果。

##### (3)文档副作用

对数据流、软件结构、模块逻辑或任何其他有关特性进行修改时,必须对相关技术文档进行相应修改。否则会导致文档与程序功能不匹配、缺少条件改变、新错误信息不正确等错误,使文档不能反映软

件当前的状态。如果对可执行软件的修改没有反映在文档中,就会产生文档副作用。

#### 10. 维护的困难性有哪些?

答:软件维护的困难性是由于软件需求分析和开发方法的缺陷。软件生存周期中的开发阶段没有严格而又科学的管理和规划,就会引起软件运行时的维护困难。这种困难表现在如下几种:

##### (1) 读懂别人的程序是困难的

要修改别人编写的程序,首先要看懂、理解别人的程序,而理解别人的程序是非常困难的。这种困难程序随着程序文档的减少而很快的增加,如果没有相应的文档,困难就达到非常严重的地步。一般程序员都有这样的体会:修改别人的程序,还不如自己重新编程序。

##### (2) 文档的不一致性

文档不一致性是维护工作困难的又一因素。它会导致维护人员不知所措,不知根据什么进行修改。这种不一致表现在各种文档之间的不一致以及文档与程序之间的不一致。这种不一致是由于开发过程中文档管理不严所造成的。在开发中经常会出现修改程序却遗忘了修改与其相关的文档,或某一文档做了修改,却没有修改与其相关的另一文档这类现象。要解决文档不一致性,就要加强开发工作中的文档版本管理工作。

##### (3) 软件开发和软件维护在人员和时间上的差异

如果软件维护工作是由该软件的开发人员进行,则维护工作就变得容易,因为他们熟悉软件的功能、结构等。但通常开发人员与维护人员是不同的,这种差异会导致维护的困难。由于维护阶段持续时间很长,正在运行的软件可能是十几、二十年前开发的,开发工具、方法、技术与当前的工具、方法、技术差异很大,这又是维护困难的另一因素。

##### (4) 软件维护不是一项吸引人的工作

由于维护工作的困难性,维护工作经常遭受挫折,而且很难出成果,不像软件开发工作那样吸引人。

#### 11. 维护的流程是什么?

答:软件维护的流程如下:

制定维护申请报告。

审查申请报告并批准。

进行维护并做详细记录。

复审。

#### 12. 试述维护过程?

答:一个维护申请提出之后,经评审需要维护,则按下列过程实施维护:

(1) 首先确定要进行维护的类型。有许多情况,用户可以把一个请求看作校正性维护,而软件开发者可以把这个请求看作适应性或完善性维护,此时,对不同观点就要协商解决。

(2) 对校正性维护从评价错误的严重性开始。如果存在一个严重的错误,例如一个系统的重要功能不能执行,则由管理者组织有关人员立即开始分析问题。如果错误并不严重,则校正性维护与软件其他任务一起进行,统一安排,按计划进行维护工作。甚至会有这样一种情况:申请是错误的,因此经审查后发现并不需要修改软件。

(3) 对适应性和完善性维护。如同它是另一个开发工作一样,建立每个请求的优先权,安排所需求的工作。若设置一个极高的优先权,当然也就意味着要立即开始此项维护工作了。

(4) 实施维护任务。不管维护类型如何,大体上要开展相同的技术工作。这些工作包括修改软件设计、必要的代码修改、单元测试、集成测试、确认测试以及复审。每种维护类型的侧重点不一样。

(5) “救火”维护。存在着并不完全适合上面所述的经过仔细考虑的维护申请。

#### 13. 长期维护小组有哪些人员组成?

答:对长期运行的复杂系统需要一个稳定的维护小组。维护小组由以下成员组成。

(1)组长。维护小组组长是该小组的技术负责人,负责向上级主管部门报告维护工作。

(2)副组长。副组长是组长的助手。在组长缺席时完成组长的工作,具有与组长相同的业务水平和工作经验。副组长还执行同开发部门或其他维护小组联系的任务。

(3)维护负责人。维护负责人是维护小组的行政负责人。他通常管理几个维护小组的人事工作,负责维护小组成员的人事管理工作。

(4)维护程序员。维护程序员负责分析程序改变的要求和执行修改工作。维护程序员不仅具有软件开发方面的知识和经验,也应具有软件维护方面的知识和经验,还应熟悉程序应用领域的知识。

#### 14. 维护的特点有哪些?

答:(1)维护的费用高

维护的主要特点是维护费用高、维护困难。造成维护费用高的主要原因是软件维护工作效率非常低。这是因为要修改软件,首先要理解原来系统的设计思想、原来程序设计代码的功能,要解释原来的数据结构,要分析接口设计和设计约束。而要看懂别人的程序是困难的,要花很多代价。在搞清楚这些问题之后,才能进行修改。这样就导致生产率非常低下,因而费用非常高。

#### (2)维护的困难性

维护的困难性主要是由开发的软件可维护性差引起的,即软件结构性差、文档不齐全、开发不规范,再加之原来开发的技术较陈旧、维护工作不吸引人等,所以维护工作就很难进行。

## 第九章 软件开发的增量模型

了解增量模型的基本思想、特点;了解模型的分类和各种模型的表示;了解快速原型的基本原理。

理解快速原型的开发技术、构造原型的建议和开发环境。

深刻理解快速原型开发过程和步骤。

掌握的技能是:能够应用快速原型模型开发软件项目。

### 考试要求

#### 1. 概述

瀑布模型的局限性,要求达到识记层次。

增量模型的基本思想,要求达到识记层次。

增量模型的分类,要求达到识记层次。

#### 2. 渐增模型

增量构造模型,要求达到领会层次。

演化提交模型,要求达到领会层次。

#### 3. 快速原型模型

基本思想,要求达到识记层次。

快速原型模型表示,要求达到识记层次。

原型开发过程,要求达到识记层次。

#### 4. 快速原型开发技术和开发环境

构造原型技术,构造原型建议及开发环境,要求达到领会层次。

#### 5. 增量模型的评价

原型的作用,原型使用建议,原型的优点及原型存在的问题,要求达到识记层次。

### 知识重点

#### (一) 增量模型

增量模型是在瀑布模型的基础上加以修改而形成的,它和瀑布模型之间的本质区别是:瀑布模型属于整体开发模型,它规定在开始下一个阶段的工作之前,必须完成前一阶段的所有细节。而增量模型属于非整体开发模型,它推迟某些阶段或所有阶段中的细节,从而较早的产生工作软件。

增量模型是在项目的开发过程中以一系列的增量方式开发系统。增量方式包括增量开发和增量提交。增量开发是指在项目开发周期内,以一定的时间间隔开发部分工作软

件;增量提交是指在项目开发周期内,以一定的时间间隔增量方式向用户提交工作软件及相应文档。增量开发和增量提交可以同时使用,也可单独使用。

根据增量的方式和形式的不同,分为渐增模型和原型模型。

### 1 .渐增模型

这种模型是瀑布模型的变种,有两类渐增模型:

(1)增量构造模型。它在瀑布模型基础上,对一些阶段进行整体开发,对另一些阶段进行增量开发,也就是说在前面的开发阶段按瀑布模型进行整体开发,后面的开发阶段按增量方式开发。

(2)演化提交模型。它在瀑布模型的基础上,所有阶段都进行增量开发,也就是说不仅是增量开发,也是增量提交。

### 2 .原型模型

又称快速原型模型,它是增量模型的另一种形式。它是在开发真实系统之前,构造一个原型,在该原型的基础上,逐渐完成整个系统的开发工作。根据原型的不同作用,有三类原型模型:

(1)探索型原型。这种类型的原型模型是把原型用于开发的需求分析阶段,目的是要弄清用户的需求,确定所期望的特性,并探索各种方案的可行性。它主要针对开发目标模糊,用户与开发者对项目都缺乏经验的情况,通过对原型的开发来明确用户的需求。

(2)实验型原型。这种原型主要用于设计阶段,考核实现方案是否合适,能否实现。对于一个大型系统,若对设计方案心中没有把握时,可通过这种原型来证实设计方案的正确性。

(3)演化型原型。这种原型主要用于及早向用户提交一个原型系统,该原型系统或者包含系统的框架,或者包含系统的主要功能,在得到用户的认可后,将原型系统不断扩充演变为最终的软件系统。它将原型的思想扩展到软件开发的全过程。

## (二)渐增模型

### 1 .增量构造模型

在该模型中,需求分析阶段和设计阶段都是按瀑布模型的整体方式开发,但是编码阶段和测试阶段是按增量方式开发。在这种模型的开发中,用户可以及早看到部分软件功能,可以及早发现全面问题,以便在开发其它软件功能时及时解决问题。

### 2 .演化提交模型

在该模型中,项目开发的各个阶段都是增量方式。先对某部分功能进行需求分析,然后顺序进行设计、编码、测试,把该功能的软件交付给用户,然后再对另一部分功能进行开发利用,提交用户直至所有功能全部增量开发完毕。

该模型是增量开发的极端形式,它不仅是增量开发也是增量提交,用户将最早收到部分工作软件,及时发现问题更彻底,修改扩充更容易。

## (三)快速原型模型

### 1 .基本思想

### (1)原型

原型是指模拟某种产品的原型模型。软件开发中的原型是软件的一个早期可运行的版本,它反映了最终系统的重要特性。

(2)快速原型思想的产生。是在研究需求阶段的方法和技术中产生的。

### (3)快速原型的原理

快速原型是利用原型辅助软件开发的一个新思想。经过简单快速分析,快速实现一个原型,用户与开发者在试用原型过程中加强通信与反馈,通过反复评价和改进原型,减少误解,弥补遗漏,适应变化,最终提高软件质量。

### (4)原型运用方式

由于运用原型的目的和方式不同,在使用原型时可采取的策略有抛弃策略和附加策略。

抛弃策略是将原型用于开发过程的某一阶段,促进该阶段的开发结果更加完整、准确、一致、可靠,该阶段结束后,原型随之作废。探索型和实验型快速原型就是采用此策略的。

附加策略是将原型用于开发的全过程,原型由最基本的核心开始,逐步增加新的功能和新的需求,反复修改反复扩充,最后发展为用户满意的最终系统。演化型快速原型就采用此策略。

## 2.快速原型模型表示

对于探索型,用原型过程来代替需求分析,把原型作为需求说明的补充形式,运用原型尽可能使需求说明完整、一致、准确、无二义性,但在整体上仍采用瀑布模型。

对于实验型,用原型过程来代替设计阶段,即在设计阶段引入原型,快速分析实现方案,快速构造原型,通过运行,考察设计方案的可行性与合理性,原型成为设计的总体框架或设计结果的一部分。

对于演化型,用原型过程来代替全部开发阶段。这是典型的演化提交模型的形式,它在强有力的软件工具和环境支持下,通过原型过程的反复循环,直接得到软件系统。不强调开发的严格阶段性和高质量的阶段性文档,不追求理想的开发模式。

## 3.原型开发过程

### (1)原型构造要求

原型不同于最终系统,因此构造原型时, 必须注意功能性能的取舍,忽略一切暂时不关心的部分以加速原型的实现,同时又要充分体现原型的作用。满足评价原型的要求。

构造之前,必须明确运用原型的目的,从而解决分析与构造内容的取舍,还要根据原型的目的确定考核、评价原型的内容。

### (2)原型的特征分类

根据原型的目的、方式及内容的取舍不同,原型特征可分为四类;系统的界面形式、系统的总体结构、系统的主要处理功能和性能、数据库模式。

### (3)原型开发步骤

快速分析。迅速确定系统的基本需求。

构造原型。根据基本需求说明、尽快实现一个可运行的系统。



运行原型。这是发现问题、消除误解、开发者与用户充分协调的一个步骤。

评价原型。在运行的基础上,考核评价原型的特性,纠正错误,增添新的要求。

修改。根据评价原型的结果进行修改。

修改过程代替了初始的快速分析,从而形成原型开发的循环过程。用户与开发者在这种循环过程中不断接近系统的最终要求。

#### (四) 快速原型的开发技术和开发环境

##### 1. 构造原型的技术

(1)可执行的规格说明。这是一种使要求说明过程自动化的技术,通过可执行的规格说明语言来描述预期的行为“做什么”,人们可以从直接观察中用规格说明语言来规定任何系统行为。

(2)基于脚本的设计。此方法主要用于解决要求的验证问题。一个脚本将模拟在系统运行期间用户经历的事件,它提供了输入——处理——输出的屏幕,以及有关对话的一个模型,开发者能够给用户显示一个系统的逼真视图。

(3)采用非常高级语言或专门语言。这是一些建模的语言,使用应用领域中的术语,方便了用户和开发者在计划中的系统的特性的思想交流。

(4)能重用软件。能重用成分是一些具体应用中共同出现的一些程序设计模式,包括输入/输出规格说明、控制结构、一般问题/解法描述等。

##### 2. 构造原型建议

(1)暂不考虑速度、空间等性能效率方面的要求。

(2)暂不考虑错误恢复和处理。

(3)可降低可靠性和软件质量标准。

(4)原型界面部分要设计得简单易学,最好能与最终系统的界面相容。

(5)根据不同的软件类型和应用领域,可使用不同风格的高级语言来构造原型。

##### 3. 原型的开发环境

(1)交互式系统。(2)数据库管理系统。(3)通用输入/输出条件。(4)重用代码库。

#### (五) 增量模型的评价

##### 1. 原型的作用

(1)为软件系统提供明确的需求说明。

(2)原型可作为新颖设计思想的实现工具及高风险开发的安全因素,证实设计的可行性。

(3)原型模型支持软件产品的演化,对开发过程中的问题和错误具有应付变化的机制。

(4)原型模型鼓励用户参与开发过程,参与原型的运行和评价,能充分地与开发者协调一致。

开发期间,原型可作为终端用户的教学环境。

##### 2. 使用原型的建议

能够使用原型的情况：

- (1)开发周期很长的项目。
- (2)系统的使用可能变化较大,不能相对稳定。
- (3)用户对系统的需求较为模糊,对某种要求缺乏信心。
- (4)开发者对系统的某种设计方案的实现无信心或无十分的把握。

不宜使用原型的情况：

- (1)缺乏开发工具,或对原型的可用工具不了解的时候。
- (2)用户不愿意参与开发。
- (3)用户的数据资源没有很好地组织和管理的时候。
- (4)用户的软件资源没有被组织和管理起来的时候。

### 3.原型的优点

- (1)可及早为用户提供有用的产品。
- (2)可及早发现问题,随时纠正错误。
- (3)减少技术、应用风险,缩短开发时间,减少费用,提高生产率。
- (4)通过实际运行原型,提供直接评价系统的方法,促使用户主动参与开发活动,加强了信息反馈,促进各类人员的协调,减少误解,适应需求的变化,能有效地提高系统的质量。

### 4.存在问题

- (1)缺乏丰富而强有力的软件工具和开发环境。
- (2)缺乏有效的管理机制,还未建立起自己的开发标准。
- (3)对设计人员水平及开发环境要求较高。
- (4)在多次重复改变原型的过程中,程序员会感到厌倦。
- (5)系统的易变性对测试有一定影响,难于做到彻底测试,更新文档较为困难。

## 反馈测试题解

### 一、名词解释

#### 1. 软件原型

答:软件的一个早期可运行的版本,反映了最终系统的重要特性。

#### 2. 增量开发

答:在项目开发周期内,以一定的时间间隔开发部分工作软件。

#### 3. 增量提交

答:项目开发周期内,以一定的时间间隔增量方式向用户提交工作软件及相应文档。

#### 4. 原型

答:模拟某种产品的原始模型。

#### 5. 探索型原型

答:这种类型的原型模型是把原型用于开发的需求分析阶段,目的是要弄清用户的需求,确定所期望的特性,并探索各种方案的可行性。它主要针对开发目标模糊,用户与开发者对项目都缺乏经验的情

况,通过对原型的开发来明确用户的需求。

#### 6. 实验型原型

答:这种原型主要用于设计阶段,考核实现方案是否合适,能否实现。对于一个大型系统,若对设计方案心中没有把握时,可通过这种原型来证实设计方案的正确性。

#### 7. 演化型原型

答:这种原型主要用于及早向用户提交一个原型系统,该原型系统或者包含系统的框架,或者包含系统的主要功能,在得到用户的认可后,将原型系统不断扩充演变为最终的软件系统。它将原型的思想扩展到软件开发的全过程。

## 二、填空题

1 增量模型是在项目的开发过程中以一系列的\_\_\_\_\_开发系统。增量方式包括\_\_\_\_\_和\_\_\_\_\_。

答:增量方式 增量开发 增量提交

2 增量开发是指在项目开发周期内,以一定的时间间隔\_\_\_\_\_;增量提交是指在项目开发周期内,以一定的时间间隔\_\_\_\_\_向用户提交\_\_\_\_\_及\_\_\_\_\_。

答:开发部分工作软件 增量方式 工作软件 相应文档

3 增量构造模型是在瀑布模型基础上,对一些阶段进行\_\_\_\_\_,对另一些阶段进行\_\_\_\_\_。

答:整体开发 增量开发

4 增量构造模型是指,在前面的开发阶段按\_\_\_\_\_进行\_\_\_\_\_,后面的开发阶段按\_\_\_\_\_开发。

答:瀑布模型 整体开发 增量方式

5 原型模型又称\_\_\_\_\_,它是\_\_\_\_\_的另一种形式。

答:快速原型模型 增量模型

6 演化提交模型在\_\_\_\_\_的基础上,所有阶段都进行\_\_\_\_\_,也就是说不仅是\_\_\_\_\_,也是\_\_\_\_\_。

答:瀑布模型 增量开发 增量开发 增量提交

7 对于探索型,用原型过程来代替\_\_\_\_\_,把原型作为\_\_\_\_\_的补充形式,运用原型尽可能使需求说明完整、一致、准确、无二义性,但在整体上仍采用\_\_\_\_\_。

答:需求分析 需求说明 瀑布模型

8 对于实验型,用原型过程来代替\_\_\_\_\_阶段,即在\_\_\_\_\_阶段引入原型,快速分析\_\_\_\_\_,快速构造原型,通过运行,考察设计方案的\_\_\_\_\_,原型成为设计的\_\_\_\_\_或\_\_\_\_\_的一部分。

答:设计 设计 实现方案 可行性与合理性 总体框架 设计结果

9 原型开发步骤包括\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_。

答:快速分析 构造原型 运行原型 评价原型 修改

10 对于演化型,用原型过程来代替\_\_\_\_\_阶段。这是典型的演化提交模型的形式,它是在强有力的\_\_\_\_\_支持下,通过原型过程的\_\_\_\_\_,直接得到\_\_\_\_\_。

答:全部开发 软件工具和环境 反复循环 软件系统

11 软件开发中的原型是软件的一个\_\_\_\_\_,它反映了最终系统的\_\_\_\_\_。

答:早期可运行的版本 重要特性

12. 由于运用原型的目的和方式不同, 在使用原型时也采取不同的策略, 有\_\_\_\_\_和\_\_\_\_\_。

答: 抛弃策略 附加策略

13. 快速原型是利用原型辅助软件开发的一种新思想。经过简单\_\_\_\_\_, \_\_\_\_\_一个原型, 用户与开发者在\_\_\_\_\_过程中加强通信与反馈, 通过\_\_\_\_\_, 减少误解, 弥补遗漏, 适应变化, 最终提高软件质量。

答: 快速分析 快速实现 试用原型 反复评价和改进原型

14. 探索型类型的原型模型是把原型用于开发的\_\_\_\_\_阶段, 目的是要弄清\_\_\_\_\_, 确定\_\_\_\_\_, 并探索各种方案的\_\_\_\_\_。

答: 需求分析 用户的需求 所期望的特性 可行性

15. 探索型类型的原型模型主要针对开发\_\_\_\_\_, \_\_\_\_\_的情况, 通过对原型的开发来明确\_\_\_\_\_。

答: 目标模糊 用户与开发者对项目都缺乏经验 用户的需求

16. 实验型原型主要用于\_\_\_\_\_阶段, 考核\_\_\_\_\_是否合适和\_\_\_\_\_。

答: 设计 实现方案 能否实现

17. 在同一个应用中的共享是指在同一应用的类层次结构中, 存在继承关系的各相似子类中, 存在数据结构和行为的\_\_\_\_\_, 使各相似子类共享共同的\_\_\_\_\_。使用继承来实现\_\_\_\_\_。

答: 继承 结构和行为 代码的共享

18. 属性指的是类中对象所具有的\_\_\_\_\_。不同对象的同一属性可以具有相同或不同的\_\_\_\_\_。

答: 性质(数据值) 属性值

19. 操作是类中对象所使用的一种\_\_\_\_\_。类中的各对象可以\_\_\_\_\_操作, 方法是类的操作的\_\_\_\_\_。

答: 功能或变换 共享 实现步骤

20. 事件可以看成是信息从一个对象到另一个对象的单向传送, 各事件将信息从一个对象传到另一个对象中去, 因此要确定各事件的\_\_\_\_\_和\_\_\_\_\_。\_\_\_\_\_用来表示事件、事件的接收对象和发送对象。

答: 发送对象 接收对象 事件跟踪图

21. 状态图反映了\_\_\_\_\_与\_\_\_\_\_的关系。状态图确定了由事件序列引起的\_\_\_\_\_。

答: 状态 事件 状态序列

22. 状态是\_\_\_\_\_的抽象。状态指明了对象对\_\_\_\_\_的响应。

答: 对象属性值 输入事件

23. 快速原型模型根据原型的作用, 有\_\_\_\_\_型原型、\_\_\_\_\_型原型、\_\_\_\_\_型原型三类原型模型。

答: 探索 实验 演化

24. 增量模型根据增量的方式和形式的不同, 分为\_\_\_\_\_模型和\_\_\_\_\_模型。

答: 渐增 原型

25. 若原型不满足需求说明, 则根据明确的要求修改\_\_\_\_\_, 若原型不满足用户需求, 则修改和规定\_\_\_\_\_, 重新构造原型。

答: 原型 需求说明

26. 构造原型时, 必须注意\_\_\_\_\_取舍, 忽略一切暂时不关心的部分。

答: 功能性能

27. 用原型过程来代替设计阶段,考察设计方案的可行性与合理性,这是快速原型的\_\_\_\_\_原型。

答:实验型

28. 瀑布模型\_\_\_\_\_适应需求可变的软件开发,只有到\_\_\_\_\_才能见到整个软件系统。

答:不 开发结束

29. 快速原型的思想是在研究\_\_\_\_\_的方法和技术中产生的。

答:需求分析

30. 瀑布模型本质上是一种\_\_\_\_\_顺序模型。

答:线性

31. 用原型过程来代替需求分析,使需求说明完整、一致、准确,这种快速原型是\_\_\_\_\_原型。

答:探索型

32. 瀑布模型属于\_\_\_\_\_开发模型,增量模型属于\_\_\_\_\_开发模型。

答:整体 非整体

33. 用原型过程来代替全部开发阶段,这种快速原型是\_\_\_\_\_原型。

答:演化型

34. 快速原型模型是在\_\_\_\_\_基础上,逐渐完成整个系统的开发工作。

答:原型

35. 增量模型在开发过程中以一系列\_\_\_\_\_开发系统,推迟某阶段的\_\_\_\_\_,从而\_\_\_\_\_产生工作软件。

答:增量方式 细节 尽早

36. 增量构造模型在\_\_\_\_\_阶段按整体方式开发,但是在\_\_\_\_\_阶段按增量方式开发。

答:需求分析、设计 编码、测试

37. 软件原型是软件的\_\_\_\_\_可运行版本,反映最终系统的\_\_\_\_\_。

答:早期 重要特性

38. 演化提交模型中,项目开发各阶段都是\_\_\_\_\_开发的。

答:增量方式

39. 构造原型时,忽略一切暂时不关心的部分,必须注意\_\_\_\_\_取舍。

答:功能性能

40. 演化提交模型中,项目开发各阶段都是按\_\_\_\_\_开发的。

答:增量方式

41. 瀑布模型不适应需求可变的软件开发,只有到开发结束时才能见到\_\_\_\_\_。

答:整个软件系统

42. 增量构造模型在分析和设计阶段按整体方式开发,按增量方式开发的是\_\_\_\_\_阶段。

答:编码和测试

43. 渐增模型有\_\_\_\_\_模型、\_\_\_\_\_模型两类。

答:增量构造 演化提交

44. 由于运用原型的目的和方式不同,在使用原型可采取的策略有\_\_\_\_\_策略和\_\_\_\_\_策略。

答:抛弃 附加

45. 构造原型的技术主要有\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_。

答:可执行的规格说明 基于脚本的设计 采用非常高级语言或专门语言 能重用软件 CO

46. 根据原型的目的、方式及内容的取舍不同,原型特征可分为\_\_\_\_\_四类。

答:系统的界面形式、系统的总体结构、系统的主要处理功能和性能、数据库模式

### 三、选择题

1 通常用于构造原型的技术包括可执行规格说明、( )、自动程序设计、专用语言、可复用的软件和简化假设等等。

- A .代数规格说明
- B 基于脚本的设计
- C .数据流图
- D 原型语言

答: B

2 在原型法中称( )为用户/设计者,开发人员根据有户要求不断修改原型,直到满足用户要求为止。

- A .用户
- B 开发人员
- C .系统分析员
- D 程序员

答: A

3 建立原型的目的不同,实现原型的途径也有所不同,指出下列不正确的类型 ( )

- A .用于验证软件需求的原型
- B 垂直原型
- C .用于验证设计方案的原型
- D 用于演化出目标系统的原型

答: B

4 .( )是指模拟某种产品的原始模型。在软件开发中,它是软件的一个早期的可运行的版本,它反映了最终系统的部分重要特性。

- A .模型
- B 最初模型
- C .原型
- D 进化模型

答: C

5 原型的使用和开发过程,叫做 ( )

- A .原型期
- B 原型生存期
- C .原型周期
- D 以上说法都不对

答: B

6 原型化方法是用户和设计者之间执行的一种交互过程,适用于( )系统。

- A .需求不确定性高的
- B 需求确定的
- C .管理信息
- D 实时

答: A

7 由于软件项目的特点和运行原型的目的不同,原型有三种不同的作用类型:探索性、( )和进化型。

- A .实验型
- B 经验型
- C .追加型
- D 废弃型

答: A

8 使用原型化方法,可以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( ),确认系统作为产品的结果。

- A .系统设计的可行性
- B 开发环境
- C .需求说明
- D 数据流图

答: A

9 原型化方法是一种( )型的设计过程。

- A .自外向内
- B 自顶向下

C.自内向外

D.自底向上

答:A

10. 快速原型模型的主要特点之一是

( )

A. 开发完毕才见到产品

B. 及早提供工作软件

C. 及早提供全部完整软件

D. 开发完毕才见到工作软件

答:B

11. 对于构造原型的建议,以下说法不正确的是

( )

A. 暂不考虑速度、空间等性能效率方面的要求

B. 暂不考虑错误恢复和处理

C. 可降低可靠性和软件质量标准

D. 对于原型界面部分的设计,暂不考虑与最终系统的界面相容

E. 根据不同的软件类型和应用领域,可使用不同风格的高级语言来构造原型

答:D

12. 对于不宜使用原型的情况,以下说法不正确的是

( )

A. 用户对系统的需求较为模糊,对某种要求缺乏信心时,不宜使用原型

B. 用户不愿意参与开发的时候,不宜使用原型

C. 用户的数据资源没有很好地组织和管理的时候,不宜使用原型

D. 用户的软件资源没有被组织和管理起来的时候,不宜使用原型

答:A

13. 以下说法错误的是

( )

A. 快速原型思想是在研究概要设计阶段的方法和技术中产生的

B. 探索型和实验型快速原型采用的是抛弃策略

C. 演化型快速原型采用附加策略

D. 快速原型是利用原型辅助软件开发的一种新思想

答:A

14. 对于原型的使用建议,以下说法不正确的是

( )

A. 开发周期很长的项目,能够使用原型

B. 在系统的使用可能变化较大,不能相对稳定时,能够使用原型

C. 缺乏开发工具,或对原型的可用工具不了解的时候,能够使用原型

D. 开发者对系统的某种设计方案的实现无信心或无十分的把握,能够使用原型

答:C

15. 以下说法错误的是

( )

A. 增量模型是在瀑布模型的基础上加以修改而形成的

B. 增量模型推迟某些阶段或所有阶段中的细节,从而较早的产生工作软件

C. 瀑布模型和增量模型都属于整体开发模型

D. 瀑布模型规定在开始下一个阶段的工作之前,必须完成前一阶段的所有细节

答:C

16. 以下说法正确的是

( )

A. 对于探索型,用原型过程来代替全部开发阶段

B. 对于实验型,用原型过程来代替设计阶段

C. 对于演化型,用原型过程来代替需求分析

答:B

17. 对于原型的作用,以下说法不正确的是( )
- A. 原型为软件系统提供明确的需求说明
  - B. 原型可作为新颖设计思想的实现工具及高风险开发的安全因素,证实设计的可行性
  - C. 原型模型对开发过程中的问题和错误具有应付变化的机制
  - D. 原型模型在用户不愿意参与开发过程的情况下,能充分地鼓励开发者的积极性

答:D

18. 以下说法错误的是 ( )
- A. 对于探索型,把原型作为需求说明的补充形式
  - B. 对于实验型,通过原型过程的反复循环,直接得到软件系统
  - C. 对于演化型,原型成为设计的总体框架或设计结果的一部分

答:A

19. 用于设计阶段,考查实现方案是否可行的是 ( )
- A. 探索型
  - B. 演化型
  - C. 实验型
  - D. 增量型

答:C

20. 用于整个开发阶段,及早提交一个原型系统是( )原型。
- A. 实验型
  - B. 探索型
  - C. 提交型
  - D. 演化型

答:D

21. 快速原型思想是在研究( )阶段的方法技术中产生的。
- A. 软件开发
  - B. 设计
  - C. 编码
  - D. 需求分析

答:D

22. 构造原型时,主要考虑 ( )
- A. 全部功能
  - B. 原型要体现的特征
  - C. 全部细节
  - D. 全部需求

答:B

23. 瀑布模型本质上是一种( )模型。
- A. 线性顺序
  - B. 顺序迭代
  - C. 线性迭代
  - D. 及早见产品

答:A

24. 瀑布模型的问题是 ( )
- A. 用户容易参与开发
  - B. 缺乏灵活性
  - C. 用户与开发者易沟通
  - D. 适用可变需求

答:B

25. 增量模型是一种( )模型。
- A. 整体开发
  - B. 非整体开发
  - C. 灵活性差
  - D. 较晚产生工作软件

答:B

26. 构造原型时,主要考虑 ( )
- A. 全部功能
  - B. 原型要体现的特征
  - C. 全部细节
  - D. 全部需求

答:B

27. 渐增模型是 ( )
- A. 与瀑布模型无关
  - B. 与变换模型有关
  - C. 瀑布模型的改进
  - D. 变换模型的变种



答:C

## 四、简答题

### 1 瀑布模型有何局限性?

答: 尽管传统的瀑布模型曾经给软件产业带来了巨大的进步,部分缓解了软件危机,但这种模型本质上是一种线性顺序模型,因此存在着比较明显的缺点,各阶段之间存在着严格的顺序性和依赖性,特别强调预先定义需求的重要性,在着手进行具体的开发工作之前,必须通过需求分析预先定义并“冻结”软件需求,然后再一步一步地实现这些需求。但是实际项目很少是遵循着这种线性顺序进行的。虽然瀑布模型也允许迭代,但这种改变往往给项目开发带来混乱。在系统建立之前很难只依靠分析就确定出一套完整、准确、一致、有效的用户需求,这种预先定义需求的方法更不能适应用户需求不断变化的情况。

传统的瀑布模型很难适应需求可变、模糊不定的软件系统的开发,而且在开发过程中,用户很难参与进去,只有到开发结束才能看到整个软件系统。这种理想的、线性的开发过程,缺乏灵活性,不适应实际的开发过程。

### 2 增量模型的基本思想是什么?

答: 为了克服瀑布模型的局限性,使开发过程具有一定的灵活性和可修改性,于是产生了增量模型。它是在瀑布模型的基础上加以修改而形成的。

增量模型和瀑布模型之间的本质区别是:瀑布模型属于整体开发模型,它规定在开始下一个阶段的工作之前,必须完成前一阶段的所有细节。而增量模型属于非整体开发模型,它推迟某些阶段或所有阶段中的细节,从而较早地产生工作软件。

增量模型是在项目的开发过程中以一系列的增量方式开发系统。增量方式包括增量开发和增量提交。增量开发是指在项目开发周期内,以一定的时间间隔开发部分工作软件;增量提交是指在项目开发周期内,以一定的时间间隔增量方式向用户提交工作软件及相应文档。增量开发和增量提交可以同时使用,也可单独使用。

有多种增量模型,根据增量的方式和形式的不同,分为渐增模型和原型模型。

### 3 使用快速原型方法构造原型时,对原型的要求是什么?

答: 原型不同于最终系统,两者在功能范围上的区别是最终系统要实现软件需求的全部功能,而原型只实现所选择的部分功能,最终系统对每个软件需求都要求详细实现,而原型仅仅是为了试验和演示用的,部分功能需求可以忽略或者模拟实现。

因此,在构造原型时,必须注意功能性能的取舍,忽略一切暂时不关心的部分以加速原型的实现,同时又要充分体现原型的作用,满足评价原型的要求。

在构造原型之前,必须明确运用原型的目的,从而解决分析与构造内容的取舍,还要根据构造原型的目的确定考核、评价原型的内容。

### 4 构造原型的技术有哪些?

答: 构造原型的技术有:

(1) 可执行的规格说明。这是一种使要求说明过程自动化的技术,通过可执行的规格说明语言来描述预期的行为“做什么”,人们可以从直接观察中用规格说明语言来规定任何系统行为。

(2) 基于脚本的设计。此方法主要用于解决要求的验证问题。一个脚本将模拟在系统运行期间用户经历的事件,它提供了输入—处理—输出的屏幕,以及有关对话的一个模型,开发者能够给用户显示一个系统的逼真视图。

(3) 采用非常高级语言或专门语言。这是一些建模的语言,使用应用领域中的术语,方便了用户和

开发者在计划中的系统的特性的思想交流。

(4)能重用软件。能重用成分是一些具体应用中共同出现的一些程序设计模式,包括输入/输出规格说明、控制结构、一般问题/解法描述等。

5.快速原型模型的开发步骤是什么?

答:原型开发步骤为:

(1)快速分析。在分析人员与用户紧密配合下,迅速确定系统的基本需求,根据原型所要体现的特征(如上述的特征类别)描述基本需求以满足开发原型的需要。这里关键要注意分析与描述内容的选取,围绕运用原型的目标,集中力量确定局部的需求说明,从而尽快开始构造原型。

(2)构造原型。在快速分析的基础上,根据基本需求说明尽快实现一个可运行的系统。这里要求具有强有力的软件工具支持,并忽略最终系统在某些细节上的要求,如安全性、坚固性、例外处理等等,主要考虑原型系统能够充分反映所要评价的特性,而暂时删除一切次要内容。

(3)运行原型。这是发现问题、消除误解、开发者与用户充分协调的一个步骤。由于原型忽略了很多内容,集中反映要评价的特性,外观看来不太完整,用户要在开发者的指导下运行原型,使用过程中努力发现各种不合理的部分,各类人员在共同运用原型的过程中进一步加深对系统的了解及相互之间的理解。

(4)评价原型。在运行的基础上,考核评价原型的特性,分析运行效果是否满足用户的愿望,纠正过去交互中的误解与分析中的错误,增添新的要求,并满足因环境变化或用户新想法引起的系统要求变动,提出全面的修改意见。

(5)修改。根据评价原型的活动结果进行修改。若原型未满足需求说明的要求,说明对需求说明存在不一致的理解或实现方案不够合理,则根据明确的要求迅速修改原型。若原型运行效果不满足用户要求,表明需求说明不准确、不完整、不一致或要求有所变动和增加,则修改和规定新的需求说明,重新构造原型。

修改过程代替了初始的快速分析,从而形成原型开发的循环过程。用户与开发者在这种循环过程中不断接近系统的最终要求。

上述步骤是为了描述方便而划分的。在软件工具支持下,上述各种活动往往交融在一起,或合而为一或交叉进行,运行、评价和修改有可能在各类人员共同使用和随时交互过程中交织在一起,而不再像瀑布模型那样严格地划分阶段,线性推进。

6.评价快速原型模型的优缺点。

答:快速原型模型的优点:

(1)可及早为用户提供有用的产品。

(2)可及早发现问题,随时纠正错误。

(3)减少技术、应用风险,缩短开发时间,减少费用,提高生产率。

(4)通过实际运行原型,提供直接评价系统的方法,促使用户主动参与开发活动,加强了信息反馈,促进各类人员的协调,减少误解,适应需求的变化,能有效提高系统质量。

快速原型模型的缺点:

(1)缺乏丰富而强有力的软件工具和开发环境。

(2)缺乏有效的管理机制,还未建立起自己的开发标准。

(3)对设计人员水平及开发环境要求较高。

(4)在多次重复改变原型的过程中,程序员会感到厌倦。

(5)系统的易变性对测试有一定影响,难于做到彻底测试,更新文档较为困难。

7.使用原型的建议有哪些?

答:能够使用原型的情况:

- (1)开发周期很长的项目,通过原型开发来缩短开发周期。
- (2)系统的使用可能变化较大,不能相对稳定,而原型模型具有适应变化的机制。
- (3)用户对系统的需求较为模糊,对某种要求缺乏信心。
- (4)开发者对系统的某种设计方案的实现无信心或无十分的把握。

上述这些情况均适合于使用原型模型来开发。

不宜使用原型的情况:

- (1)缺乏开发工具,或对原型的可用工具不了解的时候。
- (2)用户不愿意参与开发。
- (3)用户的数据资源没有很好地组织和管理的时候,因为快速原型需要快速寻找和存取数据。
- (4)用户的软件资源没有被组织和管理起来的时候,因为 MIS 中的模型、模块、使用设施和程序的难易程度对原型使用很关键。

#### 8. 原型的作用是什么?

答:(1)为软件系统提供明确的需求说明,当用户要求含糊不清、不完全、不稳定时,通过原型执行、评价,使用户要求明确。

(2)原型可作为新颖设计思想的实现工具,也可作为高风险开发的安全因素,从而证实设计的可行性。

(3)原型模型支持软件产品的演化,对开发过程中的问题和错误具有应付变化的机制。

(4)原型模型鼓励用户参与开发过程,参与原型的运行和评价,能充分地开发者协调一致。

开发期间,原型可作为终端用户的教学环境。

#### 9. 原型的开发环境是什么?

答:除了上述的构造原型的技术和建议外,还应该有开发环境来辅助原型的开发。

- (1)交互式系统。能快速地响应使用者的要求。
- (2)数据库管理系统。能够提供很多工具,可以定义、建立、查询、加工信息资源。
- (3)通用输入/输出软件。容易使用的数据编辑,屏幕格式化软件等对原型设计和开发都有很大帮助。
- (4)重用代码库。可减少重复劳动。

#### 10. 构造原型的建议有哪些?

答:(1)暂不考虑速度、空间等性能效率方面的要求。

(2)暂不考虑错误恢复和处理。

(3)可降低可靠性和软件质量标准。

(4)原型界面部分要设计得简单易学,最好能与最终系统的界面相容。因为原型的界面是与用户通信的窗口,通过这个窗口,用户最容易获取信息和发表自己的意见。

(5)根据不同的软件类型和应用领域,可使用不同风格的高级语言来构造原型。

以上是构造原型的一些建议,它可以减少构造原型的开销,达到快速分析,快速实现的目的。

#### 11. 试述原型开发步骤是什么?

答:(1)快速分析。

在分析人员与用户紧密配合下,迅速确定系统的基本需求,根据原型所要体现的特征(如上述的特征类别)描述基本需求以满足开发原型的需要。这里关键要注意分析与描述内容的选取,围绕运用原型的目标,集中力量确定局部的需求说明,从而尽快开始构造原型。

(2)构造原型。

在快速分析的基础上,根据基本需求说明尽快实现一个可运行的系统。这里要求具有强有力的软件工具支持,并忽略最终系统在某些细节上的要求,如安全性、坚固性、例外处理等等,主要考虑原型系

统能够充分反映所要评价的特性,而暂时删除一切次要内容。例如,如果构造原型的目的在于确定输入界面的形式,则可借助于输入界面自动生成工具(如 FormGenerator),由界面形式的描述和数据域的定义立即生成简单的输入模块,而暂时忽略有关善后处理工作及参照检查、值域检查等内容,从而迅速提供用户使用。如果要利用原型确定系统的总体结构,可借助于菜单生成器迅速实现系统的控制结构,忽略转贮、恢复等维护功能,用户通过运行菜单了解系统的总体结构。总之,在此阶段要求快速实现,尽快投入运行和演示。

### (3)运行原型。

这是发现问题、消除误解、开发者与用户充分协调的一个步骤。由于原型忽略了很多内容,集中反映要评价的特性,外观看来不太完整,用户要在开发者的指导下运行原型,使用过程中努力发现各种不合理的部分,各类人员在共同运用原型的过程中进一步加深对系统的了解及相互之间的理解。

### (4)评价原型。

在运行的基础上,考核评价原型的特性,分析运行效果是否满足用户的愿望,纠正过去交互中的误解与分析中的错误,增添新的要求,并满足因环境变化或用户新想法引起的系统要求变动,提出全面的修改意见。

### (5)修改。

根据评价原型的活动结果进行修改。若原型未满足需求说明的要求,说明对需求说明存在不一致的理解或实现方案不够合理,则根据明确的要求迅速修改原型。若原型运行效果不满足用户要求,说明需求说明不准确、不完整、不一致或要求有所变动和增加,则修改和规定新的需求说明,重新的构造原型。

修改过程代替了初始的快速分析,从而形成原型开发的循环过程。用户与开发者在这种循环过程中不断接近系统的最终要求。

上述步骤是为了描述方便而划分的。在软件工具支持下,上述各种活动往往交织在一起或合而为一或交叉进行。运行、评价和修改有可能在各类人员共同使用和随时交互过程中交织在一起,而不再像瀑布模型那样严格的阶段划分,线性推进。

## 12. 什么是快速原型模型?分哪几类?

答:快速原型模型是另一类增量模型。它是以原型为基础的增量开发模型。可以在原型的基础上,按功能增量方式开发,也可把原型思想用于某一开发阶段来进行开发。原型模型有三类:探索原型模型、实验原型模型、演化原型模型。

## 13. 快速原型的基本思想是什么?

答:快速原型是利用原型辅助软件开发的一种新思想,经过简单快速分析来实现一个原型,用户与开发者在试验原型过程中加强通信与反馈,通过反复评价和改进原型,来减少误解、弥补遗漏、适应变化,最终提高软件质量。

原型思想的第一个要点是原型的目的。软件开发的原型是软件的一个早期可运行版本,只反映最终系统的某些重要特征。如可用原型来解决系统的人机交互界面的结构,或确定系统的体系结构,或确定系统的功能性能,或确定系统的数据结构系统等。

原型要反映系统的某些重要特征,则要进行功能取舍,不能实现软件需求的全部功能,忽略一切暂不关心的部分,但要充分体现原型的作用,满足评价的要求。

原型思想的第二个要点是要快速分析、快速构造原型。根据原型所要体现的特征,迅速确定系统的需求。在快速分析的基础上,尽快实现一个可运行的系统。为了多次对原型进行评价修改,必须要快速分析、快速构造原型,以便进行多次迭代。原型思想的第三个要点是用户参与运行、评价原型。原型是否达到预期目标,必须要求用户参加原型的运行与评价,同时在运行与评价过程中,加强用户与开发者之间的通信,增进思想交流,加强信息反馈,消除误解。

#### 14. 什么是探索原型模型？

答:这是一种把原型思想用于需求分析阶段的快速原型模型。通过对原型的开发,来明确用户的需求。适用于一个开发目标模糊、用户与开发者均缺乏这种项目开发经验的软件项目。一旦通过原型的开发运行,明确了用户的需求,可按瀑布模型的设计、编码、测试进行开发。这种原型模型是在研究需求分析阶段的方法技术中产生的。

#### 15. 什么是实验原型模型？

答:这是一种把原型思想用于设计阶段的快速原型模型。若对于一个大型软件系统的设计方案心中没有把握时,要考核设计方案是否正确、能否实现,可通过原型模型的开发运行来证实。一旦通过原型模型的开发运行,证实设计方案是正确、能实现的,可废弃这个原型,按照开发过程来开发。

#### 16. 什么是演化原型模型？

答:这是一种把原型思想用于整个开发阶段的快速原型模型。以原型为基础,按功能增加方式,把原型系统逐步演化为最终软件产品。其主要目的是及早向用户提交一个原型系统。该原型系统或者包括系统的框架,或者包含系统主要功能。当用户认可后,将原型不断扩充、演变为最终的软件产品,它是在不断与用户交流过程中开发出来的,不断得到用户的反馈意见,随时根据用户意见进行修改完善,因而能满足用户需求。

#### 17. 渐增模型有哪几类？

答:渐增模型是在瀑布模型基础上加以改进而来的增量模型。它是以瀑布模型为基础,按功能增量方式进行增量开发。渐增模型有两类:增量构造模型和演化提交模型。

##### (1) 增量构造模型

增量构造模型是部分阶段进行整体开发,部分阶段进行增量开发。也就是需求分析阶段和设计阶段与瀑布模型一样,按照瀑布模型方式进行开发,而编码和测试阶段不采用瀑布模型方式,而采用增量方式开发,即先对部分功能进行编码,进行测试,将这部分功能提交用户,然后再对另一部分功能进行编码、测试、提交用户。直到把全部功能按这种方式开发完毕。在这种模型中,用户可以及早看到部分软件功能,可以及早发现问题,以便在其他软件功能开发时及时解决。

##### (2) 演化提交模型

演化提交模型是全部阶段进行增量开发。也就是说把整体软件系统按功能分为若干部分,对每一部分功能用瀑布模型来开发。例如,一个数据处理系统有数据采集和编辑功能、查询功能、统计功能三大部分,按演化模型开发,即先对数据采集与编辑功能进行需求分析、设计、编码、测试,然后将这些功能提交用户,征求用户意见,可及时得到用户的反馈意见;再对查询功能进行需求分析、设计、编码、测试、提交用户;最后对统计功能进行开发。这是一种更彻底的增量开发,可及早发现和解决问题。

#### 18. 原型运用方式有哪些？

答:由于运用原型的目的和方式不同,在使用原型时也采取不同的策略,有抛弃策略和附加策略。

(1) 抛弃策略是将原型用于开发过程的某一阶段,促使该阶段的开发结果更加完整、准确、一致、可靠,该阶段结束后,原型随之作废。探索型和实验型快速原型就是采用此策略的。

(2) 附加策略是将原型用于开发的全过程,原型由最基本的核心开始,逐步增加新的功能和新的需求,反复修改反复扩充,最后发展为用户满意的最终系统,演化型快速原型就采用此策略。

采用何种形式、何种策略运用快速原型主要取决于软件项目的特点、人员素质、可供支持的原型开发工具和技术等,这要根据实际情况的特点来决定。

#### 19. 快速原型的原理是什么？

答:快速原型是利用原型辅助软件开发的一种新思想。经过简单快速分析,快速实现一个原型,弥补遗漏,适应变化,最终提高软件质量。

#### 20. 如何对原型的特征分类？

答: 根据原型的目的和方式不同,构造原型的内容的取舍不同,体现出原型特征有如下类别:

- (1) 系统的界面形式,用原型来解决系统的人机交互界面的结构;
- (2) 系统的总体结构,用原型来确定系统的体系结构;
- (3) 系统的主要处理功能和性能,用原型来实现系统的主要功能和性能;
- (4) 数据库模式,用原型来确定系统的数据库结构。

#### 21. 构造原型的技术有哪些?

- (1) 可执行的规格说明。

这是一种使要求说明过程自动化的技术,通过可执行的规格说明语言来描述预期的行为“做什么”,人们可以从直接观察中用规格说明语言来规定任何系统行为。

- (2) 脚本的设计。

此方法主要用于解决要求的验证问题。一个脚本将模拟在系统运行期间用户经历的事件,它提供了输入——处理——输出的屏幕,以及有关对话的一个模型,开发者能够给用户显示一个系统的逼真视图。

- (3) 采用非常高级语言或专门语言。

这是一些建模的语言,使用应用领域中的术语,方便了用户和开发者在计划中的系统特性的思想交流。

- (4) 能重用软件。

能重用成分是一些具体应用中共同出现的一些程序设计模式,包括输入/输出规格说明、控制结构、一般问题/解法描述等。

#### 22. 原型的优点是什么?

答: (1) 可及早为用户提供有用的产品。

- (2) 可及早发现问题,随时纠正错误。

- (3) 减少技术、应用风险,缩短开发时间,减少费用,提高生产率。

(4) 通过实际运行原型,提供直接评价系统的方法,促使用户主动参与开发活动,加强了信息反馈,促进各类人员的协调,减少误解,适应需求的变化,能有效提高系统质量。

## 第十章 面向对象的方法

软件工程中的传统开发方法虽然提高了软件可靠性、可维护性和生产率,但是并没有解决软件工程所面临的各种问题,于是一种新的面向对象的开发方法应运而生,它对解决软件工程所面临的各种问题有较大的突破,它将日益完善、成熟,并将成为软件开发方法的主流。

本章总的要求是:掌握面向对象的基本思想、基本概念、基本原理,掌握三种模型的基本概念和构造方法,掌握面向对象的分析、面向对象的设计、面向对象的实现的过程。

了解传统开发方法的各种局限性,了解面向对象语言、面向对象技术、面向对象开发法的发展过程。

理解面向对象分析、面向对象设计、面向对象实现的内容、方法和步骤。

深刻理解对象、类、类的层次结构、方法和消息的实质,深刻理解对象模型、动态型、功能模型的元素、结构和构造方法。

掌握的技能是:画对象图、画状态图、画数据流程图。

熟练掌握的技能是:确定对象类、确定关联、确定属性、识别继承。

### 考试要求

#### 1. 面向对象概述

传统开发方法存在的问题,要求达到识记层次。

面向对象的概念,要求达到识记层次。

面向对象的开发方法,要求达到识记层次。

#### 2. 面向对象的模型

##### (1) 对象模型

对象和类、关联、类的层次结构,要求达到领会层次。

对象图,要求达到简单应用层次。

##### (2) 动态模型

事件、状态、行为,要求达到领会层次。

状态图,要求达到简单应用层次。

(3) 功能模型,要求达到领会层次。

#### 3. 面向对象的分析

面向对象分析过程,要求达到领会层次。

建立对象模型,要求达到简单应用层次。

建立动态模型,要求达到简单应用层次。

建立功能模型,要求达到简单应用层次。

#### 4. 面向对象的设计

面向对象设计的准则,要求达到领会层次。

面向对象设计的启发规则,要求达到领会层次。

系统设计,要求达到识记层次。

对象设计,要求达到识记层次。

#### 5. 面向对象实现

程序设计语言,要求达到识记层次。

类的实现,要求达到领会层次。

应用系统的实现,要求达到领会层次。

面向对象的测试,要求达到识记层次。

## 知识重点

### (一) 面向对象的概念

#### 1 .对象

对象是人们要进行研究的任何事物,对象不仅能表示结构化的数据,而且能表示抽象的事件、规则以及复杂的工程实体。对象具有很强的表达能力和描述功能。对象具有状态和操作。对象实现了数据和操作的结合,使数据和操作封装于对象的统一体中。

#### 2 .类

具有相同或相似性质的对象的抽象就是类。因此,对象的抽象是类,类的具体化就是对象,也可以说类的实例是对象。类具有属性,它是对象的状态的抽象,用数据结构来描述类的属性。类具有操作,它是对象的行为的抽象,用操作名和实现该操作的方法来描述。

#### 3 .类的结构

在客观世界中有若干类,这些类之间有一定的结构关系。通常有两种主要的结构关系,即“一般—具体”结构关系,“整体—部分”结构关系。

#### 4 .消息和方法

对象之间进行通信的构造叫做消息。在对象的操作中,当一个消息发送给某个对象时,消息包含接收对象去执行某种操作的信息。接收消息的对象经过解释,然后给予响应,这种通信机制称为消息传递。

类中操作的实现过程叫做方法。当一个对象接收一条消息后,它所含的方法决定对象怎样动作。方法也可以发送消息给其他对象。请求执行某一动作或提供信息。由于对象的内部对用户是密封的,所以消息只是对象同外部世界连接的管道。对象内部的数据只能被自己的方法所操纵。

#### 5 .面向对象开发方法

当今,面向对象开发方法有 Coad 方法、Booch 方法、OMT 方法和 OOSE 方法等。



## (二) 对象模型

用面向对象方法开发软件,通常需要建立三种形式的模型,它们分别是描述系统数据结构的对象模型、描述系统控制结构的动态模型和描述系统功能的功能模型。这三种模型都涉及到数据、控制和操作等共同的概念,只不过每种模型描述的侧重点不同。这三种模型从三个不同但又密切相关的角度模拟目标系统,它们各自从不同的侧面反映了系统的实质性内容,综合起来则全面反映了对目标系统的需求。

对象模型表示了静态的、结构化的系统数据性质,描述了系统的静态结构,它是从客观世界实体的对象关系角度来描述,表现了对象的相互关系。该模型主要关心系统中对象的结构、属性和操作,使用了对象图的工具来刻画,它是分析阶段三个模型的核心,也是其他两个模型的框架。

动态模型是与时间和变化有关的系统性质,该模型描述了系统的控制结构,它表示了瞬时的、行为化的系统控制性质,它关心的是系统的控制,操作的执行顺序,它从对象的事件和状态的角度出发,表现了对对象的相互行为。该模型描述的系统属性是触发事件、事件序列、状态、事件与状态的组织。使用状态图作为描述工具。

功能模型描述了系统的所有计算。功能模型指出发生了什么,动态模型确定什么时候发生,而对象模型确定发生的客体,功能模型表明一个计算如何从输入值得到输出值,它不考虑所计算的次序。功能模型由多张数据流图组成。数据流图说明数据流是如何从外部输入、经过操作和内部存储输出到外部的。功能模型也包括对象模型中值的约束条件。

功能模型说明对象模型中操作的含义,动态模型中动作的意义以及对象模型中约束的意义。

## (三) 面向对象的分析

面向对象分析的目的是对客观世界的系统进行建模。为了做到这种模型化,必须调查所有需求,分析所有需求的实质含义,并重新严格定义。分析模型应该是问题的精确而又简洁的表示,后继的设计阶段必须参考模型的内容,更重要的是开发早期的错误可以通过分析模型来修正。

## (四) 面向对象的设计

面向对象设计是把分析阶段得到的需求转变成符合成本和质量要求的、抽象的系统实现方案的过程。从面向对象分析到面向对象设计(通常缩写为 OOD),是一个逐渐扩充模型的过程。或者说,面向对象设计是面向对象观点建立求解域模型的过程。

尽管面向对象分析和面向对象设计的定义有明显区别,但是在实际的软件开发过程中二者的界限是模糊的。许多分析结果可以映射成设计结果,而在设计过程中又往往会加深和补充对系统需求的理解,从而进一步完善分析结果。因此,分析和设计活动是一个多次反复迭代的过程。面向对象方法学在概念和表示方法上的一致性,保证了在各项开发活动之间的平滑(无缝)过渡,领域专家和开发人员能够比较容易地跟踪整个系统开发

过程,这是面向对象方法与传统方法比较起来所具有的一大优势。

瀑布模型把设计进一步划分成概要设计和详细设计两个阶段,类似地,也可以把面向对象设计再细分为系统设计和对象设计。系统设计确定实现系统的策略和目标系统的高层结构。对象设计确定空间中的类、关联、接口形式及实现操作的算法。

对象设计要确定实现用到的类、关联的完整定义、接口的形式以及实现操作方法的算法,可以增加实现必须的内部对象,对数据结构和算法进行优化。

对象设计中,必须按照系统设计中确定的设计策略进行设计,完善相应的细节,设计工作的重心必须从强调应用域的概念转到强调计算机实现概念上来。分析中得到的对象可作为设计的框架,要选择相应的方式来实现这个框架。选择方法的标准是尽可能减少执行时间、占用内存少、开销小。分析中得到类、属性和关联等都必须用具体的数据结构来实现,还必须引入新的类来存储中间结果,从而避免重复计算。

### (五)面向对象的实现

面向对象的实现主要包括两项工作:把面向对象设计结果翻译成用某种程序设计语言书写的面向对象程序;测试并调试面向对象的程序。

在开发过程中,类的实现是核心问题。在用面向对象风格所写的系统中,所有的数据都封装在类的实例中。而整个程序则被封装在一个更高级的类中。在使用既存部件的面向对象系统中,可以只花费少量时间和工作量来实现软件。在实现一个特定的软件之前,可以把许多必要的功能事先设计在类中。只要增加类的实例,开发少量的新类和实现各个对象之间互相通信的操作,就能建立需要的软件。

## 反馈测试题解

### 一、名词解释

#### 1. 事件

答:事件是指定时刻发生的某种事情,它是某事情发生的信号。它没有持续时间,是一种相对的快速事件,如移动鼠标、键入命令、拨号、上班、出发等等。事件是引起对象状态变化的控制信息。它是引起对象从一种状态转换成另一状态的现实世界中事情的抽象。事件也是信息从一个对象到另一个对象的单向传送。

#### 2. 属性

答:一个类具有的静态性质和特征,用数据结构来描述。

#### 3. 类

答:具有相同或相似性质的对象的抽象就是类。因此,对象的抽象是类,类的具体化就是对象,也可以说类的实例是对象。

(1)类具有属性,它是对象的状态的抽象,用数据结构来描述类的属性。

(2)类具有操作,它是对象的行为的抽象,用操作名和实现该操作的方法来描述。

#### 4. 操作

答:操作是类中对象所使用的一种功能或变换。类中的各对象可以共享操作,每个操作都有一个目

标对象作为其隐含参数。

#### 5. 方法

答:方法是类的操作实现步骤。例如文件这个类,可以有打印操作,可以设计不同的方法来实现 ASCII 文件的打印、二进制文件的打印、数字图像文件的打印,所有这些方法逻辑上均是做同一工作,即打印文件。因此可以用类中 print 操作去执行它们,但每个方法均是由不同的一段代码来实现。

#### 6. 关联

答:关联表示类之间的一种关系,就是一些可能的链的集合。

#### 7. 链

答:链表示对象间的物理与概念联结,如张三为通达公司工作。

#### 8. 行为

答:一个对象具有的动态性质和特征,用操作来描述。

#### 9. 受限关联

答:受限关联由两个类及一个限定词组成,限定词是一种特定的属性,用来有效地减的重数,限定词在关联的终端对象集中说明。

#### 10. 状态

答:一个对象具有静态性质和特征,用数据值来描述。

#### 11. 角色

答:角色说明类在关联中的作用,它位于关联的端点。二元关联有两个角色,每个角色有各自的角色名称,角色名是用来唯一标识端点的。不同类的关联角色可有可无,同类的关联角色不能省略。

#### 12. 消息传递

答:一个对象向另一对象发送消息,接受消息的对象经过解释,然后给予响应。这种通信机制称为消息传递。通过消息传递实现了对象之间的交互作用。

#### 13. 属性

答:属性指的是类中对象所具有的性质(数据值)。不同对象的同一属性可以具有相同或不同的属性值。类中的各属性名是唯一的。

#### 14. 关系

答:关系是类之间的静态结构关系。有两种结构关系,即一般具有关系和整体部分关系。

#### 15. 多态性

答:多态性是指同一个操作可以作用于不同类的对象上,并获得不同的结果。也就是说,不同对象收到同一消息,可以产生不同的结果。多态性允许每个对象以适合自身的方式去响应共同的消息。

多态性增强了软件的灵活性和重用性,允许更为明确、易懂的方式去建立通用软件,多态性与继承性相结合,使软件具有更广泛的重用性和可扩充性。

#### 16. 状态图

答:它是用于建立动态模型工具,反映了状态和事件的关系。当接收到一事件时,下一状态就取决于当前状态和所接收的该事件。由该事件引起的状态变化称为转换。

在状态图中,状态用圆圈来表示,圆圈内有关状态名,在该状态上可以附有活动,活动表示为“do: 活动名”。转换用带箭头弧线来表示,在弧线上标有事件名,该事件可以附有动作,动作名放在事件名之后,用“/ 动作名”来表示。

#### 17. 操作

答:操作是对象到达某一个状态时要进行的处理。这种处理可以是动作,也可以是活动。动作没有持续时间,活动有持续时间,所以活动依附于状态上,动作依附于事件上。

#### 18. 方法

答:操作的实现过程。

#### 19. 状态

答:状态是对象属性值的抽象,它是对象的某个特定时期所处的某种情形。对象的属性值按照影响对象显著行为的性质,将其并到一个状态中去。状态指明了对象对输入事件的影响。

#### 20. 继承性

答:继承性是指一般具体关系中,子类自动共享父类的属性和操作的限制。

#### 21. 消息

答:对象要执行的操作的一个说明,有三部分:接受消息的对象名、要执行的操作名及相应的参数。消息是对象之间进行通信的构造。

#### 22. 操作

答:一个类具有的动态性质和特征,用操作名和方法来描述。

#### 23. 关联

答:类之间的静态语义联系。若干类之间,除了一般具体和整体部分关系外,还有一种语义上的联系。

## 二、填空题

1 现实世界中,各对象之间相互触发,一个触发行为就是一个\_\_\_\_\_。对事件的响应取决于\_\_\_\_\_的状态,响应包括\_\_\_\_\_的改变或形成一个新的\_\_\_\_\_。

答:事件 接受该触发的对象 状态 触发

2 活动是一种有时间间隔的操作,它是依附于\_\_\_\_\_。动作是一种瞬时操作,它是与\_\_\_\_\_联系在一起的操作。

答:状态的操作 事件

3 在向对象方法中,信息隐蔽通过对象的\_\_\_\_\_来实现。类结构分离了\_\_\_\_\_与\_\_\_\_\_,从而支持了信息隐蔽。

答:封装性 接口 实现

4 功能模型描述了系统的\_\_\_\_\_。功能模型指出\_\_\_\_\_,动态模型确定\_\_\_\_\_,而对对象模型确定\_\_\_\_\_。功能模型由\_\_\_\_\_组成。

答:所有计算 发生了什么 什么时候发生 发生的客体 多张数据流图

5 面向对象的特征是\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_。

答:对象惟一性 分类性 继承性 多态性(多形性)

6 “整体—部分”结构称为\_\_\_\_\_,它们之间的关系是一种\_\_\_\_\_关系,或者是\_\_\_\_\_关系。

答:组装结构 “与” “has a”

7 在类层次中,子类只继承一个父类的数据结构和方法,则称为\_\_\_\_\_。子类继承了多个父类的数据结构和方法,则称为\_\_\_\_\_。

答:单重继承 多重继承

8 抽象是指强调实体的\_\_\_\_\_,忽略一些无关紧要的属性。类实现了对象的\_\_\_\_\_和\_\_\_\_\_的抽象,它是对象的共性的抽象。

答:本质 内在的属性 数据(即状态) 行为

9 面向对象有三个基本要素,它们是\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_。

答:抽象 封装性(信息隐蔽) 共享性

10. 面向对象技术在三个级别上促进了共享。它们是\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_。  
答: 同一个类中的共享 在同一个应用中的共享 在不同应用中的共享
11. 封装性是指所有软件部件内部都有明确的\_\_\_\_\_以及清楚的\_\_\_\_\_。每个软件部件都有友好的\_\_\_\_\_, 软件部件的\_\_\_\_\_与\_\_\_\_\_分离。  
答: 范围 外部边界 界面接口 内部实现 外部可访问性
12. 同一个类中的共享是指同一个类中的对象有着相同\_\_\_\_\_和相同的\_\_\_\_\_。  
答: 数据结构 行为特征
13. 面向对象方法认为系统是由应用域的\_\_\_\_\_组成。  
答: 对象
14. 面向对象程序设计语言与其他程序设计语言的最主要差别是它具有\_\_\_\_\_。  
答: 继承性
15. 对象具有状态, 描述对象的状态用它的\_\_\_\_\_。  
答: 属性值
16. 动态模型描述了系统的\_\_\_\_\_。  
答: 动态行为
17. 对象具有封装性, 实现了\_\_\_\_\_的结合。  
答: 数据和操作
18. 类的实例化是\_\_\_\_\_。  
答: 对象
19. 不同应用中信息共享的这种机制和构造是通过\_\_\_\_\_来实现的。  
答: 类库
20. 类具有属性, 描述类的属性用\_\_\_\_\_。  
答: 数据结构
21. 对象模型的主要元素是类、关联和\_\_\_\_\_。  
答: 关系
22. 子类只继承一个父类的属性和操作, 这称为\_\_\_\_\_。  
答: 单重继承
23. 子类自动共享父类的属性和操作的机制称为\_\_\_\_\_。  
答: 继承
24. 说明一个状态可采用\_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_等内容描述。  
答: 状态名 状态目的描述 产生该状态的事件序列 表示状态特征的事件 在状态中接收的条件
25. 面向对象开发方法有: \_\_\_\_\_方法、\_\_\_\_\_方法、\_\_\_\_\_方法和\_\_\_\_\_方法等。  
答: Booch Coad OMT OOSE
26. 面向对象设计的准则是\_\_\_\_\_。  
答: 模块化 抽象 信息隐蔽 低耦合 高内聚
27. 一般说来, 对面向对象软件的测试可分为\_\_\_\_\_四个层次进行。  
答: 算法层 类层 模板层 系统层
28. 在面向对象的系统设计中, 常见的系统种类有\_\_\_\_\_。大多数问题只是上述结构的变种, 许多问题是多种结构形式的组合。  
答: 批变换 连续变换 交互式接口 动态模拟 实时系统 事务管理
29. 面向对象分析的目的是\_\_\_\_\_。

答:对客观世界的系统进行建模

30. 聚集是一种“\_\_\_\_\_”关系。聚集最重要的性质是\_\_\_\_\_,也具有\_\_\_\_\_。

答:整体——部分 传递性 逆对称性

31. 功能模型由多张数据流图组成。数据流图中包含有\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_对象和\_\_\_\_\_对象。

答:处理 数据流 动作 数据存储

32. 面向对象设计可细分为\_\_\_\_\_设计和\_\_\_\_\_设计。

答:系统 对象

33. 软件系统中存在两种控制流,外部控制流是系统中对象之间外部事件的事件流,有\_\_\_\_\_三种外部事件控制流。\_\_\_\_\_控制流是一个处理内部的控制,均可结构化。

答:过程驱动序列 事件驱动序列 并发序列 内部

34. 在 C++ 和 C 中有一个\_\_\_\_\_函数,可以使用这个过程来说明构成系统主要对象的那些类的实例。

答:main()

35. 在面向对象设计中存在\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_三种内聚。

答:操作内聚 类内聚 一般——具体内聚

36. 事件跟踪图用来表示\_\_\_\_\_,\_\_\_\_\_对象和\_\_\_\_\_对象。\_\_\_\_\_和\_\_\_\_\_可用一条垂直线表示,各\_\_\_\_\_用水平箭头线表示,箭头方向是从\_\_\_\_\_对象指向\_\_\_\_\_对象。

答:事件 事件的接收 发送 接收对象 发送对象 事件 发送 接收

37. 一般化关系是“\_\_\_\_\_”的关系,有一般化类和具体类之分,一般化类又称\_\_\_\_\_类,具体类又称\_\_\_\_\_类,各\_\_\_\_\_类继承了\_\_\_\_\_类的性质,各\_\_\_\_\_类的一些共同性质和操作又归纳到\_\_\_\_\_类中。

答:一般——具体 父 子 子 父 子 父

38. 面向对象的三种分析模型是:\_\_\_\_\_模型、\_\_\_\_\_模型和\_\_\_\_\_模型。

答:对象 动态 功能

39. 对象的抽象是\_\_\_\_\_,类的具体化就是\_\_\_\_\_,也可以说类的实例是\_\_\_\_\_。

答:类 对象 对象

40. 类通常有两种主要的结构关系,即\_\_\_\_\_关系和\_\_\_\_\_结构关系。

答:一般——具体结构 整体——部分

41. 整体——部分结构称为\_\_\_\_\_结构,它们之间的关系是一种“\_\_\_\_\_”关系,或者是“\_\_\_\_\_”关系。

答:组装 与 has a

42. 一个方法有\_\_\_\_\_。

答:方法名、参数、方法体

43. 一般——具体结构关系 称为\_\_\_\_\_结构,也可以说是“\_\_\_\_\_”关系,或者是“\_\_\_\_\_”关系。

答:分类 或 is a

44. 在类的层次结构中,通常上层类称为\_\_\_\_\_类或\_\_\_\_\_类,下层类称为\_\_\_\_\_类。

答:父 超 子

45. 主要的对象类型有\_\_\_\_\_。

答:有形实体、作用、事件、性能说明

46. 对于纯面向对象的语言,在系统中的每个“事物”都是\_\_\_\_\_。在这些语言中没有“\_\_\_\_\_”

过程”,且常常是交互的。

答:对象 主

47. 结构化方法的本质是\_\_\_\_\_,其\_\_\_\_\_,可修改性和可重用性都比较差。

答:功能分解 稳定性

48. 通过\_\_\_\_\_这种机制和构造来实现不同应用中的信息共享。

答:类库

49. 对象具有很强的\_\_\_\_\_能力和\_\_\_\_\_功能。

答:表达 描述

50. 类具有属性,它是对象的\_\_\_\_\_的抽象,用\_\_\_\_\_来描述类的属性。

答:状态 数据结构

51. 对象还有\_\_\_\_\_,用于改变对象的状态。对象实现了\_\_\_\_\_和\_\_\_\_\_的结合。

答:行为 数据 操作

52. 对象具有状态,对象用\_\_\_\_\_来描述它的状态。

答:数据值

53. 对象的抽象是\_\_\_\_\_,类的实例化是\_\_\_\_\_。

答:类 对象

54. 动态模型描述了系统的\_\_\_\_\_结构。

答:控制

55. 继承性是\_\_\_\_\_自动共享父类属性和\_\_\_\_\_的机制。

答:子类 操作

56. 子类只继承\_\_\_\_\_的属性和操作,称为单重继承。

答:一个父类

57. 受限关联由两个类和一个\_\_\_\_\_组成。

答:限定词

58. 类具有\_\_\_\_\_,它是\_\_\_\_\_的行为的抽象。

答:操作 对象

### 三、选择题

1.( )应当在应用分析之前进行,因为我们在了解问题之前应当对问题敞开思想考虑,不应加以限制。

- A .论域分析                      B .高层设计                      C .实例的建立                      D .类的开发

答:A

2 .一个面向对象系统的体系结构通过它的( )的关系确定。

- A .类与对象                      B .成分对象和对象  
C .过程与对象                      D .类与界面

答:B

3 .在软件工程学中,我们把一组具有相同数据结构和相同操作的对象的集合定义为( ),此定义包括一组数据属性和在数据上的一组合法操作。

- A .类                      B .属性                      C .对象                      D .消息

答:A

4 按照传统的生命周期方法开发软件,各阶段工作自顶向下,从抽象到具体顺序进行,我们一般用

( )模型来模拟。

- A .螺线模型                      B 喷泉模型                      C 瀑布模型                      D .椭圆模型

答:B

5 对象是面向对象开发方法的基本成分,每个对象可用它本身的一组( )和它可以执行的一组操作来定义。

- A .服务                      B 参数                      C 属性                      D .调用

答:C

6 面向对象软件技术的许多强有力的功能和突出的优点,都来源于把类组织成一个层次结构的系统,一个类的上层可以有父类,下层可以有子类,这种层次结构系统的一个重要性质是( ),一个类获得其父类的全部描述(数据和操作)。

- A .传递性                      B 继承性                      C .复用性                      D .并行性

答:B

7 在考察系统的一些涉及时序和改变的状况时,要用动态模型来表示。动态模型着重于系统的控制逻辑,它包括两个图:一个是事件追踪图,另一个是( )

- A .数据流图                      B 状态图                      C 系统结构图                      D .时序图

答:B

8 在开发废弃型类时,在软件生存期中最花费时间的部分应当是软件的( )

- A .实现                      B 测试                      C .设计                      D .求精和维护

答:D

9 一个面向对象系统的体系结构通过它的成分对象和对象间的关系确定,与传统的面向数据流的结构化开发方法相比,它具有( )优点。

- A .设计稳定                      B 变换分析                      C 事务分析                      D .模块独立性

答:A

10 面向对象方法学的出发点和基本原则是尽可能模拟人类习惯的思维方式,分析、设计和实现一个软件系统的方法和过程,尽可能接近于人类认识世界解决问题的方法和过程,因此面向对象方法有许多特征,如软件系统是由对象组成的;( );对象彼此间仅能通过传递消息互相联系;层次结构的继承。

- A .开发过程基于功能分析和功能分解  
B .强调需求分析重要性  
C .把对象划分成类,每个对象类都定义一组数据和方法  
D 对既存类进行调整

答:C

11 每个对象可用它自己的一组属性和它可以执行的一组( )来表征。

- A .行为                      B 功能                      C 操作                      D .数据

答:C

12 软件开发过程中,抽取和整理用户需求并建立问题域精确模型的过程叫( )

- A .生存期                      B 面向对象设计  
C .面向对象程序设计                      D 面向对象分析

答:D

13 应用执行对象的操作可以改变该对象的( )

- A .属性                      B 功能                      C 行为                      D .数据

答:A



14.面向对象开发产生的分析文档应当( )考虑问题,在分析阶段识别的概念是高层的概念。

- A .与问题直接相关                      B .与问题不相关  
C .在更小的问题范围内                D .在更大的问题范围内

答:D

15. 功能模型中所有的( )往往形成一个层次结构。在这个层次结构中一个数据流图的过程可由下一层数据流图做进一步的说明。

- A. 数据流图      B. 概念模型图      C. 状态迁移图      D. 事件追踪图

答：A

16.所有的对象都可以成为各种对象类,每个对象类都定义了一组 ( )

- A.说明                      B.方法                      C.过程                      D.类型

答: B

17. 通过执行对象的操作改变该对象的属性,但它必须通过( )的传递。

- A. 接口                      B. 消息                      C. 信息                      D. 操作

答: B

18. 在面向对象的设计中,我们应遵循的设计准则除了模块化、抽象、低耦合、高内聚以外,还有

- A. 隐藏复杂性      B. 信息隐蔽      C. 经常类的复用      D. 类的开发

答: B

19. Rumbaugh 等人提出的对象模型技术 OMT 把分析时收集的信息构造在三类模型中,即对象模型、静态模型和 ( )。

- A. 信息模型      B. 控制模型      C. 功能模型      D. 行为模型

答:C

20. 面向对象的主要特征除对象惟一性、封装、继承外,还有 ( )

- A.多态性                      B.完整性                      C.可移植性                      D.兼容性

答:A

21. 表示对象的相互行为的模型是( )模型。

- A. 对象                      B. 动态                      C. 功能                      D. 静态

答: B

22. 以下说法错误的是 ( )

- A. 数据流图中的处理用来改变数据值
- B. 在一个计算中,用数据流来表示一中间数据值,数据流不能改变数据值
- C. 动作对象是一种主动对象,它通过生成或者使用数据值来驱动数据流图
- D. 数据流图中的数据存储是被动对象,它用来存储数据
- E. 存储和动作对象可以用文件或外部设备来实现

答: E

23. 在面向对象程序设计中,以下能够正确指出为提高健壮性应遵守的主要准则标号的序号是

- ( )

A. 预防用户的操作错误	B. 检查参数的合法性
C. 不要预先确定限制条件	D. 先测试后优化
E. 全面覆盖	F. 尽量不使用全局信息

答:C

24. 以下说法错误的是 ( )
- A. 面向对象分析与面向对象设计的定义没有明显区别
  - B. 在实际的软件开发过程中面向对象分析与面向对象设计的界限是模糊的
  - C. 面向对象分析和面向对象设计活动是一个多次反复迭代的过程
  - D. 从面向对象分析到面向对象设计,是一个逐渐扩充模型的过程

答:A

25. 在面向对象程序设计中,以下能够正确指出为提高重用性应遵守的主要准则标号的序号是 ( )
- |           |           |
|-----------|-----------|
| 提高方法的内聚   | 减少方法规模    |
| 保持方法的一致性  | 把策略与实现分开  |
| 全面覆盖      | 尽量不使用全局信息 |
| 避免使用多分支语句 | 精心确定公有方法  |

- |    |    |
|----|----|
| A. | B. |
| C. | D. |

答:B

26. 以下说法错误的是 ( )
- A. 采用面向对象方法开发软件的基本目的和主要优点是通过重用提高软件的生产率
  - B. 在面向对象程序中,对象是属性(状态)和方法(操作)的封装体
  - C. 在面向对象程序中,对象彼此间通过继承和多态性启动相应的操作
  - D. 继承和多态机制是面向对象程序中实现重用的主要手段

答:C

27. 在面向对象程序设计中,以下能够正确指出为提高可扩充性应遵守的主要准则标号的序号是 ( )
- |            |                |
|------------|----------------|
| 封装实现策略     | 不要用一个方法遍历多条关联链 |
| 避免使用多分支语句  | 精心确定公有方法       |
| 不要预先确定限制条件 | 先测试后优化         |

- |    |    |
|----|----|
| A. | B. |
| C. | D. |

答:C

28. 面向对象的实现主要包括两项工作,以下能正确指出这两项工作标号的序号是 ( )
- 把面向对象设计结果翻译成用某种程序设计语言书写的面向对象程序
  - 测试并调试面向对象的程序
  - 面向对象设计
  - 选择程序设计语言

- |    |    |    |
|----|----|----|
| A. | B. | C. |
| D. | E. | F. |

答:A

29. 在数据流图中,以下说法错误的是 ( )
- A. 数据存储用一条直线表示,线段之上标注存储名
  - B. 动作对象用长方形表示,说明它是一个对象
  - C. 处理用椭圆表示,椭圆中含有对处理的描述

- D. 数据流图中的数据流将对象的输出与处理、处理与对象的输入、处理与处理联系起来
- E. 有些数据流也是对象。在数据流图中,用空三角来表示产品对象的数据流

答:A

30. 以下说法正确的是 ( )

- A. 组装结构可用来描述现实世界中的一般的抽象关系
- B. 分类结构可用来描述现实世界中的类的组成的抽象关系
- C. 面向对象的继承性是子类自动共享父类数据结构和方法的机制
- D. 面向对象的唯一性是指将具有一致的数据结构(属性)和行为(操作)的对象抽象成类

答:C

31. 以下说法正确的是 ( )

- A. 对象模型指出发生了什么
- B. 动态模型确定什么时候发生
- C. 功能模型确定发生的客体
- D. 功能模型描述了系统的所有计算,它考虑所计算的次序

答:B

32. 以下说法正确的是 ( )

- A. 功能模型不包括对象模型中值的约束条件
- B. 功能模型说明对象模型中操作的含义
- C. 功能模型说明动态模型中约束的意义
- D. 功能模型说明对象模型中动作的意义

答:B

33. 在面向对象的系统设计中,以下说法错误的是 ( )

- A. 系统中主要的组成部分称为子系统
- B. 子系统是一个对象或一个功能
- C. 子系统是类、关联、操作、事件和约束的集合
- D. 每次分解的各子系统数目不能太多,最底层子系统称为模块

答:B

34. 以下说法错误的是 ( )

- A. 多态性防止了程序相互依赖性而带来的变动影响
- B. 多态性是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
- C. 多态性与继承性相结合使软件具有更广泛的重用性和可扩充性
- D. 封装性是保证软件部件具有优良的模块性的基础

答:A

35. 以下说法错误的是 ( )

- A. 功能模型是类似编译器之类系统的主要模型
- B. 功能模型由多张数据流图组成
- C. 数据流图不表示控制信息,控制信息在动态模型中表示
- D. 数据流图也不表示对象中值的组织,这种信息在对象模型中表示
- E. 有些数据流也是对象,把对象看成是单纯的数值和把对象看成是包含许多数值的数据存储这二者是相同的

答:E

36. 以下说法错误的是 ( )

- A. 面向对象方法不公支持过程抽象,而且支持数据抽象
- B. 某些面向对象的程序设计语言还支持参数化抽象
- C. 信息隐蔽通过对象的封装性来实现
- D. 在面向对象方法中,类是最基本的模块

答:D

37. 以下说法错误的是 ( )

- A. 对象具有很强的表达能力和描述功能
- B. 对象是人们要进行研究的任何事物
- C. 对象是封装的最基本单位
- D. 类封装比对象封装更具体、更细致

答:D

38. 以下说法正确的是 ( )

- A. 角色说明类在关联中的作用,它位于关联的端点
- B. 不同类的关联角色不能省略
- C. 同类的关联角色可有可无
- D. 在一个类层次结构中,若有多重继承,则该类层次结构是树型层次结构
- E. 在一个类层次结构中,若有单重继承,则该类层次结构是网状层次结构

答:A

39. 同一类中有相同的数据结构,这是( )级别的共享。

- A. 不同应用
- B. 同一应用
- C. 不同类
- D. 同一类

答:D

40. 类库这种机制是( )级别的共享。

- A. 同一类
- B. 不同类
- C. 同一应用
- D. 不同应用

答:D

41. 面向对象分析阶段建立的三个模型中,核心的模型是( )模型。

- A. 功能
- B. 动态
- C. 对象
- D. 分析

答:C

42. 对象模型的描述工具是 ( )

- A. 状态图
- B. 数据流图
- C. 对象图
- D. 结构图

答:C

43. 汽车有一个发动机。汽车和发动机之间的关系是( )关系。

- A. 一般具体
- B. 整体部分
- C. 分类关系
- D. isa

答:B

44. 火车是一种陆上交通工具。火车和陆上交通工业之间的关系是( )关系。

- A. 组装
- B. 整体部分
- C. hasa
- D. 一般具体

答:D

45. 面向对象程序设计语言不同于其他语言的最主要特点是 ( )

- A. 模块性
- B. 抽象性
- C. 继承性
- D. 共享性

答:C

46. 软件部分的内部实现与外部可访问性分离,这是指软件的 ( )

- A. 继承性
- B. 共享性
- C. 封装性
- D. 抽象性

答:C

47. 面向对象开发方法在概念和表示上的一致性保证了分析和设计的( )过渡。  
A. 困难                      B. 不容易                      C. 平滑                      D. 顺序  
答:C
48. 在面向对象方法中,信息隐蔽是通过对象的( )来实现的。  
A. 分类性                      B. 继承性                      C. 封闭性                      D. 共享性  
答:C
49. 动态模型的描述工具是 ( )  
A. 对象图                      B. 结构图                      C. 状态图                      D. 设计图  
答:C
50. 在只有单重继承的类层次结构中,类层次结构是( )层次结构。  
A. 树型                      B. 网状型                      C. 星型                      D. 环型  
答:A
51. 在有多重继承的类层次结构中,类层次结构是( )层次结构。  
A. 树型                      B. 网状型                      C. 环型                      D. 星型  
答:B
52. ( )模型表示了对象的相互行为。  
A. 对象                      B. 动态                      C. 功能                      D. 分析  
答:B
53. 在确定类时,所有( )是候选的类。  
A. 名词                      B. 形容词                      C. 动词                      D. 代词  
答:A
54. 常用动词或动词词组来表示 ( )  
A. 对象                      B. 类                      C. 关联                      D. 属性  
答:C
55. 在确定属性时,所有( )是候选的属性。  
A. 动词                      B. 名词                      C. 修饰性名词词组                      D. 词组  
答:C
56. 描述类中某个对象的行为,反映了状态与事件关系的是 ( )  
A. 对象图                      B. 状态图                      C. 流程图                      D. 结构图  
答:B
57. 有时间间隔的操作是 ( )  
A. 动作                      B. 活动                      C. 加工                      D. 处理  
答:B
58. 与事件联系在一起的瞬时操作是 ( )  
A. 处理                      B. 动作                      C. 活动                      D. 加工  
答:B

四、简答题

1 面向对象设计准则是什么？

答:面向对象设计准则：

(1)模块化。面向对象开发方法很自然地支持了把系统分解成模块的设计原理:对象就是模块。它

是把数据结构和操作这些数据的方法紧密结合在一起所构成的模块。

(2)抽象。面向对象方法不仅支持过程抽象,而且支持数据抽象。类实际上是一种抽象数据类型。它对外开放的公共接口构成了类的规格说明(即协议),这种接口规定了外界可以使用的合法操作,利用这些操作可以对类实例中包含的数据进行操作。使用者无须知道这些操作的实现算法和类中数据元素的具体表示方法,就可以通过这些操作使用类中定义的数据。通常把这类抽象称为规格说明抽象。此外,某些面向对象的程序设计语言还支持参数化抽象。

(3)信息隐蔽。在面向对象方法中,信息隐蔽通过对象的封装性来实现。类结构分离了接口与实现,从而支持了信息隐蔽。对于类的用户来说,属性的表示方法和操作的实现算法都应该是隐蔽的。

(4)低耦合。在面向对象方法中,对象是最基本的模块,因此,耦合主要指不同对象之间相互关联的紧密程度。低耦合是设计的一个重要标准,因为这有助于使得系统中某一部分的变化对其他部分的影响降到最低程度。在理想情况下,对某一部分的理解、测试或修改,无须涉及系统的其他部分。

如果一类对象过多地依赖其他类对象来完成自己的工作,则不仅给理解、测试或修改这个类带来很大困难,而且还将大大降低该类的可重用性和可移植性。

当然,对象不可能是完全孤立的,当两个对象必须相互联系相互依赖时,应该通过类的协议(即公共接口)实现耦合,而不应该依赖于类的具体实现细节。

(5)高内聚。在面向对象设计中存在下述三种内聚: 操作内聚。一个操作应该完成一个且仅完成一个功能。 类内聚。设计类的原则是一个类应该只有一个用途,它的属性和操作应该是高内聚的。类的属性和操作应该全都是完成该类对象的任务所必需的,其中不包含无用的属性或操作。如果某个类有多个用途,通常应该把它分解成多个专用的类。 一般—具体内聚。设计出的一般—具体结构,应该符合多数人的概念。更准确地说,这种结构应该是对相应的领域知识的正确抽取。

一般说来,紧密的继承耦合与高度的一般—具体内聚是一致的。

## 2 继承性和多态性的好处是什么?

答:继承性和多态性的好处是:

在软件开发中,类的继承性使所建立的软件具有开放性、可扩充性,这是信息组织与分类的行之有效的方法,它简化了对象、类的创建工作量,增加了代码的可重用性。采用继承性,提供了类的规范的等级结构。对单重继承,可用树结构来描述,对多重继承,可用格结构来描述。通过类的继承关系,使公共的特性能够共享,提高了软件的重用性。首先进行共同特性的设计和验证,然后自顶向下来开发,逐步加入新的内容,符合逐步细化的原则。通过继承,便于实现多态性。

多态性允许每个对象以适合自身的方式去响应共同的消息。这样就增强了操作的透明性、可理解性和可维护性。用户不必为相同的功能操作但作用于不同类型的对象而费心去识别。

多态性增强了软件的灵活性和重用性。允许使用更为明确、易懂的方式去建立通用软件。多态性与继承性相结合使软件具有更广泛的重用性和可扩充性。

## 3 为什么说用结构化方法开发的软件,其稳定性、可修改性和可重用性都比较差?

答:这是因为结构化方法的本质是功能分解,从代表目标系统整体功能的单个处理着手,自顶向下不断把复杂的处理分解为子处理,这样一层一层地分解下去,直到只剩下若干个容易实现的子处理为止,然后用相应的工具来描述各个最低层的处理。因此,结构化方法是围绕实现处理功能的“过程”来构造系统的。然而,用户需求的变化大部分是针对功能的,因此,这种变化对于基于过程的设计来说是灾难性的。用这种方法设计出来的系统结构常常是不稳定的,用户需求的变化往往造成系统结构的较大变化,从而需要花费很大代价才能实现这种变化。

## 4 系统设计的内容是什么?

答:系统设计是问题求解及建立解答的高级策略。必须制定解决问题的基本方法,系统的高层结构形式包括子系统的分解、它的固有并发性、子系统分配给硬软件、数据存储管理、资源协调、软件控制实

现、人机交互接口。

系统设计阶段先从高层入手,然后细化。系统设计要决定整个结构及风格,这种结构为后面设计阶段的更详细策略的设计提供了基础。

(1)系统分解。系统中主要的组成部分称为子系统,子系统既不是一个对象也不是一个功能,而是类、关联、操作、事件和约束的集合。每次分解的各子系统数目不能太多,最底层子系统称为模块。

(2)确定并发性。分析模型、现实世界及硬件中不少对象均是并发的。系统设计的一个重要目标就是确定哪些必须是同时动作的对象,哪些不是同时动作的对象。后者可放在一起,综合成单个控制线或任务。

(3)处理器及任务分配。各并发子系统必须分配给单个硬件单元,要么是一个一般的处理器,要么是一个具体的功能单元,必须完成下面的工作:估计性能要求和资源需求,选择实现子系统的硬软件,将软件子系统分配给各处理器以满足性能要求和极小化处理器之间的通信,决定实现各子系统的各物理单元的联结。

(4)数据存储管理。系统中的内部数据和外部数据的存储管理是一项重要的任务。通常各数据存储可以将数据结构、文件、数据库组合在一起,不同数据存储要在费用、访问时间、容量及可靠性之间做折衷考虑。

(5)全局资源的处理。必须确定全局资源,并且制定访问全局资源的策略。全局资源包括:物理资源,如处理器、驱动器等;空间,如盘空间、工作站屏幕等;逻辑名字,如对象标识符、类名、文件名等。

如果资源是物理对象,则可以通过建立协议实现对并发系统的访问,以达到自身控制;如果资源是逻辑实体,如对象标识符,那么在共享环境中存在冲突访问的可能,如独立的事务可能同时使用同一个对象标识符,则各个全局资源都必须有一个保护对象,由保护对象来控制对该资源的访问。

(6)选择软件控制机制。分析模型中所有交互行为都表示为对象之间的事件。系统设计必须从多种方法中选择某种方法来实现软件的控制。

(7)人机交互接口设计。设计中的大部分工作都与稳定的状态行为有关,但必须考虑用户使用系统的交互接口。

## 5. 简述三种分析模型的关系。

答:功能模型说明对象模型中操作的含义、动态模型中动作的意义以及对象模型中约束的意义。一些不存在相互作用的系统,如编译器系统,它们的动态模型较小,因为它们的目的是功能处理,功能模型是这类系统的主要模型。

功能模型由多张数据流图组成。数据流图用来表示从源对象到目标对象的数据值的流向。数据流图不表示控制信息,控制信息在动态模型中表示。数据流图也不表示对象中值的组织,这种信息在对象模型中表示。

## 6. 在类的实现中,利用既存类的途径是什么?

答:在类的实现中,利用既存类的途径有三种:

(1)“原封不动”重用。寻找“原封不动”使用的既存类,提供所需要的特性。此时,所需要的类已经存在,建立它的一个实例,用以提供所需要的特性。这个实例可直接为软件利用,或者它可以用来作为另一个类的实现部分。通过重用既存类,可得到不加修改就能工作的已测试的代码。

由于大多数面向对象语言的两个特性,即类的规格说明与实现的分离(信息隐蔽)和封装,这种重用一般是成功的。

(2)进化性重用。此时,一个能够完全符合要求特性的类可能并不存在。但是,如果具有类似功能的类存在,则可以通过继承,由既存类渐进式地设计新类。还可以将几个既存类的特性混合起来开发出新的类。每个既存类是某些概念的模型。混合起来则产生了一个为特定应用的具有多重概念的类。一个既存类可能会提供某些在我们的新类中需要的特性以及某些新类不需要的特性。因此,可以先

建立一个新的更抽象的类,使 成为我们要设计的类的父类,然后,修改既存类以继承新的父类。

(3)“ 废弃性 ”开发。不用任何重用来开发一个新类。虽然不需要使用既存类来演变成新类,但还是有重用的可能。在新类实现时,通过说明一些既存类的实例,可以加快一个类的实现。像表格、硬件接口都可以用来作为一个新类的局部。

#### 7. 面向对象软件的测试如何进行 ?

答:面向对象软件的测试可分为下列四个层次进行:

(1)算法层:测试类中定义的每个方法,基本上相当于传统软件测试中的单元测试。

(2)类层:测试封装在同一个类中的所有 方法与属性之间的相互作用。在面向对象软件中类是基本模块,因此可以认为这是面向对象测试中所特有的模块(单元)测试。

(3)模板层:测试一组协同工作的类之间的相互作用,大体上相当于传统软件测试中的集成测试。

(4)系统层:把各个子系统组装成完整的面向对象软件系统,在组装过程中同时进行测试。

#### 8. 如何建立对象模型 ?

答:建立一个应用问题的对象模型,就是要找出这个应用问题的所有类、类的属性、类的操作以及这些类之间的关系和关联。一个行之有效的方法是对应用问题的陈述进行词法分析,所有的名词及名词词组是可能的类;所有的修饰性名词是可能的属性;所有的描述性动词是可能的关联;描述中的“ .....由 .....组成 ”是可能的整体部分关系;描述中的“ .....分为...”是可能的一般具体关系。除了应用问题的陈述中找这些成分外,还可以在应用领域的知识背景中去找有关的成分。找出这些可能类、属性、操作、关系和关联后,还要根据有关的规则来确定是否是真的类、属性、关系和关联。确认之后,可用这些成分的图形表示来构造对象模型。

#### 9. 功能模型的元素有哪些 ?

答:功能模型中的数据流图包含有处理、数据流、动作对象和数据存储对象。

##### (1)处理

数据流图中的处理用来改变数据值。最低层的处理纯粹是函数,一张完整的数据流图是一个高层的处理。用椭圆来表示处理,椭圆中含有处理的名字,每个处理都有输入数据流和输出数据流。处理用类的方法来实现。

##### (2)数据流

数据流将对象的输出与处理、处理与对象的输入、处理与处理联系起来。数据流用带箭头的直线表示,在其上面标注数据流的名字。

##### (3)动作对象

动作对象是一种主动对象,这通过生成或者使用数据值来驱动数据流图。动作对象位于数据流图的边界上、作为输入流的源点或输出流的终点。动作对象用矩形来表示,矩形中标注动作对象的名字。

##### (4)数据存储对象

数据存储对象是一个被动对象,它用来存储数据,与动作对象不同,本身不产生任何操作,只响应存储和访问操作。数据存储对象用两条平行线来表示。

功能模型的数据流图与结构化分析方法的数据流图的表示相同,只是元素的含义不相同。

#### 10. 功能模型的特点是什么 ?

答:功能模型用来说明值是如何计算的,表明值之间的依赖关系及其相关的功能,它描述了系统的所有计算。动态模型确定什么时候发生,而功能模型指出发生了什么,对象模型确定发生的客体。功能模型表明一个计算如何从输入值到输出值,它不考虑所计算的顺序。功能模型由多张数据流图组成。

#### 11. 对象模型的特征是什么 ?

答:对象模型表示了静态的、结构化的系统数据性质,描述了系统的静态结构,它是从客观世界实体的对象关系角度来描述,表现了对对象的相互关系。该模型主要关心系统中对象的结构、属性和操作,使



用了对象图的工具来刻画。它是分析阶段三个模型的核心,也是其他两个模型的框架。

#### 12. 动态模型的特征是什么?

答:动态模型是与时间和变化有关的系统性质,该模型描述了系统的控制结构。它表示了瞬时的、行为化的系统控制性质。它关心的是系统的控制、操作的执行顺序,它从对象的事件和状态的角度出发,表现了对象的相互行为。用状态图来建立动态模型。

#### 13. 什么是对象?有哪些对象类型?

答:对象是人們要进行研究的任何事物,从最简单的整数到复杂的飞机等均可看作对象,它不仅能表示具体的事物,还能表示抽象的规则、计划或事件,主要有如下的对象类型:

(1)有形实体。指一切看得见、摸得着的实物,如计算机、机房、机器人、工件等。这些都属于有形实体,也是最容易识别的对象。

(2)作用。指人或组织所起的作用,如医生、教师、学生、工人、公司、部门等。

(3)事件。在特定时间所发生的事,如出行、演出、事故、开会等。

(4)性能说明。厂商对产品的性能的说明,如产品名字、型号,各种性能指标等。

对象不仅能表示结构化的数据,而且能表示抽象的事件、规则以及复杂的工程实体。因此,对象具有很强的表达能力和描述功能。

#### 14. 试述面向对象方法的开发过程?

答:(1)面向对象的分析

面向对象分析的目的是对客观世界的系统建立对象模型、动态模型、功能模型。在建立模型之前,设计者必须了解需求以及问题所处的环境,必须调查所有需求,分析所有需求的实质含义。在理解系统需求的基础上,按照上述建立模型的过程进行建模工作。建立模型的工作不是一次就可建好的,复杂问题的模型需要反复构造,先构造模型的子集,然后扩展到整个模型。

##### (2)面向对象的设计

面向对象的设计是把分析阶段得到的分析模型,经过逐步扩充和完善,转变为设计模型。将分析模型的问题域的概念转变为设计模型的实现概念上来。

面向对象的设计也可以细分为系统设计和对象设计。系统设计确定实现系统的策略和目标系统的高层结构,如人机界面的设计、数据管理的设计、软件控制的实现、系统的体系结构等。对象设计确定实现用到的类、关联的完整定义、接口形式及实现操作方法的算法,可以增加实现所必需的内部对象,对数据结构和算法进行优化。

面向对象的设计所遵循的基本原理和启发规则与结构化设计方法类似,但是增加了一些面向对象的新特点。

尽管面向对象分析和面向对象设计是两个阶段,但是在实际的软件开发中二者的界限是模糊的,许多分析结果可以映射为设计结果,而在设计过程中又往往会加深和补充对系统需求的理解,从而进一步完善分析的结果。因此,分析和设计是一个多次反复迭代的过程。

##### (3)面向对象的实现

面向对象的实现包含两方面的工作,把面向对象的设计结果翻译成用某种程序设计语言书写的面向对象程序,然后进行测试并调试面向对象的程序。

面向对象程序设计的质量基本上由面向对象的设计的质量决定。但是与程序设计语言的特点和程序设计风格有关系。

软件测试仍然是保证软件可靠性的主要措施,但是,面向对象的软件也给测试带来了一些新的特点和新问题,如继承性就给测试带来了困难,面向对象程序的执行与面向过程的程序的执行也大不相同,它是事件驱动的,对象彼此间通过消息传递启动相应操作,但是对象并没有明显的规定用什么次序启动它的操作才是合法的。因此传统的测试方法也不再完全适用了。

## 15. 面向对象的特征有哪些？

答: (1) 对象唯一性

每个对象都有自身唯一的标识, 通过这种标识, 可找到相应的对象。在对象的整个生命期中, 它的标识都不改变, 不同的对象不能有相同的标识。在对象建立时, 由系统授予新对象以唯一的对象标识符, 它在历史版本管理中有巨大作用。

(2) 分类性

分类性是指将具有一致的数据结构(属性)和行为(操作)的对象抽象成类。一个类就是这样一种抽象, 它反映了与应用有关的重要性, 而忽略其他一些无关内容。任何类的划分都是主观的, 但必须与具体的应用有关。每个类是具有相同性质的个体对象的集合, 而每个对象是相关类的实例。

(3) 继承性

继承性是子类自动共享父类数据结构和方法的机制, 这是类之间的一种关系。在定义和实现一个类的时候, 可以在一个已经存在的类的基础之上来进行, 把这个已经存在的类所定义的内容做为自己的内容, 并加入若干新的内容。

继承性是面向对象程序设计语言不同于其他语言的最主要的特点, 是其他语言所没有的。

在类层次中, 子类只继承一个父类的数据结构和方法, 则称为单重继承。

在类层次中, 子类继承了多个父类的数据结构和方法, 则称为多重继承。

在软件开发中, 类的继承性使所建立的软件具有开放性、可扩充性, 这是信息组织与分类的行之有效的方法, 它简化了对象、类的创建工作量, 增加了代码的可重用性。

采用继承性, 提供了类的规范的等级结构。对单重继承, 可用树结构来描述; 对多重继承, 可用格结构来描述。通过类的继承关系, 使公共的特性能够共享, 提高了软件重用性。首先进行共同特性的设计和验证, 然后自顶向下来开发, 逐步加入新的内容, 符合逐步细化的原则。通过继承, 便于实现多态性。

(4) 多态性(多形性)

多态性是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。不同的对象, 收到同一消息可以产生不同的结果, 这种现象称为多态性。如 MOVE 操作, 可以是窗口对象的移动操作, 也可以是国际象棋棋子移动的操作。

多态性允许每个对象用适合自身的方式去响应共同的消息。这样就增强了操作的透明性、可理解性和可维护性。用户不必为相同的功能操作但作用于不同类型的对象而费心去识别。

多态性增强了软件的灵活性和重用性。允许使用更为明确、易懂的方式去建立通用软件。多态性与继承性相结合使软件具有更广泛的重用性和可扩充性。

## 16. 类的结构有哪些？

答: 在客观世界中有若干类, 这些类之间有一定的结构关系。通常有两种主要的结构关系, 即一般——具体结构关系, 整体——部分结构关系。

(1) 一般——具体结构称为分类结构, 也可以说是“或”关系, 或者是“isa”关系。例如, 汽车和交通工具都是类。它们之间的关系是一种“或”关系, 汽车“是一种”交通工具。类的这种层次结构可用来描述现实世界中的一般化的抽象关系, 通常越在上层的类越具有一般性和共性, 越在下层的类越具体、越细化。

(2) 整体——部分结构称为组装结构, 它们之间的关系是一种“与”关系, 或者是“hasa”关系。例如, 汽车和发动机都是类, 它们之间是一种“与”关系, 汽车“有一个”发动机。类的这种层次关系可用来描述现实世界中的类的组成的抽象关系。上层的类具有整体性, 下层的类具有成员性。

在类的层次结构中, 通常上层类称为父类或超类, 下层类称为子类。

## 17. 如何建立功能模型？

答:确定输入值、输出值。先列出系统的输入值和输出值,它是系统与外界之间的事件的参数,然后说明这些输入是经过什么处理和内部存储得到输出的,用数据流图中的相应元素把它们表示出来,这就形成了顶层数据流图。顶层数据流图由单个处理组成,也可以由收集输入、计算机值、生成结果的一个综合处理构成。

数据流图通常按层次组织。因此,可将顶层图中的处理扩展为更低层次的数据流图。

#### 18. 面向对象的要素有哪些?

答:面向对象有一些基本要素。虽然这些要素并不是仅为面向对象系统所独有,但这些要素很适合于用来支持面向对象的系统。

##### (1) 抽象。

抽象是指强调实体的本质、内在的属性,忽略一些无关紧要的属性。在系统开发中,抽象指的是在决定如何实现对象之前的对象的意义和行为。使用抽象可以尽可能避免过早考虑一些细节。大多数语言都提供数据抽象机制,运用继承性和多态性强化了这种能力。分析阶段使用抽象仅仅涉及应用域的概念,在理解问题域之前不考虑设计与实现。合理应用抽象可以在分析、设计、程序结构、数据库结构及文档化等过程中使用统一的模型。

面向对象比其他方法技术有更高的抽象性。对象具有极强的抽象表达能力,对象可表示一切事物,可表达结构化的数据,也可表达非结构化的数据,如工程实体、图形、声音、规则等。类实现了对象的数据(即状态)和行为的抽象,它是对象的共性的抽象。

##### (2) 封装性(信息隐蔽)。

封装性是保证软件部件具有优良的模块性的基础。封装性是指所有软件部件内部都有明确的范围以及清楚的外部边界。每个软件部件都有友好的界面接口,软件的内部实现与外部可访问性分离。

面向对象的类是封装良好的模块,类定义将其说明(用户可见的外部接口)与实现(用户不可见的内部实现)显式地分开,其内部实现按其具体定义的作用域提供保护。

对象是封装的最基本单位。在用面向对象的方法解决实际问题时,要创建类的实例,即建立对象,除了应具有的特性外,还应定义仅由该对象所私有的特性。因此,对象封装比类的封装更具体、更细致。

封装防止了程序相互依赖性而带来的变动影响。面向对象的封装比传统语言的封装更为清晰、更为有力。

##### (3) 共享性。

面向对象技术在不同级别上促进了共享。

同一个类中的共享。同一个类中的对象有着相同数据结构。这是由数据成员的类型、定义顺序、继承关系等决定的;也有着相同的行为特征,这是由方法接口和实现决定的。从这个意义上讲,这些对象之间是结构、行为特征的共享关系。进一步,在某些实际应用中还会出现要求这些对象之间有状态(即数据成员值)的共享关系。例如,所有同心圆的类,各个具体圆的圆心坐标值是相同的,即共处于同一状态。

在同一个应用中的共享。在同一应用的类层次结构中,存在继承关系的各相似子类中,存在数据结构和行为的继承,使各相似子类共享共同的结构和行为。使用继承来实现代码的共享,这也是面向对象的主要优点之一。

在不同应用中的共享。面向对象不仅允许在同一应用中共享信息,而且为未来目标的可重用设计准备了条件。通过类库这种机制和结构来实现不同应用中的信息共享。

#### 19. 试述传统开发方法存在的问题。

答:(1) 软件重用性差

重用性是指同一事物不经修改或稍加修改就可多次重复使用的性质。软件重用性是软件工程追求的目标之一,也是节约费用、减少人员、提高软件生产率的重要途径。传统的开发方法,例如结构化方法

等,虽然给软件产生带来巨大进步,但是并没有解决软件重用的问题。同类型的项目,只要需求有一些变化,都要从头开始,原来的系统很难重用。

## (2) 软件可维性差

软件工程强调软件的可维护性,强调文档资料的重要性,规定最终的软件产品应该由完整、一致的配置成分组成。在软件开发过程中,始终强调软件的可读性、可修改性和可测试性是软件的重要的质量指标。但是实践证明,用传统方法开发出来的软件,维护时其费用和成本仍然很高,其原因是可修改性差,维护困难,导致可维护性差。

## (3) 开发出的软件不能满足用户需要

用传统的结构化方法开发大型软件系统涉及各种不同领域知识,在开发需求模糊或需求动态变化的系统时,所开发出的软件系统往往不能真正满足用户的需要。

### 20. 什么是消息和方法?

答:对象之间进行通信的构造叫做消息。在对象的操作中,当一个消息发送给某个对象时,消息包含接收对象去执行某种操作的信息。接收消息的对象经过解释,然后给予响应。这种通信机制称为消息传递。发送一条消息至少要包含说明接收消息的对象名、发送给该对象的消息名(即对象名、方法名),一般还要对参数加以说明,参数可以是认识该消息的对象所知道的变量名,或者是所有对象都知道的全局变量名。

类中操作的实现过程叫做方法,一个方法有方法名、参数、方法体。

### 21. 什么是对象的状态和行为?

答:对象具有状态。一个对象用数据值来描述它的状态,如某个具体的学生张三,他的姓名、年龄、性别、家庭地址、学历、所在学校等,用这些数据值来表示这个具体的学生的情况。

对象还有操作,用于改变对象的状态,对象及其操作就是对象的行为。如某个工人经过“增加工资”的操作后,他的工资额就发生变化。

### 22. 什么是关联的多重性?

答:关联的多重性是指类中有多少个对象与关联的类的一个对象相关。

### 23. 设计简单的类应该是什么?

答:应该尽量设计小而简单的类,这样便于开发和管理。当类很大的时候,要记住它的所有操作是非常困难的。经验表明,如果一个类的定义不超过一页纸(或两屏),则使用这个类是比较容易的。为了使类保持简单,应该注意以下几点:

#### (1) 避免包含过多的属性。

属性过多通常表明这个类过分复杂了,它所完成的功能可能太多了。

#### (2) 有明确的定义。

为了使类的定义明确,分配给每个类的任务应该简单,最好能用一两个简单句子描述它的任务。

#### (3) 尽量简化对象之间的合作关系。

如果需要多个对象协同配合才能做好一件事,则破坏了类的简明性和清晰性。

#### (4) 不要提供太多的操作。

一个类提供的操作过多,同样表明这个类过分复杂。一般地,一个类提供的公共操作不超过 7 个。

在开发大型软件系统时,遵循上述启发规则也会带来另一个问题:设计出大量较小的类,这同样会带来一定的复杂性。解决这个问题的办法是把系统中的类按逻辑分组,也就是划分“模板”。

### 24. 保证设计结果清晰易懂的主要因素有什么?

答:使设计结果清晰、易读、易懂是提高软件可维护性和可重用性的重要措施。显然,人们不会重用那些他们不理解的设计。保证设计结果清晰易懂的主要因素如下:

#### (1) 用词一致。

应该使名字与它所代表的事物一致,而且应该尽量使用人们习惯的名字。不同类中相似操作的名字应该相同。

(2)使用已有的协议。

如果开发同一软件的其他设计人员已经建立了类的协议,或者在使用的类库中 相应的协议,则应该使用这些已有的协议。

(3)减少消息模式的数目。

如果已有标准的消息模式,设计人员应该遵守这些模式。如果确需自己建立消息模式,则应该尽量减少消息模式的数目,只要可能,就使消息具有一致的模式,以利于读者理解。

(4)避免模糊的定义。

一个类的用途应该是有限的,而且应该从类名可以较容易地推想出它的用途。

25. 试述面向对象设计的准则。

答:模块化设计有几条基本原理,这些原理在进行面向对象设计时仍然适用,但是增加了一些与面向对象方法密切相关的新特点,从而具体化为下列的面向对象设计准则:

(1)模块化

面向对象开发方法很自然地支持了把系统分解成模块的设计原理:对象就是模块。它是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。

(2)抽象

面向对象方法不仅支持过程抽象,而且支持数据抽象。类实际上是一种抽象数据类型,它对外开放的公共接口构成了类的规格说明(即协议),这种接口规定了外界可以使用的合法操作,利用这些操作可以对类实例中包含的数据进行操作。使用者无须知道这些操作的实现算法和类中数据元素的具体表示方法,就可以通过这些操作使用类中定义的数据。通常把这类抽象称为规格说明抽象。

此外,某些面向对象的程序设计语言还支持参数化抽象。所谓参数化抽象,它是指当描述类的规格说明时并不具体指定所要操作的数据类型,而是把数据类型作为参数。这使得类的抽象程度更高,应用范围更广,可重用性更高。例如,C++语言提供的“模板”机制就是一种参数化抽象机制。

(3)信息隐蔽

在面向对象方法中,信息隐蔽通过对象的封装性来实现。类结构分离了接口与实现,从而支持了信息隐蔽。对于类的用户来说,属性的表示方法和操作的实现算法都应该是隐蔽的。

(4)低耦合

在面向对象方法中,对象是最基本的模块,因此,耦合主要指不同对象之间相互关联的紧密程度。低耦合是设计的一个重要标准,因为这有助于使得系统中某一部分的变化对其他部分的影响降到最低程度。在理想情况下,对某一部分的理解、测试或修改,无须涉及系统的其他部分。

如果一类对象过多地依赖其他类对象来完成自己的工作,则不仅给理解、测试或修改这个类带来很大困难,而且还将大大降低该类的可重用性和可移植性。显然,类之间的这种相互依赖关系是高耦合的。应该避免对象之间的高耦合,强调对象间的低耦合。

当然,对象不可能是完全孤立的,当两个对象必须相互联系相互依赖时,应该通过类的协议(即公共接口)实现耦合,而不应该依赖于类的具体实现细节。

(5)高内聚

在面向对象设计中存在下述三种内聚:

写出类的符号表示

类名
属性名:类型 = 缺省值
操作名(参数:类型,...):结果类型

类的符号表示

五、应用题

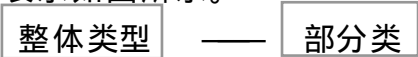
1 简述对象型的特征,举现实世界的例子,给出它的一般化关系、聚集关系的描述。

答:对象模型表示了静态的、结构化的系统数据性质,描述了系统的静态结构,它是从客观世界实体的对象关系角度来描述,表现了对对象的相互关系。该模型主要关心系统中对象的结构、属性和操作,使用了对象图的工具来刻画,它是分析阶段三个模型的核心,也是其他两个模型的框架。

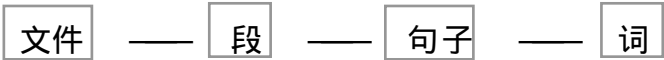
在对象模型中,定义了两种类的层次结构:聚集关系和一般化关系。

聚集是一种“整体—部分”关系。在这种关系中,有整体类和部分类之分。聚集最重要的性质是传递性,也具有逆对称性。

整体类和部分类之间的关系的表示如图所示。

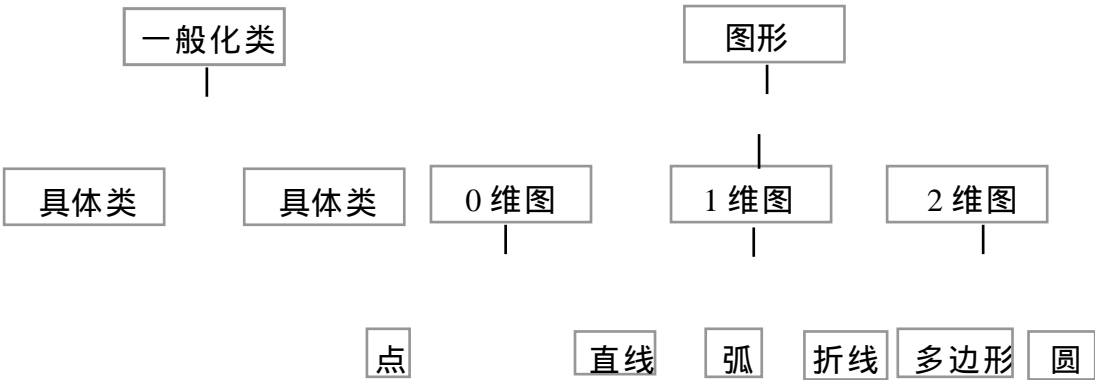


例如,一个字处理应用的对象模型中,有文件、段、句子和词,文件中有多个段,每个段又有多个句子,每个句子又有多个词。它们的关系如下图所示。



一般化关系是在保留对象差异的同时共享对象相似性的一种高度抽象方式。它是“一般—具体”的关系,有一般化类和具体类之分,一般化类又称父类,具体类又称子类,各子类继承了父类的性质,而各子类的一些共同性质和操作又归纳到父类中。因此,一般化关系和继承是同时存在的。

一般化关系的表示如下图所示。



比如,图形可为 0 维图、1 维图、2 维图、1 维图又分为直线、折线、弧,2 维图又分为多边形和圆等。它们的关系如上图所示。

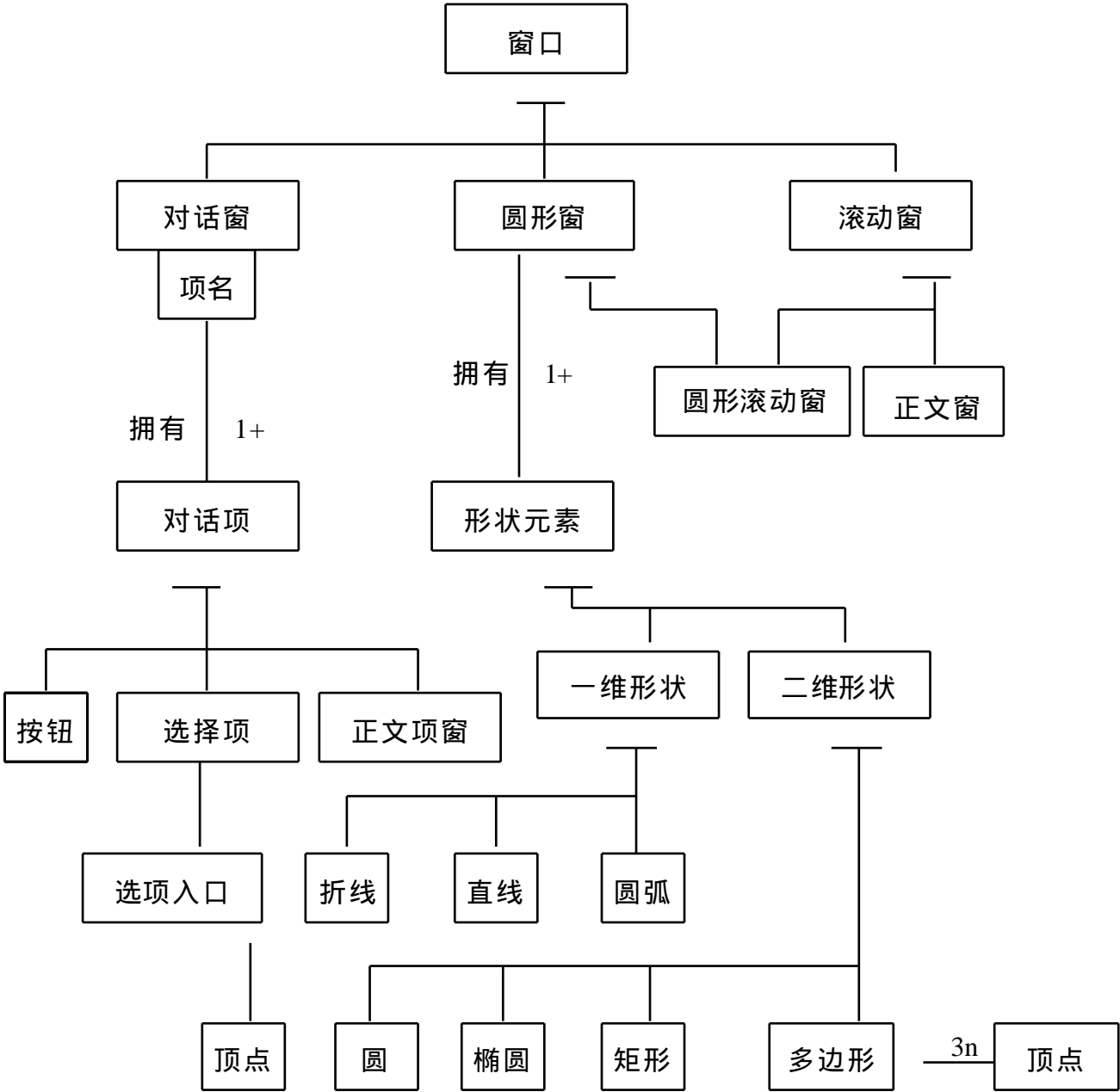
2. 建立窗口系统的对象模型。问题陈述如下:

窗口分为对话框、图形窗、流动窗三种;对话框中有干对话项,由唯一的项名字来确定,对话项分为按钮、选择项、正文项三种,选择项中有若干选择项入口;图形窗中有若干形状元素,形状元素分为一维形状和二维形状,一维形状又分为直线、圆弧、折线;二维形状分为圆、椭圆、矩形、多边形,其中多边形和折线由若干有序顶点组成,正文窗是滚动窗的一种,而图形滚动窗既是一种图形窗又是一种滚动窗。

答:该应用问题的对象模型如下图所示。该问题的类、属性、操作如下表所示。

窗口系统的数据词典

类名	属 性	操 作
窗口	x1,y1,x2,y2	显示,不显示,放大,缩小
对话框		
对话项	x,y,标签	
按钮	串	按下
选择项		
选择项入口	串,值	
正文项	最大长度,当前串	
图形窗		
形状元素	颜色,线宽	画图,擦图,移动
一维形状		
二维形状	填充色,填充式样	
直线	x1,y1,x2,y2	
圆弧	x,y,r,x1,y1,x2,y2	
折线		
圆	x,y,r	
椭圆	x,y,a,b	
矩形	x1,y1,x2,y2	
多边形		
滚动窗		
正文窗	串	插入,删除
图形滚动窗		
顶点	x,y	



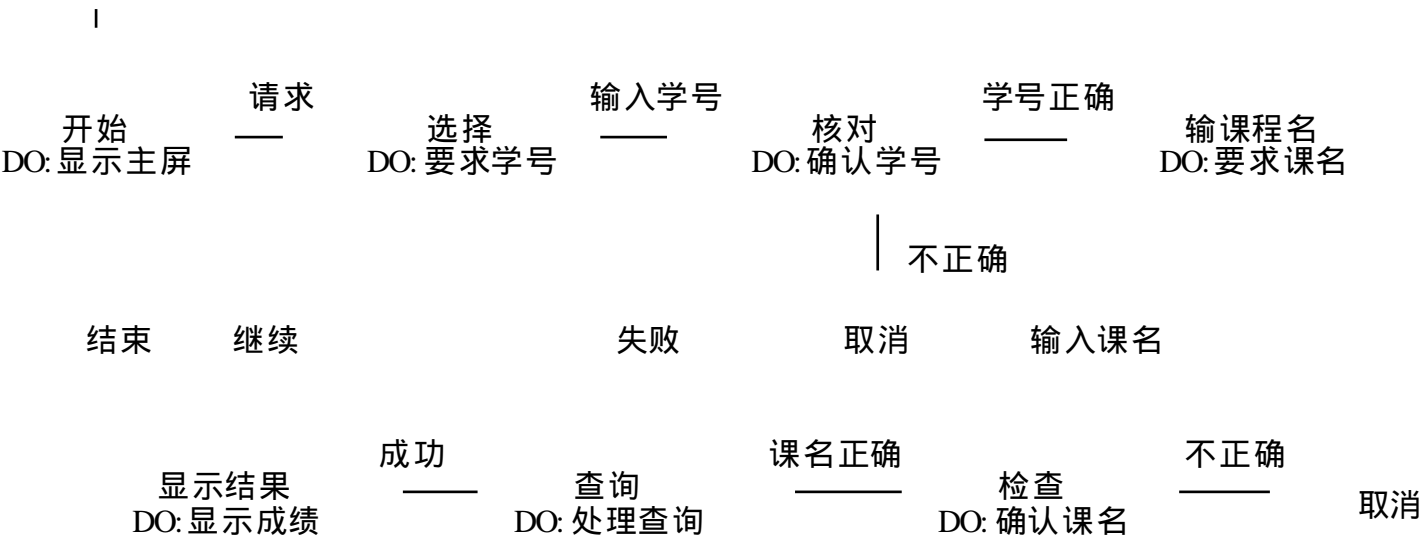
窗口系统的对象模型

3. 在学校教学管理系统中,学生查询成绩就是系统中的一次交互,请用状态图来描述这种查询的交互行为。

答: 该问题的事件跟踪表如下图(a)所示。该问题的状态图下图(b)所示。

用户	终端	学生	成绩
	请求查询		
	要求学号		
	输入学号		
		确认学号	
		学号正确	
	要求课程名		
	输入课程名		
			输入课程名
			课程名正确
			查询
			查询成功
	显示成绩		

图(a) 教学管理系统成绩查询的事件跟踪表



图(b) 教学管理系统成绩查询的状态图



# 第十一章 软件质量与质量保证

软件质量是贯穿软件生存周期的一个极为重要的问题。在软件生存期中特别要重视软件质量保证,以生成高质量的软件产品。

本章总的要求是系统了解软件质量的定义,软件质量的度量与评价、软件质量保证的基本概念;软件质量度量模型、软件复杂性、软件可靠性、设计质量的评审和程序质量评审的具体内容,软件容错技术的概念。

理解软件复杂性的几种常用度量方法;软件质量度量模型;实现容错软件的一般方法和容错软件的实现过程。

深刻理解使用软件质量度量模型、软件复杂性度量模型、软件可靠性模型、容错软件设计过程与软件评审的方法对保证软件质量所起的作用。

## 考试要求

### 1. 概述

软件质量的定义,要求达到领会层次。

软件质量的度量与评价,要求达到识记层次。

软件质量保证,要求达到识记层次。

### 2. 质量度量模型

McCall 质量度量模型,要求达到领会层次。

ISO 质量度量模型,要求达到领会层次。

### 3. 软件复杂性

软件复杂性的基本概念,要求达到识记层次。

软件复杂性的度量方法,要求达到识记层次。

### 4. 软件可靠性

软件可靠性定义,要求达到识记层次。

软件可靠性指标,要求达到识记层次。

软件可靠性模型,要求达到识记层次。

### 5. 软件评审,要求达到识记层次。

### 6. 软件容错技术

容错软件定义,要求达到识记层次。

容错的一般方法,要求达到领会层次。

容错软件的设计过程,要求达到识记层次。

# 知识重点

## (一) 软件质量的概念

### 1. 软件质量

软件质量是与所确定的功能和性能需求的一致性;与所成文的开发标准的一致性;与所有专业开发的软件所期望的隐含特性的一致性。

### 2. 软件质量保证

软件的质量保证就是向用户及社会提供满意的高质量的产品,确保软件产品从诞生到消亡为止所有阶段的的质量的活动,即确定、达到和维护需要的软件质量而进行的所有有计划、有系统的管理活动。它包括的主要功能有:质量方针的制定;质量保证方针和质量保证标准的制定;质量保证体系的建立和管理;明确各阶段的质量保证工作;各阶段的质量评审;确保设计质量;重要质量问题的提出与分析;总结实现阶段的质量保证活动;整理面向用户的文档、说明书等;产品质量鉴定、质量保证系统鉴定;质量信息的收集、分析和使用。

## (二) 质量度量模型

影响较大的软件质量模型是 McCall 质量度量模型和 ISO 的软件质量评价模型。

McCall 质量度量模型是 McCall 等人于 1979 年提出的软件质量模型。针对面向软件产品的运行、修正、转移,软件质量概念包括 11 个特性,McCall 定义了一些评价准则,使用它们对反映质量特性的软件属性分级,以此来估计软件质量特性的值。

## (三) 软件复杂性

软件度量的一个重要分支就是软件复杂性度量。软件复杂性与质量属性有着密切的关系,从某些方面反映了软件的可维护性、可靠性等质量要素。

软件复杂性主要表现在程序的复杂性。程序的复杂性主要指模块内程序的复杂性。它直接关系到软件开发费用的多少、开发周期长短和软件内部潜伏错误的多少。同时它也是软件可理解性的另一种度量。

程序复杂性主要有以下几种度量方法:

### 1. 代码行度量法

此方法的基本考虑是统计一个程序的源代码行数,并以源代码行数作为程序复杂性的度量。代码行度量法只是一个简单的、估计得很粗糙的方法。

### 2. McCabe 度量法

M McCabe 度量法是由 Thomas McCabe 提出的一种基于程序控制流的复杂性度量方法。McCabe 复杂性度量又称环路度量,它认为程序的复杂性很大程度上取决于控制的复杂性。这种方法以图论为工具,先画出程序图,然后用该图的环路数作为程序复杂性的度量值。程序图是退化的程序流程图。也就是说,把程序流程图中每个处理符号都退化成

一个结点,原来连结不同处理符号的流线变成连接不同结点的有向弧,这样得到的有向图就叫做程序图。

计算环路复杂性的方法是,在一个强连通的有向图  $G$  中,环的个数  $V(G)$  由以下公式给出:

$$V(G) = m - n + 2p$$

其中,  $V(G)$  是有向图  $G$  中路数,  $m$  是图  $G$  中弧数,  $n$  是图  $G$  中结点数,  $p$  是  $G$  中的强连通分量个数。

当分支或循环的数目增加时,程序中的环路也随之增加,因此 McCabe 环路复杂度量值实际上是为软件测试的难易程度提供了一个定量度量的方法,同时也间接地表示了软件的可靠性。实验表明,源程序中存在的错误数以及为了诊断和纠正这些错误所需的时间与 McCabe 环路复杂度量值有明显的关系。

尽管 McCabe 复杂度量法有许多缺点,但它容易使用,而且在选择方案和估计排错费用等都有很有效的。

#### (四) 软件可靠性

软件可靠性表明了一个程序按照用户的要求和设计的目标,执行其功能的正确程度。一个可靠的程序就要求是正确的、完整的、一致的和健壮的。但在现实中,一个程序要达到完全可靠是不实际的,要精确地度量它也不现实。在一般情形下只能通过程序的测试去度量程序的可靠性。软件可靠性是指在给定的时间内,在规定的环境条件下系统完成所指定的功能的概率。

软件可靠性与可用性的常用定量指标,是平均失效等待时间 MTTF 与平均失效间隔时间 MTBF, MTTF 是一个描述失效模型或一组失效特性的指标量。MTBF 是指两次相继失效之间的平均时间。MTBF 在实际使用时通常是指当  $n$  很大时,系统第  $n$  次失效与第  $n + 1$  次失效之间的平均时间。

对软件可靠性数学理论的研究尝试已经导致产生了一些有希望的可靠性模型,但关于软件可靠性模型的研究工作尚在初始阶段。

#### (五) 软件评审

在软件生存期每个阶段的工作中都可能引入人为的错误。在某一阶段中出现的错误,如果得不到及时纠正,就会传播到开发的后续阶段中去,并在后续阶段中引出更多的错误。对软件工程过程来说,软件评审是一个“过滤器”,在软件开发的各个阶段都要采用评审的方法,以揭露软件中的缺陷,然后加以改正。

软件质量包括“设计质量”和“程序质量”。设计质量是指设计的规格说明书要符合用户的要求。程序质量是指程序要按照设计规格说明所规定的情况正确执行。

设计质量评审的对象是在需求分析阶段产生的软件需求规格说明、数据需求规格说明,在软件概要设计阶段产生的软件概要设计说明书等,评审内容为:软件的规格说明是否合乎用户的要求;可靠性;保密措施实施情况;操作特性实施情况;性能实现情况;软件是否具有可修改性、可扩充性、可互换性和可移植性;软件是否具有可测试性;软件是否具

有复用性。

程序质量评审通常是从开发者的角度进行评审,直接与开发技术有关。它是着眼于软件本身的结构、与运行环境的接口、变更带来的影响而进行的评审活动。

## (六) 软件容错技术

提高软件质量和可靠性的技术大致可分为两类,一类是避开错误技术,即在开发的过程中不让差错潜入软件的技术;另一类是容错技术,即对某些无法避开的差错,使其影响减至最小的技术。避开错误技术是进行质量管理、实现产品应有质量所必不可少的技术。但是,无论使用多么高明的避开错误技术,也无法做到完美无缺和绝无错误,这就需要采用即使错误发生也不影响系统的特性的容错技术,或使错误发生时对用户影响限制在某些允许的范围内。

实现容错技术的主要手段是冗余。冗余是指实现系统规定功能是多余的那部分资源,包括硬件、软件、信息和时间。由于加入了这些资源,有可能使系统的可靠性得到较大的提高。通常冗余技术分为结构冗余、信息冗余、时间冗余和冗余附加技术。容错消耗了资源,但换来对系统正确运行的保护。这与那种由于设计不当而造成资源浪费的冗余不同。

# 反馈测试题解

## 一、名词解释

### 1. 可移植性

答:将一个软件系统从一个计算机系统或环境移植到另一个计算机系统或环境中运行时所需的工作量。

### 2. 冗余

答:冗余是指实现系统规定功能是多余的那部分资源,包括硬件、软件、信息和时间。

### 3. 可重用性:

答:一个软件(或软件的部件)能再次用于其他相关应用的程度。

### 4. 效率

答:为了完成预定功能,软件系统所需的计算机资源和程序代码数量的程度。

### 5. 适用性

答:修改或改进一个已投入运行的软件所需工作量的程度。

### 6. 软件评审

答:为保证软件质量,在软件生存期每个阶段都要进行评审。软件评审包括软件设计质量评审与软件程序质量评审。

### 7. 软件可靠性

答:软件可靠性是最重要的软件特性。通常它衡量在规定的条件与时间内,软件完成规定功能的能力。

软件的可靠性一般情形只通过程序的测试去度量程序的可靠性。因此可以将可靠性定义为软件在给定的时间间隔的环境条件下,按设计要求,成功地运行程序的概率。

## 8. 软件复杂性

答:对于软件复杂性,至今尚无一种公认的精确定义。软件复杂性与质量属性有着密切的关系,从某些方面反映了软件的可维护性、可靠性等质量要素。软件复杂性度量的参数很多,主要有:规模、难度、结构与智能度。

程序复杂性主要有以下几种度量方法:代码行度量法、McCabe 度量法。

## 二、填空题

1 设计质量评审的对象是在\_\_\_\_\_阶段产生的\_\_\_\_\_,在\_\_\_\_\_阶段产生的\_\_\_\_\_等。

答:需求分析 软件需求规格说明 数据需求规格说明 软件概要设计 软件概要设计说明书

2 软件的规格说明分为外部规格说明和内部规格说明。内部规格说明是为了实现外部规格的更详细的规格,即软件模块结构与模块处理过程的设计(在\_\_\_\_\_阶段进行)。内部规格说明是从\_\_\_\_\_角度来看的规格说明。

答:概要设计与详细设计 开发者

3 软件的规格说明分为外部规格说明和内部规格说明。外部规格说明是从用户角度来看的规格,包括硬件/软件系统设计(在\_\_\_\_\_进行)、功能设计(在\_\_\_\_\_进行)。

答:分析阶段 需求分析阶段与概要设计阶段

4 软件复杂性主要表现在\_\_\_\_\_。\_\_\_\_\_主要指模块内程序的复杂性。它直接关系到\_\_\_\_\_、\_\_\_\_\_和\_\_\_\_\_。

答:程序的复杂性 程序的复杂性 软件开发费用的多少 开发周期长短 软件内部潜伏错误的多少

5 McCabe 质量度量模型,针对面向软件产品的运行、\_\_\_\_\_、\_\_\_\_\_。

答:修正 转移

6 软件质量必须在\_\_\_\_\_中加以保证。如果工程能力不够,或者由于各种失误导致产生软件差错,其结果就会产生软件失效。为了确保每个开发过程的质量,防止把软件差错传递到下一个过程,必须进行\_\_\_\_\_。因此须在软件开发工程的各个阶段\_\_\_\_\_。检验的实施有两种形式:\_\_\_\_\_和\_\_\_\_\_。可在各开发阶段中结合起来使用。

答:设计和实现过程 质量检验 实施检验 实际运行检验(即白盒测试和黑盒测试) 鉴定

7 软件质量定义为:(1)与所确定的\_\_\_\_\_的一致性。(2)与所成文的\_\_\_\_\_的一致性。(3)与所有专业开发的软件所期望的\_\_\_\_\_的一致性。

答:功能和性能需求 开发标准 隐含特性

8 为了提高软件的质量,软件质量保证的任务大致可归结为以下八类:\_\_\_\_\_;

\_\_\_\_\_;

\_\_\_\_\_。

答:正确定义用户要求 技术方法的应用 提高软件开发的工程能力 软件的复用 发挥每个开发者的能力 组织外部力量协作 排除无效劳动 提高计划和管理质量

9 软件质量保证应从\_\_\_\_\_开始,直到投入使用和售后服务的软件生存期的每一阶段中的每一步骤。

答:产品计划 and 设计

10 \_\_\_\_\_就是向用户及社会提供满意的高质量的产品,确保软件产品\_\_\_\_\_的所有阶段的质量的活动,即确定、达到和\_\_\_\_\_需要的软件质量而进行的所有有计划、有系统的管理活动。

答:软件的质量保证 从诞生到消亡为止 维护

11. 把质量保证工作重点放在过程管理上,对制造过程的每一道工序都进行\_\_\_\_\_。

答:质量控制

12. 软件复杂性主要表现在程序的复杂性。程序的复杂性主要指\_\_\_\_\_程序的复杂性。

答:模块内

13. 程序质量评审通常是从开发者的角度进行评审,直接与开发技术有关。它是着眼于\_\_\_\_\_变更带来的影响而进行的评审活动。

答:软件本身的结构与运行环境的接口

14. 软件复杂性度量的参数主要有\_\_\_\_\_、\_\_\_\_\_、结构和智能度。

答:规模 难度

15. McCabe 复杂性度量又称\_\_\_\_\_。

答:环路度量

16. 冗余是指实现系统规定功能是多余的那部分资源,包括硬件、软件、\_\_\_\_\_和\_\_\_\_\_。

答:信息 时间

17. 软件的规格说明分为\_\_\_\_\_。

答:外部规格说明和内部规格说明

18. 质量度量准则是为定量度量\_\_\_\_\_而规定的一些检查项目。

答:质量

19. 我们将评审“设计的规格说明书是否符合用户的要求”称为\_\_\_\_\_评审。

答:设计质量

20. 软件产品从诞生到消亡所有阶段的软件质量的活动是\_\_\_\_\_。

答:软件质量保证

21. 软件可靠性是指在给定的条件与时间内,在规定的环境条件下系统完成所指定的软件功能的\_\_\_\_\_。

答:概率

22. 贯穿软件生存期中的一个极为重要的问题是\_\_\_\_\_。

答:软件质量

23. 在软件的设计中占有极其重要地位的是软件功能结构,它是联系\_\_\_\_\_跟开发者的规格说明。

答:用户

24. 当程序的分支数目或循环数目增加时其复杂度也增加,因此环路的复杂度取决于\_\_\_\_\_的复杂度。

答:程序控制结构

25. 为减少软件开发费用,缩短软件开发周期,减少软件内部潜在错误,可以减少程序复杂性,以提高软件的简单性和\_\_\_\_\_。

答:可理解性

26. 质量保证策略的发展大致可以分为:\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_三个阶段。

答:以检测为重 以过程管理为重 以新产品开发为重

27. 在 McCall 质量度量模型中,针对面向软件产品的运行、修正、转移,软件质量概念包括 11 个特性,面向软件产品操作的五个特性是:\_\_\_\_\_。面向软件产品修改的三个特性是\_\_\_\_\_。面向软件产品适应的三个特性是\_\_\_\_\_。

答:正确性、可靠性、效率、完整性、可用性 可维护性、可测试性、适应性 可移植性、可重用性、可互操作性

28. 提高软件质量和可靠性的技术大致可分为两类,一类是\_\_\_\_\_技术,即在开发的过程中不让差错潜入软件的技术;另一类是\_\_\_\_\_,即对某些无法避开的差错,使其影响减至最小的技术。

答:避开错误 容错技术

29. 为了确保每个开发过程的质量,防止把软件差错传递到下一个过程,必须进行质量检验。检验的实施有两种形式:\_\_\_\_\_和\_\_\_\_\_。可在各开发阶段中结合起来使用。

答:实际运行检验(即白盒测试和黑盒测试) 鉴定

30. 结构冗余按其工作方式,它分为\_\_\_\_\_,\_\_\_\_\_和\_\_\_\_\_冗余三种。

答:静态 动态 混合

31. 常借用硬件可靠性的定量度量方法来度量软件的可靠性与可用性。常用指标有\_\_\_\_\_与\_\_\_\_\_。

答:平均失效等待时间 MTTF 平均失效间隔时间 MTBF

32. 通常,把“质量”理解为“用户满意程度”。为使得用户满意,有两个必要条件:(1)设计的规格说明书要符合用户的要求;(2)程序要按照设计规格说明所规定的情况正确执行;也可把上述条件(1)称为“\_\_\_\_\_质量”,把条件(2)称为“\_\_\_\_\_质量”。

答:设计 程序

33. 混合冗余兼有\_\_\_\_\_冗余和\_\_\_\_\_冗余的长处。

答:静态 动态

34. 静态冗余常用的有:\_\_\_\_\_冗余 TMR 和\_\_\_\_\_冗余。

答:三模 多模

35. 实现容错技术的主要手段是\_\_\_\_\_。通常冗余技术分为\_\_\_\_\_冗余、\_\_\_\_\_冗余、\_\_\_\_\_冗余、冗余\_\_\_\_\_四类。

答:冗余 结构 信息 时间 附加技术

36. 软件的规格说明分为\_\_\_\_\_规格说明和\_\_\_\_\_规格说明。设计质量是由\_\_\_\_\_规格说明决定的,程序质量是由\_\_\_\_\_规格说明决定的。

答:外部 内部 外部 内部

37. 动态冗余的主要方式是\_\_\_\_\_。

答:多重模块待机储备

48. 假如  $n$  个相同的系统(硬件或软件)进行测试,它们的失效时间分别是  $t_1, t_2, \dots, t_n$ ,则平均失效等待时间  $MTTF =$ \_\_\_\_\_。

答:  $\frac{1}{n} \sum_{i=1}^n t_i$

49. 提高软件质量和可靠性的技术大致可分为两类,一类是避开错误技术,另一类是\_\_\_\_\_。

答:容错技术

50. 环路复杂度取决于\_\_\_\_\_的复杂度。当程序的分支数目或循环数目增加时其复杂度也增加。

答:程序控制结构

51. 减少程序复杂性,可提高软件的简单性和\_\_\_\_\_,并使软件开发费用减少,开发周期缩短,软件内部潜藏错误减少。

答:可理解性

52. 软件功能结构是联系\_\_\_\_\_和开发者的规格说明,它在软件的设计中占有极其重要的地位。

答:用户

53. 影响软件质量的因素有\_\_\_\_\_和\_\_\_\_\_两大类。

54. 软件产品制成后才进行检测,这种检测只能判断产品的质量,不能提高\_\_\_\_\_。

55. 软件质量保证就是向用户及社会提供满意的高质量产品,确保软件产品从\_\_\_\_\_到\_\_\_\_\_的所有阶段的质量活动。

56. 设计质量的评审是针对需求分析阶段产生的软件需求、规格说明书、数据需求规格说明,在软件概要设计阶段产生的\_\_\_\_\_。

57. ISO 的软件质量评价模型分为三个层次,其中第一层称为\_\_\_\_\_层;第二层称为\_\_\_\_\_层;第三层称为\_\_\_\_\_层。

58. 为了作好软件质量评价,必须在\_\_\_\_\_定义其质量需求。

### 三、选择题

答：A

答:D

答:D

答:D

答:C

• 210 •



- A .开发方法
- B 测试能力
- C .计划和管理质量
- D 测试和维护的效率

答:C

7 软件质量保证即为了确定、达到和( )需要的软件质量而进行的所有有计划、有系统的管理活动。

- A .测试
- B 维护
- C .质量
- D 效率

答:B

8 软件可靠性表明了一个程序按照用户的要求和设计的目标,执行其功能的正确程度。即“软件可靠性是软件在给定的时间间隔及给定的设计要求,成功地运行程序的( )”。

- A .可靠性
- B 适应性
- C .概率
- D .可移植性

答:C

9 .( )是指能够以数字概念来描述可靠性的数学表达式中所使用的量。

- A .硬件可靠性的定量度量
- B 软件可靠性的定量指标
- C .系统的定量度量
- D .可靠性的度量

答:B

10 .软件质量保证的主要任务有:力争不重复劳动,掌握开发新软件的方法,用户要求定义,组织外部力量协作排除无效劳动,发挥每个开发者的能力,提高软件开发的( ),提高计划和管理质量。

- A 开发方法
- B .工程能力
- C 测试能力
- D .测试和维护的效率

答:B

11 .为了提高软件的质量和( ),软件质量保证的主要任务有:力争不重复劳动,掌握开发新软件的方法等八类任务。

- A 测试
- B .维护
- C 质量
- D .效率

答:D

12 .为了提高软件的质量和效率,软件质量保证的主要任务有:力争不重复劳动,掌握开发新软件的方法,用户要求定义,组织外部力量协作排除( ),发挥每个开发者的能力,提高软件开发的工程能力,提高计划和管理质量。

- A 冗余
- B .无效劳动
- C .开发方法
- D .测试

答:B

13 .为使得用户满意,有两个必要条件: 设计的规格说明符合用户的要求; 程序要按照设计规格说明所规定的情况正确执行。把条件 称为 ( )

- A 程序流程
- B .程序质量
- C .设计要求
- D .设计质量

答:B

14 .人们常借用( )方法来度量软件的可靠性。

- A 硬件可靠性的定量度量
- B .软件可靠性的定量指标
- C 系统的定量度量
- D .可靠性的度量

答:A

15 .程序能够满足规格说明和完成用户业务要求的质量特性称为 ( )

- A. 可靠性
- B. 可用性
- C. 正确性
- D. 完整性

答:C

16. 软件需求是度量软件质量的基础,不符合需求的软件就不具备 ( )

- A. 软件特点
- B. 质量
- C. 软件产品
- D. 功能

答:B

17. 软件特性中,程序能够满足规格说明和完成用户业务目标的程度,称作 ( )

- A. 正确性
- B. 移植性
- C. 可靠性
- D. 完整性

答:A

18. 程序能够按要求的精确度实现与其功能的程度,称作 ( )

- A. 正确性
- B. 移植性
- C. 可靠性
- D. 完整性

答:D

19. 软件或数据不受未授权人控制的程度,称作 ( )

- A. 正确性
- B. 移植性
- C. 可靠性
- D. 完整性

答:A

20. 软件可靠性是最重要的软件特性,通常用它来衡量在规定的条件和时间内,软件完成 ( ) 的能力。

- A. 需求分析
- B. 规定功能
- C. 概要设计
- D. 软件测试

答:B

21. 在 McCall 质量度量模型中,对于以下软件质量概念的解释正确的是 ( )

- A. 正确性。软件按照设计要求,在规定时间和条件下不出故障,持续运行的程度
- B. 可靠性。软件满足设计规格说明及用户预期目标的程度
- C. 效率。为了完成预定功能,软件系统所需的计算机资源和程序代码数量的程度
- D. 完成性。用户熟悉、使用及准备输入和解释输出所需工作量的大小
- E. 可用性。对非授权人访问软件或数据行为的控制程度

答:C

22. 以下说法错误的是 ( )

- A. 质量保证策略的发展大致可以分为:以检测为重、以过程管理为重、以用户监督为重
- B. 以检测为重,这种检测只能判断产品的质量,不能提高产品质量
- C. 以过程管理为重,以制造过程的每一道工序都进行质量控制
- D. 软件质量保证应从产品计划 and 设计开始

答:A

23. 以下说法错误的是 ( )

- A. 当程序的分支数目或循环数目增加时其环路复杂度也增加
- B. 环路复杂性与程序中覆盖的路径条数无关
- C. 若模块 A 的复杂度为 3,模块 B 的复杂度为 4,则模块 A 与模块 B 的复杂度是 7
- D. 对于复杂度超过 10 的程序,应分成几个小程序,以减少程序中的错误

答:B

24. 在 McCall 质量度量模型中,对于以下软件质量概念的解释正确的是 ( )
- A. 可维护性。修改或改进一个已投入运行的软件所需工作量的程度
  - B. 可测试性。找到并改正程序中的一个错误所需代价的程度
  - C. 适应性。将一个系统耦合到另一个系统所需的工作量
  - D. 可重用性。一个软件(或软件的部件)能再次用于其它相关应用的程度。

答:D

25. 以下不属于设计质量评审对象的是 ( )
- A. 在需求分析阶段产生的软件需求规格说明
  - B. 在需求分析阶段产生的数据需求规格说明
  - C. 在软件概要设计阶段产生的软件概要设计说明书
  - D. 在软件详细设计阶段产生的软件的功能结构和功能的通用性

答:D

26. 以下说法错误的是 ( )
- A. 软件复杂性的参数很多,主要有:规模、难度、结构、智能度
  - B. 软件复杂性主要表现在程序的复杂性
  - C. 软件度量就是软件复杂性度量
  - D. 程序的复杂性主要指模块内程序的复杂性

答:C

27. 以下说法错误的是 ( )
- A. McCabe 度量法对于不同种类的控制流的复杂性不能区分
  - B. McCabe 度量法将简单 IF 语句与循环语句的复杂性分别看待
  - C. McCabe 度量法对于嵌套 IF 语句与简单 CASE 语句的复杂性是一样的
  - D. McCabe 度量法将模块接口当成一个简单分支一样处理
  - E. McCabe 度量法看待一个具有 1000 行的顺序程序与一行语句的复杂性相同

答:B

28. 以下说法错误的是 ( )
- A. 程序图描述了程序内部、外部的控制流程
  - B. 程序图完全不表现对数据的具体操作以及分支和循环的具体条件
  - C. 程序图往往把一个简单的 IF 语句与循环语句的复杂性看成是一样的
  - D. 程序图往往把嵌套的 IF 语句与 CASE 语句的复杂性看成是一样的

答:A

29. 以下说法错误的是 ( )
- A. MTTF 是一个描述失效模型或一组失效特性的指标量
  - B. MTBF 是指两次相继失效之间的平均时间
  - C. MTBF 在实际使用时通常指当  $n$  很大时,系统第  $n$  次失效与第  $n+1$  次失效之间的平均时间
  - D. 对于失效率为常数和修复时间很短的情况,MTTF 与 MTBF 差别很大

答:D

30. 在屏蔽软件错误的冗错系统中,冗余附加件的构成不包括的内容是 ( )
- A. 冗余备份程序的存储及调用
  - B. 实现错误检测和错误恢复的程序
  - C. 实现容错误软件所需的固化程序

D. 关键程序和数据冗余存储和调用

答:D

31. 以下说法正确的是 ( )

- A. 外部规格说明是从开发者角度看的规格说明
- B. 内部规格说明是从用户角度看的规格
- C. 程序质量是由外部规格说明决定的
- D. 设计质量是由内部规格说明决定的
- E. 内部规格说明是为了实现外部规格的更详细的规格

答:E

32. 软件质量必须在 ( ) 加以保证。

- A. 设计与实现过程
- B. 开发之前
- C. 开发之后
- D. 开发期间

答:A

33. 为了确保每个开发过程的质量,防止把软件差错传递到下一个过程,必须进行 ( )

- A. 质量检验
- B. 软件容错
- C. 软件维护
- D. 系统容错

答:A

34. 软件质量是贯穿软件 ( ) 的一个极为重要的问题。

- A. 开发
- B. 生存期
- C. 度量
- D. 测试

答:B

35. 可被用于与其实现功能相关的其他应用问题的程序称为 ( )

- A. 可重用性
- B. 可移植性
- C. 可互操作性
- D. 通信共用性

答:A

36. 在软件开发和维护的过程中,为了定量地评价软件质量,必须对软件特性进行 ( )

- A. 测试
- B. 度量
- C. 评审
- D. 维护

答:B

37. 只有高水平的 ( ) 能力才能生产出高质量的软件产品。因此须在软件开发环境或软件工具箱的支持下,运用先进的开发技术、工具和管理方法提高开发软件的能力。

- A. 组织
- B. 开发
- C. 设计
- D. 软件工程

答:D

38. 系统因错误而发生错误时,仍然能在一定程度上完成预期的功能,则把该软件称为 ( )

- A. 容错软件
- B. 系统软件
- C. 测试软件
- D. 操作系统

答:B

39. 对白盒测试和黑盒测试补充的一种有效方法是加强阶段 ( )

- A. 调试
- B. 评审
- C. 维护
- D. 自测试

答:B

40. 在软件开发和维护的过程中,为了定量地评价软件质量,必须对( )进行度量,以测定软件具有要求质量特性的程度。

- A. 软件需求
- B. 软件质量特性
- C. 软件质量
- D. 软件特性

答:C

41. 许多产品的质量问题的源于新产品的开发设计阶段,因此在产品( )阶段就应采取有力措施来消灭由于设计原因而产生的质量隐患。

- A. 软件评审
- B. 软件
- C. 开发设计
- D. 软件度量

答:C

42. 软件特性中,程序能够满足规格说明书,称作( );软件或数据不受未授权人控制,称作( ),程序能够按要求的精确度实现其功能,称作( )

- A. 正确性
- B. 移值性
- C. 可靠性
- D. 完整性

答:A D C

43. 对软件产品,一般有4个方面影响着产品的质量,即( )、过程质量、人员素质及成本、时间和进度等条件。

- A. 概要设计说明书
- B. 需求规格说明书
- C. 详细设计说明书
- D. 开发技术

答:D

44. 质量保证是为保证产品和服务充分满足( )要求需进行的有计划、有组织的活动。

- A. 开发者
- B. 生产者
- C. 测试者
- D. 消费者

答:D

45. McCabe 复杂性度量又称( )

- A. 代码行度量
- B. 环路度量
- C. 程序量度量
- D. 功能性度量

答:D

## 四、简答题

1 软件质量的含义是什么？

答:软件质量是贯穿软件生存期的一个极为重要的问题,关于软件质量的定义有多种说法,从实际应用来说,软件质量定义为:(1)与所确定的功能和性能需求的一致性。(2)与所成文的开发标准的一致性。(3)与所有专业开发的软件所期望的隐含特性的一致性。

上述软件质量定义反映了以下三个方面的问题:(1)软件需求是度量软件质量的基础。不符合需求的软件就不具备质量。(2)专门的标准中定义了一些开发准则,用来指导软件人员用工程化的方法来开发软件。如果不遵守这些开发准则,软件质量就得不到保证。(3)往往会有一些隐含的需求没有明确地提出来。例如,软件应具备良好的可维护性。如果软件只满足那些精确定义了的需求而没有满足这些隐含的需求,软件质量也不能保证。软件质量是各种特性的复杂组合。它随着应用的不同而不同,随着用户提出的质量要求不同而不同。

2 影响软件质量的因素有哪些？

答:影响软件质量的因素可以分为两大类:(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。

### 3 软件质量保证的主要功能是什么?

答:软件的质量保证包括的主要功能有:质量方针的制定;质量保证方针和质量保证标准的制定;质量保证体系的建立和管理;明确各阶段的质量保证工作;各阶段的质量评审;确保设计质量;重要质量问题的提出与分析;总结实现阶段的质量保证活动;整理面向用户的文档、说明书等;产品质量鉴定、质量保证系统鉴定;质量信息的收集、分析和使用。

### 4 说明 McCall 软件质量度量模型。

答:McCall 质量度量模型:

这是 McCall 等人于 1979 年提出的软件质量模型。针对面向软件产品的运行、修正、转移,软件质量概念包括 11 个特性,其定义如下:

#### (1)面向软件产品操作。

正确性。软件满足设计规格说明及用户预期目标的程度。

可靠性。软件按照设计要求,在规定时间和条件下不出故障、持续运行的程度。

效率。为了完成预定功能,软件系统所需的计算机资源和程序代码数量的程度。

完整性。对非授权人访问软件或数据行为的控制程度。

可用性。用户熟悉、使用及准备输入和解释输出所需工作量的大小。

#### (2)面向软件产品修改。

可维护性。找到并改正程序中的一个错误所需代价的程度。

可测试性。测试软件以确保其能够执行预定功能所需工作量的程度。

适应性。修改或改进一个已投入运行的软件所需工作量的程度。

#### (3)面向软件产品适应。

可移植性。将一个软件系统从一个计算机系统或环境移植到另一个计算机系统或环境中运行时所需的工作量。

可重用性。一个软件(或软件的部件)能再次用于其他相关应用的程度。

可互操作性。将一个系统耦合到另一个系统所需的工作量。

通常,对以上各个质量特性直接进行度量是很困难的,在有些情况下甚至是不可能的。因此,McCall 定义了一些评价准则,使用它们对反映质量特性的软件属性分级,以此来估计软件质量特性的值。软件属性一般分级范围从 0(最低)到 10(最高)。主要评价准则定义如下:

可跟踪性。跟踪一个设计说明或一个实际程序部件到原始需求(可追溯)的能力。

完备性。所需功能实现的程度。

一致性。在整个软件开发项目中使用统一的设计和文档编制技术的程度。

安全性。防止软件受到意外的或蓄意的存取、使用、修改、毁坏,或防止失密的程度。

容错性。系统出错时,能以某种预定方式,作出适当处理,得以继续执行和恢复系统的能力。它又称健壮性。

准确性。能达到的计算或控制精度。它又称为精确性。

可审查性。检查与标准是否符合的难易程度。

可操作性。软件操作的难易程度。

可训练性。软件使新用户使用该系统的辅助程度。

简洁性。在不复杂、可理解的方式下,定义和实现软件功能的程度。

简明性,又称可理解性。软件易读的程度。

模块性。软件系统内部接口达到的高内聚、低耦合的程度。

自描述性。对软件功能进行自身说明的程度。

通用性。软件功能覆盖面宽广的程度。

可扩充性。软件的体系结构、数据设计和过程设计的可扩充的程度。

硬件独立性。不依赖于某个特定设备及计算机而能工作的程度。

通信共用性。使用标准接口、协议和带宽的程度。

数据共用性。使用标准数据结构和数据类型的程度。

#### 5 软件复杂性度量的主要参数有哪些？

答: 软件复杂性与质量属性有着密切的关系, 从某些方面反映了软件的可维护性、可靠性等质量要素。软件复杂性度量的参数很多, 主要有: (1) 规模, 即总共的指令数, 或源程序行数。(2) 难度, 通常由程序中出现的操作数的数目所决定的量来表示。(3) 结构, 通常用与程序结构有关的度量来表示。(4) 智能度, 即算法的难易程度。

#### 6 程序复杂性的度量方法有哪些？

答: 程序复杂性的度量方法:

(1) 代码行度量法。此方法的基本考虑是统计一个程序的源代码行数, 并以源代码行数作为程序复杂性的度量。代码行度量法只是一个简单的、估计得很粗糙的方法。

(2) McCabe 度量法。McCabe 度量法是由 Thomas McCabe 提出的一种基于程序控制流的复杂性度量方法。McCabe 复杂性度量又称环路度量, 它认为程序的复杂性很大程度上取决于控制的复杂性。

这种方法以图论为工具, 先画出程序图, 然后用该图的环路数作为程序复杂性的度量值。程序图是退化的程序流程图。也就是说, 把程序流程图中每个处理符号都退化成一个结点, 原来连结不同处理符号的流线变成连接不同结点的有向弧, 这样得到的有向图就叫做程序图。

根据图论, 在一个强连通的有向图  $G$  中, 环的个数  $V(G)$  由以下公式给出:

$$V(G) = m - n + 2p$$

其中,  $V(G)$  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  $G$  中的强连通分量个数。在一个程序中, 从程序图的入口点总能到达图中任何一个结点, 因此, 程序总是连通的, 但不是强连通的。为了使图成为强连通图, 从图的入口点到出口点加一条用虚线表示的有向边, 使图成为强连通图。这样就可以使用上式计算环路复杂性了。

#### 7 McCabe 度量法的缺点是什么？

答: McCabe 度量法的缺点是: (1) 对于不同种类的控制流的复杂性不能区分。(2) 简单 IF 语句与循环语句的复杂性同等看待。(3) 嵌套 IF 语句与简单 CASE 语句的复杂性是一样的。(4) 模块间接口当成一个简单分支一样处理。

#### 8 说明容错软件的定义。

答: 容错软件的定义, 有以下四种:

(1) 规定功能的软件, 在一定程度上对自身错误的作用 (软件错误) 具有屏蔽能力, 则称此软件为具有容错功能的软件, 即容错软件。

(2) 规定功能的软件, 在一定程度上能从错误状态自动恢复到正常状态, 则称之为容错软件。

(3) 规定功能的软件, 在因错误而发生错误时, 仍然能在一定程度上完成预期的功能, 则把该软件称为容错软件。

(4) 规定功能的软件, 在一定程度上具有容错能力, 则称之为容错软件。

#### 9 说明容错系统的设计过程。

答: 容错系统的设计过程包括以下设计步骤:

(1) 按设计任务要求进行常规设计, 尽量保证设计的正确。按常规设计得到非容错结构, 它是容错系统构成的基础。在结构冗余中, 不论是主模块还是备用模块的设计和实现, 都要在费用许可的条件

下,用调试的方法尽可能提高可靠性。

(2)对可能出现的错误分类,确定实现容错的范围。对可能发生的错误进行正确的判断和分类,例如,对于硬件的瞬时错误,可以采用指令复执和程序复算;对于永久错误,则需要采用备份替换或者系统重构。对于软件来说,只有最大限度地弄清错误发生和暴露的规律,才能正确地判断和分类,实现成功的容错。

(3)按照“成本—效率”最优原则,选用某种冗余手段(结构、信息、时间)来实现对各类错误的屏蔽。

(4)分析或验证上述冗余结构的容错效果。如果效果没有达到预期的程度,则应重新进行冗余结构设计。如此反复,直到有一个满意的结果为止。

#### 10. 软件可靠性指标有哪些?

答:软件可靠性与可用性的定量指标,是指能够以数字概念来描述可靠性的数学表达式中所使用的量。人们常借用硬件可靠性的定量度量方法来度量软件的可靠性与可用性。下面主要讨论常用指标平均失效等待时间 MTTF 与平均失效间隔时间 MTBF。

##### (1)MTTF(Mean Time To Failure)

假如我们对  $n$  个相同的系统(硬件或者软件)进行测试,他们的失效时间分别是  $t_1, t_2, \dots, t_n$ , 则平均失效等待时间 MTTF 定义为:

$$MTTF = \frac{1}{n} \sum_{i=1}^n t_i$$

对于软件系统来说,这相当于同一系统在  $n$  个不同的环境(即使用不同的测试用例)下进行测试。因此,MTTF 是一个描述失效模型或一组失效特性的指标量。这个指标的目标值应由用户给出,在需求分析阶段纳入可靠性需求,作为软件规格说明提交给开发部门。在运行阶段,可把失效率函数  $\lambda(t)$  视为常数,则平均失效等待时间 MTTF 是失效率  $\lambda$  的倒数:  $MTTF = 1/\lambda$ 。

##### (2)MTBF(Mean Time Between Failures)

MTBF 是平均失效间隔时间,它是指两次相继失效之间的平均时间。MTBF 在实际使用时通常是指当  $n$  很大时,系统第  $n$  次失效与第  $n+1$  次失效之间的平均时间。对于失效率  $\lambda(t)$  为常数和修复时间(MTTR)很短的情况,MTTF 与 MTBF 几乎相等。

#### 11. 如何提高软件质量和可靠性?

答:提高软件质量和可靠性的技术大致可分为两类,一类是避开错误技术,另一类是容错技术。现在许多高可靠性、高稳定性的系统都非常重视应用容错技术。

#### 12. 实现容错技术的主要手段是什么?

答:实现容错的主要手段是冗余。冗余是指所有对于实现系统规定功能来说是多余的那部分资源,包括硬件、软件、信息与时间。通常冗余的技术有:结构冗余、信息冗余、时间冗余和冗余附加技术。

#### 13. 什么是质量保证?有哪些策略?

答:计算机软件不同于硬件或其他产品,它是一种复杂、抽象的逻辑实体,它所固有的一些特性:抽象性、复杂性、多样性、易变性、难把握性,增加了软件开发的困难。因此,软件质量是贯穿软件生存期的一个极为重要的问题,要很好重视软件质量的问题。质量保证是为了保证产品和服务充分满足消费者要求的质量需进行的有计划、有组织的活动。它包括的主要功能有:质量方针的制定;质量保证方针和质量保证标准的制定;质量保证体系的建立和管理;各阶段的质量评审;确保设计质量;重要质量问题的提出与分析等。

质量保证策略的发展大致可以分为三个阶段:以检测为重;以过程管理为重;以新产品开发为重。软件质量保证应从产品计划 and 设计开始,直到投入使用和售后服务的软件生存期的每一阶段的每一步骤。

#### 14. 质量保证的主要任务是什么?



答:为了提高软件的质量,软件质量保证的任务大致可归结为以下几点:

(1)正确定义用户要求。软件质量保证人员必须正确定义用户的要求。必须十分重视领导全体开发人员收集和积累有关用户业务领域的各种业务资料和技术技能。

(2)技术方法的应用。开发新软件的方法,最普遍公认的成功方法就是软件工程学的方法。标准化、设计方法论、工具化等都属此列。应当在开发新软件的过程中大力使用和推行软件工程学中所介绍的开发方法和工具。

(3)提高软件开发的工程能力。只有高水平的软件工程能力才能生产出高质量的软件产品。因此须在软件开发环境或软件工具箱的支持下,运用先进的开发技术、工具和管理方法提高开发软件的能力。

(4)软件的复用。利用已有的软件成果是提高软件质量和软件生产率的重要途径。为此,不要只考虑如何开发新软件,而首先应考虑哪些已有软件可以复用,并在开发过程中,随时考虑所生产软件的复用性。

(5)发挥每个开发者的能力。软件生产是人的智能生产活动,它依赖于开发组织团队的能力。开发者必须有学习各专业业务知识、生产技术和和管理技术的能动性。管理者或产品服务者要制定技术培训计划、技术水平标准,以及适用于将来需要的中长期技术培训计划。

(6)组织外部力量协作。一个软件自始至终由一软件开发单位来开发也许是最理想的。但在现实中常常难以做到。因此需要改善对外部协作部门的开发管理。必须明确规定进度管理、质量管理、交接检查、维护体制等各方面的要求,建立跟踪检查的体制。

(7)排除无效劳动。最大的无效劳动是因需求规格说明有误、设计有误而造成的返工。定量记录返工工作量,收集和分析返工劳动花费的数据非常重要。另一种较大的无效劳动是重复劳动,即相似的软件在几个地方同时开发。这多是因软件开发计划不当,或者开发信息不流畅造成的。为此,要建立互相交流、信息往来通畅、具有横向交流特征的信息流通网。

(8)提高计划和管理质量。对于大型软件项目来说,提高工程项目管理能力极其重要。它必须重视项目开发初期计划阶段的项目计划评价,计划执行过程中及计划完成报告的评价。将评价、评审工作在工程实施之前就列入整个开发工程的工程计划之中。

## 第十二章 软件工程管理

软件工程管理是对软件生存周期一切活动的管理,尤其是对软件项目开发过程的管理,它对保证高质量的软件产品有着重要意义。

本章总的要求是了解软件产品的各种特点与软件工程管理的内容、软件项目计划内容、软件开发成本估算、软件项目进度安排、软件配置管理等概念。

理解软件开发成本估算的 COCOCM 模型和 Putnan 估算模型;软件项目进度安排中的软件开发任务的并行性,Gantt 图与工程网络图;文档的作用与分类。

深入理解软件项目计划的重要性及各个环节之间的联系,软件工程标准化的意义。

### 考试要求

#### 1. 软件工程管理概述

软件产品的特点,要求达到识记层次。

软件工程管理的重要性,要求达到识记层次。

软件工程管理的内容,要求达到识记层次。

#### 2. 软件项目计划

软件项目计划概念,要求达到领会层次。

软件项目计划内容,要求达到领会层次。

制定软件工程规范,要求达到领会层次。

软件开发成本估算,要求达到领会层次。

风险分析,要求达到识记层次。

软件项目进度安排,要求达到领会层次。

软件质量保证,要求达到简单应用层次。

#### 3. 软件配置管理

基线概念,要求达到识记层次。

软件配置项,要求达到领会层次。

版本控制,要求达到领会层次。

变更控制,要求达到领会层次。

#### 4. 软件工程标准化与软件文档

软件工程标准化的意义,要求达到识记层次。

软件工程标准的层次,要求达到识记层次。

文档的作用与分类,要求达到领会层次。

## 知识重点

### (一) 软件工程管理的内容

软件工程管理的具体内容包括对开发人员、组织机构、用户、文档资料等方面的管理。

软件开发人员一般分为:项目负责人、系统分析员、高级程序员、初级程序员、资料员和其他辅助人员。软件生存期各个阶段的活动既要有分工又要互相联系。因此,要求各类人员既能胜任工作,又要能相互很好地配合。

这里的组织机构要求好的组织结构,合理的人员分工,有效的通讯。软件开发的组织机构没有统一的模式。目前主要有主程序员、专家组、民主组织三种组织机构。

软件是为用户而开发的,在开发过程中自始至终必须得到用户的密切合作和支持。作为项目负责人,要特别注意与用户保持联系,掌握用户的心理和动态,防止来自用户的各种干扰和阻力。

控制包括进度控制、人员控制、经费控制和质量控制。为保证软件开发按预定的计划进行,对开发过程要实施以计划为基础。

软件工程管理很大程度上是通过对文档资料管理来实现的。因此,要把开发过程中的一切初步设计、中间过程、最后结果建立成一套完整的文档资料。文档标准化是文档管理的重要方面。

### (二) 软件项目计划

在软件项目管理过程中一个关键的活动是制定项目计划,它是软件开发工作的第一步。项目计划的目标是为项目负责人提供一个框架,使之能合理地估算软件项目开发所需资源、经费和开发进度,并控制软件项目开发过程按此计划进行。软件项目计划是可行性研究阶段的结果产品。

在做计划时,必须就需要的人力、项目持续时间及成本作出估算。软件项目计划包括两个任务:研究与估算。即通过研究确定该软件项目的主要功能、性能和系统界面,估算项目开发所需的经费、所要使用的资源以及开发进度。

### (三) 软件配置管理

软件配置管理,简称 SCM (Software Configuration Management),它用于整个软件工程过程。其主要目标是:标识变更;控制变更;确保变更正确地实现;报告有关变更。SCM 是一组管理整个软件生存期各阶段中变更的活动。

#### 1. 基线

基线是软件生存期中各开发阶段的一个特定点,它的作用是把开发各阶段工作的划分更加明确化,使本来连续的工作在这些点上断开,以便于检查与肯定阶段成果。因此基线可作为一个检查点,在开发过程中,当采用的基线发生错误时,我们可以知道处于的位置,返回到最近和最恰当的基线上。

## 2. 软件配置项

软件配置项 (Software Configuration Item, SCI) 是软件工程中产生的信息项, 它是配置管理的基本单位, 对已成为基线的 SCI, 虽然可以修改, 但必须按照一个特殊的、正式的过程进行评估, 确认每一处的修改。

## 3. 版本控制与变更控制

随着软件生存期向前推进, SCI 的数量在不断增多, 一些文档经过转换生成另一些文档, 并产生一些信息。另一方面又随时会有新的变更出现, 形成新的版本。

软件工程过程中某一阶段的变更, 均要引起软件配置的变更, 这种变更必须严格加以控制和管理, 保持修改信息, 并把精确、清晰的信息传递到软件工程过程的下一步骤。变更控制包括建立控制点和建立报告与审查制度。对于一个大型软件来说, 不加控制的变更很快会引起混乱。因此变更控制是一项最重要的软件配置任务。

## (四) 软件工程标准化与软件文档

就一个软件开发项目来说, 有许多层次。不同分工的人员相互配合, 在开发项目的各个部分以及各开发阶段之间也都存在着许多联系和衔接问题。如何把这些错综复杂的关系协调好, 需要有一系列统一的约束和规定。

在软件开发项目取得阶段成果或最后完成时, 需要进行阶段评审和验收测试。投入运行的软件, 其维护工作中遇到的问题又与开发工作有着密切的关系。软件的管理工作则渗透到软件生存期的每一个环节。所有这些都要求提供统一的行动规范和衡量准则, 使得各种工作都能有章可循。

根据软件工程标准制定的机构与适用的范围, 它分为国际标准、国家标准、行业标准、企业规范及项目 (课题) 规范五个等级。

文档是指某种数据媒体和其中所记录的数据。在软件工程中, 文档用来表示对需求、工程或结果进行描述、定义、规定、报告或认证的任何书面或图示的信息。它们描述和规定了软件设计及实现的细节, 说明使用软件的操作命令。文档也是软件产品的一部分, 没有文档的软件就不成为软件。软件文档的编制在软件开发工作中占有突出的地位和相当大的工作量。高质量文档对于转让、变更、修改、扩充和使用文档, 对于发挥软件产品的效益有着重要的意义。

# 反馈测试题解

## 一、名词解释

### 1. 算式估算法

答: 专家估算法和类推估算法的缺点在于, 它们依靠带有一定盲目和主观的猜测对项目进行估算。算式估算法则是企图避免主观因素的影响。用于估算的方法有两种基本类型: 由理论导出和由经验得出。

### 2. 自顶向下估算方法

答:估算人员参照以前完成的项目所耗费的总成本(或总工作量),来推算将要开发的软件的总成本(或总工作量),然后把它们按阶段、步骤和工作单元进行分配,这种方法称为自顶向下估算方法。

### 3. 控制

答:控制包括进度控制、人员控制、经费控制和质量控制。为保证软件开发按预定的计划进行,对开发过程要实施以计划为基础。由于软件产品的特殊性和软件工程的不成熟,控制软件进度计划比较困难。通常把一个大的开发任务分为若干期工程。例如,分一期工程、二期工程。然后再制定各期工程的具体计划,这样才能保证计划实际可行,便于控制。在制定计划时要适当留有余地。

### 4. 开发人员

答:软件开发人员一般分为:项目负责人、系统分析员、高级程序员、初级程序员、资料员和其他辅助人员。根据项目的规模大小,有可能一人兼数职,但职责必须明确。不同职责的人,要求的素质不同如项目负责人需要有组织能力、判断能力和对重大问题能做出决策的能力;系统分析员需要有概括能力、分析能力和社交活动能力;程序员需要有熟练的编程能力等。人员要少而精,选人要慎重。软件生存期各个阶段的活动既要有分工又要互相联系。因此,要求选择各类人员既能胜任工作,又要能相互很好地配合,没有一个和谐的工作环境很难完成一个复杂的软件项目。

### 5. 软件工程管理

答:软件工程包含软件开发技术和软件工程管理两大部分内容。软件工程管理是对软件项目的开发管理。具体地说,就是对整个软件生存期的一切活动进行管理。对任何工程来说,工程的成败,都与管理的好坏有密切的关系,软件工程更不例外。由于软件产品的独特性,软件工程管理不同于其他工程管理,它对保证高质量的软件产品更具有极为重要的意义。

### 6. 软件工程标准化

答:随着软件工程学的发展,人们对计算机软件的认识逐渐深入。软件工程的范围从只是使用程序设计语言编写程序,扩展到整个软件生存期。诸如软件概念的形成、需求分析、设计、实现、测试、安装和检验,运行和维护,直到软件淘汰(为新的软件所取代)。同时还有许多技术管理工作(如过程管理、产品管理、资源管理)以及确认与验证工作(如评审和审计、产品分析、测试等)常常是跨软件生存期各个阶段的专门工作。所有这些工作都应当逐步建立其标准或规范来。由于计算机发展迅速,未形成标准之前,在行业中先使用一些约定,然后逐渐形成标准。

另一方面,软件工程标准的类型也是多方面的。它可能包括过程标准(如方法、技术、质量等)、产品标准(如需求、设计、部件、描述、计划报告等)、专业标准(如职别、道德准则、认证、特许、课程等),以及记法标准(如术语、表示法、语言等)。

### 6. 变更控制

答:软件工程过程中某一阶段的变更,均要引起软件配置的变更,这种变更必须严格加以控制和管理,保持修改信息,并把精确、清晰的信息传递到软件工程过程的下一步骤。

变更控制包括建立控制点和建立报告与审查制度。对于一个大型软件来说,不加控制的变更很快就会引起混乱。

### 7. 软件配置项

答:随着软件工程过程的进展,软件配置项(Software Configuration Item,SCI)是软件工程中产生的信息项,它是配置管理的基本单位,对已成为基线的SCI,虽然可以修改,但必须按照一个特殊的、正确的过程进行评估,确认每一处的修改。

### 8. 基线

答:基线是软件生存期中各开发阶段的一个特定点,它的作用是把开发各阶段工作的划分更加明确化,使本来连续的工作在这些点上断开,以便于检查与肯定阶段成果。因此基线可以作为一个检查点,在开发过程中,当采用的基线发生错误时,我们可以知道处于的位置,返回到最近和最恰当的基线上。

## 9. 软件配置管理

答:在软件建立时变更是不可避免的,而变更时由于没有进行变更控制,可能加剧了项目中的混乱。为协调软件开发使得混乱减到最小,使用配置管理技术,使变更所产生的错误达到最小并最有效地提高生产率。

软件配置管理,简称 SCM(Software Configuration Management),它用于整个软件工程过程。其主要目标是:标识变更;控制变更;确保变更正确地实现;报告有关变更。SCM 是一组管理整个软件生存期各阶段中变更的活动。

## 10. 工程网络图

工程网络图是一种有向图,该图中用圆表示事件(事件表示一项子任务的开始与结束),有向弧或箭头表示子任务的进行,箭头上的数字称为权,该权表示此子任务的持续时间,箭头下面损耗号中的数字表示该任务的机动时间,图中的圆表示与某个子任务开始或结束事件的时间点。圆的左边部分中数字表示事件号,右上部分中的数字表示前一子任务结束或后一个子任务开始的最早时刻,右下部分中的数字则表示前一子任务结束或后一子任务开始的最迟时刻。对工程网络图只有一个开始点和一个终止点,开始点没有流入箭头,称为入度为零。终止点没有流出箭头,称为出度为零。中间的事件圆表示在它之前的子任务已经完成,在它之后的子任务可以开始。圆的表示如下:

## 11. Gantt 图

答:Gantt 图常用水平线段来描述把任务分解成子任务,以及每个子任务的进度安排,该图表示方法简单易懂,一目了然,动态反映软件开发进度情况,是进度计划和进度管理的有力工具,在子任务之间依赖关系不复杂的情况下常使用此种方法。

## 12. 风险分析

答:风险分析实际上就是贯穿在软件工程中的一系列风险管理步骤,其中包括风险识别、风险估计、风险管理策略、风险解决和风险监督,它能让人们去主动“攻击”风险。

## 13. COCOMO 估算模型

答:结构性成本模型 COCOMO(Constructive Cost Mode)是最精确、最易于使用的成本估算方法之一。该模型分为:基本 COCOMO 模型,是一个静态单变量模型,它是对整个软件系统进行估算;中级 COCOMO 模型,是一个静态多变量模型。中级 COCOMO 模型将软件系统模型分为系统和部件两个层次,系统是由部件构成的,它把软件开发所需人力(成本)看作是程序大小和一系列“成本驱动属性”的函数,用于部件级的估算,更精确些;详细 COCOMO 模型,将软件系统模型分为系统、子系统和模块三个层次,它除包括中级模型中所考虑的因素外,还考虑了在需求分析、软件设计等每一步的成本驱动属性的影响。

## 14. 自底向上估算方法

答:自底向上估算方法是将待开发的软件细分,分别估算每一个子任务所需要的开发工作量,然后将它们加起来,得到软件的总开发量。这种方法的优点是对每一部分的估算工作交给负责该部分工作的人来做,所以估算较为准确。其缺点是其估算往往缺少与软件开发有关的系统级工作量,如集成、配置管理、质量管理、项目管理等,所以估算往往偏低。

## 15. 差别估算方法

答:差别估算是将开发项目与一个或多个已完成的类似项目进行比较,找出与某个相类似项目的若干不同之处,并估算每个不同之处对成本的影响,导出开发项目的总成本。该方法的优点是可以提高估算的准确度,缺点是不容易明确“差别”的界限。

## 16. 专家估算法

答:依靠一个或多个专家对要求的项目做出估算,其精确性取决于专家对估算项目的定性参数的了解和他们的经验。

## 17. 类推估算法

答:自顶向下的方法中,它是将估算项目的总体参数与类似项目进行直接比较相比得到结果。自底向上方法中,类推是在两个具有相似条件的工作单元之间进行。

## 18. 主程序员组织机构

答:由一位高级工程师(主程序员)主持计划、协调和复审全部技术活动;一位辅助工程师协助主程序员工作,并在必要时代替主程序员工作;若干名技术人员负责分析和开发活动;可以有一位或几位专家和一位资料员协助软件开发机构的工作。资料员非常重要,负责保管和维护所有的软件文档资料,帮助收集软件的数据,并在研究、分析、评价文档资料的准备方面进行协助工作。

主程序员的制度突出了主程序员的领导,责任集中在少数人身上,有利于提高软件质量。

## 19. 专家组织机构

答:它是由若干专家组成一个开发机构,强调每个专家的才能,充分发挥每个专家的作用。这种组织机构虽然能发挥所有工作人员的积极性,但往往有可能出现协调上的困难。

## 20. 民主组织机构

答:民主组织由从事各方面工作的人员轮流担任组长。很显然,这种组织机构对调动积极性和个人的创造性是很值得称道的。但是,由于过多地进行组长信息“转移”,不符合软件工程化的方向。

## 二、填空题

1 软件配置管理,简称 SCM,它用于整个软件工程过程。其主要目标是:\_\_\_\_、\_\_\_\_、\_\_\_\_、\_\_\_\_。SCM 是一组管理整个软件生存期各阶段中\_\_\_\_的活动。

答:标识变更 控制变更 确保变更正确地实现 报告有关变更 变更

2 软件配置项(SCI)是软件工程中产生的\_\_\_\_,它是配置管理的\_\_\_\_。

答:信息项 基本单位

3 软件工程过程中某一阶段的变更,均要引起\_\_\_\_的变更,这种变更必须严格加以控制和管理,保持\_\_\_\_,并把精确、清晰的信息传递到软件工程过程的\_\_\_\_。

答:软件配置 修改信息 下一步骤

4 变更控制包括建立\_\_\_\_和建立\_\_\_\_。

答:控制点 报告与审查制度

5 在变更控制的过程中,“检出”和“登入”处理实现了两个重要的变更控制要素,即\_\_\_\_和\_\_\_\_。\_\_\_\_管理各个用户存取和修改一个特定软件配置对象的权限。\_\_\_\_可用来确保由不同用户所执行的并发变更。

答:存取控制 同步控制 存取控制 同步控制

6 软件开发项目生存期需求分析阶段应包括的文档是\_\_\_\_。

答:软件需求说明书 数据要求说明书 确认测试计划 初步用户操作手册

7 软件开发项目生存期概要设计阶段应包括的文档是\_\_\_\_。

答:概要设计说明书 数据库设计说明书 补充的用户操作手册 修订的测试计划

8 软件开发的组织机构没有统一的模式。主要有\_\_\_\_、\_\_\_\_、\_\_\_\_三种组织机构。

答:主程序员 专家组织机构 民主组织机构

9 在软件工程管理中,控制包括\_\_\_\_、\_\_\_\_、\_\_\_\_和\_\_\_\_。

答:进度控制 人员控制 经费控制 质量控制

10 软件项目计划是由\_\_\_\_和\_\_\_\_共同经过\_\_\_\_阶段后制定的。

答:系统分析员 用户 可行性研究与计划

11. 软件项目计划是\_\_\_\_\_阶段的结果产品。但由于可行性研究是在高层次进行系统分析,未能考虑软件系统开发的细节情况,因此软件项目计划一般在\_\_\_\_\_阶段完成后才定稿的。

答:可行性研究 需求分析

12. 软件项目计划包括两个任务:研究与估算。即通过研究确定\_\_\_\_\_,估算\_\_\_\_\_。

答:该软件项目的主要功能、性能和系统界面 项目开发所需的经费、所要使用的资源以及开发进度

13. 风险分析实际上就是贯穿在软件工程中的一系列风险管理步骤,其中包括\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_,\_\_\_\_\_和\_\_\_\_\_。

答:风险识别 风险估计 风险管理策略 风险解决 风险监督

14. 软件开发项目生存期详细设计阶段应包括的文档是\_\_\_\_\_。

答:详细设计说明书

15. 软件开发过程中,严格控制变更,保留变更的有关信息,这种管理是由( )来完成的。

答:变更管理

16. 要成功地完成软件开发工作的一个主要决定性因素是\_\_\_\_\_。

答:软件管理

17. 能协调软件开发,使得混乱减少到最小的方法是使用\_\_\_\_\_。

答:软件配置管理

18. 在软件的生产过程中,总是有大量各种信息要记录,因此,\_\_\_\_\_在产品的开发过程中起着重要的作用。

答:软件文档

19. 软件工程管理的具体内容包括对开发人员、组织机构、用户、\_\_\_\_\_等方面的管理。

答:文档资料

20. 差别估算的缺点是不容易明确“差别”的界限,但它的优点是可以提高\_\_\_\_\_。

答:估算的准确度

21. 主程序员组织机构的制度突出了主程序员的领导,责任集中在少数人身上,有利于提高\_\_\_\_\_。

答:软件质量

22. 基线的作用是把各阶段的开发工作划分得更加明确,便于检查与确认阶段成果。因此,基线可以作为项目的一个\_\_\_\_\_。

答:检查点

23. 在一个大系统的开发过程中,由于\_\_\_\_\_失误造成的后果要比程序错误造成的后果更为严重。

答:管理

24. 在一个软件项目的开发过程中要自始至终得到\_\_\_\_\_的密切合作与支持。

答:用户

25. 软件项目计划的第一项活动是确定\_\_\_\_\_。

答:软件范围

26. 软件配置管理,简称\_\_\_\_\_。软件配置项简称\_\_\_\_\_。

答:SCM SCI

27. 国家标准由政府或国家级的机构制定或批准,适合于全国范围的标准。中华人民共和国国家技术监督局是中国的最高标准化机构,它所公布实施的标准简称为“\_\_\_\_\_”,用“\_\_\_\_\_”标识;NSI



指\_\_\_\_\_ :BS 指\_\_\_\_\_ ;IN 指\_\_\_\_\_ ;JS 指\_\_\_\_\_。

答: 国标 GB 美国国家标准协会 英国国家标准 德国标准协会 日本工业标准

28. 软件项目计划包括\_\_\_\_\_与\_\_\_\_\_两个任务。

答: 研究 估算

29. 根据软件工程标准制定的机构与适用范围, 它分为\_\_\_\_\_五个等级。

答: 国际标准、国家标准、行业标准、企业规范及项目(课题)规范

30. 工程网络只有一个开始点和一个终止点, 开始点没有流入箭头, 称为\_\_\_\_\_为零。终止点没有流出箭头, 称为\_\_\_\_\_为零。

答: 入度 出度

31. 行业标准是由行业机构学术团体或国防机构制定的适合某个行业的标准。IEEE 指\_\_\_\_\_ : GIB 指\_\_\_\_\_ ;DOD - STD 指\_\_\_\_\_ ;MIL - S 指\_\_\_\_\_。

答: 美国电气与电子工程师学会 中华人民共和国国家军用标准 美国国防部标准 美国军用标准

32. 工程网络图是一种\_\_\_\_\_图, 该图中用\_\_\_\_\_表示事件, 有向弧或箭头表示子任务的进行, 箭头上的数字称为\_\_\_\_\_, 箭头下面括号中的数字表示该任务的\_\_\_\_\_。图中的圆表示与某个子任务开始或结束事件的时间点。圆的左边部分中数字表示\_\_\_\_\_, 右上部分中的数字则表示前一子任务结束或后一个子任务开始的\_\_\_\_\_时刻, 右下部分中的数字则表示前一子任务结束或后一子任务开始的\_\_\_\_\_时刻。

答: 有向 圆 权 机动时间 事件号 最早 最迟

33. 软件工程的分解是从\_\_\_\_\_和\_\_\_\_\_即\_\_\_\_\_和\_\_\_\_\_两个方面进行的。

答: 横向 纵向 空间 时间

34. 软件开发人员一般分为\_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_和其他\_\_\_\_\_人员。

答: 项目负责人 系统分析员 高级程序员 初级程序员 资料员 辅助

35. 软件工程包含\_\_\_\_\_和\_\_\_\_\_两在部分内容。

答: 软件开发技术 软件工程管理

36. 目前软件工程规范可分为三级: \_\_\_\_\_、\_\_\_\_\_和\_\_\_\_\_。

答: 国家标准与国际标准 行业标准与工业标准 企业标准与开发小组标准

37. 差别估算的优点是可以提高\_\_\_\_\_, 缺点是不容易明确“差别”的界限。

答: 估算的准确度

38. 在软件项目管理过程中一个关键的活动是\_\_\_\_\_, 它是软件开发工作的第一步。

答: 制定项目计划

39. 成本估算是在软件项目开发前, 估算项目开发所需的\_\_\_\_\_。

答: 经费、资源以及开发进度

40. 软件工程管理不同于其他工程管理, 它对保证高质量的\_\_\_\_\_产品更具有极为重要的意义。

答: 软件

41. 成本估算方法中, 有自顶向下估算方法、自底向上估算方法和\_\_\_\_\_方法。

答: 差别估算

42. 软件工程的分解是从横向和纵向, 即\_\_\_\_\_两个方面进行的。

答: 空间和时间

43. \_\_\_\_\_的制度突出了主程序员的领导, 责任集中在少数人身上, 有利于提高软件质量。

答: 主程序员组织机构

44. 软件工程包含\_\_\_\_\_和\_\_\_\_\_两大部分内容。

答:软件开发技术 软件工程管理

45. 在软件开发和维护过程中一个软件往往有许多版本,版本控制工具用来存储、更新、恢复和管理一个软件的\_\_\_\_\_。

答:多个版本

46. 事实证明,由\_\_\_\_\_失误造成的后果要比程序错误造成的后果更为严重。

答:管理

47. 工程网络图是一种有向图,该图中用圆表示\_\_\_\_\_,有向弧或箭头表示\_\_\_\_\_的进行,箭头上的数字称为权。

答:事件 子任务

48. 软件开发人员一般分为:\_\_\_\_\_,系统分析员、高级程序员、初级程序员、资料员和其他辅助人员。

答:项目负责人

49. 参照以前完成的项目所耗费的总成本,来推算将要开发的软件的总成本,然后把它们按阶段、步骤和工作单元进行分配,这种方法称为\_\_\_\_\_方法。

答:自顶向下估算

### 三、选择题

1. 在变更控制中,( )管理各个用户存取和修改一个特这下软件配置对象的权限。

A .异步控制                      B .同步控制                      C .存取控制                      D .基线控制

答:C

2. 在配置管理中,“检出”和“登入”处理实现了两个重要的变更控制要素,即( )和同步控制。

A .异步控制                      B .同步控制                      C .基线控制                      D .存取控制

答:D

3. 按照软件配置管理的原始指导思想,受控制的对象应是 ( )

A .软件元素                      B .软件配置项                      C .软件项目                      D .软件过程

答:B

4. 软件配置项是软件配置管理的对象,指的是软件工程过程中所产生的 ( )

A .接口                              B .软件环境                      C .信息项                      D .版本

答:C

5. 模块可以有多种实现,即有多种( ),称它们构成一个模块家族。

A .设计                              B .版本                              C .结构化                      D .分析

答:B

6. ( )应该考虑这一下系统的具体版本进行描述和生成。

A .成本控制                      B .需求分析                      C .系统设计                      D .版本控制

答:D

7. ( )是指某种数据媒体和其中所记录的数据。

A .数据库                              B .软件的文档                      C .文件                              D .信息库

答:B

8. 版本用来定义软件配置项的 ( )

A .演化阶段                      B .环境                              C .要求                              D .软件工程过程

答:A

9.变更控制是一项最重要的软件配置任务,其中“检出”和( )处理实现了两个重要的变更控制要素,即存取控制和同步控制。

- A.登入
- B.管理
- C.填写变更要求
- D.审查

答:A

10.在变更控制中,( )可用来确保由不同用户所执行的并发变更。

- A.异步控制
- B.同步控制
- C.存取控制
- D.基线控制

答:B

11.一个项目是否开发,从经济上来说是否可行,归根结底是取决于对( )

- A.成本的估算
- B.项目计划
- C.工程管理
- D.工程网络图

答:A

12.软件管理比其他工程管理更为( )

- A.容易
- B.困难
- C.迅速
- D.迟缓

答:B

13.只有高水平的软件工程能力才能生产出高质量的软件产品。因此,须在软件开发环境或软件工具箱的支持下,运用先进的开发技术、工具和管理方法来提高( )能力。

- A.组织软件
- B.软件质量
- C.设计软件
- D.开发软件

答:D

14.软件开发规范的体现和指南是( )

- A.文档
- B.程序
- C.需求分析
- D.详细设计

答:A

15.为使得开发人员对软件产品的各阶段工作都进行周密的思考,从而减少返工,所以( )的编制是很重要的。

- A.需求说明
- B.概要说明
- C.软件文档
- D.测试大纲

答:C

16.以下说法错误的是( )

- A.软件配置管理简称 SCI
- B.软件配置项是配置管理的基本单位
- C.软件配置实际上是一动态为概念
- D.软件工程过程中某一阶段的变更均要引起软件配置的变更

答:A

17.以下说法错误的是( )

- A.软件项目计划是由程序员与用户单位共同经过“可行性研究与计划”阶段后制定的
- B.软件项目计划是可行性研究阶段为结果产品
- C.项目计划的目标是为项目负责人提供一个框架
- D.软件项目计划中的研究,即通过研究确定该软件项目的主要功能,性能和系统界面
- E.估算是在软件项目开发前,估算项目开发所需的经费、所要使用的资源以及开发进度

答:A

18.以下说法错误的是( )

- A.GB 指中华人民共和国国家军用标准

- B. ANSI 指美国国家标准协会
- C. BS 指英国国家标准
- D. DIN 指德国标准协会
- E. JIS 指日本工业标准

答:A

19. 以下不属于软件项目进度安排的主要方法的是 ( )

- A. 工程网络图
- B. cantt 图
- C. 任务资源表
- D. IFD 图

答:D

20. 以下说法错误的是 ( )

- A. IEEE 指美国电气与电子工程师学会
- B. GB 指中华人民共和国国家军用标准
- C. DOD - STD 指美国国防部标准
- D. MIL - S 指美国军用标准

答:B

21. 以下说法错误的是 ( )

- A. 文档仅仅描述和规定了软件的使用范围及相关的操作命令
- B. 文档也是软件产品的一部分,没有文档的软件就不成为软件
- C. 软件文档的编制在软件开发工作中占有突出的地位和相当大的工作量
- D. 高质量文档对于发挥软件产品的效益有着重要的意义

答:A

22. 以下说法错误的是 ( )

- A. 自顶向下估算方法的缺点是往往不清楚低级别上的技术性困难问题
- B. 自底向上估算方法的缺点是其估算往往缺少与软件开发有关的系统级工作量
- C. 差别估算方法的缺点是不容易明确“差别”的界限
- D. 专家估算法和类推估算法的缺点在于依靠带有一定盲目和主观的猜测对项目进行估算
- E. 类推估算法用于估算的方法有两种基本类型:由理论导出和主经验得出

答:E

23. 就软件产品的特点,以下说法错误的是 ( )

- A. 软件具有高度抽象性,软件及软件生产过程具有不可见性
- B. 同一功能软件的多样性,软件生产过程中的易错性
- C. 软件在开发和维护过程中的不变性
- D. 不同开发者之间思维碰撞的易发生

答:C

24. 单元测试是发现( )错误,集成测试是发现( )错误,确认测试是发现( )错误,系统测试是发现( )错误。

- A. 接口错误
- B. 编码上的错误
- C. 性能、质量不合要求
- D. 功能错误
- E. 需求错误
- F. 设计错误

答:B A D C

25. 为使软件项目开发获得成功,必须对( )的工作范围、可能遇到的风险、需要的资源(人、硬件、软件)、要实现的任务、经历的里程碑、花费的工作量(成本)以及进度的安排等做到心中有数。

- A. 需求分析
- B. 概要设计
- C. 软件开发项目
- D. 软件开发进度

答:C

26. 软件工程管理是对软件项目的开发管理,即对整个软件( )的一切活动的管理。

- A. 软件项目
- B. 生存期
- C. 软件开发计划
- D. 软件开发

答:B

27. 在软件项目管理过程中一个关键的活动是( ),它是软件开发工作的第一步。

- A. 编写规格说明书
- B. 制定测试计划
- C. 编写需求说明书
- D. 制定项目计划

答:D

28.( )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。

- A. 开发文档
- B. 管理文档
- C. 用户文档
- D. 软件文档

答:B

29.( )是在软件开发过程中,由软件开发人员制定的需提交管理人员的一些工作计划或工作报告。

- A. 开发文档
- B. 管理文档
- C. 用户文档
- D. 软件文档

答:A

30.( )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。

- A. 开发文档
- B. 管理文档
- C. 用户文档
- D. 软件文档。

答:C

31. 自底向上估算方法的优点是对每一部分的估算工作交给负责该部分工作的人来做,所以估算( )其缺点是其估算往往缺少与软件开发有关的系统级工作量,所以估算( )

- A. 往往偏低
- B. 不太准确
- C. 往往偏高
- D. 较为准确

答:D A

32.COCOMO 估算模型是( )

- A. 模块性成本模型
- B. 结构性成本模型
- C. 动态单变量模型
- D. 动态多变量模型

答:B

33.Putnam 成本估算经验模型是( )

- A. 模块性成本模型
- B. 结构性成本模型
- C. 动态单变量模型
- D. 动态多变量模型

答:D

## 四、简答题

1.说明软件工程管理的重要性。

答:由软件危机引出软件工程,是计算机发展史上的一个重大进展。为了对付大型复杂的软件系统,须采用传统的“分解”方法。软件工程的分解是从横向和纵向即空间和时间两个方面进行的。横向分解就是把一个大系统分解为若干个小系统,小系统分解为子系统,子系统分解为模块,模块分解为过程。纵向分解就是生存期,把软件开发分为几个阶段,每个阶段有不同的任务、特点和方法。为此,软件

工程管理需要有相应的管理策略。

根据软件产品的特征,且随着软件规模的不断增大,开发人员也随着增多,开发时间也相应持续增长,这些都增加了软件工程管理的难度,同时也突出了软件工程管理的必要性与重要性。事实证明,由管理失误造成的后果比程序错误造成的后果更为严重。很少有软件项目的实施进程能准确地符合预定目标、进度和预算的,这也就足以说明软件工程管理的重要。

## 2. 软件质量保证应做好哪几方面的工作？

答:软件质量保证是软件工程管理的重要内容,软件质量保证应做好以下几方面的工作:

(1)采用技术手段和工具。质量保证活动要贯彻开发过程始终,必须采用技术手段和工具,尤其是使用软件开发环境来进行软件开发。

(2)组织正式技术评审。在软件开发的每一个阶段结束时,都要组织正式的技术评审。国家标准要求单位必须采用审查、文档评审、设计评审、审计和测试等具体手段来保证质量。

(3) 加强软件测试。软件测试是质量保证的重要手段,因为测试可发现软件中大多数潜在错误。

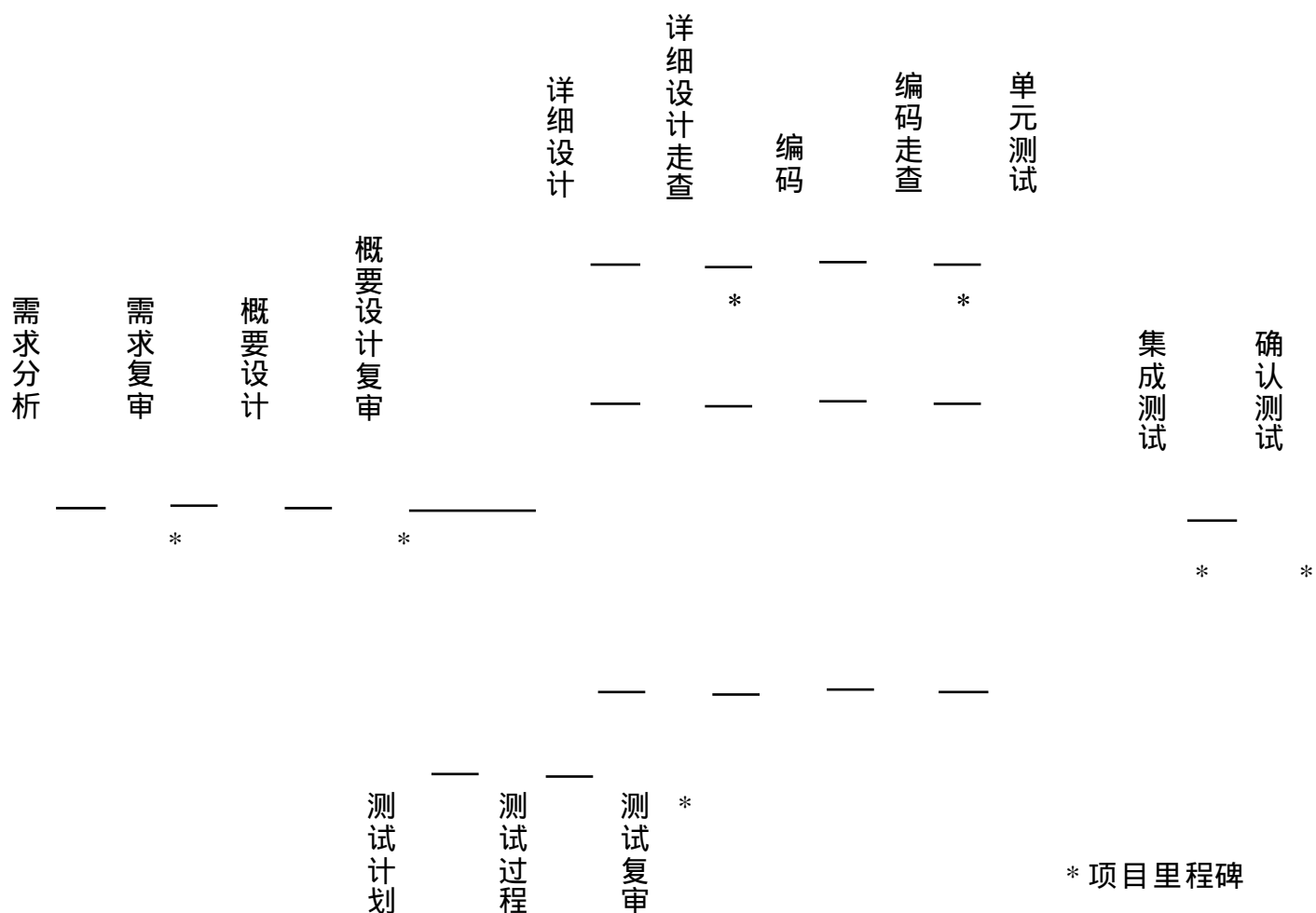
(4)推行软件工程规范(标准)。用户可以自己制定软件工程规范(标准),但标准一旦确认就应贯彻执行。

(5)对软件的变更进行控制。软件的修改和变更常常会引起潜伏的错误,因此必须严格控制软件的修改和变更。

(6)对软件质量进行度量。即对软件质量进行跟踪,及时记录和报告软件质量情况。

3. 画出表示软件任务开发并行性的任务网络图。

答:表示软件任务开发并行性的网络图如图所示:

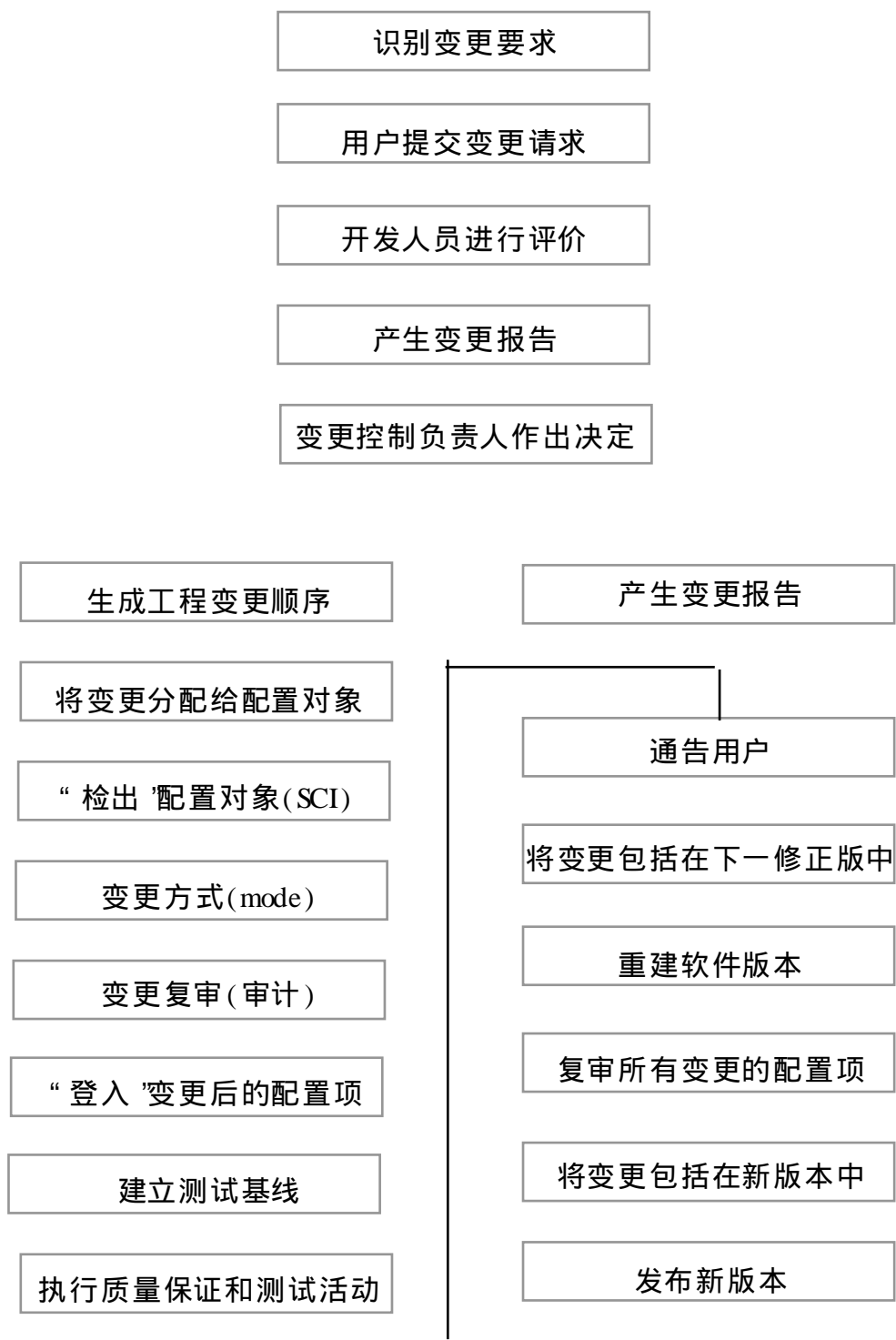


### 典型软件开发任务的并行图

4. 请叙述软件工程过程中版本控制与变更控制处理过程。

答: 软件工程过程中某一阶段的变更,均要引起软件配置的变更,这种变更必须严格加以控制和管理,保持修改信息,并把精确、清晰的信息传递到软件工程过程的下一步骤。

变更控制包括建立控制点和建立报告与审查制度。对于一个大型软件来说,不加控制的变更很快就会引起混乱。因此变更控制是一项最重要的软件配置任务,变更控制的过程如图所示。



变更控制的过程

其中“检出”和“登入”处理实现了两个重要的变更控制要素,即存取控制和同步控制。存取控制管理各个用户存取和修改一个特定软件配置对象的权限。同步控制可用来确保由不同用户所执行的并发变更。

5 软件工程管理包括哪些内容？

答: 软件工程管理的具体内容包括对开发人员、组织机构、用户、文档资料等方面的管理。

(1) 开发人员。软件开发人员一般分为:项目负责人、系统分析员、高级程序员、初级程序员、资料员和其他辅助人员。软件生存期各个阶段的活动既要有分工又要互相联系。因此,要求各类人员既能胜

任工作,又要能相互很好地配合,没有一个和谐的工作环境很难完成一个复杂的软件项目。

(2)组织机构。组织机构要求好的组织结构、合理的人员分工、有效的通讯。软件开发的组织机构没有统一的模式。主要有主程序员、专家组、民主组织三种组织机构。

(3)用户。软件是为用户而开发的,在开发过程中自始至终必须得到用户的密切合作和支持。作为项目负责人,要特别注意与用户保持联系,掌握用户的心理和动态,防止来自用户的各种干扰和阻力。

(4)控制。控制包括进度控制、人员控制、经费控制和质量控制。为保证软件开发按预定的计划进行,对开发过程要实施以计划为基础。

(5)文档资料。软件工程管理很大程度上是通过对文档资料管理来实现的。因此,要把开发过程中的一切初步设计、中间过程、最后结果建立成一套完整的文档资料。文档标准化是文档管理的重要方面。

#### 6. 软件开发成本估算方法有哪些?

答:对于一个大型的软件项目,由于项目的复杂性,开发成本的估算不是一件简单的事,要进行一系列的估算处理。一个项目是否开发,从经济上来说是否可行,归根结底是取决于对成本的估算。成本估算方法有:自顶向下估算方法、自底向上估算方法、差别估算方法。此外还有专家估算法、类推估算法与算式估算法。

介绍几种成本估算模型:COCOMO 估算模型和 Putnam 成本估算经验模型。

#### 7. 试述软件工程标准化的重要性。

答:在开发项目的各个部分以及各开发阶段之间也都存在着许多联系和衔接问题。如何把这些错综复杂的关系协调好,需要有一系列统一的约束和规定,因此,软件工程标准化在软件开发项目中是十分重要的。

#### 8. 试述软件产品的特点。

答:软件是非物质性的产品,而且是知识密集型逻辑思维的产品,它具有以下特性:

- (1)软件具有高度抽象性,软件及软件生产过程具有不可见性。
- (2)同一功能软件的多样性,软件生产过程中的易错性。
- (3)软件在开发和维护过程中的易变性。
- (4)不同开发者之间思维碰撞的易发性。

#### 9. 文档的作用是什么?

答:在软件工程中,文档用来表示对需求、工程或结果进行描述、定义、规定、报告或认证的任何书面或图示的信息。它们描述和规定了软件设计和实现的细节,说明使用软件的操作命令。文档也是软件产品的一部分,没有文档的软件就不成为软件。

#### 10. 软件工程标准化的等级有哪些?

答:根据软件工程标准制定的机构与适用的范围,它分为国际标准、国家标准、行业标准、企业规范及项目(课题)规范 5 个等级。

##### (1)国际标准

由国际标准化组织 ISO 制定和公布,供世界各国参考的标准。该组织有很大的代表性和权威性,它所公布的标准有很大权威性。ISO9000 是质量管理和质量保证标准。

##### (2)国家标准

由政府或国家级的机构制定或批准,适合于全国范围的标准。主要有:GB(国际);ANSI(美国国家标准协会)。

##### (3)行业标准

由行业机构、学术团体或国防机构制定的适合某个行业的标准。主要有:IEEE(美国电气与电子工程师学会);GJB(中华人民共和国国家军用标准)。



#### (4)企业规范

大型企业或公司所制定的适用于本部门的规范。

#### (5)项目(课题)规范

某一项目组织为该项目专用的软件工程规范。

#### 11. 试述软件项目计划内容

答:在软件项目管理过程中一个关键的活动是制定项目计划,它是软件开发工作的第一步。软件项目计划内容如下:

(1)范围。对该软件项目的综合描述,定义其所要做的工作以及性能限制,它包括项目目标、主要功能、性能限制、系统接口、特殊要求、开发概述等。

(2)资源。包括人力资源、硬件资源、软件资源、其他等。

(3)进度安排。进度安排的好坏往往会影响整个项目的按期完成,因此这一环节是十分重要的。制定软件进度与其他工程没有很大的区别,其主要的方法有:工程网络网、Gantt图、任务资源表。

(4)成本估算。为使开发项目能在规定的时间内完成,且不超过预算,成本估算是很重要的。软件成本估算是一门不成熟的技术,国外已有的技术只能作为我们的借鉴。

(5)培训计划。为用户各级人员制定培训计划。

#### 12. 软件配置管理有哪些内容?

答:(1)基线

基线是软件生存期中各开发阶段的一个特定点,它的作用是把开发各阶段工作的划分更加明确化,使本来连续的工作在这些点上断开,以便于检查与肯定阶段成果。因此基线可以作为一个检查点,在开发过程中,当采用的基线发生错误时,我们可以知道处于的位置,返回到最近和最恰当的基线上。

#### (2)软件配置项

软件配置项(SCI)是软件工程中产生的信息项,它是配置管理的基本单位。

#### (3)版本控制管理

版本控制管理是对系统不同版本进行标识与跟踪的过程。版本标识的目的是便于对版本加以区分、检索和跟踪,以表明各个版本之间的关系。

#### (4)变更控制

变更控制包括建立控制点和建立报告与审查制度。对于一个大型软件来说,不加控制地变更很快就会引起混乱。因此变更控制是一项最重要的软件配置任务。

## 第十三章 软件开发环境

软件开发环境主要目标是提高软件生产率,改善软件质量和减低软件成本,而这些目标的实现只能依靠软件工具的广泛应用,所以软件工具是开发环境中最主要的组成部分,对软件工具开发、设计和使用的研究是十分重要的。使用 CASE 进行软件开发可以提高软件开发效率、改善软件质量。

本章总的要求是了解软件开发环境概念、软件开发环境的分类、软件工具、软件工具分类、工具间集成、计算机辅助软件工程 CASE 的基本概念。

理解软件环境的特点和分类,软件工具的分类与评价,工具间集成性与灵活性,CASE 的分类与生命周期。

理解集成化 CASE,CASE 生存周期,软件工程环境;软件工程方法的研究在软件开发中的作用。

### 考试要求

#### 1. 软件开发环境

软件开发环境概论,要求达到识记层次。

软件开发环境的分类,要求达到识记层次。

#### 2. 软件工具

软件工具的基本概念与特点,要求达到识记层次。

软件工具的分类,要求达到领会层次。

#### 3. 计算机辅助软件工程(CASE)

CASE 定义,要求达到识记层次。

CASE 分类,要求达到识记层次。

CASE 的集成,要求达到领会层次。

CASE 生存期,要求达到领会层次。

CASE 工作台,要求达到领会层次。

### 知识重点

#### (一) 软件开发环境

软件开发环境是相关的一组软件工具集合,它支持一定的软件开发方法或按照一定的软件开发模型组织而成。虽然目前提出的对软件开发环境的定义表面上不尽相同,但实质上是一致的,它们都强调:

(1) 软件开发环境是一组相关工具的集合。

(2)这些相关工具按一定的开发方法或一定的开发处理模型组织起来。

(3)这些相关工具支持整个软件生存期的各阶段或部分阶段。软件开发环境的目标应是提高软件开发的生产率和软件产品的质量。因而理想的软件开发环境应是能支持整个软件生存期阶段的开发活动,并能支持各种处理模型的软件方法学,同时实现这些开发方法的自动化。

软件开发环境是与软件生存期、软件开发方法和软件处理模型紧密相关的。其分类方法很多。

按解决的问题分类,软件开发环境可分为程序设计环境、系统合成环境、项目管理环境。

按软件开发环境的演变趋向分类,分为以语言为中心的环境、工具箱环境和基于方法的环境。

按集成化程度分类,当前国内外软件工程把软件开发环境分为三代:

(1)第一代。建立在操作系统之上;工具是通过一个公用框架集成;工具不经修改即可由调用过程来使用;工具所使用的文件结构不变,而且成为环境库的一部分;人机界面图形能力差,多使用菜单技术。

(2)第二代。具有真正的数据库,而不是文件库,多采用 E - R 模式;在更低层次集成工具,工具和文件都作为实体保存在数据库中;现有工具要进行适当修改或定制方可加入;人机界面采用图形、窗口等。

(3)第三代。建立在知识库系统上;出现集成化工具集,用户不用在任务之间切换不同的工具;采用形式化方法和软件重用等技术;采用多窗口技术。这一代软件集成度最高,利用这些工具,实现了软件开发的自动化,大大提高了软件开发的质量和生产率,缩短软件开发的周期,并可减低软件的开发成本。

## (二) 软件工具

软件工具是指为支持计算机软件的开发、维护、模拟、移植或管理而研制的程序系统。所以软件工具是一个程序系统。它是为专门目的而开发的,在软件工程范围内也就是为实现软件生存期中的各种处理活动(包括管理、开发和维护)的自动化和半自动化而开发的程序系统,开发软件工具的主要目的是为了提高软件生产率和改善软件的质量。

软件工具通常由工具、工具接口和工具用户接口三部分构成。工具通过工具接口与其他工具、操作系统或网络操作系统,以及通信接口、环境信息库接口等进行交互作用。当工具需要与用户进行交互作用时则通过工具的用户接口。

当前,软件工具的发展具备以下几个特点:软件工具由单个工具向多个工具集成化方向发展;重视用户界面的设计;不断地采用新理论和新技术;软件工具的商品化推动了软件产业的发展,而软件产业的发展,又增加了对软件工具的需求,促进了软件工具的商品化进程。

软件工具种类繁多,如何分类,一直是人们研究的热点。Reifer 和 Trattner 将软件工具分为六类:模拟工具、开发工具、测试和评估工具、运行和维护工具、性能测量工具和程序设计支持工具。

### (三) 计算机辅助软件工程(CASE)

计算机辅助软件工程,缩写为 CASE。CASE 是一组工具和方法的集合,可以辅助软件开发生命周期各阶段进行软件开发。从学术研究角度讲,CASE 把软件开发技术、软件工具和软件开发方法集成到一个统一而一致的框架中。从产业角度讲,CASE 是种类繁多的软件开发和系统集成的产品及软件工具的集合。其中,软件工具不是对任何软件开发方法的取代,而是对方法的辅助,它旨在提高软件开发的效率,增强软件产品的质量。

CASE 系统所涉及到的技术有两类:一类是支持软件开发过程本身的技术,如支持规约、设计、实现、测试等等。采用这类技术的 CASE 系统研制时间较长,已有许多产品上市。另一类是支持软件开发过程管理的技术,如支持建模、过程管理等等。这类技术不很成熟,采用这类技术的 CASE 系统会调用前一类技术的 CASE 系统。

## 反馈测试题解

### 一、名词解释

#### 1. CASE

答:CASE 是一组工具和方法的集合,可以辅助软件开发生存周期各阶段进行软件开发。CASE 把软件开发技术、软件工具和软件开发方法集成到一个统一而一致的框架中,并且吸收了 CAD(计算机辅助设计)、软件工程、操作系统、数据库、网络和许多其他计算机领域的原理和技术,因而,CASE 领域是一个应用、集成和综合的领域。

#### 2. 软件工具

答:软件工具是指为支持计算机软件的开发、维护、模拟、移植或管理而研制的程序系统,软件工具是一个程序系统。

#### 3. 软件开发环境

答:软件开发环境是支持软件产品开发的软件系统,它是由软件工具集合和环境集成构成。软件工具集用以支持软件开发过程、活动和任务;环境集成为工具集成和软件开发、维护和管理提供统一的支持。

### 二、填空题

1 软件开发环境的目标应是提高\_\_\_\_\_和\_\_\_\_\_。因而理想的件开发环境应是能支持整个软件生存期阶段的开发活动,并能支持各种处理模型的\_\_\_\_\_,同时实现这些开发方法的\_\_\_\_\_。

答:软件开发的生产率 软件产品的质量 软件方法学 自动化

2 \_\_\_\_\_是指为支持计算机软件的开发、维护、模拟、移植或管理而研制的程序系统。

答:软件工具

3 软件开发环境是相关的一组 \_\_\_\_\_ 集合,它支持一定的 \_\_\_\_\_ 或按照一定的 \_\_\_\_\_ 组织而成。

答:软件工具 软件开发方法 软件开发模型

4 软件工具通过\_\_\_\_\_与其他工具、操作系统或网络操作系统以及通信接口、环境信息库接口

等进行交互作用。当工具需要与用户进行交互作用时则通过\_\_\_\_\_。

答:工具接口 工具的用户接口

5 软件工具的发展特点是软件工具由单个工具向\_\_\_\_\_方向发展。重视\_\_\_\_\_的设计,不断地采用新理论和新技术。软件工具的商品化推动了软件产业的发展,而软件产业的发展,又增加了对软件工具的需求,促进了软件工具的商品化进程。

答:多个工具集成化 用户界面

6 \_\_\_\_\_是指工具运行在相同的硬件/操作系统平台上。\_\_\_\_\_是指工具使用共享数据模型来操作。\_\_\_\_\_是指工具提供相同的用户界面。

答:平台集成 数据集成 表示集成

7 一个 CASE 工作台是一组\_\_\_\_\_,支持像设计、实现或测试等特定的软件开发阶段。工作台工具能通过\_\_\_\_\_、\_\_\_\_\_或\_\_\_\_\_来集成。

答:工具集 共享文件 共享仓库 共享数据结构

8 在 CASE 集成中,\_\_\_\_\_集成支持工作台或环境中一个工具对系统中其他工具的访问。\_\_\_\_\_集成意指 CASE 系统嵌入了关于过程活动、阶段、约束和支持这些活动所需的工具的知识。

答:控制 过程

9 表示集成或用户界面集成意指一个系统中的工具使用共同的风格,以及采用共同的用户交互标准集,工具有一个相似的外观。目前,表示集成有如下三种不同级别:\_\_\_\_\_、\_\_\_\_\_和\_\_\_\_\_。

答:窗口系统集成 命令集成 交互集成

10 数据集成指不同软件工程能相互交换数据。因而,一个工具的结果能作为另一个工具的输入。有三个级别的数据集成:\_\_\_\_\_、\_\_\_\_\_和\_\_\_\_\_。

答:共享文件 共享数据结构 共享仓库

11 CASE 把\_\_\_\_\_、\_\_\_\_\_和\_\_\_\_\_集成到一个统一而一致的框架中,并且吸收了 CAD(计算机辅助设计)、软件工程、操作系统、数据库、网络和许多其他计算机领域的原理和技术。

答:软件开发技术 软件工具 软件开发

12 开发过程管理包括\_\_\_\_\_和\_\_\_\_\_等。

答:项目计划和控制 任务管理

13 用来辅助软件开发、运行、维护、管理、支持等过程中的活动的软件称为\_\_\_\_\_。

答:软件工具

14 开发软件工具的主要目的是为了\_\_\_\_\_。

答:软件的质量

15 CASE 可以辅助软件开发生命周期各阶段进行软件开发,它是一组\_\_\_\_\_。

答:工具和方法的集合

16 CASE 一术语的英文为:\_\_\_\_\_。

答:Computer - Aided Software Engineering

17 软件开发环境按解决的问题分类,可分为:\_\_\_\_\_环境、\_\_\_\_\_环境、\_\_\_\_\_环境。

答:程序设计 系统合成 项目管理

18 Reifer 和 Trattner 将软件工具分为:\_\_\_\_\_等六类。

答:模拟工具、开发工具、测试和评估工具、运行和维护工具、性能测量工具和程序设计支持工具

19 对 CASE 工具分类的标准可分为\_\_\_\_\_。

答:功能、支持的过程、支持的范围

20 支持分析和设计的工作台有时称为\_\_\_\_\_ CASE 工具,它们支持软件开发的早期过程。程序

设计工作台则称为\_\_\_\_\_CASE 工具。

答:上游 下游

21. 根据支持的范围,CASE 工具可分为\_\_\_\_\_。

答:窄支持、较宽支持和一般支持工具

22. 软件工具通常由\_\_\_\_\_、\_\_\_\_\_和\_\_\_\_\_三部分构成。

答:工具 工具接口 工具用户接口

23. 请填写以下名称的英文缩写或英文简字:

软件开发环境\_\_\_\_\_

软件工程环境\_\_\_\_\_

软件支持环境\_\_\_\_\_

项目支持环境\_\_\_\_\_

自动开发环境\_\_\_\_\_

集成化程序设计环境\_\_\_\_\_

工具箱\_\_\_\_\_

工具箱\_\_\_\_\_

答: SED SEE SSE PSE ADE IPE Toolbox Toolkit

24. 按软件开发环境的演变趋向分类,可分为\_\_\_\_\_的环境、\_\_\_\_\_环境、\_\_\_\_\_的环境。

答:以语言为中心 工具箱 基于方法

25. CASE 系统所涉及到的技术有两类:一类是支持\_\_\_\_\_的技术;另一类是支持\_\_\_\_\_的技术。从 CASE 系统产生方式来看,还有一种特殊的 CASE 技术,即\_\_\_\_\_,它是生成 CASE 系统的生成器所采用的技术。

答:软件开发过程本身 软件开发过程管理 元 - CASE 技术

26. 根据支持的过程,CASE 工具可分为\_\_\_\_\_等。

答:设计工具、编程工具、维护工具

27. 一个组织中的 CASE 系统遵循从初始需求到完全废弃这一生存期,CASE 生存期各步骤如下:

(1)CASE \_\_\_\_\_;(2)CASE \_\_\_\_\_;(3)CASE \_\_\_\_\_;(4)CASE \_\_\_\_\_;(5)CASE \_\_\_\_\_;  
(6)CASE \_\_\_\_\_。

答:需求 剪裁 引入 操作 演化 废弃

28. 1990 年 Wasserman 讨论软件工程环境的集成时,提出一个五级模型,即\_\_\_\_\_。

答:平台集成、数据集成、表示集成、控制集成、过程集成

29. 1993 年, Fuggetta 根据 CASE 系统对软件过程的支持范围,提出 CASE 系统可分为:支持\_\_\_\_\_的工具、工作台支持\_\_\_\_\_活动、环境支持\_\_\_\_\_三类。

答:单个过程任务 某一过程所有活动或某些 软件过程所有活动或至少大部分

30. 软件开发环境是相关的一组\_\_\_\_\_的集合,它支持一定的软件开发方法或按照一定的软件开发模型组织而成。

答:软件工具

31. 环境集成主要有\_\_\_\_\_、界面集成、控制集成、\_\_\_\_\_、过程集成。

答:数据集成 平台集成

32. 软件开发环境是与软件生存期、\_\_\_\_\_和软件处理模型紧密相关的。

答:软件开发方法

33. 软件工具通常由工具、工具接口和\_\_\_\_\_三部分构成。

答:工具用户接口

34. CASE 是多年来在软件开发管理、软件开发方法、软件开发环境和\_\_\_\_\_等方面研究和发展的产物。

答: 软件工具

35. 需求分析工具主要包括:正文和数据流图工具、\_\_\_\_\_工具、面向对象的分析工具。

答: 数据字典

36. 软件开发环境就是围绕着软件开发的一定目标而组织在一起的相关一组\_\_\_\_\_的有机集合。

答: 软件工具

37. 计算机辅助软件工程这一术语的英文为:\_\_\_\_\_。

答: Computer - Aided Software Engineering。

38. 软件开发环境的主要目标是提高\_\_\_\_\_、\_\_\_\_\_和降低软件成本。

答: 软件开发的生产率 改善软件质量

39. 基于方法的环境可分为两大类:\_\_\_\_\_与\_\_\_\_\_。

答: 支持软件开发周期特定阶段的管理 开发过程管理

40. 产品管理包括\_\_\_\_\_和\_\_\_\_\_。

答: 版本管理 配置管理

### 三、选择题

1. ( ) 意指 CASE 系统嵌入了关于过程活动、阶段、约束和支持这些活动所需的工具的知识。

- |         |         |
|---------|---------|
| A .控制集成 | B .平台集成 |
| C .界面集成 | D 过程集成  |

答:D

2 软件开发环境是相关的一组( ) 集合。

- |         |        |
|---------|--------|
| A .软件环境 | B 软件过程 |
| C .软件工程 | D 软件工具 |

答:D

3 数据集成指不同软件工程能相互交换数据。有三种级别的数据集成:共享文件、共享数据结构和共享仓库。( ) 采用所有工具都能识别的文件格式,在 UNIX 中广泛使用。

- |         |         |
|---------|---------|
| A .共享单元 | B .共享文件 |
| C .消息共享 | D .共享仓库 |

答:B

4 表示集成意指一个系统中的工具使用共同的风格,以及采用共同的用户交互标准集。表示集成有三种不同级别:窗口系统集成、命令集成和 ( )

- |         |         |
|---------|---------|
| A .数据集成 | B .平台集成 |
| C .界面集成 | D 交互集成  |

答:D

5. ( ) 支持工作台或环境中一个工具对系统中其他工具的访问。

- |         |         |
|---------|---------|
| A .过程集成 | B 控制集成  |
| C .平台集成 | D .界面集成 |

答:B

6 数据集成指不同软件工程能相互交换数据。有三种级别的数据集成:共享文件、共享数据结构和 ( )

- A .共享单位
- B 继承
- C .消息共享
- D .共享仓库

答:D

- 7 .支持计算机软件的开发、维护、模拟、移植或管理而研制的程序系统称为 ( )
- A .软件工具
  - B 软件环境
  - C .软件过程
  - D 软件模型

答:A

- 8 软件开发环境支持一定的( )或按照一定的软件开发模型组织而成。
- A .软件生存周期
  - B 软件过程
  - C .软件开发方法
  - D 软件开发模型

答:C

- 9 .在软件的开发与维护过程中,用来存储、更新、恢复和管理一个软件的多版本,它是( )工具。
- A. 文档分析
  - B. 项目管理
  - C. 成本估算
  - D. 版本控制

答:D

10. 支持像设计、实现或测试等特定的软件开发阶段的 CASE 工作台是一组 ( )
- A. 工具集
  - B. 软件包
  - C. 平台集
  - D. 程序包

答:A

11. 软件开发环境中最主要的组成部分是 ( )
- A. 软件工程
  - B. 项目管理工具
  - C. 软件工具
  - D. 需求分析工具

答:C

12. 测试工作台包括的工具 ( )
- |       |         |
|-------|---------|
| 测试管理器 | 测试数据生成器 |
| 预测器   | 报告生成器   |
| 文件比较器 | 动态分析器   |
| 模拟器   | 加载器     |
| 静态分析器 |         |

- A.
- B.
- C.
- D.

答:B

13. 根据支持的范围,CASE 工具可分为窄支持、较宽支持和一般支持,则以下解释正确的是 ( )
- A. 窄支持指支持特定过程阶段
  - B. 较宽支持是指支持过程中特定的任务
  - C. 一般支持是指支持覆盖软件过程的全部阶段或大多数阶段
  - D. 较宽支持是指支持覆盖软件过程的大多数阶段

答:C

14. 以下说法错误的是 ( )
- A. 大多数开发系统都采用基于文件集成的策略
  - B.CASE 开放式工作台或者提供控制集成机制,或者可剪裁,其数据集成或协议是独立的
  - C. 在封闭式系统中,系统的集成的约定是该工作台开发商独有的



D. 许多工作台都是封闭式系统,因为这允许更紧密地数据集成、表示集成和控制集成  
答:B

15. 一般分析和设计工作台的构成为 ( )
- |           |           |
|-----------|-----------|
| 图表编辑器     | 设计分析和核实工具 |
| 仓库查询语言    | 数据字典      |
| 报告定义和生成工具 | 代码生成器     |
| 按格式打印     | 动态分析器     |
| 交互式调试器    |           |
- A. B.  
C. D.

答:B

16. Westinghouse 公司于 1992 年公布了 13 类软件工具分类标准和该类的范例工具以及例子,以下说法错误的是 ( )

- A. 代码生成程序属设计工具
- B. 结构图属需求追踪工具
- C. 连接程序属编码和单元测试工具
- D. 测试驱动程序属测试和集成工具

答:B

17. 程序设计工作台由支持程序开发过程的一组工具组成。以下解释错误的是 ( )
- A. 交叉引用:产生一个交叉引用列表,显示所有的程序名是在哪里声明和使用的
  - B. 静态分析器:程序执行之前,显示程序的工作状态
  - C. 动态分析器:产生带附注的一个源文件代码表
  - D. 交互式调试器:允许用户来控制程序的执行次序,显示执行期间的程序状态

答:B

18. 组成程序设计工作台的工具可能为 ( )
- |       |           |
|-------|-----------|
| 语言编译器 | 结构化编辑器    |
| 连接器   | 加载器       |
| 交叉引用  | 静态分析器     |
| 数据字典  | 报告定义和生成工具 |
| 代码生成器 |           |
- A. B.  
C. D.

答:B

19. Westinghouse 公司于 1992 年公布了 13 类软件工具分类标准和该类的范例工具以及例子,以下说法错误的是 ( )

- A. 桌面出版系统属文档工具
- B. 计划和进度属项目管理工具
- C. 文件和修改管理属配置管理工具
- D. 格式管理系统属质量保证工具

答:D

20. Westinghouse 公司于 1992 年公布了 13 类软件工具分类标准和该类的范例工具以及例子,以下说法错误的是 ( )

- A. 动画工具属系统模拟和模型工具
- B. 数据字典工具属设计工具
- C. 编辑程序属需求追踪工具
- D. 面向对象分析工具属需求分析工具

答:B

21. 程序设计工作台由支持程序开发过程的一组工具组成。以下解释错误的是 ( )

- A. 语言编译器:将源代码程序转换成目标码
- B. 结构化编辑器:结合嵌入的程序设计语言知识,对程序的源代码文本进行编辑
- C. 连接器:将已编译的程序目标代码模块连起来
- D. 加载器:程序执行之前将它加载到计算机内存

答:B

22. 1990 年 Wasserman 讨论软件工程环境的集成时,提出一个五级模型。这一模型也适用于工作台。则以下说法正确的是 ( )

- A. 平台集成:工具在一个过程模型和“过程机”的指导下使用
- B. 数据集成:工具激活后能控制其它工具的操作
- C. 表示集成:工具提供相同的用户界面
- D. 控制集成:工具使用共享数据模型来操作
- E. 过程集成:工具运行在相同的硬件/操作系统平台上

答:C

23. Westinghouse 公司于 1992 年公布了 13 类软件工具分类标准和该类的范例工具以及例子,以下说法错误的是 ( )

- A. 追踪和状态报告属项目管理工具
- B. 检查表属质量保证工具
- C. 代码质量度量属度量工具
- D. 数据管理属软件再用工具

答:D

24. 表示集成是指工具提供相同的 ( )

- |       |         |
|-------|---------|
| A. 语言 | B. 用户界面 |
| C. 命令 | D. 操作系统 |

答:B

25. 控制集成是指工具激活后能控制其他( )的操作。

- |       |       |
|-------|-------|
| A. 工具 | B. 系统 |
| C. 软件 | D. 应用 |

答:A

26. 数据集成指不同软件工程能相互 ( )

- |       |         |
|-------|---------|
| A. 合作 | B. 交换数据 |
| C. 交流 | D. 通讯   |

答:B

27. 一个 CASE 工作台是一组( ),支持设计、实现或测试等特定的软件开发阶段。

- |        |        |
|--------|--------|
| A. 工具集 | B. 软件包 |
| C. 平台集 | D. 程序包 |

答:A

28. 软件开发环境是支持软件产品开发的软件系统,它是由软件开发工具集和环境集成机制构成。前者用于支持( )相关过程、活动和任务;后者为( )和软件开发、维护和管理提供统一的支持。

- A. 软件开发
- B. 软件系统
- C. 开发环境集成
- D. 工具集成

答:A D

29. 早期的软件工具只能完成一件特定的任务,后来出现了工作台,它将一组( )组合在一起,对软件开发过程的某些方面提供支持。( )是工作台的发展,其目的在于为软件开发提供完整的和一致的支持。

- A. 软件开发环境
- B. 软件
- C. 工具
- D. 工作台

答:C A

30. CASE 系统所涉及到的技术有两类:一类是支持软件( )的技术;另一类是支持软件( )的技术。

- A. 开发过程本身
- B. 开发方法管理
- C. 开发方法本身
- D. 开发过程管理

答:A D

31. 平台集成是指工具运行在相同的( )平台上。

- A. 硬件
- B. 软件
- C. 硬件/ 操作系统
- D. 硬件/ 软件

答:C

## 四、简答题

1 什么是 CASE ? CASE 工具有哪些分类 ?

答:CASE 是一组工具和方法的集合,可以辅助软件开发生命周期各阶段进行软件开发。从学术研究角度讲,CASE 是多年来在软件开发管理、软件开发方法、软件开发环境和软件工具等方面研究和发展的产物。CASE 把软件开发技术、软件工具和软件开发方法集成到一个统一而一致的框架中,并且吸收了 CAD(计算机辅助设计)、软件工程、操作系统、数据库、网络和许多其他计算机领域的原理和技术。从产业角度讲,CASE 是种类繁多的软件开发和系统集成产品及软件工具的集合。其中,软件工具不是对任何软件开发方法的取代,而是对方法的辅助,它旨在提高软件开发的效率,增强软件产品的质量。

随着 CASE 的发展,出现了各种各样的 CASE 工具。1993 年,Fuggetta 根据 CASE 系统对软件过程的支持范围,提出 CASE 系统可分为三类:

(1)支持单个过程任务的工具,例如检查一个设计的一致性,编译一个程序,比较测试结果等等。工具可能是通用的、独立的,或者也可能归组到工作台。

(2)工作台支持某一过程所有活动或某些活动,例如规约、设计等等。他们一般以或多或少的集成度组成工具集。

(3)环境支持软件过程所有活动或至少大部分。它们一般包括几个不同的工作台,将这些工作台以某种方式集成起来。

2 “软件开发环境应是高度集成的一体化的系统”的含义是什么 ?

答:软件开发环境应是 高度集成的一体化的系统。其含义是:(1)应该支持软件生存期各个阶段的活动,从需求分析、系统设计、编码和调试、测试验收到维护等各阶段的工作。(2)应该支持软件生存期各个阶段的管理和开发两方面的工作。(3)应协调一致地支持各个阶段和各方面的工作。具有统一形

式的内部数据表示。(4)整个系统具有一致的用户接口和统一的文档报表生成系统。

3 软件开发环境应具有高度的通用性。在此,通用性包括哪些方面?

答:软件开发环境应具有高度的通用性。这是指:(1)能适应最常用的几种语言。(2)能适应和支持不同的开发方法。(3)能适应不同的计算机硬件及其系统软件,对这些方面应具有最小的依赖性(尤其是对硬件)。(4)能适应开发不同类型的软件。(5)能适应并考虑到不同用户的需要(如程序员、系统分析员、项目经理、质量保证人员、初学者与熟练人员)。

4 “软件开发环境应易于定制、裁剪或扩充以符合用户要求”,在此,“定制”、“裁剪”、“扩充”的含义是什么?

答:软件开发环境应易于定制、裁判或扩充以符合用户要求,即软件开发环境应具有高度的适应性和灵活性。

定制是指软件开发环境应能符合项目特性、过程和用户的爱好。裁剪是指环境应能自动按用户需要建立子环境,即构成适合具体硬件环境、精巧的、很少冗余的工作环境。扩充是指环境能向上扩展,根据用户新的需求或软件技术的新发展(如加入新工具,引入智能新机制)对原有的环境进行更新和扩充。

5.CASE 的工作台有哪些?

答:(1)程序设计工作台。由支持程序设计的一组工具组成。如将编辑器、编译器和调试器等集成在一个宿主机上构成程序设计工作台供开发人员使用。

(2)分析和设计工作台。支持软件过程的分析和设计阶段。如支持结构化方法的工作台,支持面向对象的方法进行分析和设计的工作台。

(3)测试工作台。趋于支持特定的应用和组织机构,常具有较好的开放性。

(4)交叉开发工作台。这些工作台支持在一种机器上开发软件,而在其他系统上运行所开发的软件。

(5)配置管理(CM)工作台。这些工作台支持配置管理。如版本管理工具、变更跟踪工具、系统建造(装配)工具等。

(6)文档工作台。这些工具支持高质量文档的制作,如字处理器、图表图象编辑器、文档浏览器等。

(7)项目管理工作台。支持项目管理活动,有项目规划和质量、开支评估和预算追踪工具。

软件工具的分类:软件工具种类繁多、涉及面广,如编辑、编译、正文格式处理、静态分析、动态追踪、需求分析、测试、模拟和图形交互等。

6.CASE 的集成有哪些?

答:(1)平台集成:工具运行在相同的硬件/操作系统平台上。

(2)数据集成:工具使用共享数据模型来操作。

(3)表示集成:工具提供相同的用户界面。

(4)控制集成:工具激活后能控制其他工具的操作。

(5)过程集成:工具在一个过程模型和“过程机”的指导下使用。

7. 试述软件环境的发展。

答:在 20 世纪 80 年代中期软件开发环境研究成为热点,其显著的特点是采用信息库,软件开发环境的核心是一个共享的信息库,使得不同的工具可以以各种方式结合在一起。20 世纪 80 年代后期软件开发环境研究显著的特点是发展了面向对象的分析和设计方法,它克服了结构化技术的缺点。在 90 年代主要是进行系统集成方法与集成系统的研究,所研究的集成 CASE 环境可以加快开发复杂信息系统的速度,确保用户软件开发成功,提高软件质量,降低投资成本和开发风险。出现一系列集成的 CASE 软件产品,用以实现需求管理、应用程序分析设计和建模、编码、软件质量保证和测试过程的项目管理及文档生成管理等软件开发工作的规范化、工程化和自动化。

8. 软件工具的分类

答: 软件工具种类繁多、涉及面广,如编辑、编译、正文格式处理、静态分析、动态追踪、需求分析、设计分析、测试、模拟和图形交互等。

如何对软件工具进行分类,一直是人们研究的热点,自 90 年代以来掀起了新的热潮。Reifer 和 Trattner 将软件工具分为六类:模拟工具、开发工具、测试和评估工具、运行和维护工具、性能测量工具和程序设计支持工具。Westinghouse 公司于 1992 年公布了以下 13 类软件工具分类标准和该类的范例工具以及例子。

系统模拟和模型工具:结构和数据流模型;算法模拟;定时和大小工具;动画工具。

需求追踪工具:编辑程序;数据库管理系统;在 DBMS 上的应用运行工具。

需求分析工具:正文和数据流图工具;数据字典工具;面向对象的分析工具。

设计工具:结构图;模块规格说明;伪码;代码生成程序;语言敏感的编辑程序。

编码和单元测试工具:编码程序;语言敏感的编辑程序;语言;代码格式化程序;交叉编辑程序;连接程序;源码层次的调试程序。

测试和集成工具:测试驱动程序;覆盖分析程序;回归测试;测试床。

文档工具:桌面出版系统;文档模板;格式管理系统。

项目管理工具:计划和进度;追踪和状态报道;成本估算和代码行估算。

配置管理工具:访问和版本控制机构;产品基线;文件和修改管理。

① 质量保证工具:检查表;直方图;图形;表格。

1 度量工具:行计数;代码质量度量;管理度量;其他标准度量。

2 软件再用工具。

3 其他:数据管理、通信、电子公告牌、活页等。

9. 软件工具的发展有以下特点:

答:(1)软件工具由单个工具向多个工具集成化方向发展。如将编辑、编译、运行结合在一起构成集成工具。注重工具间的平滑过渡和互操作性。如微软公司的 Office 工具。

(2)重视用户界面的设计。交互式图形技术及高分辨率图形终端的发展,为友好方便的用户图形提供了物质基础。多窗口管理、鼠标器使用、图形资源的表示等技术,极大地改善了用户界面的质量,改善了软件的感觉。

(3)不断的采用新理论和新技术。如许多软件工具的研制中采用了数据库技术、交互图形技术、网络技术、人工智能技术和形式化技术等。

(4)软件工具的商品化推动了软件产业的发展,而软件产业的发展,又增加了对软件工具的需求,促进了软件工具的商品化进程。

10. 试述软件开发环境的分类。

答:(1)按解决的问题分类

程序设计环境。解决如何将规范说明转换成可工作的程序问题,它包括两个重要部分:方法与工具。

系统合成环境。主要考虑把很多子系统集成为一个大系统的问题。所有的大型软件系统都有两个基本特点:他们是由一些较小的、较易理解的子系统组成的,因此,需要有一个系统合成环境来辅助控制子系统及其向大系统的集成。

项目管理环境。大型软件系统的开发和维护必然会有许多人员在一段时间内协同工作,需要对人与人之间的交流和合作进行管理。项目管理环境的责任是解决由于软件产品的规模大、生存期长、人们的交往多而造成的问题。

(2)按软件开发环境的演变趋向分类

以语言为中心的环境。

这类环境的特点是:强调支持某特定语言的编程;包含支持某特定语言编程所需的工具集;环境采取高度的交互方式;仅支持与编程有关的功能(如编码和调试),不支持项目管理等功能。

工具箱环境。

这类环境的特点是由一整套工具组成,供程序设计选择之用,如有窗口管理系统,各种编辑系统,通用绘图系统,电子邮件系统,文件传输系统,用户界面生成系统等。用户可以根据个人需要对整个环境的工具进行剪裁,以产生符合自己需要的个人的系统环境。其次这类环境特点是独立于语言的。这类环境的例子有:UNIX,Window,APSE 的接口集 CAIS,SPICE 等。

另外还可以按集成化程度将环境分成第一代、第二代和第三代集成化环境,以及分布式环境和智能环境等,在此不一一讨论了。

基于方法的环境。

这类环境专门用于支持特定的软件开发方法。这些方法可分为两大类:支持软件开发周期特定阶段的管理与开发过程。前者包括规格说明、设计、确认、验证和重用。后者又可细分为支持产品管理与支持开发和维护产品的过程管理。产品管理包括版本管理和配置管理。开发过程管理包括项目支持和控制、任务管理等。这类环境的例子有: Cornell 程序综合器,支持结构化方法; SmallTalk—80 支持面向对象方法。

### (3)按集成化程度分类

环境的形成与发展主要体现在各工具的集成化的程度上,当前国内外软件工程把软件开发环境分为三代,各代之间的主要特征为:

第一代,建立在操作系统之上;工具是通过一个公用框架集成;工具不经修改即可由调用过程来使用;工具所使用的文件结构不变,而且成为环境库的一部分;人机界面图形能力差,多使用菜单技术。例如 70 年代 UNIX 环境以文件为库集成核心。

第二代,具有真正的数据库,而不是文件库。多采用 E - R 模式;在更低层次集成工具,工具和文件都作为实体保存在数据库中;现有工具要作适当修改或定制方可加入;人机界面采用图形、窗口等。例如 Ada 程序设计环境(APSE)以数据库为集成核心。

第三代,建立在知识库系统上;出现集成化工具集,用户不用在任务之间切换不同的工具;采用形式化方法和软件重用等技术;采用多窗口技术。这一代软件集成度最高,利用 这些工具,实现了软件开发的自动化,大大提高了软件开发的质量和生产率,缩短了软件开发的周期,并可减低软件的开发成本。例如 80 年代 CASE 与目前的 CASE 集成化产品。

### 11. 对软件开发环境的要求有哪些?

答:软件开发环境的目标应是提高软件开发的生产率和软件产品的质量。因而理想的软件开发环境应是能支持整个软件生存期阶段的开发活动,并能支持各种处理模型的软件方法学,同时实现这些开发方法的自动化。比较一致的观点,认为软件开发环境的基本要求如下:

#### (1)软件开发环境应是高度集成的一体化的系统。其含义是:

应该支持软件生存期各个阶段的活动,从需求分析、系统设计、编码和调试、测试验收到维护等各阶段工作。

应该支持软件生存期各个阶段的管理和开发两方面的工作。

应协调一致的支持各个阶段和各方面的工作。具有统一形式的内部数据表示。

整个系统具有一致的用户接口和统一的文档报表生成系统。

#### (2)软件开发环境应具有高度的通用性。这是指:

能适应最常用的几种语言。

能适应和支持不同的开发方法。

能适应不同的计算机硬件及其系统软件,对这些方面应具有最小的依赖性(尤其是对硬件)。

能适应开发不同类型的软件。

能适应并考虑到不同用户的需要(如程序员、系统分析员、项目经理、质量保证人员、初学者与熟练人员)。

(3)软件开发环境应易于定制、裁剪或扩充以符合用户要求,即软件开发环境应具有高度的适应性和灵活性。

其定制是指软件开发环境应能符合项目特性、过程和用户的爱好。裁剪是指环境应能自动按用户需要建立子环境,即构成适合具体硬件环境的、精巧的、很少冗余的工作环境。扩充是指环境能向上扩展,根据用户新的需求或软件技术的新发展(如加入新工具,引入智能新机制)对原有的环境进行更新和扩充。

(4)软件开发环境不但可应用性更好,而且是易使用的、经济高效的系统。为此,它应该:

易学、易用、响应时间合理和用户喜爱。

能支持自然语言处理。

能支持交互式和分布式协作开发。

降低环境用户和环境本身的资源花费。

(5)软件开发环境应有辅助开发向半自动开发和自动开发逐步过渡的系统。半自动和自动开发的含义是:

各个阶段的文档之间要能半自动变换和跟踪。

应该注重使用形式化技术。

不同程度的,逐步地采用“软件构件”的集成组装技术,并建立其可扩充的,可再用的“软件构件”库。

采用人工智能技术,逐步包含支持开发的专家系统。

当然符合以上要求的软件开发环境,目前在世界上还不存在,但是软件工程发展很快,这些目标相信是可以实现的。

# 软件工程考前模拟试题(一)

一、名词解释题(每小题 3 分,共 15 分)

1. 面向对象的方法
2. 白盒测试
3. 系统设计说明书
4. 技术可行性
5. 决策树

## 二、填空题(每小题 2 分,共 20 分)

- 1 硬件与\_\_\_\_\_一起构成完整的计算机系统。
- 2 软件是一种\_\_\_\_\_产品,它与物质产品有很大区别。
- 3 瀑布模型是将\_\_\_\_\_各活动规定为依\_\_\_\_\_联接的若干阶段的模型。
- 4 喷泉模型是一种以\_\_\_\_\_为动力,以\_\_\_\_\_为驱动的模式。
- 5 结构化方法由结构化分析、\_\_\_\_\_、结构化程序设计构成,它是一种面向\_\_\_\_\_的开发方法。
- 6 软件结构是以\_\_\_\_\_为基础而组成的一种控制层次结构。
- 7 反映软件结构的基本形态特征是\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_和\_\_\_\_\_。
- 8 一个模块把数值作为参数传送给另一个模块,这种耦合方式称为\_\_\_\_\_。
- 9 两个模块都使用同一张表,这种耦合称为\_\_\_\_\_。
- 10 IPO 图描述分层图中一个模块的输入、输出和\_\_\_\_\_内容。

### 三、选择题(每小题 1 分,共 20 分)

1. 在结构化分析方法中,用以表达系统内数据的运动情况的工具有 ( )
  - A. 数据流图
  - B. 数据词典
  - C. 结构化英语
  - D. 判定树与判定表
2. 功能模型中所有的 ( ) 往往形成一个层次结构。在这个层次结构中一个数据流图的过程可以由下一层的数据流图做进一步的说明。
  - A. 数据流图
  - B. 概念模型图
  - C. 状态迁移图
  - D. 事件追踪图
3. 模块 ( ) ,则说明模块的独立性越强。
  - A. 耦合越强
  - B. 扇入数越高
  - C. 耦合越弱
  - D. 扇入数越低
4. 在设计测试用例时, ( ) 是用的最多的一种黑盒测试方法。
  - A. 等价类划分
  - B. 边值分析
  - C. 因果图
  - D. 判定表
5. 面向对象的主要特征除对象惟一性、封装、继承外,还有 ( )
  - A. 多态性
  - B. 完整性
  - C. 可移植性
  - D. 兼容性



6 Jackson 方法是一种面向( )的方法。

- A .对象  
B .数据结构  
C .数据流  
D .控制流

7 软件质量保证即为了确定、达到和( )需要的软件质量而进行的所有有计划、有系统的管理活动。

- A 测试                      B 维护                      C 质量                      D 效率

8. 原型的使用的开发过程,叫做 ( )

- A .原型期                                      B .原型生存期  
C .原型周期                                    D .以上说法都不对

9 结构化分析方法(SA)是一种面向( )的需求分析方法。

- A. 对象                      B. 数据结构                      C. 数据流                      D. 控制流

10. 模块( )定义为受该模块内一个判断影响的所有模块集合。

- A .控制域                      B .作用域                      C .宽度                      D .接口

11. 纯收入是累计效益现在值与投资之 ( )

- A .和                                  B .差  
C .积                                  D .商

12. 软件可靠性的定量指标,常借用硬件可靠性的定量度量方法来度量软件的可靠性,其中 MTTF 是( ), MTBF 是( )

- A .平均等待时间  
B .平均间隔时间  
C .平均失效等待时间  
D .平均失效间隔时间

13. 提高软件质量和可靠性的技术大致分为两大类( )和 ( )

- A .重用技术                      B .避开错误  
C .容错技术                      D .模块化设计

14. 版本管理是对系统不同的版本进行( )的过程。

- A .标识与跟踪                      B .标识变更  
C .发布变更                         D .控制变更

15. 一个项目是否开发,从经济上来说是否可行,归根结底是取决于对 ( )

- A .成本的估算                                      B 项目计划  
C .工程管理                                         D .工程网络图

16. 自顶向下估算方法的主要优点是对( )工作的重视,所以估算中不会遗漏系统级的成本估算,估算工作量小、速度快。它的缺点是往往不清楚( )上的技术性困难问题,而往往这些困难将会使成本上升。

- A .成本估算  
B .系统级  
C .低级别  
D .工程管理

17. 软件工具是指为支持计算机软件的开发、维护、模拟、移植或管理而研制的程序系统, 所以软件工具是一个 ( )

- A .软件工程                      B 项目管理工具  
C .程序系统                     D .需求分析工具

18 .CASE 是一组( )的集合。

- A .工具  
B .工具和方法  
C .方法  
D .程序

19.描述结构化系统分析方法的工具不包括 ( )

- |         |         |
|---------|---------|
| A .数据流图 | B 组织结构图 |
| C .数据词典 | D 结构化语言 |

20 .决策树 ( )

- A .能用来代替程序流程图
- B .是程序流程图的辅助手段
- C .是描述基本加工的逻辑功能的有效工具
- D .A 和 B

#### 四、简答题( 每小题 5 分,共 20 分)

- 1 .试述系统开发生命周期。
- 2 .什么是基本加工？描述表达基本加工逻辑功能的结构化工具的特点？
- 3 .开放式工作台有什么优点？
- 4 .增量模型的基本思想是什么？

#### 五、应用题( 共 25 分)

1 交通工具分为空中、陆上、水上交通工具,空中交通工具分为客机、货机、专用轻型机;陆上交通工具分为火车、汽车,火车和汽车又分客车、货车,水上交通工具有轮船,轮船分为客轮、货轮、客货混合轮。建立交通工具的对象模型。

2 某学校对学生成绩的评定办法为:若期末考试成绩大于等于 90 分,作业情况为好的成绩为优,而作业成绩为差的定为良;否则,若期末考试成绩大于等于 75,作业情况为好的成绩定为良,而作业情况为差的成绩定为及格;若期末考试成绩大于等于 60 分,作业情况为好的成绩定为及格,而作业情况为差的成绩定为不及格,期末考试成绩低于 60 分,成绩定为不及格。上述功能请用决策树和决策表表示。

3 用 PAD 图描述下面问题的控制结构。

有一个表  $A(1), A(2), \dots, A(N)$  按递增顺序排列。给定一个 Key 值,在表中用折半法查找。若找到,将表位置  $i$  送入  $x$ ,否则将零送到  $x$ ,同时将 Key 值插入表中。

# 软件工程考前模拟试题(一)参考答案

## 一、名词解释题

- 1 是使描述问题的问题空间与解决问题的方法空间在结构上尽可能一致。
- 2 也称结构测试,是机器测试的一种方式,即将软件看作白盒子,按照程序的内部结构和处理逻辑来选定测试用例,对软件的逻辑路径及过程进行测试,检查它与设计是否相等。
- 3 系统设计说明书是从系统总体的角度出发对系统建设中各主要技术方面的设计进行说明,是系统设计阶段的产物,其着重点在于阐述系统设计的指导思想以及所采用的技术路线和方法,编写系统设计说明书将为后续的系统开发工作从技术和指导思想上提供必要的保证。
- 4 对现有技术进行评价,分析系统是否可以用现有技术来实施以及技术发展对系统建设有什么影响。
- 5 又称判断树,是一种图形工具,适合于描述加工中具有多个策略,而且每个策略和若干条件有关的逻辑功能。

## 二、填空题

- 1 软件
- 2 逻辑
- 3 生存周期 线性顺序
- 4 用户需求 对象
- 5 结构化设计 数据流
- 6 模块
- 7 深度 宽度 扇入 扇出
- 8 数据耦合
- 9 公共耦合
- 10 处理

## 三、选择题

- |         |         |         |       |       |
|---------|---------|---------|-------|-------|
| 1 .A    | 2 .A    | 3 .C    | 4 .A  | 5 .A  |
| 6 .B    | 7 .B    | 8 .B    | 9 .C  | 10 .B |
| 11 .B   | 12 .C、D | 13 .B、C | 14 .A | 15 .A |
| 16 .B、C | 17 .C   | 18 .B   | 19 .B | 20 .C |

## 四、简答题

- 1 系统开发是管理信息系统建设中最重要的一個阶段,从项目开发开始到结束的整个过程,称为系统开发生命周期。  
系统开发生命周期一般分为以下阶段:  
(1)系统分析

这一阶段的主要任务是明确用户的信息需求,提出新系统的逻辑方案。需要进行的工作有系统的初步调查,可行性研究,现行系统的详细调查及新系统逻辑模型的提出等。

## (2)系统设计

这一阶段的主要任务是根据新系统的逻辑方案进行软、硬件系统的设计,其中包括总体结构设计、输出设计、输入设计、处理过程设计、数据存储设计和计算机系统方案的选择等。

## (3)系统实施

这一阶段将设计的系统付诸实施,主要工作有软件的程序编制与软件包的购置、计算机与通讯设备的购置、系统的安装、调试与测试、新旧系统的转换等。

### 2 数据流图中所有不进一步分解的加工,称为基本加工。

描述表达基本加工逻辑功能的结构化工具主要有:

#### (1)自然语言的文字叙述

#### (2)结构化语言

#### (3)决策树

#### (4)决策表

#### (5)数学公式

#### (6)上述各工具的联合使用

其特点如下所述:

(1)自然语言语义丰富,语法灵活,可描述十分广泛而复杂的问题,表达人们丰富的感情和智慧。

(2)结构化语言没有严格的语法规则,使用的词汇比形式化的计算机语言广泛,但使用的语句类型很少,结构规范,表达的内容清晰、准确、易理解,不易产生歧义。

(3)决策树清晰地表达了在什么情况下应采取什么策略,不易产生逻辑上的混乱。是描述基本加工的逻辑功能的有效工具。

(4)决策表将比较复杂的决策问题简洁、明确、一目了然地描述出来。它是描述条件比较多的决策问题的有效工具。

### 3 (1)易将某个工具加入到开放式工作台,还可以用新的工具取代已有的工具。

(2)可以由一个配置管理系统来管理由工具输出的文件。

(3)能不断增强工作台的功能,不断发展工作台。

(4)工作台不依赖于某个供应商,而能从不同销售商处购买工具。如果一个工具开发商不提供支持了,最多只影响该工作台的一部分工具,其余的工具还可以继续使用。

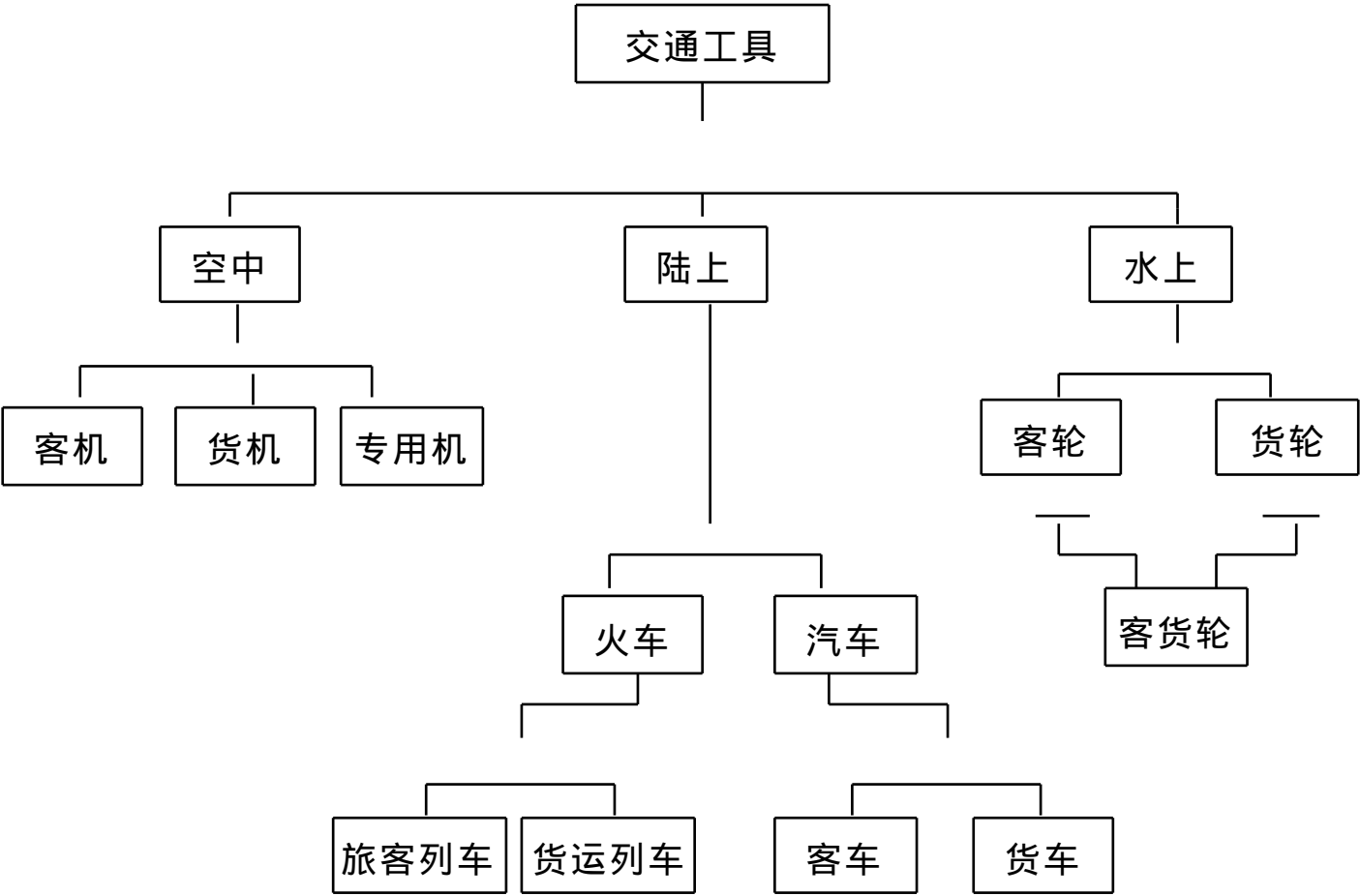
4 为了克服瀑布模型的局限性,使开发过程具有一定的灵活性和可修改性,于是产生了增量模型。它是在瀑布模型的基础上加以修改而形成的。

增量模型和瀑布模型之间的本质区别是:瀑布模型属于整体开发模型,它规定在开始下一个阶段的工作之前,必须完成前一阶段的所有细节。而增量模型属于非整体开发模型,它推迟某些阶段或所有阶段中的细节,从而较早的产生工作软件。

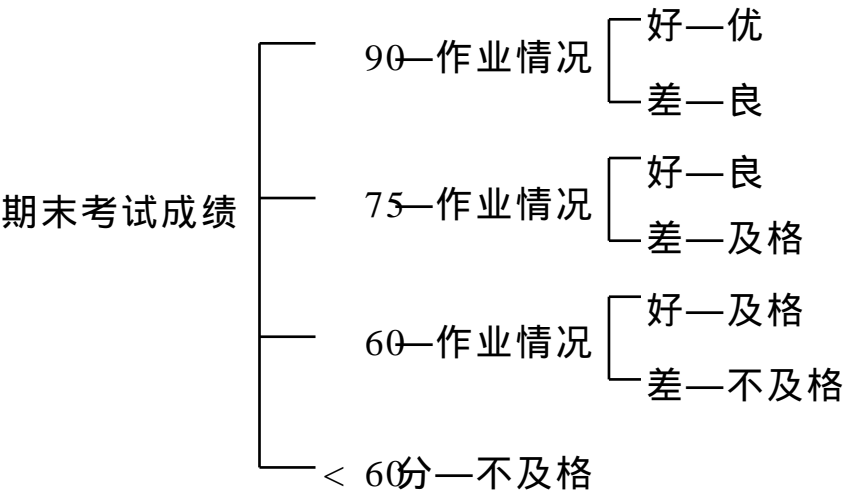
增量模型是在项目的开发过程中以一系列的增量方式开发系统。增量方式包括增量开发和增量提交。增量开发是指在项目开发周期内,以一定的时间间隔开发部分工作软件;增量提交是指在项目开发周期内,以一定的时间间隔增量方式向用户提交工作软件及相应文档。增量开发和增量提交可以同时使用,也可单独使用。

## 五、应用题

### 1 交通工具的对象模型如图所示。



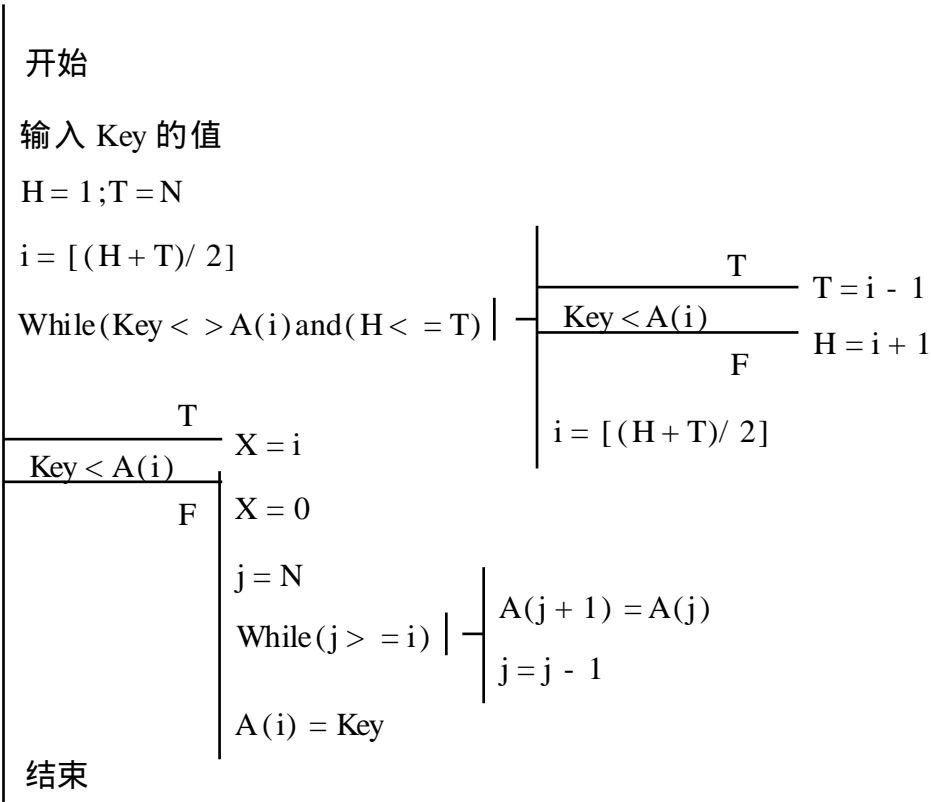
2 决策树如下：



决策表如下：

期末考试成绩 90	Y	Y	N	N			
期末考试成绩 75			Y	Y		N	N
期末考试成绩 60					Y	Y	N
作业情况好	Y	N	Y	N	Y	N	
优	*						
良		*	*				
及格				*	*		
不及格						*	*

3 .PAD 图：





- 6 软件开发环境支持一定的( )或按照一定的软件开发模型组织而成。  
A 软件生存周期  
B 软件过程  
C 软件开发方法  
D 软件开发模型
- 7 块间的信息可以作‘控制信息’用,也可以作为( )使用。  
A 控制流  
B 数据结构  
C 控制结构  
D 数据
- 8 在考察系统的一些涉及时序和改变的状况时,要用动态模型来表示。动态模型着重于系统的控制逻辑,它包括两个图:一个是事件追踪图,另一个是( )  
A 数据流图  
B 状态图  
C 系统结构图  
D 时序图
- 9 软件开发过程中,抽取和整理用户需求并建立问题域精确模型的过程叫( )  
A 生存期  
B 面向对象设计  
C 面向对象程序设计  
D 面向对象分析
- 10 在进行软件结构设计时应遵循的最主要的原理是( )原理。  
A 抽象  
B 模块化  
C 模块独立  
D 信息隐藏
- 11 项目开发计划是一个( )文档。  
A 技术性  
B 管理性  
C 需求分析  
D 设计
- 12 年利率为  $i$ , 现存入  $P$  元, 不计复利,  $n$  年后本金利息为 ( )  
A  $P \times (1 + i)^n$   
B  $P \times (1 + i^n)$   
C  $P \times (1 + i \times n)$   
D  $P \times (1 + i) \times n$
- 13 软件是不可见的复杂的逻辑实体,不同于任何其他制造业的产品。使得软件质量难于把握的一个因素是( )  
A 软件需求  
B 硬件需求  
C 软件配置  
D 硬件配置
- 14 基线是软件生存期中各开发阶段的一个特定点,它可作为一个检查点,当采用的基线发生错误时,我们可以返回到最近和最恰当的( )上。  
A 配置项  
B 程序  
C 基线  
D 过程
- 15 ( )是软件产品的重要组成部分,它在产品的开发过程起着重要的作用。  
A 需求说明  
B 概要说明  
C 软件文档  
D 测试大纲
- 16 以语言为中心的程序设计环境支持软件生存期( )活动,特别强调对编程、调试和测试活动的支持。  
A 前期  
B 后期  
C 中期  
D 初期
- 17 ( )是软件开发环境中最主要的组成部分。  
A 软件工程  
B 项目管理工具  
C 软件工具  
D 需求分析工具
- 18 下面对可行性研究报告描述正确的是( )  
A 是系统开发任务是否下达的决策依据



- B .是系统分析阶段的工作总结
  - C .是系统分析人员和用户交流的主要手段
  - D .是系统设计阶段工作的依据
- 19 .系统分析阶段的主要成果是 ( )
- A .DFD 图
  - B .系统流程图
  - C .详细调查报告
  - D .系统说明书
- 20 .可行性研究的工作结果是 ( )
- A .项目开发的初步方案
  - B .确定新项目开发有无必要和可能
  - C .提供当前现存信息系统的概括
  - D .可行性研究报告和系统设计任务书

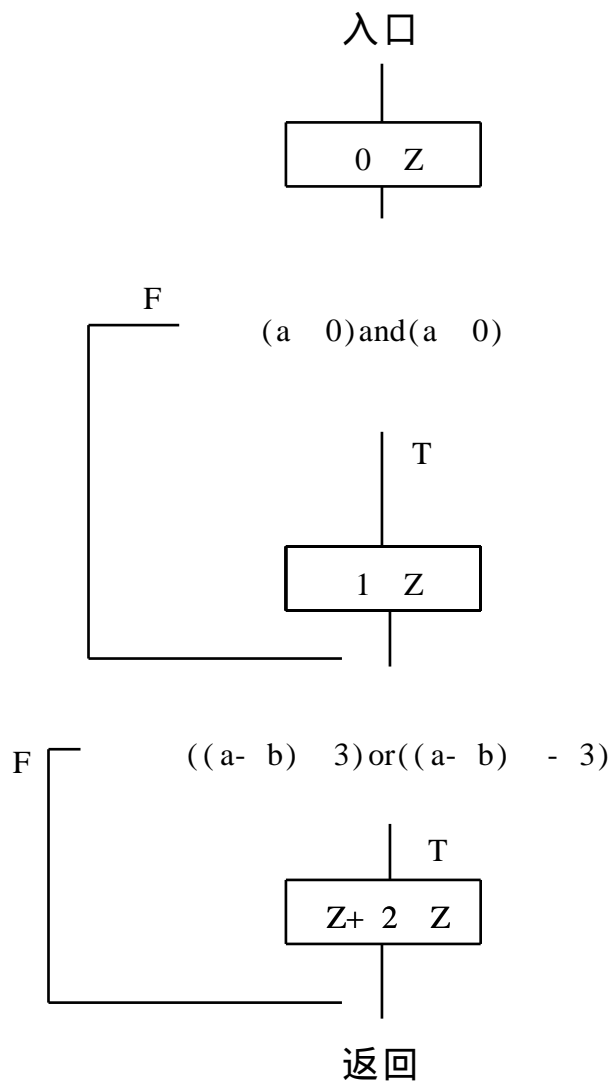
#### 四、简答题( 每小题 5 分,共 20 分)

- 1 系统分析的含义是什么 ?
- 2 试述结构化方法的基本含义和主要内容。
- 3 结构冗余按其工作方式分那三种 ?
- 4 试述容错软件定义

#### 五、应用题( 共 25 分)

1 .下图中描述了某个子程序的处理流程,根据判定覆盖、条件覆盖、判定/ 条件覆盖、条件组合覆盖、路径覆盖等五种覆盖标准,从供选择的答案中分别找出满足相应标准的最小的测试数据组。

- |         |         |         |         |
|---------|---------|---------|---------|
| a = 5   | b = 1   | a = 5   | b = - 1 |
| a = 5   | b = 1   | a = 5   | b = 1   |
| a = - 5 | b = - 1 | a = 0   | b = - 1 |
| a = 5   | b = - 1 | a = 5   | b = 1   |
| a = - 5 | b = 1   | a = 0   | b = 0   |
| a = - 5 | b = - 1 | a = - 5 | b = - 1 |
| a = 5   | b = 1   | a = 5   | b = 1   |
| a = 0   | b = 1   | a = 0   | b = - 1 |
| a = 0   | b = - 1 | a = - 5 | b = 1   |
| a = - 5 | b = 1   | a = - 5 | b = - 1 |

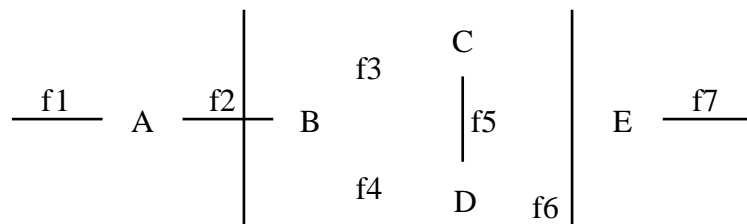


流程图

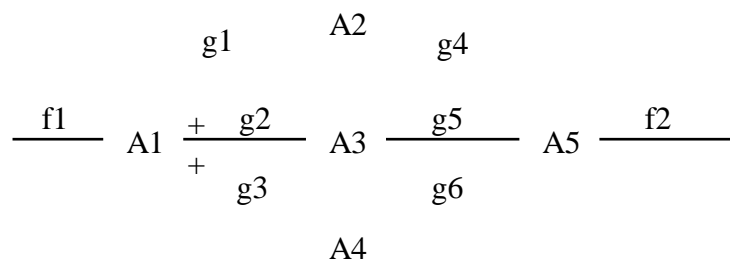
2 .一本书的组成有一个封面、一个目录、一个前言、若干章,每章有若干节,每节有若干段,每段有若干句子,每节有 0 个或多个插图,还有 0 个或多个表格,最后有一个附录。建立该书的对象模型。

3 .请将下图的 DFD 转换为软件结构图(注:图中的 表示“ 或者 ”)

主图:



子图 A:



# 软件工程考前模拟试题(二)参考答案

## 一、名词解释题

- 1 .计算机辅助软件工程或者计算机辅助系统工程的简称,其目的是实现系统开发生命周期内各阶段工作的基于计算机的自动化。
- 2 对组织的经济状况和投资能力进行分析,对系统建设,运行和维护费用进行估算,对系统建成后可能取得的社会及经济效益进行估计。
- 3 又称判断表,是一种图形工具,适合于描述加工判断的条件较多,各条件又相互组合的逻辑功能,它共分四大部分:条件、状态、决策方案和决策规则。
- 4 也称功能测试,是机器测试的一种方式,即将软件看作黑盒子,在完全不考虑程序的内部结构的特性的情况下,测试软件的外部特性。
- 5 基于系统生命周期的结构化方法,为管理信息系统建设提供了规范的步骤、准则与工具,以弥补早期方法的不足。

## 二、填空题

- 1 经济 软件项目
- 2 有形 无形
- 3  $R \times P \times (1 + n \times i)$
- 4  $S \times F \times (1 + n \times i)$
- 5 累计的经济效益 项目投资
- 6 功能需求
- 7 需求规格说明书
- 8 分解
- 9 DFD DD
- 10 数据流

## 三、选择题

- |       |       |       |       |       |
|-------|-------|-------|-------|-------|
| 1 .C  | 2 .C  | 3 .D  | 4 .A  | 5 .D  |
| 6 .C  | 7 .D  | 8 .B  | 9 .D  | 10 .C |
| 11 .B | 12 .C | 13 .A | 14 .C | 15 .C |
| 16 .B | 17 .C | 18 .A | 19 .D | 20 .D |

## 四、简答题

- 1 “分析”通常是指对现有系统的内、外情况进行调查、分析、研究、分解、剖析,以明确问题或机会所在,认识解决这些问题或把握这些机会的必要性,为确定有关活动的目标和可能的方案提供科学依据。本章所讨论的系统分析(systems analysis),是指在管理信息系统开发的生命周期中系统分析阶段的各项

活动和方法。系统分析也指应用系统思想和系统科学的原理进行分析工作的方法与技术。

2 结构化的含义是用一组规范的步骤、准则和工具来进行某项工作。

结构化方法是把整个系统开发过程分为若干阶段,每个阶段进行若干活动,每项活动应用一系列标准、规范、方法和技术,完成一个或多个任务,形成符合规范的产品。

其主要原则为:

(1)用户参与的原则

(2)“先逻辑,后物理”的原则

(3)“自顶向下”的原则

(4)工作成果描述标准化原则

3 结构冗余是通常用的冗余技术。按其工作方式,它分为静态、动态和混合冗余三种。

(1)静态冗余。常用的有:三模冗余 TMR(Triple Modular Redundancy)和多模冗余。静态冗余通过表决和比较来屏蔽系统中出现的错误。如三模冗余,是对三个功能相同但由不同的人采用不同的方法开发出现的模块的运行结果通过表决,以多数结果作为系统的最终结果。即如果模块中有一个出错,这个错误能够被其他模块的正确结果“屏蔽”。由于无需对错误进行特别的测试,也不必进行模块的切换就能实现容错,故称为静态容错。

(2)动态冗余。动态冗余的主要方式是多重模块待机储备,当系统检测到某工作模块出现错误时,就用一个备用的模块来顶替它并重新运行。这里须有检测、切换和恢复过程,故称其为动态冗余。每当一个出错模块被其备用模块顶替后,冗余系统相当于进行了一次重构。各备用模块在其待机时,可与主模块一样工作,也可不工作。前者叫做热备份系统,后者叫做冷备份系统。在热备份系统中备用模块在待机过程中其失效率为 0。

(3)混合冗余。它兼有静态冗余和动态冗余的长处。

4 归纳容错软件的定义,有以下四种:

(1)规定功能的软件,在一定程度上对自身错误的作用(软件错误)具有屏蔽能力,则称此软件为具有容错功能的软件,即容错软件。

(2)规定功能的软件,在一定程度上能从错误状态自动恢复到正常状态,则称之为容错软件。

(3)规定功能的软件,在因错误而发生错误时,仍然能在一定程度上完成预期的功能,则把该软件称为容错软件。

(4)规定功能的软件,在一定程度上具有容错能力,则称之为容错软件。

## 五、应用题

1 达到判定覆盖为

达到条件覆盖为

达到判定/条件覆盖为

达到条件组合覆盖为

达到路径覆盖为

2 .书的对象模型如图所示。



# 软件工程考前模拟试题(三)

## 一、名词解释题(每小题 3 分,共 15 分)

- 1 对象
- 2 类
- 3 数据字典
- 4 消息
- 5 信息隐蔽

## 二、填空题(每小题 2 分,共 20 分)

1. \_\_\_\_\_是描述软件开发过程中各种活动如何执行的模型。
- 2 维护的副作用有编码副作用、\_\_\_\_、文档副作用三种。
- 3 可行性研究需要从以下三个方面分析研究每种解决方案的可行性:技术可行性、经济可行性、\_\_\_\_\_。
- 4 在需求分析阶段要进行以下几方面的工作:问题识别、\_\_\_\_、编写文档。
- 5 在 SA 方法的需求描述工具中,数据流图描述系统的分解,即描述系统由哪几部分组成,各部分之间有什么联系等等。数据字典定义了数据流图中每一个图形元素;结构化语言、判定表或判定树则详细描述数据流图中不能被再分解的\_\_\_\_\_。
- 6 在一个模块中,\_\_\_\_\_反映模块的外部特性,逻辑反映它的内部特性。
- 7 详细描述处理过程常用三种描述工具:图形、表格和\_\_\_\_\_。
- 8 软件测试时需要三类信息:软件配置、\_\_\_\_\_和测试工具。
- 9 任何程序都可由顺序、选择、\_\_\_\_\_三种基本控制结构构造。
- 10 测试用例应由输入数据和预期的\_\_\_\_\_两部分组成。这样便于对照检查。

## 三、选择题(每小题 1 分,共 20 分)

- 1 为了提高软件的质量和( ),软件质量保证的主要任务有:力争不重复劳动,掌握开发新软件的方法等八类任务。  
A 测试                      B 维护                      C 质量                      D 效率
- 2 质量保证是为了保证产品和服务充分满足消费者要求的质量而进行的有计划、有组织的活动。质量保证是为了使产品实现( )的功能。  
A 系统分析员                      B 程序员  
C 软件开发者要求                      D 用户要求
- 3 ( )数据处理问题的的工作过程大致分为三步,即取得数据、变换数据和给出数据。  
A 变换型                      B 事务型                      C 结构化                      D 非结构化
- 4 原型化方法是一种( )型的设计过程。  
A 自外向内                      B 自顶向下  
C 自内向内                      D 自底向下
- 5 ( )是以提高软件质量为目的的技术活动。

- A .技术创新  
C .技术改造
- B 测试  
D 技术评审
- 6 因果图方法是根据( )之间的因果关系来设计测试用例的。  
A .输入与输出  
B .设计与实现  
C .条件与结果  
D 主程序和子程序
- 7 由于软件项目的特点和运行原型的目的不同,原型有三种不同的作用类型:探索型、( )和进化型。  
A .实验型  
B 经验型  
C 追加型  
D .废弃型
- 8 在进行软件测试时,首先应当进行单元测试,然后再进行( ),最后再进行有效性测试。  
A .组合测试  
B 集成测试  
C .有效性测试  
D 确认测试
- 9 在实现阶段要完成的工作之一是单元测试,单元测试要根据在( )阶段中的规格说明进行。  
A .可行性研究和计划  
B 需求分析  
C .概要设计  
D 详细设计
- 10 按照软件配置管理的原始指导思想,受控制的对象应是 ( )  
A .软件元素  
B 软件配置项  
C .软件项目  
D 软件过程
- 11 在结构测试用例设计中,有语句覆盖、条件覆盖、判定覆盖(即分支覆盖)、路径覆盖等,其中( )是最强的覆盖准则。  
A .语句覆盖  
B 条件覆盖  
C .判定覆盖  
D 路径覆盖
- 12 软件的开发与维护划分为八个阶段,其中单元测试是在( )阶段完成的。  
A .概要设计  
B 详细设计  
C .编码  
D 测试
- 13 模块的内部过程描述就是模块内部的( ),它的表达形式就是详细设计语言。  
A .模块化设计  
B 算法设计  
C .程序设计  
D 详细设计
- 14 进行需求分析可使用多种工具,但( )是不适用的。  
A .数据流图  
B 判定表  
C .PAD 图  
D 数据词典
- 15 原型化方法是用户和设计者之间执行的一种交互过程,适用于( )系统。  
A .需求不确定性高的  
B 需求确定的  
C .管理信息  
D 实时
- 16 为了最终实现目标系统,必须设计出组成这个系统的所有程序和文件,通常分为两个阶段完成,即( )和过程设计。  
A .程序设计  
B 结构设计  
C .系统设计  
D 详细设计
- 17 Jackson 方法根据( )来导出程序结构。  
A .数据结构  
B 数据间的控制结构  
C .数据流图  
D .IPO 图
- 18 版本用来定义软件配置项的 ( )  
A .演化阶段  
B 环境

C.要求

D.软件工程过程

19.支持计算机软件的开发、维护、模拟、移植和管理而研制的程序系统称为 ( )

A.软件工具

B.软件环境

C.软件过程

D.软件模型

20.面向数据流的设计方法把( )映射成软件结构。

A.数据流

B.系统结构

C.控制结构

D.信息流

#### 四、简答题(每小题 5 分,共 20 分)

- 1.什么是软件工程?软件工程的性质是什么?
- 2.什么是结构化分析方法?该方法使用什么描述工具?
- 3.什么是确认测试?该阶段有哪些工作?
- 4.快速原型模型有几种?各有何特点?

#### 五、应用题(共 25 分)

1.某电器集团公司下属一个成套厂(产品组装)和若干零件厂等单位,成套厂下属技术科、生产科、供应科等基层单位。现要建立一个计算机辅助企业管理系统,其中,生产科的任务是:

- (1)根据销售公司转来的内部合同(产品型号、规格、数量、交货日期)制定车间月生产计划;
- (2)根据车间实际生产日报表、周报表调整月生产计划;
- (3)以月生产计划为依据,制定产品设计(结构、工艺)及产品组装月计划;
- (4)将产品的组装计划传达到技术科,将组装月计划分解为周计划,下达给车间。

技术科的任务是:

(1)根据生产科转来的组装计划进行产品结构设计,产生产品装配图给生产科,产生外购需求计划给供应科,并产生产品自制件物料清单;

(2)根据组装计划进行产品工艺设计(冲压、喷漆、焊接等),根据产品自制件物料清单产生工艺流程图给零件厂。

供应科的任务是:

- (1)根据技术科的外购需求计划和仓库的缺货通知单及月盘存表制定采购计划给采购员;
- (2)对采购来的材料进行库管理(登录、查询、修改、删除)。

请根据以上文字叙述画出企业管理系统的分层 DFD 图。

2.某图书借阅系统有以下功能:

(1)借书:根据读书的借书证查询读者档案,若借书数目未超过规定数量,则办理借阅手续(修改库存记录及读者档案),超过规定数量者不予借阅。对于第一次借阅者则直接办理借阅手续。

(2)还书:根据读者书中的条形码,修改库存记录及读者档案,若借阅时间超过规定期限则罚款。

请对以上问题,画出分层数据流图。

3.用 SA 方法画出下列问题的顶层和 0 层数据流图。

某运动会管理系统接受来自运动员的报名表、裁判的比赛项目及项目成绩,产生运动员号码单发送给运动员,项目参加者发送给裁判,单项名次、团体名次发送给发布台。该系统有两部分功能:

(1)登记报名表:接受报名表、比赛项目,产生运动员号码单、项目参加者,形成运动员名单及团体成绩表两种数据存储。

(2)统计成绩:接受项目成绩,查询运动员名单,产生单项名次,填写团体成绩,最后产生团体名次。



# 软件工程考前模拟试题(三)参考答案

## 一、名词解释题

- 1 对象是人们要进行研究的任何事物,从最简单的整数到复杂的飞机等均可看作对象,它不仅能表示具体的事物,还能表示抽象的规则、计划和事件。
- 2 具有相同或相似性质的对象的抽象就是类。
- 3 数据字典(Date Dictionary,简称 DD)就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。
- 4 对象之间进行通信的构造叫做消息。
- 5 信息隐蔽指在设计和确定模块时,使得一个模块内包含的信息(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。

## 二、填空题

- 1 软件生存周期模型
- 2 数据副作用
- 3 社会可行性
- 4 分析与综合、导出软件的逻辑模型
- 5 每一个加工
- 6 功能、状态与接口
- 7 语言
- 8 测试配置
- 9 重复
- 10 输出数据

## 三、选择题

- |       |       |       |       |       |
|-------|-------|-------|-------|-------|
| 1 .D  | 2 .D  | 3 .A  | 4 .A  | 5 .D  |
| 6 .A  | 7 .A  | 8 .B  | 9 .B  | 10 .B |
| 11 .D | 12 .C | 13 .B | 14 .C | 15 .A |
| 16 .B | 17 .A | 18 .A | 19 .A | 20 .A |

## 四、简答题

- 1 软件工程是用科学知识的技术原理来定义、开发、维护软件的一门学科。  
软件工程是一门综合性的交叉学科,它涉及计算机科学、工程科学、管理科学、数学等领域。计算机科学中的研究成果均可用于软件工程,但计算机科学着重于原理和理论,而软件工程着重于如何建造一个软件系统。软件工程要用工程科学中的观点来进行费用估算、制定进度、制定计划和方案。软件工程要用管理科学中的方法和原理进行软件生产的管理。软件工程要用数学的方法建立软件开发中的各种

模型和各种算法,如可靠性模型,说明用户需求的形式化模型等。

2 结构化分析(Structured Analysis,简称 SA),是面向数据流进行需求分析的方法。SA 是一种建模活动,该方法使用简单易读符号,根据软件内部数据传递、变换的关系,自顶向下逐层分解,描绘出满足功能要求的软件模型。

结构化分析的主要思想是采取自顶向下逐层分解的分析策略,即面对一个复杂的问题,分析人员不可能一开始就考虑到问题的所有方面以及全部细节,采取的策略往往是分解,把一个复杂的问题划分成若干小问题,然后再分别解决,将问题的复杂性降低到人可以掌握的程度。分解可分层进行,先考虑问题最本质的方面,忽略细节,形成问题的高层概念,然后再逐层添加细节,即在分层过程中采用不同程度的“抽象”级别,最高层的问题最抽象,而低层的较为具体。

结构化分析(SA)方法利用图形等半形式化的描述方式表达需求,简明易懂,用它们形成需求说明书中的主要部分。这些描述工具是:

- (1)数据流图;
- (2)数据字典;
- (3)描述加工逻辑的结构化语言、判定表、判定树。

其中,“数据流图”描述系统的分解,即描述系统由哪几部分组成,各部分之间有什么联系等等。“数据字典”定义了数据流图中每一个图形元素。结构化语言、判定表或判定树则详细描述数据流图中不能被再分解的每一个加工。

3 确认测试又称有效性测试。它的任务是检查软件的功能与性能是否与需求规格说明书中确定的指标相符合。因而需求规格说明是确认测试的基础。

确认测试阶段有两项工作:进行确认测试与软件配置审查。

(1)进行确认测试。确认测试一般是在模拟环境下运用黑盒测试方法,由专门测试人员和用户参加的测试。确认测试需要需求规格说明书、用户手册等文档,要制定测试计划,确定测试的项目,说明测试内容,描述具体的测试用例,测试用例应选用实际运用的数据。测试结束后,应写出测试分析报告。

经过确认测试后,可能有两种情况:功能、性能与规格说明一致,该软件系统是可以接受的。功能、性能与规格说明有差距,要提交一份问题报告。对这样的错误进行修改、工作量非常大,必须同用户协商。

(2)软件配置审查。软件配置审查的任务是检查软件的所有文档资料的完整性、正常性,如发现遗漏和错误,应补充和改正。同时要编好目录,为以后的软件维护工作奠定基础。

4 原型模型又称快速原型模型,它是增量模型的另一种形式。它是在开发真实系统之前,构造一个原型,在该原型的基础上,逐渐完成整个系统的开发工作。根据原型的不同作用,有三类原型模型:

(1)探索型原型。这种类型的原型模型的是把原型用于开发的需求分析阶段,目的是要弄清用户的需求,确定所期望的特性,并探索各种方案的可行性。它主要针对开发目标模糊、用户与开发者对项目都缺乏经验的情况,通过对原型的开发来明确用户的需求。

(2)实验型原型。这种原型主要用于设计阶段,考核实现方案是否合适,能否实现。对于一个大型系统,若对设计方案心中没有把握时,可通过这种原型来证实设计方案的正确性。

(3)演化型原型。这种原型主要用于及早向用户提交一个原型系统,该原型系统或者包含系统的框架,或者包含系统的主要功能,在得到用户的认可后,将原型系统不断扩充演变为最终的软件系统。它将原型的思想扩展到软件开发的全过程。

## 五、应用题

1 .

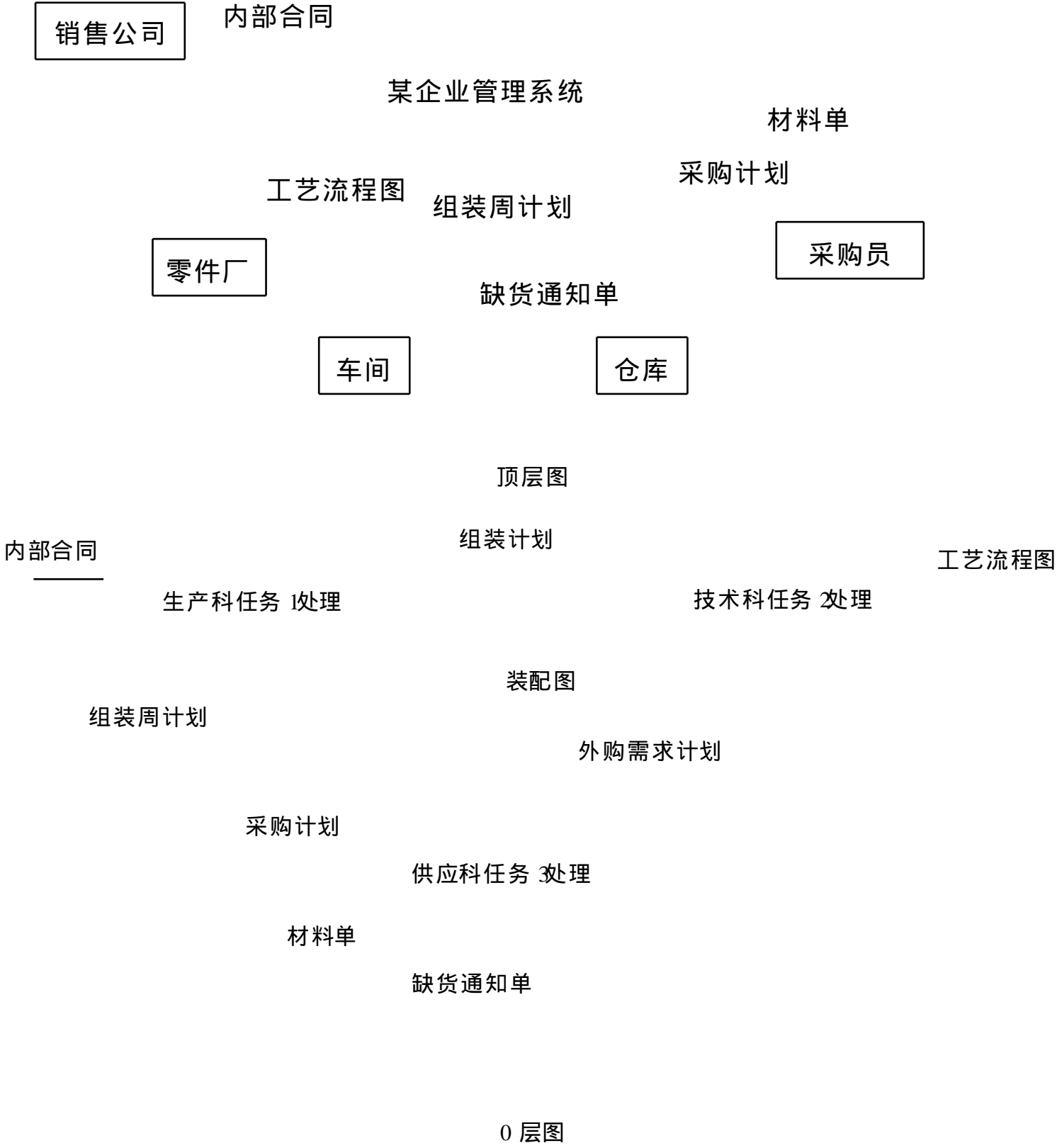


图 1

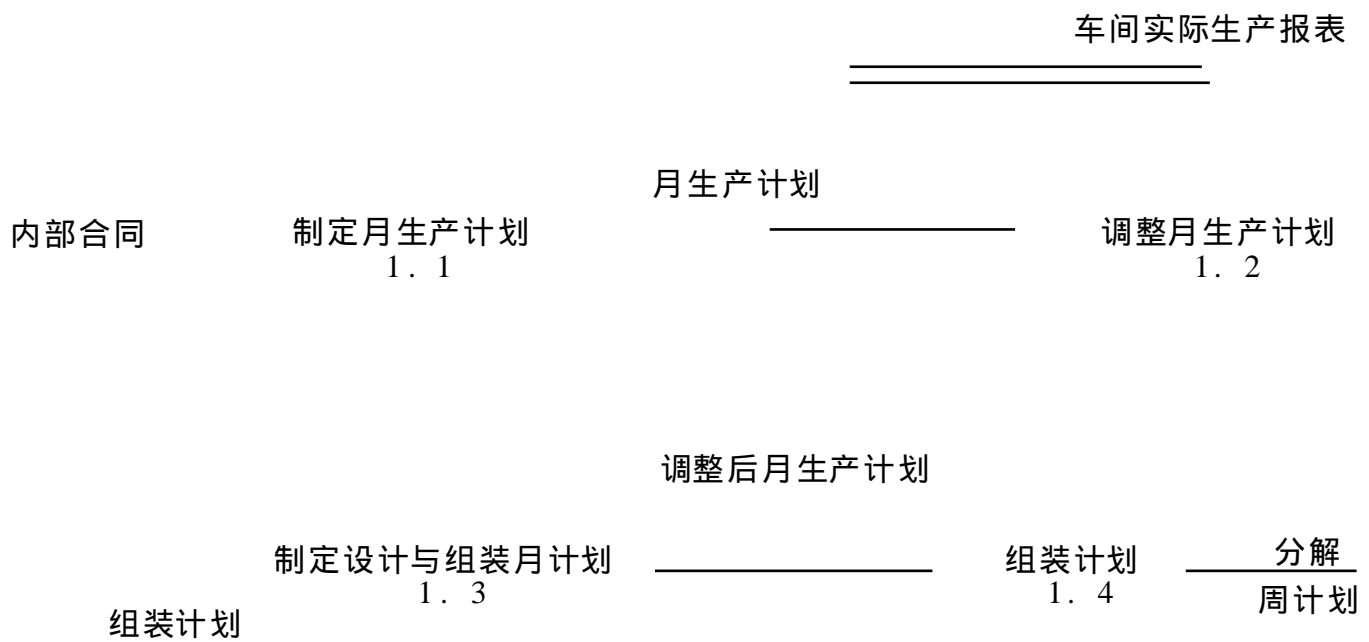
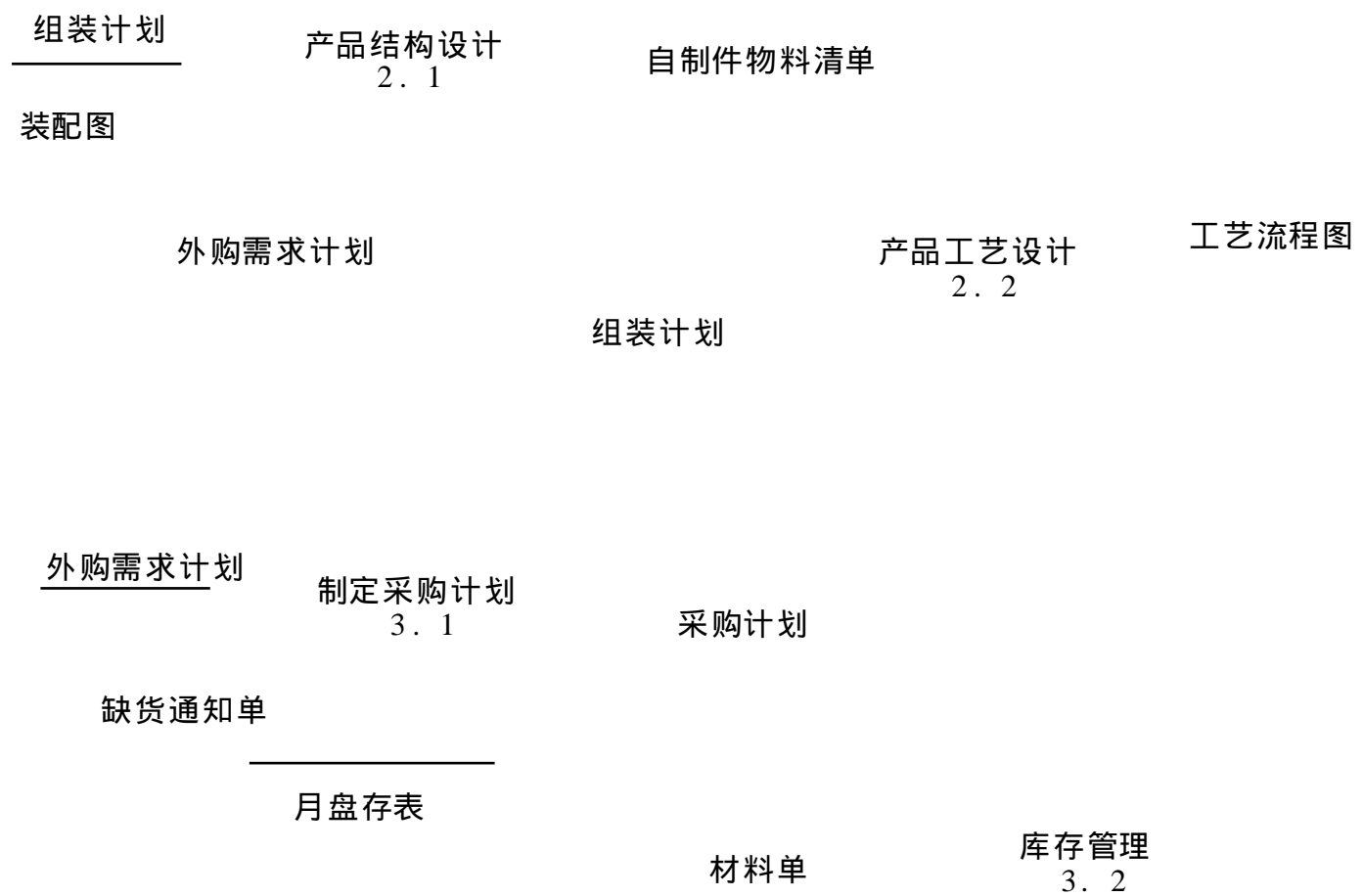
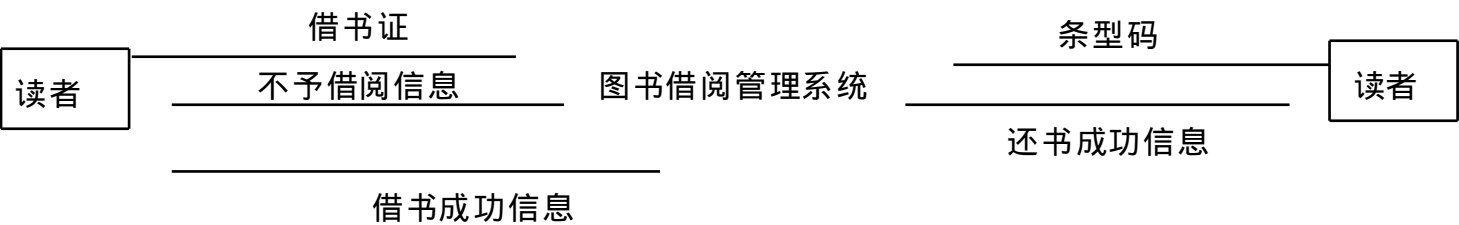


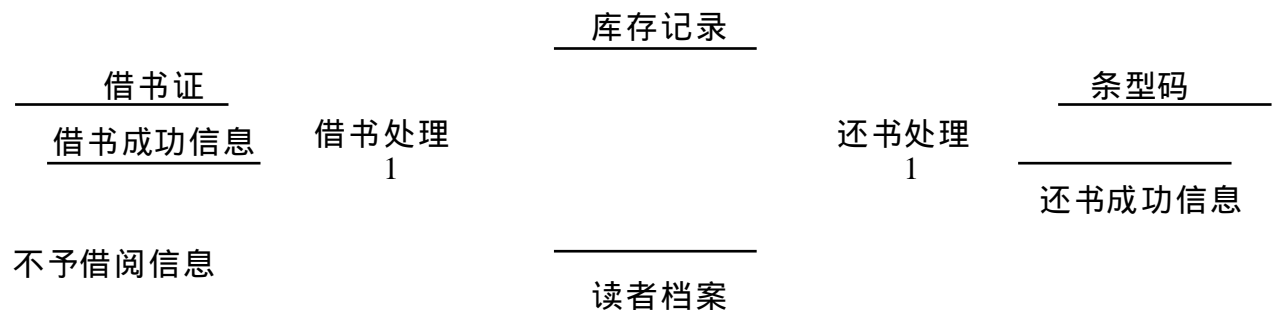
图 2



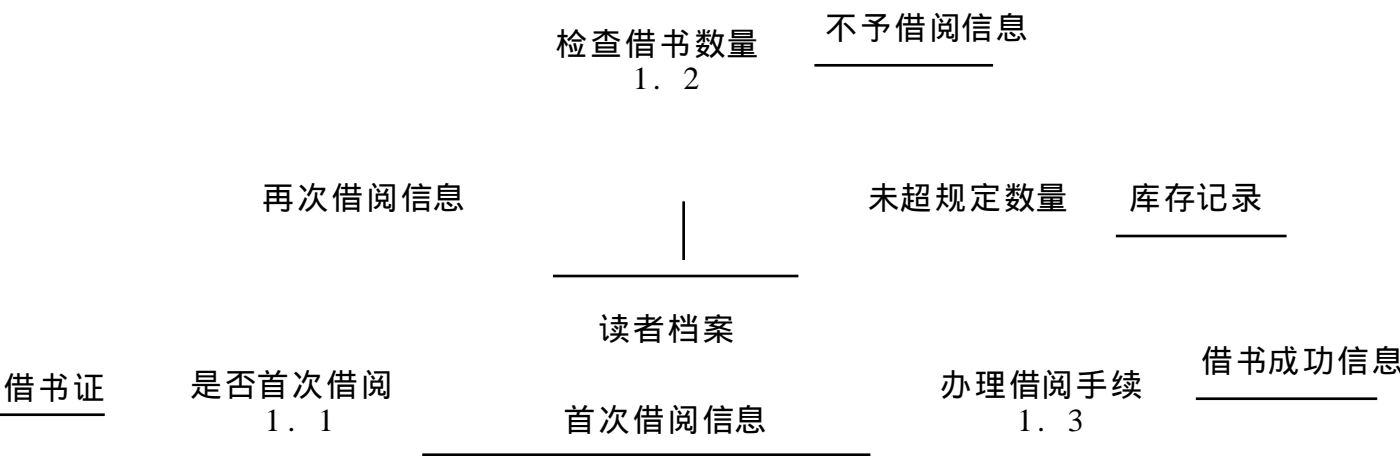
## 1 层图



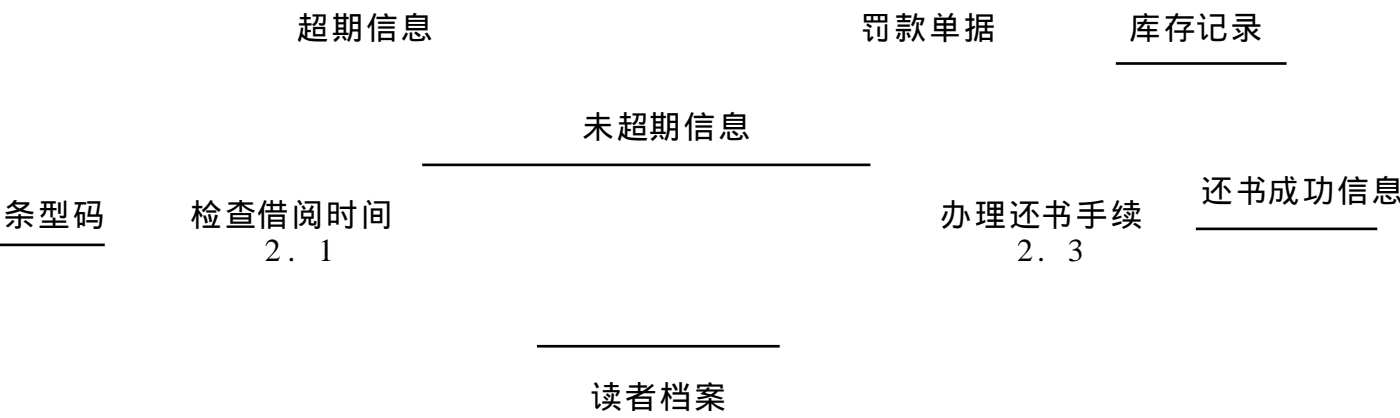
顶层图



0 层图

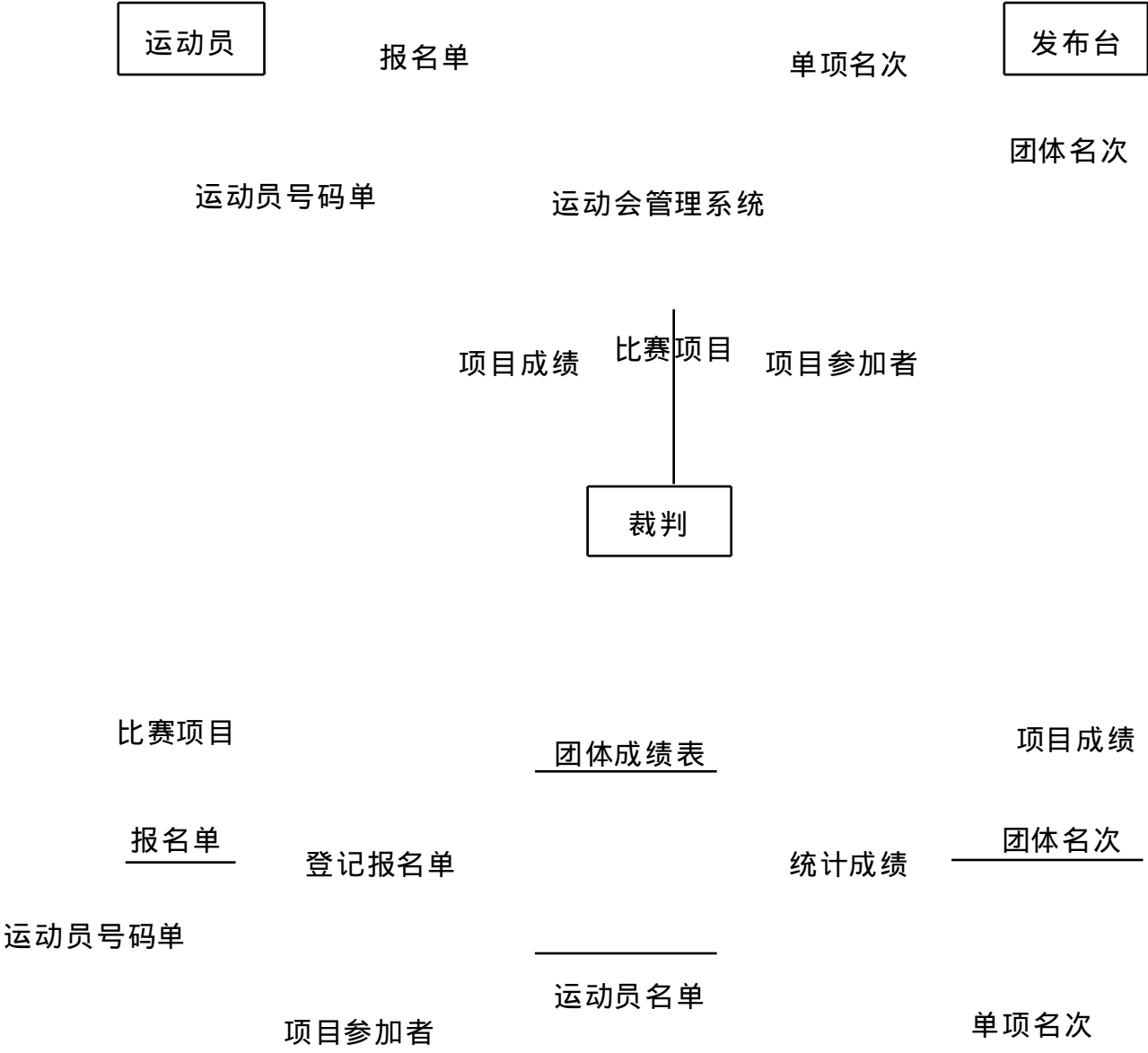


罚款处理  
2. 2



1 层图

3 .



# 软件工程考前模拟试题(四)

## 一、名词解释题(每小题 3 分,共 15 分)


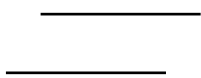
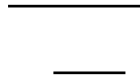
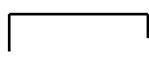
- 1 模块化
- 2 模块独立性
- 3 桩模块
- 4 原型
- 5 软件工程

## 二、填空题(每小题 2 分,共 20 分)

- 1.可行性研究的目的是用\_\_\_\_\_的代价,在尽可能\_\_\_\_\_的时间内,确定该项目是否能够\_\_\_\_\_。
- 2.可行性研究实质上是进行一项\_\_\_\_\_、压缩了的需求分析、\_\_\_\_\_过程。
- 3.可行性研究要在\_\_\_\_\_层次上以\_\_\_\_\_方式进行需求分析和设计。
- 4.可以从三方面研究可行性,即\_\_\_\_\_可行性、\_\_\_\_\_可行性、\_\_\_\_\_可行性。
- 5.系统流程图是描述\_\_\_\_\_的传统工具,用图形符号表示系统中各个\_\_\_\_\_,表达了系统中各种元素之间的\_\_\_\_\_情况。
- 6.数据字典中有四类条目,分别是\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_。
- 7.当数据流图中某个加工的一组动作存在着多个条件复杂组合的判断时,使用\_\_\_\_\_或\_\_\_\_\_较好。
8. IDEF 图是一种\_\_\_\_\_模型,表示系统功能的图形称为\_\_\_\_\_图形,连方框上的箭头有四种类型,它们分别是\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_。
- 9.为了较完整地描述用户对系统的需求,DFD 应与数据库中的\_\_\_\_\_图结合起来。
- 10.软件设计阶段产生的最重要的文档之一是\_\_\_\_\_。

## 三、选择题(每小题 1 分,共 20 分)

- 1 软件测试的目的是 ( )  
A .试验性运行软件  
B 发现软件错误  
C .证明软件正确  
D 找出软件中全部错误
- 2 .DFD 中的每个加工至少有 ( )  
A .一个输入流和一个输出流  
B .一个输入流或者一个输出流  
C .一个输入流  
D .一个输出流
- 3 .一个局部数据存储当它作为( )时,就把它画出来。  
A .某些加工的数据接口  
B .某个加工的特定输入  
C .某个加工的特定输出  
D 某些加工的数据接口或某个加工的特定输入/ 输出
- 4 对于分层的 DFD,父图与子图的平衡指子图的输入、输出数据流同父图相应加工的输入、输出数

- 据流 ( )
- A .必须一致 B 数目必须相等  
C .名字必须相同 D 数目必须不等
- 5 .IDEF 图从各个侧面反映系统 ( )
- A .怎么做 B .做什么  
C .对谁做 D .何时做
- 6 软件产品的生产主要是 ( )
- A .生产 B .再生产  
C .开发 D .研制
- 7 .个体手工劳动是( )时代的生产方式。
- A .程序设计 B 软件  
C .程序系统 D 软件工程
- 8 软件工程是一门( )学科。
- A .理论性 B 原理性  
C .工程性 D 心理性
- 9 需求规格说明书的作用不应该包括 ( )
- A .软件设计的依据  
B .用户与开发人员对软件要做什么的共同理解  
C .软件验收的依据  
D 软件可行性研究的依据
- 10 .结构化设计方法在软件开发中用于 ( )
- A .概要设计 B .详细设计  
C .程序设计 D 测试用设计
- 11 .结构化分析方法使用的描述工具“( )”描述系统由哪几部分组成,各部分之间有什么联系等等。
- A .数据流图 B 数据字典  
C .判定表 D .判定树
- 12 .软件产品的生产主要是脑力劳动,软件产品的成本主要体现在软件的( )上。
- A .复制 B .开发方式  
C .开发和研制 D .磨损、消耗
- 13 .表示人工操作的系统流程图的符号是 ( )
- A .  B.  C.  D. 
- 14 .结构化分析方法使用的描述工具“( )”定义了数据流图中每一个图形元素。
- A .数据流图 B 数据字典  
C .判定表 D .判定树
- 15 .以下说法错误的是 ( )
- A .MTTF 是一个描述失效模型或一组换效特性的指标量  
B .MTBF 是指两次相继失效之间的平均时间  
C .MTBF 在实际使用时通常指当 n 很大时,系统第 n 次失效与第 n + 1 次失效之间的平均时间  
D 对于失效率为常数和修复时间很短的情况,MTTF 与 MTBF 差别很大



- E 软件的可靠性是能够定量计算的
- 16 .以下说法错误的是 ( )
- A .GB 指中华人民共和国国家军用标准
- B .ANSI 指美国国家标准协会
- C .BS 指英国国家标准
- D .DIN 指德国标准协会
- E JIS 指日本工业标准
- 17 .组成程序设计工作台的工具可能为 ( )
- |       |           |
|-------|-----------|
| 语言编译器 | 结构化编辑器    |
| 连接器   | 加载器       |
| 交叉引用  | 静态分析器     |
| 数据字典  | 报告定义和生成工具 |
| 代码生成器 |           |
- A . B .
- C . D .
- 18 .偶然内聚指 ( )
- A .一个模块内的各处理元素之间没有任何联系
- B .指模块内执行几个逻辑上相似的功能,通过参数确定该模块完成哪一个功能
- C .把需要同时执行的动作组合在一起形成的模块为时间内聚模块
- D .指模块内所有处理元素都在同一个数据结构上操作
- 19 .以下说法错误的是 ( )
- A .适用于实时处理的语言有:汇编语言、Ada 语言
- B .编写系统软件时,可选用汇编语言、C 语言、Pascal 语言和 Ada 语言
- C .如果要完成人工智能领域内的系统,应选择 Prolog、Lisp、C 语言和 Ada 语言
- D .适用于数据处理与数据库应用的语言有:Cobol、SQL、4GL 语言
- 20 .下面说法错误的是 ( )
- A .维护申请报告由申请维护的用户填写,软件维护组织内部还要制定一份软件修改报告
- B .软件修改报告指出的问题之一是:为满足软件问题报告实际要求的工作量
- C .软件修改报告指出的另外三个问题是:要求修改的性质、优先权和关于修改的事后数据
- D .维护申请报告也称软件问题报告,用作计划维护任务的基础
- E .提出维护申请报告之后,由用户和软件维护组来评审维护请求

#### 四、简答题( 每小题 5 分,共 20 分)

- 1 软件工程目标和内容是什么 ?
- 2 说明面向对象的特征。
- 3 衡量模块独立性的两个标准是什么 ? 它们各表示什么含义 ?
- 4 非渐增式测试与渐增式测试有什么区别 ?

#### 五、应用题( 共 25 分)

- 1 某工厂人事部门,对一部分职工重新分配工作,其分配原则如下:  
对这部分职工,如果年龄不满 20 岁,初中文化程度则脱产学习,高中文化程度,则当电工,大专文化

程度当技术员;如果年龄满 20 岁但不满 40 岁,初中或者高中文化程度,若是男性,则当钳工,若是女性,则当车工,大专文化程度则当技术员;如果年满 40 岁以上,初中或高中文化程度,当材料员,大专文化程度则当技术员。

请用判定表表达以上问题的加工逻辑。

2 .一个软件公司有许多部门,分为开发部门和管理部门两种。每个开发部门开发多个软件产品。每个部门由部门名字唯一确定。该公司有许多员工,员工分为经理、工作人员和开发人员。开发部门有经理和开发人员,管理部门有经理和工作人员。每个开发人员可参加多个开发项目,每个开发项目需要多个开发人员,开发人员使用语言开发项目。每位经理可主持多个开发项目。建立该软件公司的对象模型。

3 .一个正文文件由若干记录组成,每个记录是一个字符串。要求统计每个记录中空格字符的个数及文件中空格字符的总个数。要求输出数据格式是每复制一行字符串之后,另起一行印出上一行字符串空格字符的个数,最后一行印出空格字符总个数。

# 软件工程考前模拟试题(四)参考答案

## 一、名词解释题

- 1 模块化是指解决一个复杂问题时自顶向下逐层把软件系统划分成若干模块的过程。每个模块完成一个特定的子功能,所有的模块按某种方法组装起来,成为一个整体,完成整个系统所要求的功能。
- 2 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。
- 3 桩模块用来代替被测试模块所调用的模块。它的作用是返回被测试模块所需的信息。
- 4 软件开发中的原型是软件的一个早期可运行的版本,它反映了最终系统的重要特性。
- 5 用科学知识和技术原理来定义、开发、维护软件的一门学科。

## 二、填空题

- 1 最小 短 开发
- 2 简化 设计
- 3 较高 较抽象
- 4 技术 经济 社会
- 5 物理模型 元素 信息流动
- 6 数据流 数据项 数据存储 加工
- 7 判定表 判定数
- 8 功能 活动 输入 输出 控制 机制
- 9 ER
- 10 概要设计说明书

## 三、选择题

- |      |      |      |      |      |
|------|------|------|------|------|
| 1 B  | 2 A  | 3 D  | 4 A  | 5 B  |
| 6 D  | 7 A  | 8 C  | 9 D  | 10 A |
| 11 A | 12 C | 13 C | 14 B | 15 D |
| 16 A | 17 B | 18 A | 19 C | 20 E |

## 四、简答题

1 软件工程的目的是成功地建造一个大型软件系统,所谓成功是要达到以下几个目标:付出较低的开发成本;达到要求的软件功能;取得较好的软件性能;开发的软件易于移植;需要较低的维护费用;能按时完成开发任务,及时交付使用;开发的软件可靠性高。

软件工程研究的主要内容是软件开发技术和软件开发管理两个方面。在软件开发技术中,主要研究软件开发方法、软件开发过程、软件开发工具和环境。在软件开发管理中,主要是研究软件管理学、软件经济学、软件心理学等。

- 2 面向对象的特征:

(1)对象惟一性。每个对象都有自身惟一的标识,通过这种标识,可找到相应的对象。在对象的整个生命期中,它的标识都不改变,不同的对象不能有相同的标识。在对象建立时,由系统授予新对象以惟一的对象标识符,它在历史版本管理中有巨大的作用。

(2)分类性。分类性是指将具有一致的数据结构(属性)和行为(操作)的对象抽象成类。每个类是具有相同性质的个体对象的集合,而每个对象是相关类的实例。

(3)继承性。继承性是子类自动共享父类数据结构和方法的机制,这是类之间的一种关系。在定义和实现一个类的时候,可以在一个已经存在的类的基础之上来进行,把这个已经存在的类所定义的内容作为自己的内容,并加入若干新的内容。

继承性是面向对象程序设计语言不同于其他语言的最主要的特点,是其他语言所没有的。在类层次中,子类只继承一个父类的数据结构和方法,则称为单重继承。在类层次中,子类继承多个父类的数据结构和方法,则称为多重继承。

(4)多态性(多形性)。多态性是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。不同的对象,收到同一消息可以产生不同的结果,这种现象称为多态性。

多态性允许每个对象以适合自身的方式去响应共同的消息。这样就增强了操作的透明性、可理解性和可维护性。用户不必为相同的功能操作但作用于不同类型的对象而费心去识别。

3 衡量模块的独立性的标准是两个定性的度量标准:耦合性和内聚性。

(1)耦合性。也称块间联系。指软件系统结构中各模块间相互联系紧密程度的一种度量。模块之间联系越紧密,其耦合性就越强,模块的独立性则越差。模块间耦合高低取决于模块间接口的复杂性、调用的方式及传递的信息。

(2)内聚性。又称块内联系。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。若一个模块内各元素(语句之间、程序段之间)的联系越紧密,则它的内聚性就越高。

耦合性与内聚性是模块独立性的两个定性标准,将软件系统划分模块时,尽量做到高内聚低耦合,提高模块的独立性,为设计高质量的软件结构奠定基础。

4 集成测试的方法主要有两种:非渐增式测试和渐增式测试。

(1)非渐增式测试:首先对每个模块分别进行单元测试,然后再把所有的模块按设计要求组装在一起进行测试。

(2)渐增式测试:逐个把未经过测试的模块组装到已经过测试的模块上去,进行集成测试。每加入一个新模块进行一个集成测试,重复此过程直至程序组装完毕。

渐增式与非渐增式测试的方法有以下区别:

(1)非渐增式方法把单元测试和集成测试分成两个不同的阶段,前一阶段完成模块的单元测试,后一阶段完成集成测试。而渐增式测试往往把单元测试与集成测试合在一起,同时完成。

(2)非渐增式需要更多的工作量,因为每个模块都需要驱动模块和桩模块,而渐增式利用已测试过的模块作为驱动模块或桩模块,因此工作量较少。

(3)渐增式可以较早地发现接口之间的错误,非渐增式最后组装时才发现。

(4)渐增式有利于排错,发生错误往往和最近加进来的模块有关,而非渐增式发现接口推迟到最后,很难判断是哪一部分接口出错。

(5)渐增式比较彻底,已测试的模块和新的模块再测试。

(6)渐增式占用的时间较多,但非渐增式需更多的驱动模块。桩模块也占用一些时间。

(7)非渐增式开始可并行测试所有模块,能充分利用人力,对测试大型软件很有意义。

## 四、应用题

1 此问题的条件有三个:性别、年龄、文化程度,每个条件取值列表如下:

条件名称	取值	含义
性别	M	男
	F	女
年龄	L	未满 20 岁
	E	满 20 岁但不满 40 岁
	G	40 岁以上
文化程序	J	初中毕业
	S	高中毕业
	U	大专毕业

那么 , 所有条件的组合为  $2 \times 3 \times 3 = 18$  , 列出判定表如下表所示 :

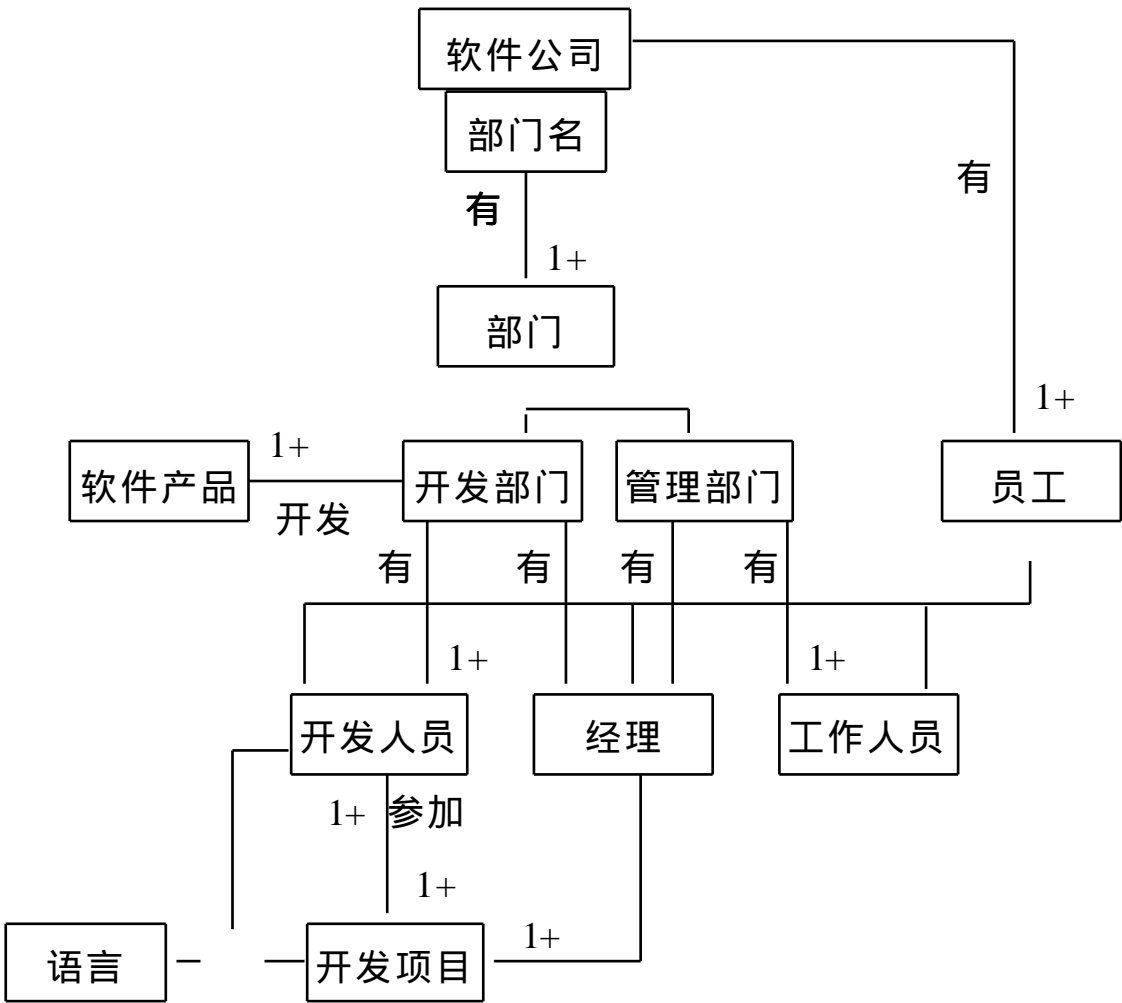
判定表																		
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
性别	M	M	M	M	M	M	M	M	M	F	F	F	F	F	F	F	F	F
年龄	L	L	L	E	E	E	G	G	G	L	L	L	E	E	E	G	G	G
文化程度	J	S	U	J	S	U	J	S	U	J	S	U	J	S	U	J	S	U
脱产学习																		
当电工																		
当钳工																		
当车工																		
当技术员																		
当材料员																		

通过合并 , 化简后的判定表如下表所示。

化简后的判定表

	1, 10	2, 11	3, 6, 9 12, 15, 18	4	5	7, 16	8, 17	13	14
性别	—	—	—	M	M	—	—	F	F
年龄	L	L	—	E	E	G	G	E	E
文化程度	J	S	U	J	S	J	S	J	S
脱产学习									
当电工									
当钳工									
当车工									
当技术员									
当材料员									

2 软件公司的对象模型如图所示。

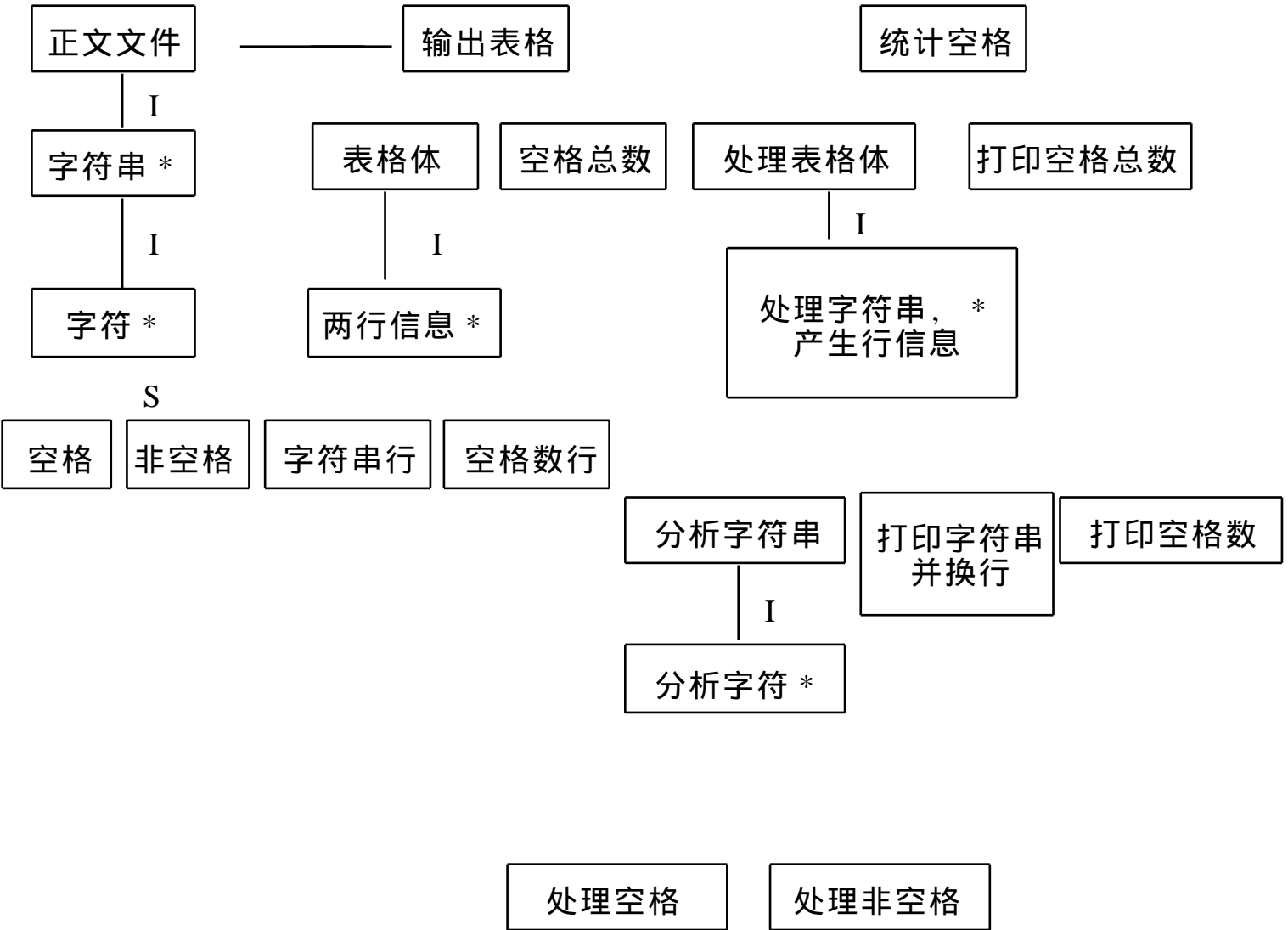


软件公司的对象模型

3 输入数据结构

输出数据结构

程序结构



# 软件工程考前模拟试题(五)

## 一、名词解释题(每小题 3 分,共 15 分)

- 1 继承性
- 2 数据流
- 3 基线
- 4 软件可靠性
- 5 软件工程过程

## 二、填空题(每小题 2 分,共 20 分)

- 1 纯收入是软件生存周期内\_\_\_\_\_与\_\_\_\_\_之差。
- 2 项目开发计划是一个\_\_\_\_\_文档。
- 3 程序设计时代的生产方式是\_\_\_\_\_,程序系统时代的生产方式是\_\_\_\_\_,软件工程时代的生产方式是\_\_\_\_\_。
- 4 软件工程是一门\_\_\_\_\_学科,计算机科学着重于\_\_\_\_\_,软件工程着重于\_\_\_\_\_。
- 5 软件开发划分的各阶段任务尽可能\_\_\_\_\_,同一阶段任务性质尽可能\_\_\_\_\_。
- 6 数据流图中,每个加工至少有\_\_\_\_\_个输入流和\_\_\_\_\_个输出流。
- 7 一个模块内部各程序段都在同一张表上操作,这个模块的内聚性称为\_\_\_\_\_。
- 8 结构化设计以\_\_\_\_\_为基础映射成软件结构。
- 9 结构化设计对数据流有两种分析方法,它们是\_\_\_\_\_设计和\_\_\_\_\_设计。
- 10 基于 IDEF0 图的设计也是结构化设计技术之一,它以系统的\_\_\_\_\_和信息结构为基础设计软件结构。

## 三、选择题(每小题 1 分,共 20 分)

- 1 软件结构使用的图形工具,一般采用( )图。  
A .DFD                      B .PAD                      C .SC                      D .ER
- 2 JSP 方法是一种面向\_\_\_\_\_的设计方法。  
A .对象                      B 数据流                      C .控制结构                      D .数据结构
- 3 程序设计语言的技术特性不应包括 ( )  
A .数据结构的描述性                      B 抽象类型的描述性  
C .数据库的易操作性                      D 软件的可移植性
- 4 软件测试中,白盒法是通过分析程序的( )来设计测试用的。  
A .应用范围                      B .内部逻辑                      C .功能                      D .输入数据
- 5 软件维护费用高的主要原因是 ( )  
A .人员少                      B 人员多                      C .生产率低                      D .生产率高
- 6 瀑布模型本质上是一种( )模型。  
A .线性顺序                      B 顺序迭代                      C .线性迭代                      D .及早见产品
- 7 软件质量必须在( )加以保证。





- 3 结构化程序设计基本要点是什么？
- 4 如何控制因修改而引起的副作用？

### 五、应用题(共 25 分)

1 某“调整工资”处理模块接受一个“职称”的变量,根据职称的不同(助教、讲师、副教授、教授)作不同的处理,其中若是助教还必须输入工龄,只有工龄超过两年才能调整工资。请用等价类划分法设计测试用例。

2 建立以下有关“微机”的对象模型。

一台微机有一个显示器,一个主机,一个键盘,一个鼠标,汉王笔可有可无。主机包括一个机箱,一个主板,一个电源及存储器等部件。存储器又分为固定存储器和活动存储器两种,固定存储器为内存和硬盘,活动存储器为软件盘和光盘。

3 画出下面用 PDL 写出的程序的 PAD 图。

```
WHILE      P      DO
  IF    A > 0    THEN    A1    ELSE    A2    ENDIF;
  S1;
  IF    B > 0    THEN    B1;
    WHILE    C    DO    S2;S3    ENDWHILE;
  ELSE    B2
  ENDIF;
  B3
ENDWHILE;
```

# 软件工程考前模拟试题(五)参考答案

## 一、名词解释题

- 1 继承性是子类自动共享父类数据结构和方法的机制,这是类之间的一种关系。
- 2 数据流是数据在系统内传播的路径,因此由一组成分固定的数据项组成。
- 3 基线是软件生存期中各开发阶段的一个特定点,它的作用是使开发阶段工作的划分更加明确化,使本来连续的工作在这些点上断开,以便于检查与肯定阶段成果。
- 4 软件按照设计要求,在规定时间内和条件下不出故障、持续运行的程度。
- 5 软件工程过程规定了获取、供应、开发、操作和维护软件时要实施的过程、活动和任务。

## 二、填空题

- 1 累计经济效益 投资
- 2 管理性
- 3 个体手工 作坊式小团体 工程化
- 4 综合性交叉 理论和原理 建造软件系统
- 5 相对独立 相同
- 6 1 1
- 7 通信内聚
- 8 数据流
- 9 变换分析 事务分析
- 10 功能模型

## 三、选择题

- |      |      |      |      |      |
|------|------|------|------|------|
| 1 C  | 2 D  | 3 D  | 4 B  | 5 C  |
| 6 A  | 7 D  | 8 A  | 9 A  | 10 B |
| 11 B | 12 D | 13 A | 14 C | 15 A |
| 16 D | 17 C | 18 A | 19 A | 20 B |

## 四、简答题

- 1 质量保证策略的发展大致可以分为以下三个阶段:
    - (1)以检测为重。产品制成后才进行检测,这种检测只能判断产品的质量,不能提高产品质量。
    - (2)以过程管理为重。把质量保证工作重点放在过程管理上,对制造过程的每一道工序都进行质量控制。
    - (3)以新产品开发为重。许多产品的质量问题的源于新产品的开发设计阶段,因此在产品开发设计阶段就应采取有力措施来消灭由于设计原因而产生的质量隐患。
- 由上可知,软件质量保证应从产品计划的设计开始,贯穿于投入使用和售后服务的软件生存期的每一阶段中的每一步骤。

2 .有许多软件需求分析与说明的方法(如结构化分析方法和面向对象分析方法),每一种分析方法都有独特的观点和表示法,但都适用下面的基本原则。

(1)必须能够表达和理解问题的数据域和功能域。数据域包括数据流(即数据通过一个系统时的变化方式)、数据内容和数据结构,而功能域反映上述三方面的控制信息。

(2)可以把一个复杂问题按功能进行分解并可逐层细化。通常软件要处理的问题如果太大太复杂就很难理解,划分成几部分,并确定各部分间的接口,就可完成整体功能。在需求分析过程中,软件领域中的数据、功能、行为都可以划分。

(3)建模。建立模型可以帮助分析人员更好地理解软件系统的信息、功能、行为,这些模型也是软件设计的基础。

3 结构化程序设计方法的基本要点有三点：

(1)采用自顶向下、逐步求精的程序设计方法。在需求分析、概要设计中,都采用了自顶向下、逐层细化的方法。在详细设计中,虽然处于“具体”设计阶段,但在设计某个模块内部处理过程中,仍可以逐步求精,降低处理细节的复杂程度。

(2)使用三种基本控制结构程序。任何程序都可由顺序、选择、重复三种基本控制结构构造。这三种基本结构的共同点是单入口、单出口,不但能有效的限制使用 GOTO 语句,还创立了一种新的程序设计思想、方法和风格,同时为自顶向下、逐步求精的设计方法提供了具体的实施手段。

(3)主程序员的组织形式。指开发程序的人员组织方式应采用由一个主程序员(负责全部技术活动)、一个后备程序员(协调、支持主程序员)和一个程序管理员(负责事务性工作,如收集、记录数据、文档资源管理等)三人为核心,再加上一些专家(如通信专家、数据库专家)和其他技术人员组成小组。这种组织形式突出了主程序员的领导,设计责任集中在少数人身上,有利于提高软件思想方法。

4 .为了控制因修改而引起的副作用,要做到：(1)按模块把修改分组；(2)自顶向下地安排被修改模块的顺序；(3)每次修改一个模块；(4)对每个修改了的模块,在安排修改下一个模块之前要确定这个修改的副作用,可使用交叉引用表、存储映像表、执行流程跟踪等。

五、应用题

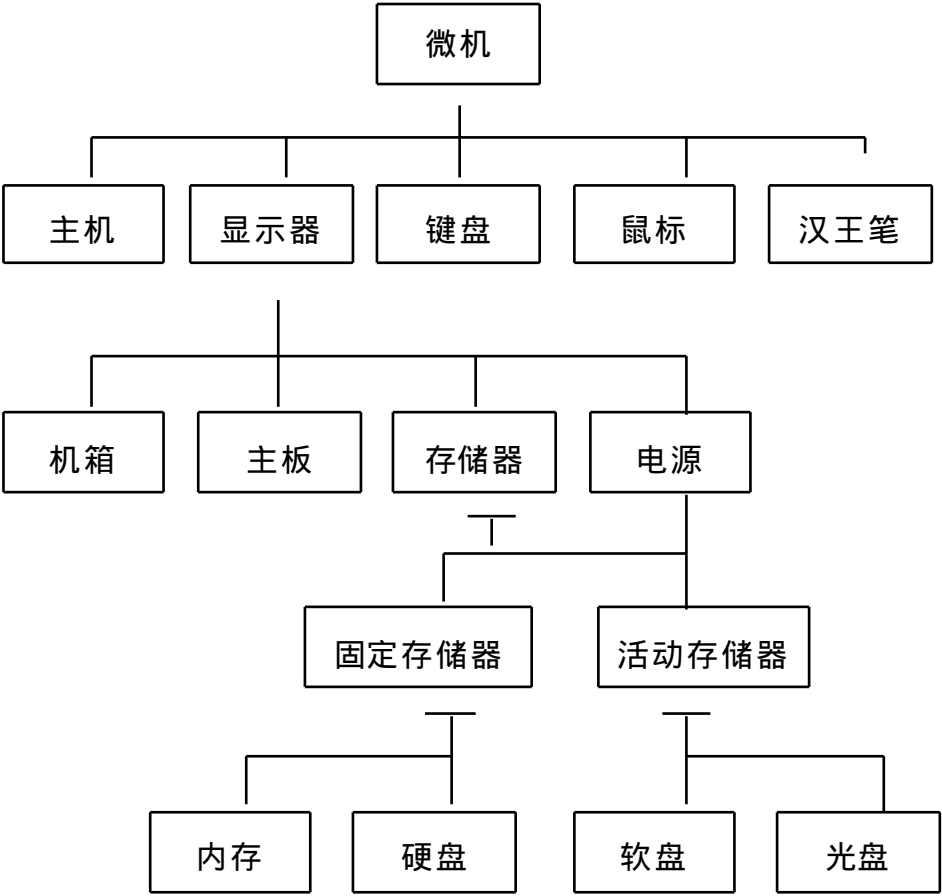
1 .划分等价类：

输入条件	合理等价类	不合理等价类
职称	教授 副教授 讲师	四种职称之外任意一种
职称兼工龄	助教兼工龄大于 2 年	助教兼工龄等于 2 年 助教兼工龄小于 2 年

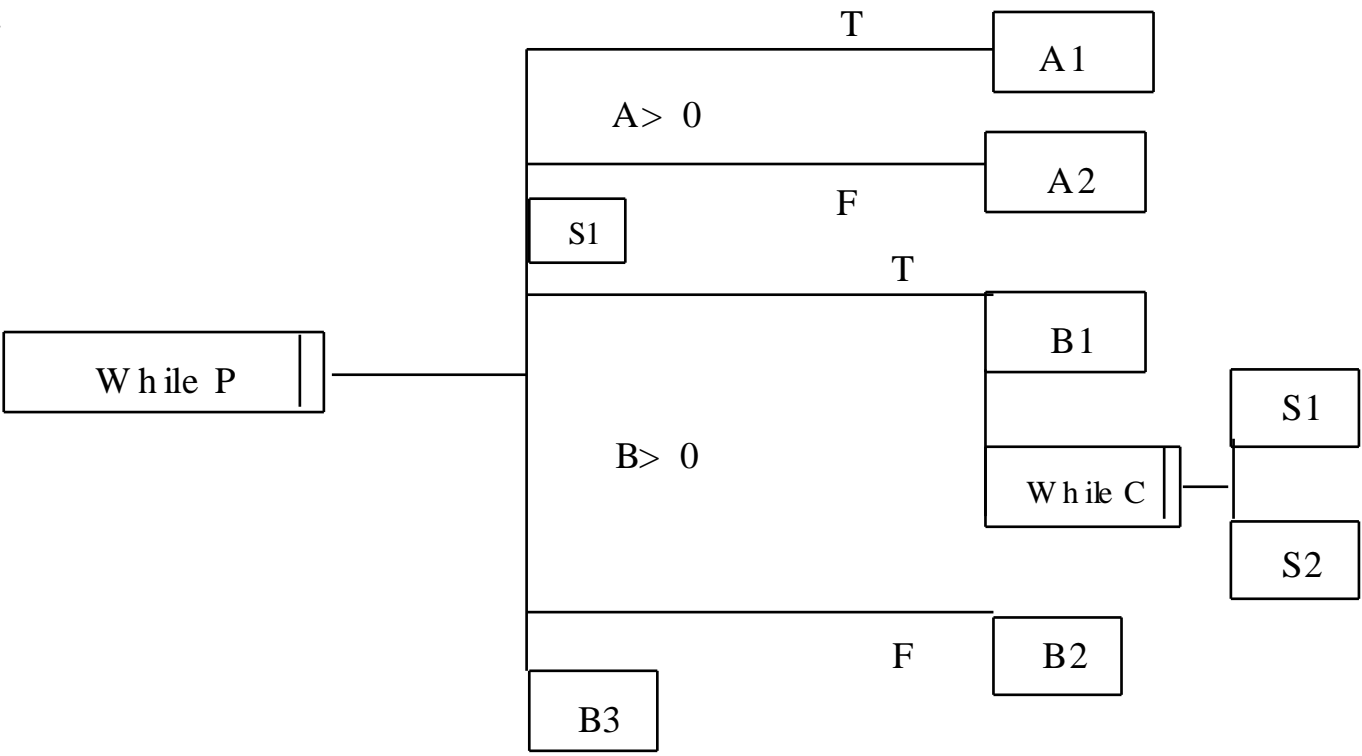
设计测试用例：

输入数据	预期结果	覆盖范围
教授	输入有效,进行调整工资处理	
副教授	输入有效,进行调整工资处理	
讲师	输入有效,进行调整工资处理	
助教 3	输入有效,进行调整工资处理	
助教 2	输入有效,不调整工资	
助教 1	输入有效,不调整工资	
工程师	输入无效	

2 .



3 .



## 2001 年 10 月全国高等教育自学考试

# 软件工程试题及参考答案

## 第一部分 选择题

一、单项选择题(每小题 1 分,共 20 分)在每小题的四个选项中只有一个选项是符合题目要求的,请将正确选项前的字母填在题后的括号内。

- 在下列工具与环境中的( )属于较早期的 CASE。  
A .基于信息工程 CASE  
B .人工智能 CASE  
C .结构的基于图形 CASE  
D .集成的 CASE 环境
- Putnam 成本估算模型是一个( )模型。  
A .静态单变量  
B .动态单变量  
C .静态多变量  
D .动态多变量
- 在 McCall 软件质量度量模型中,( )属于面向软件产品修改。  
A .可靠性  
B .可重用性  
C .适应性  
D .可移植性
- ISO 的软件质量评价模型由 3 层组成 ,其中用于评价设计质量的准则是 ( )  
A .SQIC  
B .SQMC  
C .SQRC  
D .SQDC
- 软件复杂性度量的参数包括 ( )  
A .效率  
B .规模  
C .完整性  
D .容错性
- 对象实现了数据和操作的结合 ,使数据和操作( )于对象的统一体中。  
A .结合  
B .隐藏  
C .封装  
D .抽象
- 软件调试技术包括 ( )  
A .边界值分析  
B .演绎法  
C .循环覆盖  
D .集成测试
- 瀑布模型的存在问题是 ( )  
A .用户容易参与开发  
B .缺乏灵活性  
C .用户与开发者易沟通  
D .适用可变需求
- 软件测试方法中的静态测试方法之一为 ( )  
A .计算机辅助静态分析  
B .黑盒法  
C .路径覆盖  
D .边界值分析
- 软件生命周期中所花费用最多的阶段是 ( )  
A .详细设计  
B .软件编码  
C .软件测试  
D .软件维护
- 第一个体现结构化编程思想的程序设计语言是 ( )  
A .FORTRAN 语言  
B .Pascal 语言  
C .PLI 语言  
D .Python 语言
- 程序的三种基本控制结构是 ( )  
A .过程、子程序和分程序  
B .顺序、选择和重复  
C .递归、堆栈和队列  
D .调用、返回和转移
- 在详细设计阶段 ,经常采用的工具有 ( )  
A .PAD  
B .SA  
C .SC  
D .DFD

- 14 .详细设计的结果基本决定了最终程序的 ( )  
A .代码的规模 B .运行速度 C .质量 D .可维护性
- 15 .需求分析中开发人员要从用户那里了解 ( )  
A .软件做什么 B .用户使用界面  
C .输入的信息 D .软件的规模
- 16 .结构化程序设计主要强调的是 ( )  
A .程序的规模 B .程序的效率  
C .程序设计语言的先进性 D .程序易读性
- 17 .IDEF。图反映系统 ( )  
A .怎么做 B .对谁做 C .何时做 D .做什么
- 18 .经济可行性研究的范围包括 ( )  
A .资源有效性 B .管理制度 C .效益分析 D .开发风险
- 19 .可行性分析是在系统开发的早期所做的一项重要的论证工作,它是决定该系统是否开发的决策依据,因必须给出( )的回答。  
A .确定 B .行或不行 C .正确 D .无二义
- 20 .需求分析阶段的任务是确定 ( )  
A .软件开发方法 B .软件开发工具  
C .软件开发费 D .软件系统的功能

## 第二部分 非选择题

### 二、填空题(每空 2 分,共 20 分)

- 21 .在软件开发过程中要产生大量的信息,要进行大量的修改,\_\_\_\_\_能协调软件开发,并使混乱减到最低程度。
- 22 .规定功能的软件,在一定程度上对自身错误的作用(软件错误)具有屏蔽能力,则称此软件具有\_\_\_\_\_的软件。
- 23 .McCall 提出的软件质量模型包括\_\_\_\_\_个软件质量特性。
- 24 .软件可维护性度量的七个质量特性是可理解性、可测试性、可修改性、可靠性、\_\_\_\_\_,可使用性和效率。
- 25 .为了便于对照检查,测试用例应由输入数据和预期的\_\_\_\_\_两部分组成。
- 26 .程序设计语言的心理特性主要表现在\_\_\_\_\_,简洁性、传统性、局部性和顺序性。
- 27 .软件结构是以\_\_\_\_\_为基础而组成的一种控制层次结构。
- 28 .在结构化分析中,用于描述加工逻辑的主要工具有三种,即:结构化语言、判定表、\_\_\_\_\_。
- 29 .结构化语言是介于自然语言和\_\_\_\_\_之间的一种半形式语言。
- 30 .若年利率为  $i$ ,不计复利, $n$  年后可得钱数为  $F$ ,则现在的价值  $P =$ \_\_\_\_\_。

### 三、名词解释题(每小题 3 分,共 15 分)

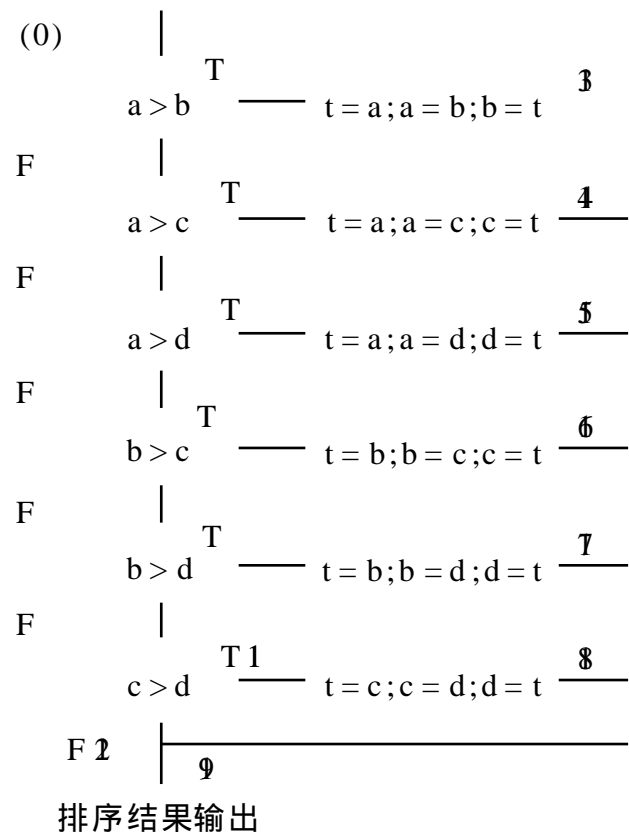
- 31 .软件生存周期模型
- 32 .数据字典(DD)
- 33 .内聚性
- 34 .JSP 方法

四、简答题( 每小题 5 分,共 20 分)

- 36 .简述容错技术的四种主要手段,并解释。
- 37 .以 G J .Myers 的观点,简述对软件测试的目的。
- 38 .就程序设计语言的工程特性而言,对程序编码有哪些要求 ?
- 39 .模块的内聚性包括哪些类型 ?

五、应用题( 第 40 小题 7 分,第 41 小题 8 分,第 42 小题 10 分,共 25 分)

40 .下面是某程序的流程图:



- (1) 计算它的环路复杂性。
- (2) 为了完成基本路径测试,求它的一组独立的路径。

41 .根据下列条件使用等价划分法设计测试用例。

某一 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。

42 .图书馆的预定图书子系统有如下功能:

- (1)由供书部门提供书目给订购组;
- (2)订书组从各单位取得要订的书目;
- (3)根据供书目录和订书书目产生订书文档留底;
- (4)将订书信息(包括数目,数量等)反馈给供书单位;
- (5)将未订书目通知订书者;
- (6)对于重复订购的书目由系统自动检查,并把结果反馈给订书者。

试根据要求画出该问题的数据流程图,并把其转换为软件结构图。



# 2001 年 10 月全国高等教育自学考试

## 软件工程试题参考答案

### 一、单项选择题(每小题 1 分,共 20 分)

- |       |       |       |       |       |
|-------|-------|-------|-------|-------|
| 1 .C  | 2 .D  | 3 .C  | 4 .D  | 5 .B  |
| 6 .C  | 7 .B  | 8 .B  | 9 .A  | 10 .D |
| 11 .B | 12 .B | 13 .A | 14 .C | 15 .A |
| 16 .D | 17 .D | 18 .C | 19 .B | 20 .D |

### 二、填空题(每空 2 分,共 20 分)

- |            |                    |
|------------|--------------------|
| 21 .软件配置管理 | 22 .容错功能           |
| 23 .11     | 24 .可移植性           |
| 25 .输出结果   | 26 .歧义性            |
| 27 .模块     | 28 .判定树            |
| 29 .形式语言   | 30 . $F/(1+(n*i))$ |

### 三、名词解释题(每小题 3 分,共 15 分)

31 .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 .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。)

### 四、简答题(每小题 5 分,共 20 分)

36 .结构冗余:包括静态冗余、动态冗余和混合冗余。

信息冗余:为检测或纠正信息在运算或传输中的错误,须外加一部分信息。

时间冗余:指重复执行指令或程序来消除瞬时错误带来的影响。

冗余附加技术:指为实现上述冗余技术所需的资源和技术。

37 .软件测试是(1)为了发现错误而执行程序的过程;(2)一个好的用例能够发现至今尚未发现的错误的测试。(3)一个成功的测试是发现至今尚未发现的错误的测试。

38 .就程序设计语言的工程特性而言,对程序编码有如下要求:

- |            |              |
|------------|--------------|
| (1)可移植性    | (2)开发工具的可利用性 |
| (3)软件的可重用性 | (4)可维护性      |

39 .模块的内聚性包括：

- (1)偶然内聚
- (2)逻辑内聚
- (3)时间内聚
- (4)通信内聚
- (5)顺序内聚
- (6)功能内聚

五、应用题(共 25 分)

40 .(1)环路复杂性 = 判断数 + 1 = 6 + 1 = 7(个)

(2)路径 1: (0) - - (13) - (19)

路径 2: (0) - - - (14) - (19)

路径 3: (0) - - - - (15) - (19)

路径 4: (0) - - - - - (16) - (19)

路径 5: (0) - - - - - (17) - (19)

路径 6: (0) - - - - - (18) - (19)

路径 7: (0) - - - - - (12) - (19)

41 .等价划分法

划分等价类并编号 ,如下表所示。

十六进制整型常量输入条件的等价类表

输入数据	合理等价类	不合理等价类
十六进制整数	1 0x 或 0X 开头 1 ~ 2 位数字串 2 以 - 0x 打头的 1 ~ 2 位数字串	3 .非 0x 或非 - 打头的串 4 .含有非数字且 (a,b,c,d,e,f) 以外字符 5 .多于 5 个字符 6 . - 后跟非 0 的多位串 7 . - 0 后跟数字串 8 . - 后多于 3 个数字
十六进制数范围	9 .在 - 7f ~ 7f 之间	10 小于 - 7f 11 大于 7f

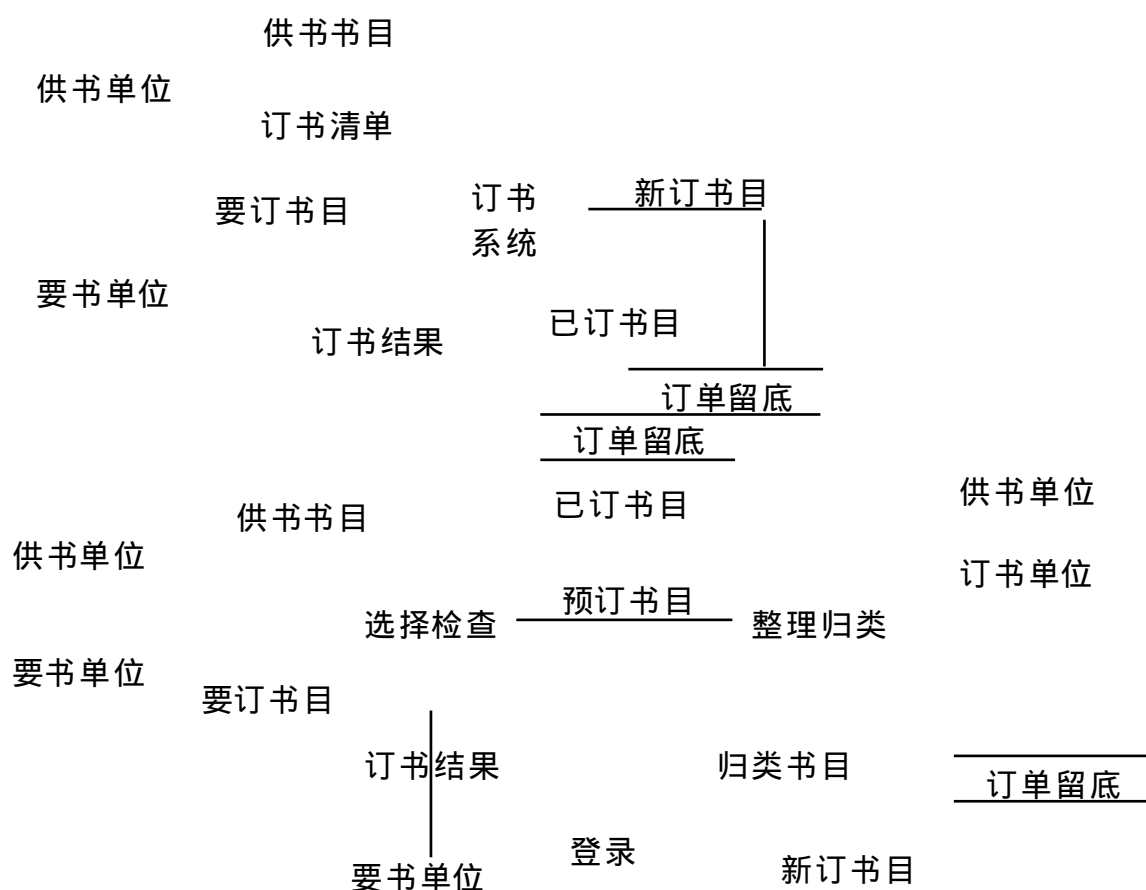
为合理等价类设计测试用例 ,表中有三个合理等价类 ,设计两个例子

测试数据	期望结果	覆盖范围
0 × 23	显示有效输入	1,9
- 0 × 15	显示有效输入	2,9

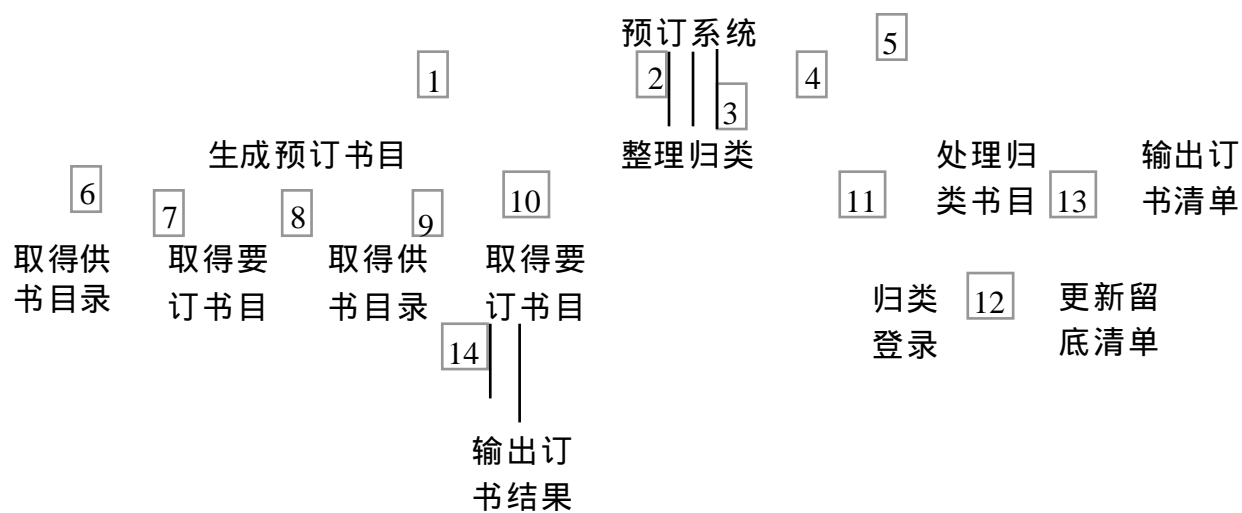
为每个不合理等价类至少设计一个测试用例

测试数据	期望结果	覆盖范围
2	显示无效输入	3
G12	显示无效输入	4
123311	显示无效输入	5
- 1012	显示无效输入	6
- 011	显示无效输入	7
- 0134	显示无效输入	8
- 0x777	显示无效输入	10
0x87	显示无效输入	11

#### 42 .(1)数据流图



## (2) 软件结构图



## 1,2—预订书目

## 5—订书清单

## 8—已订书目

12,13—更新书目

14—订书信息