**《软件开发环境与工具》--复习总结**

软件1319-张筵雍 著

**Chapter 1**

1. **软件概念：**

(1)应用软件(Application Software)：包括最终用户的计算机应用，例如文字处理软件、视频游戏和为特定用户群开发的ERP软件。

(2)中间件(Middleware)：用于控制和协调分布式系统。

(3)编程语言(Programming Languages)：定义计算机程序的语法和语义。

(4)系统软件(System Software)：包括操作系统，用于管理计算机资源。例如Windows操作系统是系统软件，负责管理计算机各类资源。

(5)测试件(Testware)：用于测试硬件和软件的套件。

(6)固件(Firmware)：是底层软件，通常存储于电子可编程存储器上。固件如同其名，被看做硬件被其他软件调用执行。

(7)设备驱动程序(Device Drivers)：使硬件之间能够方便地交换数据。建立软件与磁盘驱动器、打印机等硬件设备的数据通信界面。

(8)编程工具(Programming Tools)：用于上述软件的开发。

**2.软件的概述：**

软件是能够完成预定功能和性能的可执行的计算机程序和使程序正常运行所需要的数据，加上描述程序的操作和使用的文档(所有相关文档)。 软件=程序+数据+文档 或 软件=程序+过程+算法+文档。

1. **软件的分类：**
2. **基于软件功能的划分**

1)系统软件 2)编程软件 3)应用软件

1. **基于软件工作方式的划分**

1)实时处理软件 2)分时软件 3)交互式软件 4)批处理软件

**(3)基于软件运行环境划分**

1)单机运行软件 2)网络运行软件 3)嵌入式软件

1. **基于软件体系结构划分**

1)B/S结构 2)C/S结构 3)多层结构

1. **软件的特点：**

(1)软件是一种逻辑实体

(2)软件开发是人类智力的高度发挥

(3)软件的开发是一个复杂的过程

(4)软件维护与硬件维修有着本质的差别

(5)软件的成本非常高昂

(6)软件可以复制

**5.软件危机：**

(1)软件开发成本和进度失控： 高出一个数量级；延迟数月或数年

(2)用户对软件产品(软件系统)不满意：软件开发人员与用户沟通不够， 闭门造车。

(3)软件产品质量差：定量与定性标准不很完善。

(4)软件产品可维护性差：程序中的错误很难修改。

(5)软件缺少相应文档资料：程序与文档内容不一样，

(6)软件成本比例上升：硬件成本下降但软件成本上升

(7)软件开发生产率低：提高软件生产率是软件人员的追求

1. **软件架构：**

**6.1软件系统的架构包括：**

(1)构成软件的基本元素及其关系

(2)软件架构的文档。

**6.2软件架构概念**

软件架构是指在一定的设计原则基础上，从不同角度对组成系统的各部分进行搭配和安排，形成系统的多个结构而组成架构，它包括该系统的各个组件。

**6.3软件的多层架构：**

架构模式：C/S、B/S

软件结构多为三层结构:表示层 、应用层 、数据层

(1)表示层(Presentation Tier)：表示层位于应用的顶层，表示有关服务信息。通过与其他层通信，输出结果给浏览器/客户层和网络中的所有其他层。

(2)应用层(Application Tier)(业务逻辑、数据访问层或中间层)：逻辑层从表示层独立出来作为一层，通过执行具体的处理控制应用的功能。

(3)数据层(Data Tier)：该层是数据库服务，提供信息存储和检索功能。

**6.4软件设计分类：**

架构设计；详细设计；功能设计；初步设计。

**6.5分布式计算:**

P2P C/S B/S

**6.6 MVC——模型-视图-控制器**

**7.软件开发过程：**

**7.1软件生命周期：**

软件生命周期是指从系统设计对计算机系统提出应用要求开始，中间经过开发出一个满足需求的计算机系统，然后投入运行，直至该计算机中的软件系统退役为止。

**7.2软件开发过程：**

(1)问题的定义及规划

(2)需求分析

(3)软件设计

(4)编码和单元测试

(5)综合测试

(6)软件维护

**7.3开发模型：**

**(1)瀑布模型：**

模型中各阶段的活动从上一阶段向下一阶段逐级过渡，如同瀑布逐级下落，最终完成软件产品并交付用户使用。

**特点：**(1)阶段间具有顺序性和依赖性;(2)推迟实现的观点;(3)质量保证的观点

**缺点：**(1)由于开发阶段呈线性，当开发成果尚未经过测试阶段时，用户无法看到软件的运行效果，可能会存在需求方面的缺陷，造成开发出来的软件不是用户真正需要的软件，造成项目的返工或在维护中纠正需求的偏差。 (2)由于固定顺序，前期工作中造成的偏差如果延续到后期阶段，所造成的损失将会更大，项目的风险也就会越大。

**(2)快速原型模型(原型法)：**

该模型设计主要针对在用户需求阶段，难以给出准确、完整的用户需求的软件项目。目的是改进传统的结构化生命周期的不足，缩短开发周期，减少开发风险。

**优点：**(1)不带反馈环，软件产品的开发基本上是线性顺序进行的；(2)可以进行线性顺序开发。

**(3)增量模型(迭代增量开发)**

是利用迭代的思想，使软件开发活动逐步完善的机制。

**优点:**(1)在较短时间内向用户提交可完成一些有用的工作的产品；(2)逐步增加产品功能可以使用户有较充裕的时间学习和适应新产品，从而减少一个全新的软件可能给客户组织带来的冲击。

**缺点：**(1)在把每个新的增量构件集成到现有软件体系结构中时，必须不破坏原来已经开发出的产品；(2)必须把软件的体系结构设计得便于按这种方式进行扩充，向现有产品中加入新构件的过程必须简单、方便。也就是说，软件体系结构必须是开放的。

**(4)螺旋模型**

螺旋模型是在增量模型的基础上发展起来的。它在增量模型的基础上增加了风险分析机制。它将软件项目开发划分为制定计划、风险分析、实施开发以及客户评估四类活动。螺旋模型主要适用于大规模软件项目，特别是内部开发的大规模软件项目。

**优点:**(1)对可选方案和约束条件的强调有利于把软件质量作为软件开发的一个重要目标；(2)减少了过多测试（浪费资金）或测试不足（产品故障多）所带来的风险；(3)在螺旋模型中维护只是模型的另一个周期，在维护和开发之间并没有本质区别。

**缺点：**除非软件开发人员具有丰富的风险评估经验和这方面的专门知识，否则将出现真正的风险，当项目实际上正在走向灾难时，开发人员可能还认为一切正常。

1. **软件开发工具：**

(1)业务系统规划工具

(2)项目管理工具

(3)支持工具

(4)分析和设计工具

(5)编程工具

(6)测试与分析工具

(7)原型工具

(8)维护工具

1. **软件开发环境(SDE)：**

是指在基本硬件和宿主软件的基础之上，为支持系统软件和应用软件的工程化开发和维护而使用的一组软件。

**Chapter 2**

1. 传统的软件工程方法：

(1)结构化分析(Structrued Analysis,**SA**)

它是一种面向数据流，自顶向下、逐步求精进行需求分析的方法。它通常用数据流图表达需求,以数据字典表示数据的逻辑定义。

特点：(1)核心思想是自顶向下、逐步求精;(2)基本手段是分解和抽象。

(1.1)**数据流图：**数据流是数据在系统内的传输途径，数据流图从数据传递和加工的角度，以图形的方式刻画数据流从输入到输出的变换过程。

(1.2)数据流图的基本元素:数据流、加工、数据存取文件、输入数据的源点和输出数据的汇点



(1.3)分层数据流图:“先全局后局部，先整体后细节，先抽象后具体”。

(1.4)数据字典：

数据字典是各类数据描述的集合。它通过对数据项和数据结构的定义来描述数据流、数据存储的逻辑内容。通常包括数据项、数据结构、数据流、数据存储、处理过程和外部实体等6个部分。

(2)结构化设计(Structrued Design,**SD**)

通常把设计工作划分为概要设计和详细设计两个阶段。

**概要设计阶段的主要任务是**，通过仔细分析需求规格说明，对软件进行功能分解，从而把软件划分为模块，并且设计出完成预定功能的模块结构。**详细设计阶段的主要任务是**，详细地设计每个模块，确定完成每个模块功能所需要的算法和数据结构。

(2.1)结构化设计的主要原则:

1)抽象 2)模块化 3)信息隐蔽 4)模块独立性 高内聚低耦合

1. 详细设计工具：

图形工具：程序流程图，N-S（方框图），PAD（问题分析图）

表格工具：判定表；

语言工具：PDL（伪码）

1. 面向对象软件开发方法又称为OOSD。

OOSD包括面向对象分析（OOA）、面向对象设计（OOD）和面向对象程序设计（OOP）3个方面。

**Chapter 3**

1. **原型的定义：**

1)把系统主要功能和接口通过快速开发制作为“软件样机”，以可视化的形式展现给用户，及时征求用户意见，从而明确无误地确定用户需求。

1. 原型也可用于征求内部意见，作为分析和设计的接口之一，可方便于沟通。
2. **原型的主要价值：**
3. 原型法主要价值是可视化，强化沟通，降低风险，节省后期变更成本，提高项目成功率。
4. 对于较大型的软件来说，原型系统可以成为开发团队的蓝图。
5. 另外，原型通过与客户交流，还可以提高客户满意度。
6. **原型的基本要求：**
7. 体现产品主要的功能；
8. 提供基本的界面风格；
9. 展示比较模糊的部分，以便于确认或进一步明确，防患于未然；
10. 原型最好是可运行的，至少在各主要功能模块之间能够建立相互连接。
11. **原型的设计：**
12. 草图；
13. 低保真原型设计；
14. 高保真原型设计。
15. **低保真原型：**

初级原型指的是与最终产品不太相似的原型，根本无法使用，初级原型的优点是简单、便宜、易于制作，易于修改，这种灵活的原型非常适合我们探索、尝试、修改。低保真原型设计的特点是开发成本低、方便快捷、随时体现设计概念、能获得快速的反馈意见、同时可能会漏掉很多可用性方面的问题。

1. **高保真原型：**

高保真原型设计的特点就是包含了明确的任务操作流程与完整的功能，与视觉效果。通过低保真原型设计是为了更好，更全面的展示高保真原型设计！

1. **AxureRP-Pro原型设计工具：**

是美国 Axure Software Solution公司的旗舰产品， 是一个快速的原型工具， 主要是针对负责定义需求、 定义规格、 设计功能、 设计界面的专家， 包括用户体验设计师、 交互设计师、 业务分析师、 信息架构师、 可用性专家和产品经理。用户能快速上手，并且提供了丰富的组件样式修改，使得通过其能够创建低保真、高保真甚至接近于实际效果的界面。用来做产品原型的设计工具，主要在产品策划阶段使用，设计出来的原型仅仅代表会有什么内容、功能和大概做成什么样子，设计出来的东西主要是看的，不能拿来用的。

1. **常用工具：**
2. 主菜单和工具栏(Main Menu & Toolbar)执行常用操作，如文件打开、保存、格式化控件、输出原型、输出规格等操作。
3. 页面导航板(Sitemap Pane)对所设计的页面进行添加、删除、重命名和组织。
4. 控制面板(Widgets Pane)该面板中有线框图控件和流程图控件，用这些控件进行线框图和流程图的设计。
5. 模块面板(Masters Pane)模块是一种可以复用的特殊页面，在该面板中可进行模块的添加、删除、重命名和组织。
6. 线框图面板(Wireframe Pane)在线框图面板中可以进行页面线框图的设计，线框图面板也就是进行页面设计的工作区。
7. 控件交互面板(Interactions Pane)定义控件的交互，如：连接、弹出、动态显示和隐藏等。
8. 控件注释面板(Annotations Pane)对控件进行注释定义和控件的功能进行说明。
9. 页面交互和注释面板(Pages Notes & Page Interactions Pane)添加和管理页面的注释和交互。
10. **Axure RP支持的事件：**

OnClick：鼠标点击

OnMouseEnter：鼠标的指针移动到对象上 

OnMouseOut：鼠标的指针移动出对象外 

OnFocus：鼠标的指针进入文字输入状态（获得焦点） 

OnLostFocus：鼠标的指针离开文字输入状态 （失去焦点） 

OnPageLoad：页面或模块载入

大多对象只具备常见的三种触发事件：OnClick、OnMouseEnter 与 OnMouseOut。

一些特殊的控件可触发的事件有些不同： 

按钮控件只有 OnClick 

单选框和复选框则具有 OnFocus、 OnLostFocus

文本框、文本域、下拉框、列表框则具有 OnKeyUp、OnFocus、OnLostFocus 

页面加载或模块被载入时则发生 OnPageLoad

1. Axure 支持的动作：

Open Link in Current Window：在当前窗口打开一个页面 

Open Link in Popup Window：在弹出的窗口中打开一个页面 

Open Link in Parent Window：在父窗口中打开一个页面

Close Current Window：关闭当前窗口 

Open Link in Frame：在框架中打开一个页面 

Set Panel state(s) to State(s)：为动态面板设定要显示的状态

Show Panel(s)：显示动态面板 

Hide Panel(s)：隐藏动态面板 

Toggle Visibility for Panel(s)：切换动态面板的显示状态（显示/隐藏） 

Move Panel(s)：根据绝对坐标或相对坐标来移动动态面板

Set Variable and Widget value(s) equal to Value(s)：设定变量值或控件值 

Open Link in Parent Frame：在父页面的嵌框架中打开一个页面

Scroll to Image Map Region：滚动页面到

Image Map 所在位置

Enable Widget(s)：把对象状态变成可用状态 

Disable Widget(s)：把对象状态变成不可用状态 

Wait Time(s)：等待多少毫秒(ms)后再进行这个动作 

Other：显示动作的文字说明

**Chapter 4**

1. **数据库管理系统：**

数据库管理系统(Database Management System，

DBMS)是一种操纵和管理数据库的大型软件，用于建立、使用

和维护数据库。

**1.1 Oracle数据库：**

Oracle数据库是大型的关系型数据库管理系统，应用于

包括管理信息系统、企业数据处理、电子商务等广泛的领域。

特点：(1)可移植性 (2)集群服务 (3)数据挖掘 (4)空间数据库 (5)开发工具

**1.2 SQL Server数据库管理系统：**

SQL Server是微软公司开发的一个关系型数据库管理系统。

**1.3 Sybase数据库管理系统:**

自适应服务器企业版;优势数据库服务。

**1.3 DB2数据库管理系统：**

DB2数据库是IBM公司开发的一种大型、分布式关系型数据库管理系统，同时也是一种分布式数据库解决方案。

**1.4 MySQL数据库管理系统:**

MySQL是一个小型关系型数据库管理系统，开发者为瑞典MySQL AB公司。

**2.数据库工具主要功能：**

创建概念数据模型功能

创建物理数据模型功能

数据库连接

数据迁移与备份恢复

浏览数据库对象

数据库的库表操作

用户管理

检查设计与逆向工程

性能监视

调试存储过程

**Chapter 5**

1. **Visual C# 代码编辑窗口：**

**1.IntelliSense(智能感知)**

在编辑器中输入源代码时，IntelliSense 将显示一个包含所有 C# 关键字和 .NET Framework 类的列表框

**2.重构**

随着基本代码在开发项目过程中的不断增大和演变，有时需要更改代码，使其更具有可读性或可移植性。

**3.代码段**

是常用的 C# 源代码的小单元。在代码编辑器中单击鼠标右键，就能访问代码段菜单。通常，读者可以在 Visual C# 提供的许多代码段中浏览，也可以创建自己的代码段。

**4.波浪下画线**

红色波浪下画线标识语法错误或语义错误。

绿色波浪下画线标识潜在的编译器警告。

蓝色波浪下画线标识编译器错误。

**5.可读性帮助**

包括设置代码格式、大纲显示和着色三种类型。

**需要记住的例题：**

P92 例题 5-2；P93 例题 5-3。

**Chapter 6**

**1.Visual C#的标识符必须遵循下列语法规则：**

(1) 只能使用字母、数字和下画线组成。

(2) 必须以字母、下画线或@开始。

(3) Visual C#的标识符是大小写敏感的。

(4) 标识符不能使用C#中预定义的关键字名，但以@符号开头的标识符，允许使用关键字作为标识符。

(5) 标识符不可与Visual C#中的类库名相同。

一般情况下，变量名首字母小写，后面各单词首字母大写；而常量名、类名、方法名、属性名等首字母大写。

1. **命名空间：**

命名空间是类、函数或组件的容器，把它们按类别放入不同的命名空间中。命名空间只是一种逻辑上的划分，而不是物理上的存储分类。命名空间可以避免命名冲突。

注意：

1. 用关键字namespace声
2. 明一个命名空间时，命名空间的声明要么是源文件using语句后的第一条语句，要么作为成员出现在其他命名空间的声明之中。
3. 在同一命名空间中，不允许出现同名命名空间成员或同名的类。
4. 如果在应用程序中没有显式声明命名空间，应用程序中会创建默认命名空间。该默认的命名空间，其名称就是项目的名称。
5. **命名空间的使用：**
6. 直接定位命名空间的位置；(**在该方法中任何一个命名空间都可以在程序代码中直接使用。**)
7. 使用using关键字;(**使用using关键字，能够引用给定的命名空间或创建命名空间的别名**)
8. **命名空间类型：**

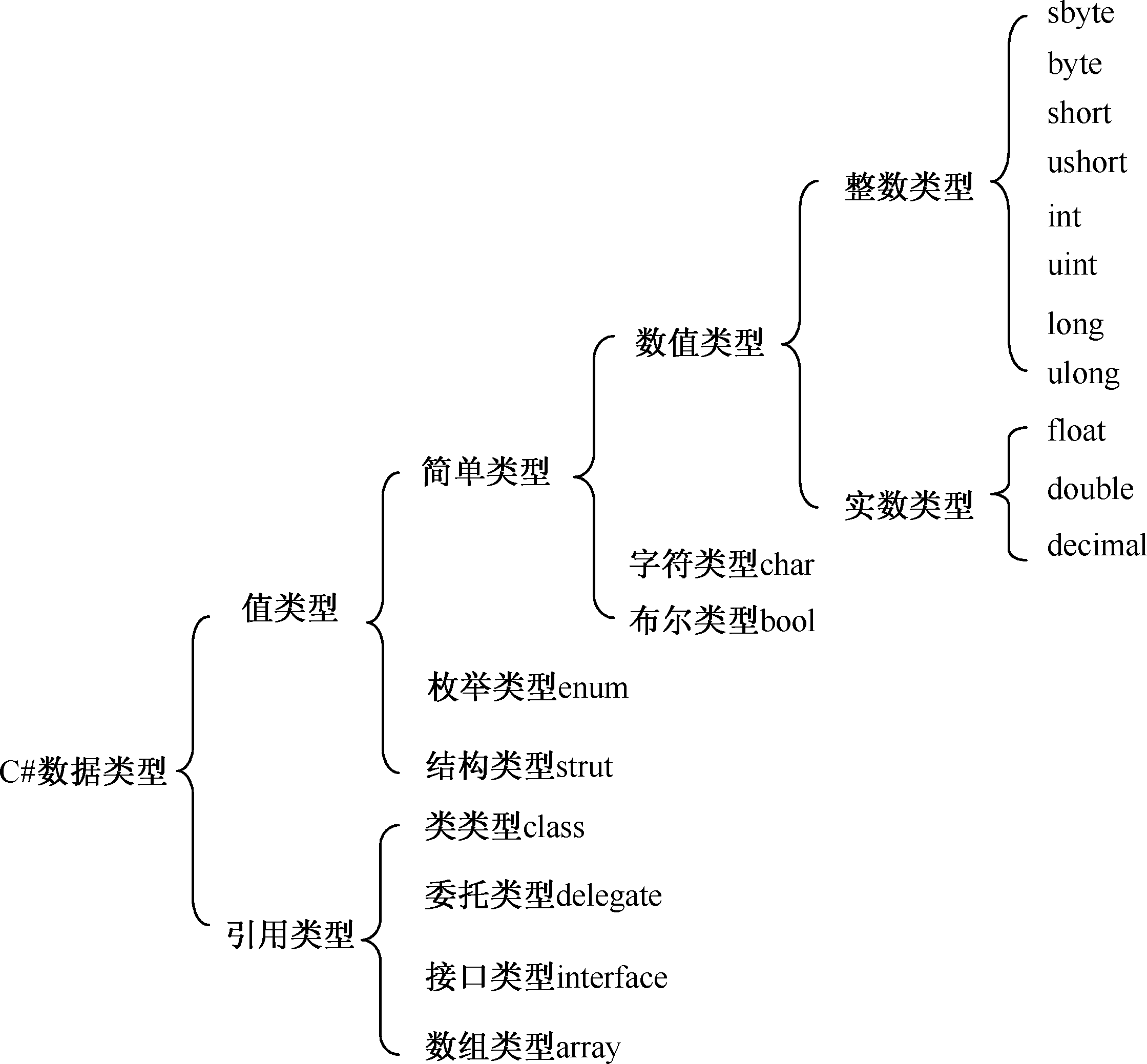
|  |  |
| --- | --- |
| 命 名 空 间 | 类 的 描 述 |
| System.Collection | 提供与集合相关的类型，如列表、队列、位数组、哈希表和字典等 |
| System.Text | 提供了操作字体集合的功能 |
| System.IO | 提供允许读写文件和数据流的接口 |
| System.Threading | 提供启用多线程的类和接口 |
| System.Drawing | 提供了对 GDI+ 基本图形功能的访问 |
| System.Data | 包含了数据访问使用的一些主要类型 |
| System.XML | 为处理 XML 提供基于标准的支持 |
| System.Net | 可为当前网络上的多种协议提供简单的编程接口 |
| System.Net.Sockets | 提供对Windows套接字的访问 |
| System.Web | 提供进行浏览器与服务器通信的类和接口 |
| System.Timers | 提供Timer组件，在指定的时间间隔引发一个事件 |
| System.Windows.Forms | 创建使用标准Windows图形接口和基于Windows的应用程序 |

1. **数据类型:**

Visual C#支持两种基本数据类型：值类型和引用类型。

值类型与引用类型的区别：

值类型变量存储的是该数据类型所代表的实际数据，变量包含其数据，可以直接访问其值，并且该变量存储在内存的栈(Stack)中；

引用类型的变量存储对数据的引用，即地址。对于引用类型，两个变量可能引用同一个对象，因此对一个变量的操作可能影响另一个变量所引用的对象。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 数 据 类 型 | 类 型 说 明 | 字  节  数 | 取 值 范 围 | 取 值 精 度 | 示    例 |
| float | 单精度浮点型 | 4 | 1.5e−45到3.4e38 | 7位 | float va=12.9F; |
| double | 双精度浮点型 | 8 | 5.0e−324到1.7e308 | 15～16位 | double va=23.45D; |
| decimal | 十进制数类型 | 16 | 1.0e-28到7.9e28 | 28～29位 | decimal va=3.33M; |

(1)布尔类型数据主要应用在流程控制中，用于判断布尔类型变量的“真”、”假“来改变流程的执行方向。

(2)枚举类型是值类型的一种特殊形式，即由一组称为枚举数列表的命名常数组成的独特类型。使用enum 关键字来声明枚举类型。

(3)Visual C#中引用类型主要包括以下几种：

 类(class)

 接口(interface)

 委托(delegate)

 数组(array)

(4)C#语言中，无论如何定义引用类型变量，都不会变为值类型变量。

(5)对象类(object类)

C#中的所有类型，包括所有值类型和引用类型，都以object类为基类，直接或间接地从object类继承而来。

(6)可以将数组分为如下3类：

 一维数组

 多维数组

 不规则数组

I.一维数组

①定义

定义一维数组的语法格式：

数组类型[] 数组名；

int[] a;

double[] b;

数组必须初始化后才能使用。

②初始化

两种方法：动态、静态。

动态初始化一维数组的格式：int[] a=new int[2]{1,2};

静态初始化一维数组的格式： int[] a={1,2};静态初始化必须和定义放在一起。

II.多维数组：定义：

string[,] str2; //引用变量str2,二维数组

Str2=new string[3,4];

Str2[1,2]=“efg”;

1. 字符串：
2. 字符串的定义：string 字符串名;将赋予的值用引号括起来；string str = “我是中国人”;
3. 查找字符串

string str="中华人民共和国";int i=str.IndexOf("民");

3)比较字符串

string str1="CDE";

string str2="cde";

int i=string.Compare(str1,str2);

4)判断是否为空字符串

string str="";string s1="不空";if(str.Length==0)

s1="空";

5)字符串大小写转换

string str="CcDdEf";

string str1=str.ToLower(); //把字符转换为小写， str值不变。

string str2=str.ToUpper(); //把字符转换为大写，str值不变。

(8)C#语言中，提供2种数据的类型转换方式：隐式转换和显式转换。

1)隐式转换

隐式转换是系统默认的，编译器不需要对转换进行详细的检查，就能安全地执行转换，该转换过程不需加以声明。在隐式转换过程中，转换一般不会失败，也不会导致信息丢失。(其实就等价于java中的非强制转化)

2)显式转换

又称强制类型转换。当明确要求编译器把数值从一种类型转换成另一种类型时，就需要明确地指定转换类型。显式转换可能导致信息丢失。(其实就等价于java中的强制转化)

(转换的类型标识符) 表达式；将“表达式”的值转换成“转换的类型标识符”的值。

(9)常量

1) 直接常量

2) 符号常量

符号常量声明的格式如下：const 类型名 常量名=常量表达式;

**6.常用类**：

(1)DateTime类 表示时间上的某一刻，通常以日期和当天的时间表示。DateTime类中的方法，可以通过下列格式进行调用：

DateTime.方法名(参数);

1. Array 类 提供创建、操作、搜索和排序数组的方法，在公共语言运行库中用做所有数组的基类。Array类中的方法，可以通过下列格式进行调用: Array.方法名(参数);
2. String 类 专门用于存储字符串，可以用它实现字符串运算功能。在String类运算中可以使用“+”运算符重载来连接两个字符串。String类中的方法，可以通过下列格式进行调用:String.方法名(参数);
3. Math类 可以用它实现常用算术运算功能。Math类中的方法，可以通过下列格式进行调用:Math.方法名(参数);
4. Convert类 可以用它将一个基本数据类型转换为另一个基本数据类型。Convert.方法名(参数);

**7.表达式和运算符：**

表达式是指定计算的操作符、操作序列，类似数学运算中的表达式。它由运算符、运算数和括号组成，可以进行嵌套。

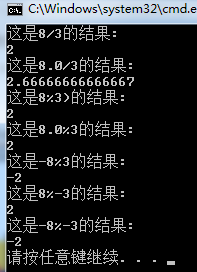
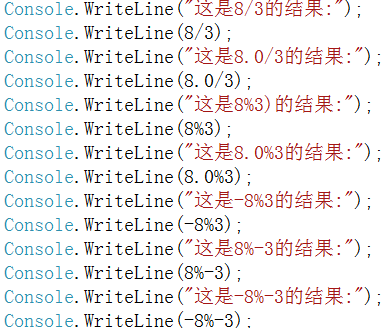
C#运算符一般分为以下3种类型：

一元运算符，即单目运算符，参与运算的操作数个数1个。

二元运算符，即双目运算符，参与运算的操作数个数2个。

三元运算符，即三目运算符，参与运算的操作数个数3个。

注意：在两个整数进行除法运算时，其运算结果也是整数，余数部分被舍去。“/”用来求除法的商，而“%”用来求除法的余数。C#中“%”运算只适应于整数数据，余数的符号与左操作数的符号相同。例如：10%3=1、-8%3= -2 。



注意：

(1)关系表达式的值只有2个，true和false。。

(2) 在使用关系运算符比较相等和不相等关系时，应区分比较的是值还是引用。对于基本数据类型的变量，关系运算符判断的是值；而对于对象数据类型变量，比较的是引用。

(3) 当比较字符串时，比较的是字符串的值，而非字符串的引用。

(4) 关系运算符是比较相同类型的数据。但要对不同类型的数据进行比较，可能会产生错误。因此，通常先把不同类型的数据转换成同一类型，再进行比较。

C#除了提供以上基本运算符之外，还包括一些特殊的运算符号。

|  |  |  |  |
| --- | --- | --- | --- |
| 运算符 | 说    明 | 运 算分 量 | 示    例 |
| ?： | 条件运算 | 三目 | x>y?x：y |
| typeof | 获得对象的类型 | 单目 | typeof(x) |
| sizeof | 获得指定类型在内存中所占字节数 | 单目 | sizeof(char) |
| is | 检查表达式是否是指定的类型 | 双目 | x is int |
| new | 创建一个新的类型实例 | 单目 | class X {}; X x =new X; |

重点PPT例题：(老师说，考试套出这上面的)

chapter 6

例题 6-2；

例题 6-3；例题 6-6；

例题 6-13；

例题 6-14；

例题 6-15；

例题 6-16；

例题 6-17；

例题 6-18；

例题 6-19；

例题 6-20；

例题 6-21；

例题 6-22；

例题6-25；

**应用举例：1. 2. 5.**

**Chapter 7**

1. **窗体设计：**

在C#中，窗体可分为普通窗体(SDI窗体)和多文档界面窗体(MDI窗体)两种类型。普通窗体又分为模式窗体和非模式窗体。

Windows窗体的属性设置分为静态设置和动态设置两种。

静态设置是在程序设计时进行的属性设置，其方法为右键单击窗体，在出现的弹出式菜单中选择“属性”菜单项，在其中找到要修改的属性栏目，修改其值。

动态设置是通过程序代码在程序运行时修改属性的值。

1. **菜单设计：**

菜单按使用形式的不同可以分为下拉式菜单(MenuStrip)和弹出式菜单(ContextMenuStrip)两种。

1. **组合框控件：**

组合框(ComboBox)控件兼有TextBox控件和ListBox控件两者的功能，用户可以通过输入文本或选择列表中的项目来进行输入**。**

1. **数据访问ADO.NET：**

ADO.NET是与数据源交互的.NET技术, 起源于ADO(ActiveX Data Objects) 。

优点：(1)互用性 (2)易维护性 (3)高性能 (4)可编程性 (5)可扩展性

**4.1 ADO.NET体系结构:**

ADO.NET模型分为数据提供程序(.NET Data Provider)和数据处理程序(DataSet数据集)两个部分。

重点记**四个对象：**

**Connection： 提供与数据源的连接功能。**

**Command： 提供运行访问数据库命令、传送数据或修改**

**数据的功能，可以运行SQL命令和存储过程**

**等。**

**DataReader： 通过Command对象运行SQL查询命令取得数**

**据流，可进行高速、只读的数据浏览。**

**DataAdapter：建立DataSet对象和数据源间的桥梁。**

**DataAdapter：使用4个Command对象来运行查询、新建、修**

**改、删除的SQL命令，把数据加载到DataSet，**

**或者把DataSet内的数据送回数据源。**

**5.几种常用的.NET Data Provider：**

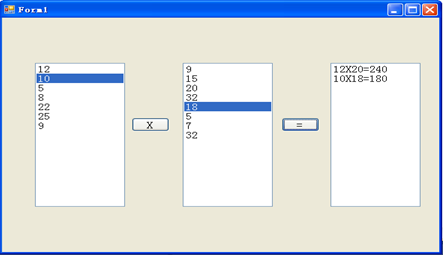
(1) SQL .NET Data Provider

(2) OLEDB .NET Data Provider

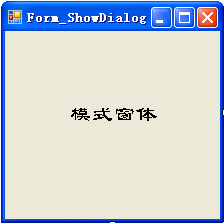
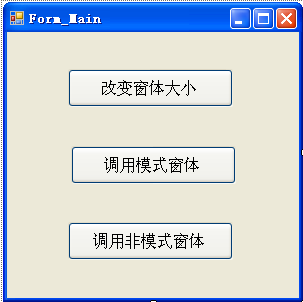
(3) ODBC .NET Data Provider

(4) ORACLE .NET Data Provider

**看看PPT上的例题**



IMG_256



**第七章的作业好好看看**