卓越工程师计划：软件工程专业系列丛书

阶梯式 GIS

软件工程实践系列教程

叶亚琴 胡茂盛 周顺平 编著

基础篇

科学出版社

卓越工程师计划：软件工程专业系列丛书

阶梯式 GIS 软件工程实践系列教程——基础篇

叶亚琴　胡茂胜　周顺平　编著

科学出版社

北京

内 容 简 介

本书是“阶梯式 GIS 软件工程实践系列教程”的基础篇。本书之所以命名为基础篇，一是本书主要面向对 C++语言和 Visual Studio 工具还不熟练的编程初学者；二是在内容方面本书重点在掌握 C++语言和 VC 工具、理解图形的基本概念和掌握绘制方法，以及点线区图形结构的定义和文件存储；三是在技能训练方面强调程序调试技巧和编程规范化，这是优秀程序员的基本功训练。

本书围绕实现一个小型点、线、区图形编辑系统，按照实习目的及要求、背景知识概述、基础编程练习和挑战编程练习四个部分进行编排。

其中基础编程练习将点、线、区图形的编辑、存储、查询和显示功能做了细致地分解，各次练习环环相扣循序渐进，且每次练习都有具体的内容、目标和上机指南。练习的内容在语言、工具和调试方法等方面有一定的重叠度，但在知识运用上越来越深，在功能实现上越来越难，目的是在逐步实现各项功能的过程中基本功得到强化训练。而上机指南则随着练习的深入越来越简略，目的是锻炼读者根据要求实现相应功能的思考能力。

本书将对初学者有一定难度的功能编排在“挑战编程练习”这一章，只提出练习内容、目标和要求，并简要说明实现过程，读者根据要求和说明思考如何一步步编程实现。更多复杂的图形编辑功能如连接、旋转、镜像等由于考虑本书篇幅和实践时间等因素，未列其中，感兴趣的读者可在本书的基础上进一步实践。

本书可作为大专院校 C++课程的实习参考书，特别是作为大一到大二期间的综合实践用书，也可供从事 GIS 基础软件开发人员参考。

图书在版编目(CIP)数据

阶梯式 GIS 软件工程实践系列教程. 基础篇/叶亚琴，胡茂胜，周顺平编著. —北京：科学出版社，2015.9

（卓越工程师计划 ：软件工程专业系列丛书）

ISBN 978-7-03-045721-9

Ⅰ. ①阶… Ⅱ. ①叶… ②胡… ③周… Ⅲ. ①地理信息系统—教材 Ⅳ. ①P208

中国版本图书馆 CIP 数据核字(2015)第 222206 号

责任编辑：张颖兵 闫 陶/责任校对：肖 婷

责任印制：高 嵘/封面设计：陈明亮

科 学 出 版 社 出版

北京东黄城根北街 16 号

邮政编码：100717

http://www.sciencep.com

武汉市首壹印务有限公司印刷

科学出版社发行 各地新华书店经销

*

开本：787×1092 1/16

2015 年 9 月第 一 版 印张：14 1/4

2015 年 9 月第一次印刷 字数：322 000

定价：32.00 元

（如有印装质量问题，我社负责调换）

卓越工程师计划：软件工程专业系列丛书

编委会

前　言

地理信息系统(Geographic Information System,GIS)是在地理学、测绘科学和计算机科学等学科基础上发展起来的交叉性新兴学科,是研究地理信息采集、存储、管理、分析和应用的技术和工具。随着GIS技术的进步,GIS在测绘、地理、地质、环保、国土、城市、农业、军事等领域得到了越来越广泛的应用,并渐渐发展成为一个产业。

GIS本质上是利用计算机技术特别是软件技术,对地理信息进行加工并服务于各行各业的一种信息技术。伴随着网络技术和移动技术的推广普及,地理信息的应用范围不断扩大,已从初期的科研和政府部门,扩大到百姓的日常生活和各种商业领域,并从室内延伸到室外;应用深度也不断细化,从地图制作和应用发展到地理信息服务,应用渗透到定位导航、生产调度、国情监测、灾害监测和其他各行各业,并涌现出了寻人、定向广告等各种个性化的应用。

随着应用领域不断扩大,各种新型的GIS应用层出不穷,推动了GIS产业的快速发展,使GIS成为一个非常具有活力和发展前景的新兴领域,同时也带来了对GIS应用人才和软件开发人才的强烈需求。

为顺应学科和产业发展的需要,自20世纪90年代以来,国内越来越多的高校开办了GIS本科专业。截至2010年,开办GIS专业的高等院校已超过200所。

各校的GIS本科专业大多建立在地学相关专业或计算机专业基础上,培养目标也大致分为两类:面向生产业务和面向软件开发。其中面向生产业务的培养目标,主要是培养能够运用GIS知识和工具开展数据处理、制图和行业应用的人才;而面向软件开发的培养目标,主要是培养具有GIS知识,能够开展GIS相关软件设计和实现的软件工程师。

各高校GIS专业在培养GIS软件开发人才的过程中,利用各自的领域优势,制定了各具特色的人才培养方案。为国土、地矿、测绘、规划、交通、物流、农业、电信等领域输送了大批具有相应背景知识的软件开发人才。

中国地质大学(武汉)信息工程学院是我国较早建立GIS专业的教学单位之一。学院结合自主研究和开发MapGIS国产地理信息系统平台软件的经验,以及学校地学背景,在专业定位上以培养面向地学信息化的软件人才为主要培养目标,十多年来为各领域输送了大批GIS软件开发人才,学生受到用人单位的普遍欢迎。

GIS软件开发与其他行业软件开发一样,是一种对动手能力要求极高的工作。众多高校为了强化GIS软件开发人才的动手能力,安排了GIS软件操作、基础开发、二次开发等不同层次的实践教学课程,实践类型更是丰富多彩,包括课内实习、课程设计、综合实习、产学研、第二课堂、项目实践和企业实习等。

虽然普遍重视实践教学,形式和层次也进行了较为合理的搭配,但GIS软件开发人才培养在实践内容方面依然普遍存在以下脱节现象。

(1) 实践内容相互脱节,缺乏主题将各门主要课程和主要实践进行有机联系:①软件开发类课程实习与GIS课程脱节,如数据结构实习与GIS各种常用结构无关,高级编程语言实习与GIS软件开发知识无关;②课程实习与课程实习之间脱节,如高级语言实习、数据结构

实习、数据库实习、网络编程实习等相互脱节；③基础实习与综合实习脱节，如数据结构、数据库等课程的课内基础练习与综合实习在练习内容、练习深度上没有很好地关联。

(2) 实践内容与实际需求脱节。各实践课程内容过于传统，与 GIS 基础知识关联度小，实现功能简单，与实际应用系统差距较大，在一定程度上降低了学生的学习兴趣。

(3)各实践课程间缺乏系统性和连贯性，不利于强化和巩固知识点，实践教学质量难于保证。

(4)综合实习没有标准化，对于要求高、综合性强的题目，对于很多学生在问题和解决方案之间存在巨大的鸿沟，有限的实习时间内难以圆满跨越。因为没有标准化，在综合实习环节也难以利用研究生等辅助教学资源。

上述问题不仅导致学生在软件开发方面的能力参差不齐，而且导致学生的软件开发能力与社会需求严重脱节。因此需要一种将各门课程的主要知识点和技能与 GIS 软件开发有机结合起来的实践，通过一系列由易到难的实践，学生在实现有强烈应用背景的功能中自然而然地运用了各种知识点和技能，从而提高学生 GIS 软件开发的能力。这就是我们希望编写一套阶梯式 GIS 软件工程实践系列教程的初衷。

该阶梯式 GIS 软件工程实践系列教程期望达到下列目标。

(1) 阶梯式。实践难度逐级提高，后面的实践基于前面的实践。

(2) 系统化。考虑到不同年级之间、不同课程之间实践内容的更好衔接；课程实践设置与 GIS 系统挂钩，既关注知识点，也关注综合运用；实践系统化与整体化。

(3) 标准化。不同级别的实践内容标准化，便于教学实施和质量控制；便于授课教师、辅导老师、助研培训与备课；便于学生准备与开展实践活动。

(4) 导向性与挑战性。阶梯式实践教学体系更具导向性，同时也能够满足创新能力强的学生的实践需求。

传统实践教学中，课程内实习是知识点的辅助练习，个性化项目实践和第二课堂则是培养创新能力的环节，该系列教程基于上述目标，旨在有效衔接和补充传统教学环节。

经过万波、叶亚琴、方芳、杨林、左泽均、胡茂盛等的努力，终于形成了阶梯式 GIS 软件工程实践系列教程的基础篇、数据库篇和网络篇。基础篇面向大一到大二的学生，重点训练学生的 GIS 软件开发基础技能，包括基础知识、编程语言、编程工具三位一体的训练；数据库篇面向大二到大三阶段的学生，重点训练 GIS 软件开发专业技能，工程、系统和专业方向三位一体的训练；网络篇面向大三到大四阶段的学生，重点是 GIS 应用软件系统开发训练，特别是基于网络和地图服务的训练。

对于期望从事 GIS 基础软件开发的学生，从基础篇开始练习是不错的选择，然后选择数据库篇以加强数据库开发技能，最后再选择网络篇。

本书是基础篇，对于偏向基础软件开发的软件工程和计算机专业的学生，也可独立选择该书作为实践指导书。

该系列实践教程是中国地质大学(武汉)信息工程学院十多年 GIS 和软件工程专业实践教学经验的总结，出版教程既为了便于开展教学，也为了与兄弟院校分享经验，衷心期望广大师生对该系列教程提出宝贵意见，以便充实改进。

感谢宴四方、熊军、廖婧、刘超群、冯庄等研究生协助完成初稿并测试所有上机指南。

编者

2014 年 10 月

目　录

第 1 章　实习目的及要求

1.1　实习目的

本教程通过开发点、线、区等图形的编辑、存储、查询和显示等功能，实现一个小型桌面图形编辑系统。

最终实现的图形编辑软件系统如图 1.1 所示。

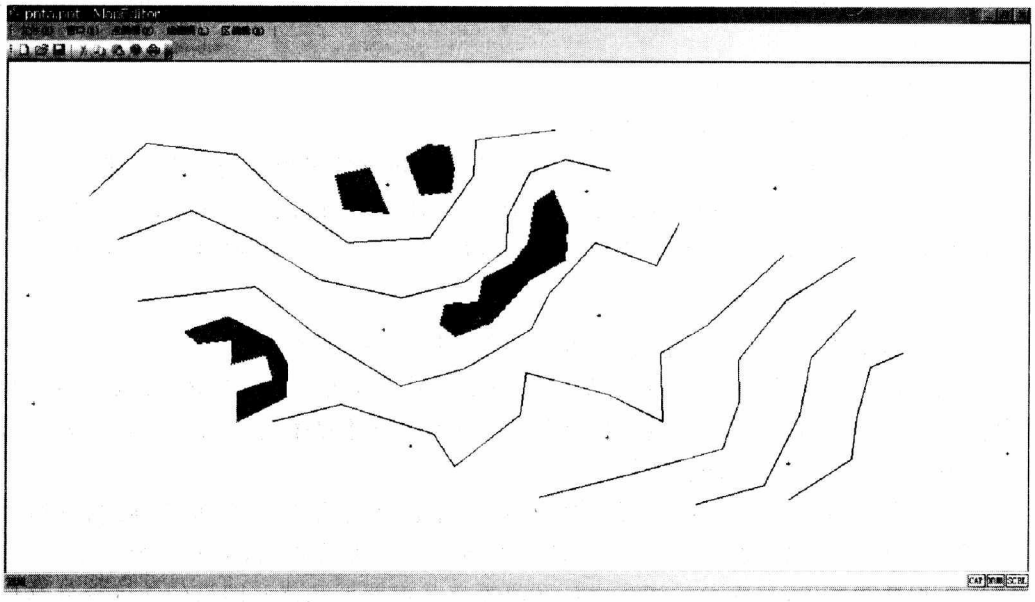


图 1.1　图形编辑软件系统

各项练习从熟悉基本编程环境和程序调试方法开始，由易到难逐步实现各项功能。在循序渐进的练习中，除了需要不断编写新代码，对已有的代码也要不断重构，以保证所实现的功能在操作上一致。因此，随着练习的深入，逐步实现目的 1.1。

目的 1.1　在编程语言、数据结构、几何图形学、Windows 绘图方法、编程工具和调试环境等方面都得到反复的综合性的训练，从而真正掌握相关背景知识和技能，并加深对软件开发过程的理解，提高自主软件开发能力。

该练习侧重于 GIS 软件开发图形编程技能训练，因此在书中没有详细讨论软件需求和设计。在使用该书时，要求学生在开始动手之前先反复阅读后面的“附录 1：C++编码规范”和“附录 2：优秀程序员的基本修炼”，然后在每个练习中对照实践，逐步实现目的 1.2。

目的 1.2　养成良好的编程习惯。

基于上述目的，建议该实习安排在完成计算机基础、C++语言、数据结构等课程学习

后，作为一项完整的课程设计来开展。建议将练习 1～30 作为必须的基础练习，鼓励学生完成练习 31 开始的挑战练习。一般学生在教学日历安排时间内可能不能顺畅地完成所有的练习，因此建议学生在课余时间开展研讨、增强功能等实践。

1.2 实习目标

通过实习，达到以下六方面目标。

1.2.1 C++语言

要求掌握 C++语言的核心内容，能够熟练运用各种概念和方法。

应该掌握的内容包括如下几部分。

(1) C 语言的基本部分，如字符集、关键字、标识符和操作符、变量和常量、表达式、语句、过程控制。

(2) 函数的定义方法和调用方法。

(3) 数组、指针、结构的定义和使用。

(4) 类的定义和使用方法，理解和掌握成员变量和成员函数的定义和使用方法。通过微软基础类库存(Microsoft Foundation Classes，MFC)的调用，充分理解类的概念，熟练掌握类的调用方法，特别是基类成员变量和成员函数，以及 this 指针的使用方法。

1.2.2 数据结构

要求理解复杂数据结构的定义，掌握其实现、排序和查找等方法。

结构是软件开发过程中频繁使用的一种自定义数据类型，正确理解结构数据实例与其中各成员的关系，熟练掌握结构的定义和使用方法，既是程序员入门级的基本要求，也有利于理解计算机存储管理的本质。

常用的数据结构主要分为线性结构(数组、线性表、栈、队列等)和非线性结构(树、图等)。为了降低软件复杂度，使之与大一期末的教学难度大致相当，本实习舍去了一些复杂的功能和性能要求，只涉及描述点、线、区等简单几何图形和这些图形的文件存储结构。即便如此，仍然希望学生通过反复练习达到掌握点、线、区等简单几何图形的数据组织方法，以及能够灵活定义结构、构造结构数组、实现排序和查找元素的目标。

1.2.3 图形绘制

要求理解 Windows 绘图原理，掌握 Windows 绘图方法。

本次实习是在 MFC 环境下完成的，MFC 将 Windows 的绘图方法封装成设备描述、画笔、刷子、绘图模式等 C++类和函数。

通过实现点、线、区等几何图形的交互式编辑、缩放、移动等功能，一方面充分理解数据坐标到窗口坐标之间的映射关系；另一方面理解 Windows 绘图原理，熟练掌握 CClientDC、CPen、CBrush 等类的使用方法，通过实现橡皮线等功能理解 DC 的绘图模式。

1.2.4 编程工具和框架

要求掌握编程工具的基本用法，理解 Visual Studio 应用程序框架。

本书全部基于 Visual Studio 2010 版进行编写，希望学生使用该版本进行练习，使用其他版本可能存在少量界面和功能的不一致。

Visual Studio 是一套集成开发环境(IDE)的开发工具,除了常规的针对代码和资源进行编辑、编译和运行所需的功能(如文件、工具、编辑、生成、调试等),还包括大量用于查看代码和资源静态状态的功能和视图(如解决方案资源管理器、类视图、属性管理器、资源视图),以及查看代码运行时资源状态和运行结果的功能和视图(如断点设置、逐语句跟踪、输出窗口、局部变量窗口、监视窗口、调用堆栈窗口和即时窗口等)。熟悉和掌握这些功能、工具和视窗的用法,是掌握 Visual Studio 集成工具的基础,也是本书希望学生掌握的基本技能。在此基础上,学生可自学 Visual Studio 提供的"体系结构"、"测试"、"分析"等高级功能。

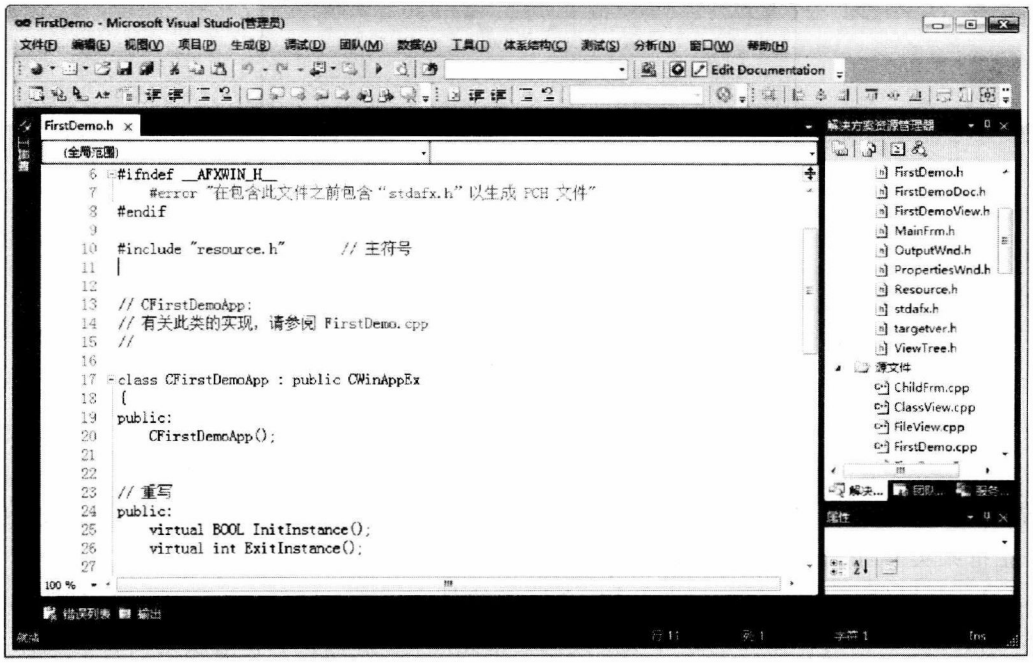


图 1.2　Visual Studio 2010

"解决方案资源管理器"提供项目及其文件的有组织的视图,并且提供对项目和文件相关命令的便捷访问。"类视图"用于显示正在开发的应用程序中定义、引用或调用的符号,阐明代码中的符号结构,并且提供对符号的便捷访问。"资源视图"用于显示工程中用到的所有非编程部件资源,并且提供对资源的快捷编辑。

Windows 是基于消息循环机制的操作系统,Windows 所有的程序都是由消息驱动的。Windows 收集和管理各类事件(如单击菜单或按钮、鼠标移动、键盘按下等)产生的消息,并将消息发送给与消息相关的应用程序,应用程序在接收到消息后根据消息类型执行相应的操作。例如,当用户单击某菜单项时,Windows 首先捕获到该事件,产生一个 WM_COMMAND 消息并发送到用户单击的应用程序的消息队列中,应用程序逐个处理消息队列中的消息,在处理 WM_COMMAND 消息时,调用相应的消息处理函数。

Visual Studio 将 Windows 操作系统的各种应用程序接口(Application Programming Interface,API)函数封装成 C++类库,统称为 MFC,MFC 中封装的类如 CWnd、CButton、

CFile、CDialog 等，MFC 还将消息处理机制封装成应用程序框架。在 MFC 应用程序框架基础上开发应用程序，程序员就可以大大简化“接收消息—调用消息处理函数”这一复杂过程，而将思维集中在文件处理、视窗操作、数据对象的设计与实现上来。因此，理解 MFC 并掌握应用程序框架，对使用 Visual Studio 开发应用程序非常重要。

MFC 应用程序框架由“应用程序类”(CWinApp)→“主框架类”(CMainFrame)→“视窗类”(CView)→“文档类”(CDocument)构成。一般一个应用程序有一个应用程序类、一个主框架类、多个视窗类、多个文档类，其关系如上述箭头所示。

MFC 应用程序框架各组成部分及其关系如图 1.3 所示。

图 1.3　MFC 应用程序框架各组成部分及其关系

图 1.3 中，“主框架窗口”是整个应用程序的窗口，即图 1.3 中黑色框内的部分。“视图窗口”是主框架窗口的一个子窗口，即图 1.3 中虚线框内的部分。主框架窗口对应的类是主框架类 CMainFrame，视图窗口对应的类是视窗类 CView。

MFC 提供了一个文档/视图结构，文档指的是文档类 CDocument，视图指的是视窗类 CView。数据的存储和加载可以由文档类来完成，数据的显示和修改则由视窗类来完成。对于以 MapEditor 命名的程序，文档类指的是 CMapEditorDoc，派生于 CDocument。视窗类指的是 CMapEditorView，派生于 CView。

本书为了更好地实践数据结构相关知识和文件操作，没有使用 CDocument 类，而是使用 CFile 类进行数据存储和管理。

对于以 MapEditor 命名的程序，应用程序类指 CMapEditorApp，派生于 CWinApp。在程序启动时首先会通过 CMapEditorApp 类完成一些初始化工作，包括窗口类的设计、注册，以及窗口的创建、显示和更新。然后再进入消息循环中，通过消息映射机制来处理各种消息。

MFC 中的所有类均派生于 CObject 类，各种派生类如图 1.4 所示。

1.2.5　程序调试

要求理解调试对于提高软件开发效率和质量的重要性，掌握程序调试方法。

掌握断点设置、单步跟踪、变量查看等调试方法，习惯使用跟踪手段检查和优化程序。

程序调试是在程序正式发布之前，程序员借助集成调试环境，用手工方式逐行或逐过

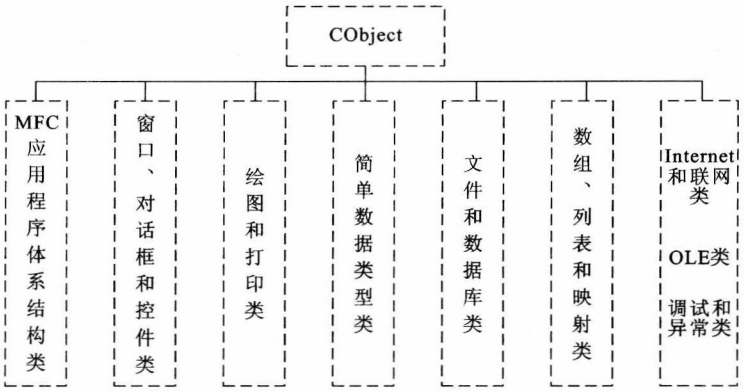


图 1.4　MFC 类汇总

程走查代码，依据各语句的执行结果，确认代码与设计的一致性，以及实际性能与需求或设计的符合程度。通过跟踪检查，发现代码算法错误或性能偏差，进而改正错误或优化代码。

程序调试本质上是一种白盒测试，是原始代码经过不断修改快速摆动到最优的有效方法，是优秀程序员必须掌握的基本技能之一。

通过本次实习，要求掌握在 Visual Studio 2010 环境下，设置和取消断点、逐语句或逐过程跟踪程序的每一步，查看各步实际运行结果，验证结果的正确性等方法。要求熟练掌握这些方法对应的按键使用方法。

通过本次实习，希望学生能够灵活运用编译器、调试器等工具和代码浏览等手段实现高效编程。

1.2.6　编程规范化

要求理解规范化编程对提高程序质量的重要性，掌握规范编程基本方法，形成良好的编程习惯。

软件规范的目的是统一软件的规范风格，提高软件源代码的可读性、可靠性和重用性，提高代码的质量和可维护性，减少软件维护成本。良好的编程规范可以改善软件质量、缩短软件开发时间、提升团队效率、简化维护工作。所以，掌握基本编程规范，形成良好的编程习惯对优秀程序员尤为重要。

通过本实习，要求了解附录 1 所列编程规范的基本要求，并在实习中反复认真应用，逐步形成使用简洁的语句编写代码、使用准确的语言编写注释的良好习惯，达到程序易读易理解的目标，降低代码歧义带来的隐形错误，从而提高编程效率、代码的质量和可靠性。

1.3　实习要求

1.3.1　对学生的要求

(1) 小步走。不要跳过任何一次练习，对于能力强的同学，可以加快实习进度，并尝试实现挑战编程所列的高级功能或自行实现更多更加复杂的功能。

(2) 重复做。实习过程中，学生一般都有足够的时间完成各项实习任务，建议学生在

第一次实现后，不再参考本书重新做一次，即所有过程和步骤完全一边回忆一边动手实现。如果重复的过程中还必须参看本书或请教别人才能完成，建议做第 3 次，以达到真正领悟和掌握。

（3）真掌握。大多数学校在教学过程中，都会根据教学内容和办学特色，安排各式各样、范围宽泛的练习，对学生动手能力的提高有利无害，但问题是很多同学并未真正掌握，没有达到实习目标甚至没有完成。

因此，理解每一项练习背后的知识、掌握练习要求的工具、过程和方法，就非常重要。要做到这一点，仅跟随教程或老师练习一遍显然不够，学生必须重复实习内容直至掌握实习内容，如果课内时间不够，学生课外要花时间，做到真理解，也就是不仅说得清，还写得出、做得对。

1.3.2 对老师的要求

（1）记录过程。要求学生在实习过程中，填写实习记录，作为实习成绩评定的部分依据。

（2）抽查代码。指导教师在实习过程中，要与学生交流，检查学生的实习情况，检查学生所写的代码，并抽取代码由学生解释其含义和用法。

（3）检查结果。通过成果演示、答辩、代码抽查、部分功能重做等形式检查和评定学生掌握的程度。

第 2 章　背景知识概述

2.1　几何图形及其结构

2.1.1　点

点是几何图形最基本的单元，是空间中只有位置、没有大小的图形。在一个平面上，通常用一个有序坐标对(x,y)来表示一个点，习惯上 x 表示水平位置，y 表示竖直位置，如图 2.1 所示。

虽然一个有序坐标对可以确定一个点的位置，但由于点是现实世界中点状地物（如电杆、灯塔、泉水、水文站、气象观测点等）的抽象，多种多样，所以除了空间位置，点还有一些属性，如种类、颜色等。

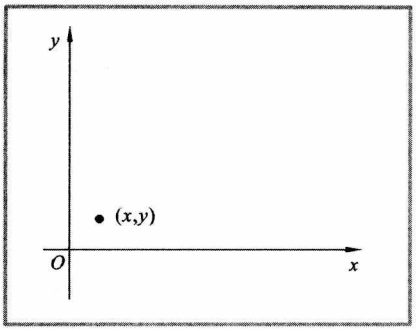


图 2.1　点坐标

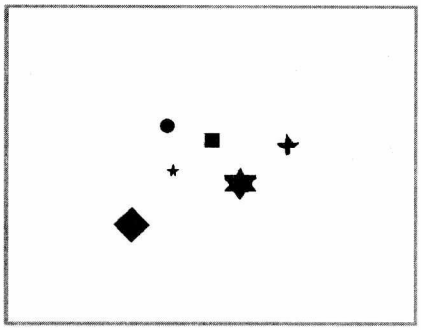


图 2.2　点状图形

在计算机中，为了记录和显示不同的点，通常对每个点给一个唯一的编号，通常称为 ID；为了显示不同的图案以表达不同的含义，还要记录图案号。因此，在本次实习中点图形的结构如下。

```
struct{
        long            ID;             //点编号
        double          x;              //点位坐标 x
        double          y;              //点位坐标 y
        COLORREF        color;          //点颜色
        int             pattern;        //点图案号
        char            isDel;          //是否被删除
    }PNT_STRU;
```

2.1.2　线

线是现实世界中线状地物（如道路、河流、航线、电力线等）的抽象。当要记录一条线（包括曲线）时，把所有数学意义上的点都记录下来显然是不必要的，仅需要记录线上的一些特征点，由这些特征点直接连接或者用函数（如弧、样条等）分段描述即可描述线的全

貌。这些特征点称为“节点”，它包括线的起点、转折点和终点。所以，在计算机中，一条线是用有限多个有序节点来表示的，如图 2.3 所示。为了降低难度，在本书中不考虑用函数描述的曲线。

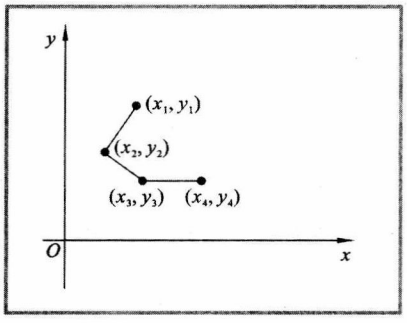


图 2.3　线坐标

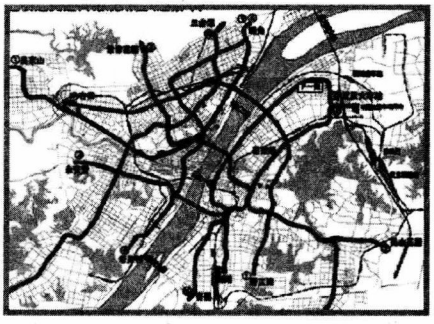


图 2.4　线状图形

与点类似，线的种类也是多种多样的，除了节点序列，还有颜色、线型、种类等更丰富的属性。为了区分不同的线，每条线同样要分配一个唯一的 ID。

因为不同的线节点数不同，所以为了提高存储和检索效率，将每条线分两部分存储，一部分是长度固定的索引，另一部分是长度变化的“节点数组”。

线索引结构如下。

```
struct {
        long            ID;             //线编号
        char            isDel;          //是否被删除
        COLORREF        color;          //线颜色
        int             pattern;        //线型(号)
        long            dotNum;         //线节点数
        long            datOff;         //线节点数据存储位置
    }LIN_NDX_STRU;
```

单个节点数据结构如下。

```
struct {
        double          x;              //节点 x 坐标
        double          y;              //节点 y 坐标
    }D_DOT;
```

2.1.3　区

区是现实世界中面状地物(如地块、湖泊、行政区等)的抽象。在计算机中，区是由平面上三个和三个以上的节点连接而成的封闭图形，即通过有序描述区边界的节点来描述一个最简单的区(因为有孔的区结构过于复杂，本实习不讨论此类区)，这样，最简单的区就是有限多个有序坐标点，如图 2.5 所示。

与点、线类似，区的种类也是多种多样的，除了节点序列，还有颜色、填充图案、种类等属性。为了区分不同的区，每个区同样要分配一个唯一的 ID。

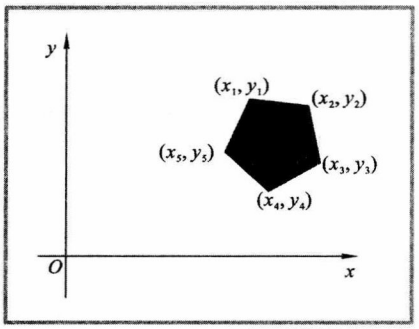
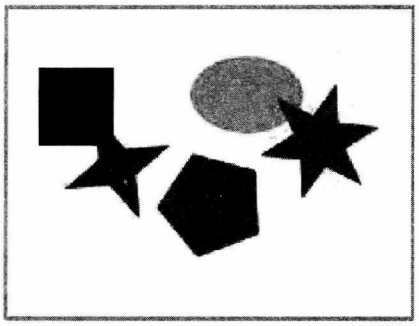


图 2.5　区坐标图



2.6　面状图形

因为不同区的边界节点数不同，所以为了提高存储和检索效率，与线类似，将每个区分两部分存储，一部分是长度固定的索引，另一部分是长度变化的边界“节点数组”。

区索引结构如下。

```
struct{
        long            ID;         //区编号
        charis          Del;        //是否被删除
        COLORREF        color;      //区颜色
        int             pattern;    //图案(号)
        long            dotNum;     //边界节点数
        long            datOff;     //边界节点数据存储位置
    }REG_NDX_STRU;
```

单个节点数据结构如下。

```
struct{
        double          x;          //节点 x 坐标
        double          y;          //节点 y 坐标
    }D_DOT;
```

从上述说明可以看出，在本实习中简单区的结构与线相似。

2.2　Windows 图形编程

2.2.1　图形绘制方法

Visual C++所编写的 Windows 应用程序通常在视图类 OnDraw 函数中添加绘图代码来完成图形绘制。OnDraw 函数是 CView 类的虚拟成员函数，它在 CView 类的派生类中被重新定义，在接到 WM_PAINT 消息后就会通过消息映射函数 OnPaint 调用它。WM_PAINT 消息是在某个视图窗口需要重画或刷新其显示内容时发出的。如果程序的数据被改变，则可以调用视图的 Invalidate 成员函数使视图窗口无效而发出 WM_PAINT 消息，并最终导致 OnDraw 函数被调用来完成绘图。

在 Windows 平台上，应用程序的图形设备接口(Graphics Device Interface，GDI)被抽象为 DC。DC 也称为设备描述表，是 GDI 中的重要组成部分，是一种数据结构，它定义了一系列图形对象和图形对象的属性以及图形输出的图形模式。图形对象包括画线用的