



普通高等教育“十一五”规划教材

程序设计基础

李晓林 张俊 主编 姬涛 副主编

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

程序设计基础

责任编辑: 李小军 徐盼欣 封面设计: 付 巍 封面制作: 白 雪



中国铁道出版社 计算机图书批销部
地址: 北京市宣武区右安门西街8号
邮编: 100054

网址: <http://edu.tqbooks.net>
读者热线电话: (010) 63583215
销售服务电话: (010) 83550290/91 83550580

ISBN 978-7-113-08560-5



9 787113 085605 >

ISBN 978-7-113-08560-5 /TP·2689 定价: 36.00 元

普通高等教育“十一五”规划教材

程序设计基础

李晓林 张俊 主编

姬涛 副主编

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

内 容 简 介

本书根据非计算机专业计算机基础课程教学指导分委员会提出的“关于进一步加强高等学校计算机基础教学的意见”(简称“白皮书”)中程序设计基础大纲为依据,并结合近年来C语言程序设计基础和C++面向对象程序设计教学实践经验编写而成。

本书以结构化程序设计思想和方法为主,以C++语言为工具,详细讲述了结构化程序设计的主要内容:数据类型、运算符、表达式、控制结构、函数、数组、指针、结构、流I/O及文件等。本书力求从C结构化程序设计平滑过渡到C++面向对象程序设计,同时兼顾了计算机等级考试二级C++大纲的要求。

全书共分为9章,涵盖全面,实例丰富,内容由浅入深,通俗易懂。为帮助学生对教材内容的深入理解,并培养学生的实践能力,本书还配有试验指导教材——《程序设计基础习题及上机指导》。

本书以培养基本的程序设计能力和计算机应用能力为目的,适合作为各专业计算机公共基础课程教材,也可作为计算机专业程序设计课程的入门教材。

图书在版编目(CIP)数据

程序设计基础 / 李晓林, 张俊主编. —北京: 中国铁道出版社, 2008. 1

普通高等教育“十一五”规划教材

ISBN 978-7-113-08560-5

I. 程… II. ①李… ②张… III. 程序设计—高等学校—教材 IV. TP311

中国版本图书馆CIP数据核字(2008)第014218号

书 名: 程序设计基础

作 者: 李晓林 张 俊 姬 涛

出版发行: 中国铁道出版社(100054, 北京市宣武区右安门西街8号)

策划编辑: 严晓舟 徐海英

责任编辑: 李小军 徐盼欣

封面设计: 付 巍

封面制作: 白 雪

印 刷: 化学工业出版社印刷厂

开 本: 787×1 092 1/16 印张: 24 字数: 564 千

版 本: 2008年1月第1版 2008年1月第1次印刷

印 数: 1~4 200 册

书 号: ISBN 978-7-113-08560-5/TP·2689

定 价: 36.00 元

版权所有 侵权必究

凡购买铁道版的图书,如有缺页、倒页、脱页者,请与本社计算机图书批销部调换。

前 言

计算机科学与技术是当今世界上发展最快、应用最广泛的科学技术之一。计算机技术已经渗透到各学科专业和社会的各个领域。计算机的应用能力成为衡量现代人才素质的一项重要内容。计算机程序设计体现了是否具备利用计算机开展创新型工作的能力,特别是对于从事理工类专业的人员更加重要。

本套教材分为《程序设计基础》和《程序设计基础习题及上机指导》,力求理论与实践相结合,以培养学生分析问题和解决问题的能力。《程序设计基础》教材以程序设计的基础理论为主,重点讲解程序设计的基本理论和方法。《程序设计基础习题及上机指导》教材与《程序设计基础》配套,包括实验指导、实验思考题、练习题等。

本书以非计算机专业计算机基础课程教学指导分委员会提出的“关于进一步加强高等学校计算机基础教学意见”(简称“白皮书”)中程序设计基础大纲为依据,结合近年来 C 语言程序设计基础和 C++ 面向对象程序设计教学实践经验编写而成。本教材力求从结构化的程序设计 C 语言平滑地过渡到面向对象的程序设计 C++,并兼容了计算机等级考试大纲的要求。

本教材以培养大学生掌握程序设计的基本理论和方法为目的,以培养计算机应用编程能力为目标,其内容涵盖了结构化程序设计和面向对象的程序设计的基本理论、基本概念、方法和规范,重点为结构化程序设计,给出了问题求解过程分析,并用流程图和 N-S 图描述程序,通过举例引入了面向对象程序设计的基础知识。教材中的例题可在 Visual C++ 6.0 集成开发环境下编译运行。本教材既可以作为计算机程序设计初学者的入门教材,也可供读者自学。

本教材共分为九章,内容包括:

第 1 章 程序设计概述。介绍计算机程序设计的基本概念、程序设计的方法和程序设计的过程。

第 2 章 数据类型、表达式和基本运算。介绍字符集、关键字、标识符、数据类型、运算符、表达式的概念,以及各种相关类型的常量、变量的说明、使用方法和表达式求值的运算规则。

第 3 章 程序设计的控制结构。介绍结构程序设计的基本概念和结构化程序设计的方法,结构化程序设计的常用工具流程图和 N-S 图, C/C++ 语言的顺序控制结构、选择控制结构、循环控制结构语句以及应用问题求解的方法。

第 4 章 函数。介绍用函数实现模块化程序设计的思想,函数原型、函数定义、函数调用的基本概念,函数参数传递的机制,递归函数、内联函数、函数重载、变量的作用域和存储类型等。

第 5 章 数组。介绍数组的概念,一维数组的定义、初始化以及常见的应用,二维数组的定义与使用,字符串与字符数组等概念及应用。

第 6 章 指针。介绍指针的基本概念及相关运算,指针用作函数参数、函数返回指针、函数指针等,指针与数组的相互表示,指针数组、指向数组的指针、指向指针的指针等概念,用 const 关键字修饰指针,动态内存分配。

第7章 结构、枚举与联合。介绍结构、枚举、联合等自定义数据类型的机制，结构的基本概念及操作，结构与数组、结构与指针、结构与函数的应用，链表的概念及常用操作，枚举、联合的概念及应用。

第8章 流 I/O 与文件。介绍流和文件的基本概念，C++中标准输入输出流和对象及运算符，C语言中标准输入输出流函数，文件流 I/O 及其常用函数。

第9章 面向对象程序设计。介绍面向对象程序设计中的基本概念、主要思想和重要语法，包括类定义的语法，构造函数和析构函数，运算符重载的概念及其实现方式，继承的概念、语法和方式，虚基类，虚函数、纯虚函数和抽象类等概念，模板的概念，函数模板和类模板，I/O 流的概念，标准流和文件流。

附录包括常用的 ASCII 码表、常用运算符的优先级和结合性、常用数学库函数、常用字符串处理库函数。

教学安排建议：课堂教授授课 32 学时，课内实验 32 学时，课外实验 64 学时。由于学时有限，在组织教学时重点应放在结构化程序设计和应用方面，可根据学生的特点适当取舍，部分内容可安排自学。

本书第 1~3 章由李晓林编写，第 4、5 章由姬涛编写，第 6~8 章由张俊编写，第 9 章和附录由李晓林和张俊共同编写。在本书编写过程中，得到了张彦铎教授、赵彤洲副教授的大力支持和热情指导，在此表示衷心的感谢！

本书源于程序设计基础教育的教学实践，凝集了任课教师多年的教学实践经验。由于计算机科学技术发展迅速，程序设计的教学内容、方法和手段日新月异，加之编者水平有限，书中不足之处在所难免，敬请读者批评指正，以便在今后进一步完善。

编者

2007 年 12 月

目 录

第 1 章 程序设计概述	1
1.1 程序设计概述	1
1.1.1 程序设计的概念	1
1.1.2 程序设计语言的发展	2
1.2 程序设计的方法	4
1.2.1 结构化程序设计	4
1.2.2 面向对象程序设计	4
1.2.3 程序设计的一般过程	5
1.3 程序设计语言	6
1.3.1 结构化程序设计语言 C	6
1.3.2 面向对象的程序设计语言 C++	8
1.4 C/C++程序设计语言	9
1.4.1 C/C++程序的组成	9
1.4.2 C/C++程序的书写格式	9
1.4.3 C/C++程序的基本结构	10
1.4.4 C/C++程序的实现	10
1.5 Visual C++ 6.0 语言开发环境	10
1.6 本章小结	14
练习题	15
第 2 章 数据类型、表达式和基本运算	16
2.1 字符集与标识符	16
2.1.1 字符集	16
2.1.2 保留字和词法约定	17
2.2 数据类型	18
2.2.1 数据类型	18
2.2.2 数据类型的分类	19
2.3 常量和变量	20
2.3.1 常量	20
2.3.2 变量	22
2.3.3 类型转换	24
2.4 运算符和表达式	25
2.4.1 运算符与表达式	25
2.4.2 算术运算	26
2.4.3 赋值运算	30
2.4.4 关系运算	31

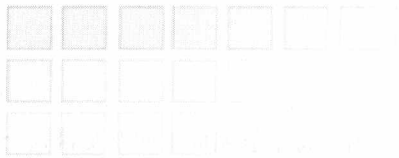
2.4.5	逻辑运算符	33
2.4.6	条件运算符	35
2.4.7	逗号运算符	35
2.4.8	sizeof 运算符	37
2.4.9	位运算符	37
2.4.10	指针和地址运算	39
2.5	表达式的优先级和结合性	41
2.6	本章小结	43
	练习题	43
第 3 章	程序设计的控制结构	46
3.1	结构化程序设计	46
3.1.1	结构化程序设计思想	46
3.1.2	流程图	47
3.1.3	N-S 图	48
3.1.4	结构化程序设计	49
3.2	语句	49
3.3	顺序控制结构	51
3.3.1	顺序控制结构	51
3.3.2	应用举例	51
3.4	选择控制结构	53
3.4.1	选择控制结构	53
3.4.2	if 语句	53
3.4.3	条件运算符	64
3.4.4	switch 语句	65
3.5	循环控制结构	69
3.5.1	循环控制结构	69
3.5.2	while 循环结构	69
3.5.3	do-while 循环结构	74
3.5.4	for 循环结构	76
3.5.5	嵌套结构	79
3.6	其他语句	83
3.6.1	break 语句与 continue 语句	83
3.6.2	goto 语句与标号	85
3.7	应用举例	86
3.8	本章小结	94
	练习题	95
第 4 章	函数	98
4.1	函数概述	98

4.1.1	函数的概念	98
4.1.2	库函数	99
4.2	函数定义和调用	101
4.2.1	函数定义	101
4.2.2	函数声明	102
4.2.3	函数调用	103
4.2.4	函数返回类型和返回值	106
4.3	函数的参数传递	107
4.3.1	形式参数和实际参数	107
4.3.2	传值调用	108
4.3.3	传地址调用	110
4.3.4	函数的默认参数值	110
4.4	递归调用	111
4.4.1	递归的基本概念	111
4.4.2	举例	113
4.5	内联函数	115
4.6	函数重载	116
4.7	变量的作用域	117
4.7.1	内部变量	118
4.7.2	外部变量	118
4.7.3	同名变量作用域规则	120
4.8	变量的存储类型	121
4.8.1	变量的生存期	121
4.8.2	自动变量	121
4.8.3	静态变量	122
4.8.4	寄存器变量	123
4.8.5	外部变量	123
4.9	预处理	125
4.9.1	文件包含	125
4.9.2	宏定义	126
4.9.3	条件编译	127
4.10	本章小结	129
	练习题	130
第 5 章	数组	132
5.1	数组的概念	132
5.2	一维数组	133
5.2.1	一维数组定义	133
5.2.2	一维数组初始化	134

5.2.3	一维数组元素引用	134
5.2.4	一维数组作为函数参数	136
5.2.5	一维数组应用	138
5.3	二维数组	145
5.3.1	二维数组定义	145
5.3.2	二维数组初始化	146
5.3.3	二维数组元素引用	146
5.3.4	二维数组应用	147
5.4	字符数组	149
5.4.1	字符串与字符数组	149
5.4.2	字符串函数	151
5.4.3	字符数组	153
5.4.4	字符数组应用举例	155
5.5	本章小结	157
	练习题	157
第 6 章	指针	159
6.1	指针与指针变量	159
6.1.1	指针的概念	159
6.1.2	指针定义	161
6.1.3	指针初始化	162
6.1.4	指针运算	163
6.2	指针与函数	173
6.2.1	指针作为函数参数	173
6.2.2	函数返回指针	179
6.2.3	函数指针	180
6.3	指针与数组	184
6.3.1	指针与数组	184
6.3.2	字符数组、字符串常量与字符指针	187
6.3.3	指针数组、指针的指针与指向数组的指针	192
6.4	指针与 const	199
6.4.1	const 关键字	199
6.4.2	用 const 限定指针	200
6.5	动态内存分配	205
6.5.1	基本概念	205
6.5.2	malloc/free	206
6.5.3	new/delete	208
6.6	应用举例	210
6.7	本章小结	213
	练习题	214

第 7 章 结构、枚举与联合	216
7.1 结构	216
7.1.1 结构的概念	216
7.1.2 结构类型的定义	217
7.1.3 结构变量的定义及初始化	219
7.1.4 结构成员访问	221
7.2 结构与数组	225
7.2.1 结构数组	225
7.2.2 应用举例	226
7.3 结构与指针	228
7.3.1 结构指针	228
7.3.2 链表	231
7.4 结构与函数	247
7.4.1 参数为结构类型	247
7.4.2 返回结构类型	251
7.5 枚举	253
7.5.1 枚举的概念	253
7.5.2 枚举类型的定义	253
7.5.3 枚举变量的应用	255
7.6 联合	258
7.6.1 联合的概念	258
7.6.2 联合类型的定义	259
7.6.3 联合变量的应用	261
7.7 typedef 定义类型	264
7.8 应用举例	266
7.9 本章小结	272
练习题	273
第 8 章 流 I/O 与文件	274
8.1 基本概念	274
8.1.1 流	274
8.1.2 文件	275
8.1.3 流缓冲区	276
8.2 标准输入输出流	277
8.2.1 C++ 的流 I/O	277
8.2.2 C 的流 I/O	278
8.3 文件流	284
8.3.1 文件指针	284
8.3.2 文件 I/O	285

8.4 应用举例.....	301
8.5 本章小结.....	308
练习题	309
第9章 面向对象程序设计	311
9.1 类与对象.....	312
9.1.1 类的定义	313
9.1.2 数据成员	314
9.1.3 成员函数	315
9.1.4 综合举例分析	318
9.1.5 对象的定义及创建.....	319
9.1.6 构造函数和析构函数.....	321
9.2 运算符重载.....	327
9.2.1 概述	327
9.2.2 成员函数形式重载.....	328
9.2.3 友元函数形式重载.....	330
9.3 类的继承与派生	332
9.3.1 继承的定义	332
9.3.2 派生类	333
9.3.3 单继承	337
9.3.4 基类指针与派生类指针	340
9.3.5 虚基类	341
9.4 多态性	344
9.4.1 多态性的概念	344
9.4.2 虚函数机制	344
9.4.3 纯虚函数与抽象类.....	347
9.5 模板	350
9.5.1 模板的概念	350
9.5.2 函数模板	351
9.5.3 类模板	354
9.6 I/O 流库	356
9.6.1 概述	356
9.6.2 标准流	357
9.6.3 文件流	358
9.6.4 应用举例	363
9.7 本章小结.....	366
练习题	366
附录 A.....	367



第 1 章

程序设计概述

计算机以其独特的方式为人们日常工作、学习和生活带来了许多便利和帮助。人们通常将一些繁杂的事务交给计算机处理，极大地提高了工作效率。计算机处理任何事物都是按照一定的序列指令来执行的。如何按照事务处理的要求来编写计算机执行的指令序列，即程序设计。设计计算机能够识别的指令序列常常借助于程序设计工具，如 Visual C++ 6.0 集成开发环境。为了使计算机运行稳定、可靠、效率高，需要有效的程序设计方法，如结构化程序设计方法和面向对象的程序设计方法。

本章从程序设计的概念出发，介绍了结构化程序设计方法和面向对象的程序设计方法的基本概念，以 C/C++ 程序设计语言为例介绍了程序设计的基本概念和结构，通过举例简要介绍了 Visual C++ 6.0 开发环境以及一个简单程序的设计实例。

通过对本章的学习，读者应该掌握程序设计的基本概念、程序设计的过程、程序设计语言的定义、程序的基本结构，以及了解 Visual C++ 6.0 集成开发环境。

1.1 程序设计概述

自 20 世纪 40 年代第一台计算机问世以来，计算机就以其独特的方式影响着人们的生活和工作，人们不仅被它近乎完美的工作效率和方式所吸引，而且为它的发展投入了大量的精力和劳动，创造出一个又一个人间神话。这些神话大多围绕着程序设计这个中心而展开，在今天的网络和信息时代，情况更是如此。

1.1.1 程序设计的概念

1. 程序与程序设计

在一般意义上，程序是指“事情进行的先后次序”或者“按照时间先后或依次安排的工作步

骤”。它有程式、顺序、方式、手续、步骤等含义。在日常生活中有许多事务都是与“程序”分不开的，如平常所说的“工作流程”、“会议议程”等。例如，与朋友打电话交谈这个事务一般需要按以下步骤进行：

- (1) 拿起电话话筒，拨朋友的电话号码。
- (2) 等待电话线路接通。
- (3) 等待朋友接电话。
- (4) 与朋友交谈。
- (5) 交谈结束后，挂上电话。

在上面这个事务中，每个步骤的次序一般是不能颠倒的，否则，将不能完成与朋友交谈这个事务。如果不拨朋友的电话号码，朋友不会知道有通话请求；同样，线路不通对方也不知道有通话请求；线路接通了，但没有人接听电话，电话交谈也不可能进行下去。因此，在这个事务中，一定要按照电话通信的流程一步一步地执行才能完成。

计算机程序是指计算机完成一项任务所要执行指令的序列。指令是计算机能够识别的命令。而完成一项任务可能是执行几条指令的序列，也可能是执行成千上万条指令的序列。简单来讲，计算机程序就是由计算机指令构成的序列，计算机按照程序中的指令逐条执行就可以完成相应的任务。

程序设计是人们为了完成某项具体的任务而编写的一系列指令，并将这一系列指令交给计算机去执行的过程。程序设计要求编写程序的人员首先对需要完成任务有一个比较清晰的认识；然后按照计算机可以识别的方式来组织这些指令构成一个指令系列，即程序；最后将描述这个任务的程序交给计算机去执行，从而完成该任务。

实际上，计算机自己不会做任何工作，它所做的工作都是由人们事先编好的程序来控制的，即使是人们常说的计算机智能，如机器人等，也是按照事先编好的程序执行，只不过这种智能机器人执行的程序比一般的程序更加复杂。

2. 程序设计语言

要想让计算机能够正确地执行人们所编写的一系列指令（程序），就需要其能够理解这些程序的含义。程序设计语言就是用计算机能够理解的语言来表达设计程序的含义，是人们与计算机之间进行交流和通信的工具。

人类语言是由语法和词汇构成的，同样，计算机语言也是由语法和词汇构成的。所谓语法就是规则的集合，规定什么是允许的，什么是不允许的；什么是正确的，什么是错误的。词汇也就是符号，它是构成语言的基本要素。

1.1.2 程序设计语言的发展

目前，通用的计算机还不能识别自然语言，而只能识别特定的计算机语言。也就是说，能够使计算机直接识别的语言只有一种，即机器语言。所谓机器语言，就是指机器能够识别的指令的集合，即指令系统。机器语言是由二进制数 0 和 1 组成的指令系列。

计算机语言一般分为低级语言和高级语言。

低级语言直接依赖于计算机硬件，不同的机型所使用的低级语言是完全不一样的。高级语言

则不依赖于计算机硬件,用高级语言编写的程序可以方便地、几乎不加修改地在不同类型的计算机上运行。

无论何种语言所编写的程序,其最终在计算机上的执行都是由 CPU 所提供的机器指令来完成的。机器指令是用二进制表示的指令集,每种类型的 CPU 都有自己的指令集。

1. 低级语言

低级语言包括机器语言和汇编语言。

(1) 机器语言

机器语言是直接由二进制指令来编写程序的语言。使用机器语言编写程序时,必须准确无误地牢记每一条指令的二进制编码。例如,以下是某计算机的两条机器指令:

加法指令: 1 0 0 0 0 0 0 0

减法指令: 1 0 0 1 0 0 0 0

不同类型的计算机所使用的机器语言不同。

用机器语言编写的程序计算机可以直接执行,执行效率高。但机器语言的指令不直观,难认、难记、难理解,且较烦琐,容易出错,写出的程序不能通用。编写机器语言程序时,要求程序员必须相当熟悉计算机结构,因此,目前很少直接用机器语言编程。

(2) 汇编语言

20 世纪 50 年代中期,为了减轻使用机器语言编程的负担,人们开始采用一些“助记符号”来表示机器语言中的机器指令,这样便形成了汇编语言。助记符一般采用一个操作的英文字母的缩写,与机器语言相比,其便于识别和记忆。例如,上例中的两条指令用汇编语言描述如下:

ADD A, B

SUB A, B

不过,计算机不能直接执行用汇编语言编写的程序,它必须经过一个称为汇编程序的系统软件翻译成机器语言程序后才能执行。我们称前者为源程序,称后者为目标程序。

汇编语言指令和机器语言指令之间有一一对应的关系。因而,不同的计算机其汇编语言也不尽相同,并且程序编写时仍需要对计算机内部比较熟悉,依然比较烦琐。但相对于机器语言而言,汇编语言要好得多。因此,在实际程序设计中,如果程序运行时间要求比较严格,程序与硬件操作联系紧密,人们还是常用汇编语言编写有关程序来解决这些问题。

2. 高级语言

高级语言是一种比较接近自然语言和数学语言的程序设计语言。高级语言的出现大大提高了程序员的工作效率,降低了程序设计的难度,并提高了程序的质量。用高级语言编写的程序看起来更像英语,很容易理解,不但使程序具有良好的可读性和可维护性,而且使更多的人掌握了程序设计方法,从而使计算机技术得到迅速的应用和普及。

高级语言的一条语句相当于多条汇编语言指令或机器指令,表达能力强。在使用算法语言时,程序设计人员不需要熟悉计算机的指令系统,可以把精力集中在研究问题的求解方法步骤(过程)上。因此,高级语言也称为面向过程的程序设计语言。同时,高级语言不依赖于计算机硬件。为某种类型的计算机编写的算法语言程序,可以很方便地移植到其他类型的计算机上运行。

当然,计算机也不能直接执行高级语言程序。如同汇编语言一样,高级语言程序(源程序)

也必须先经过编译程序（或解释程序）翻译成机器语言程序（目标程序）后，才能由计算机执行。常用的高级语言有 FORTRAN 语言、ALGOL 语言、Pascal 语言、C 语言、BASIC 语言等。

1.2 程序设计的方法

1.2.1 结构化程序设计

在程序设计语言的发展过程中，程序设计语言的方法先后经历了三个主要阶段：非结构化程序设计阶段、结构化程序设计阶段和面向对象的程序设计阶段。

非结构化程序设计方法由于其诸多的弊端，现已基本淘汰；面向对象技术已经成为程序设计的主流，但在面向对象的设计过程中，对一些细节问题的考虑又离不开结构化程序设计方法，C 语言就是属于结构化程序设计语言。

结构化程序设计（Structured Programming）是建立在面向过程的程序设计基础上的，所谓面向过程是指将问题求解过程看作对数据进行加工的过程，采用这种方法的程序设计语言称为面向过程的程序设计语言。

程序设计可以看作是解决问题步骤的计算机实现，而每一步骤都可以看作是要完成一个功能。因此，面向过程设计思想就是设计一系列彼此独立的功能，每一功能的实现都是通过对数据进行不同的加工以形成新的数据，最后将这些功能整合在一起共同构成一个实际问题的解决方案。

面向过程的程序设计的内容包括数据的组织和数据的加工两个方面。数据的组织也就是所谓的数据结构设计；数据的加工也就是设计算法。

结构化程序设计是指面向过程的程序设计中仅仅使用顺序、选择和循环三种基本的控制结构来实现软件功能的程序设计方法。

1.2.2 面向对象程序设计

结构化程序设计已经无处不在，几乎每种程序设计语言都具备支持结构化程序设计的机制。然而，随着程序规模与复杂性的增长，程序设计中的数据结构变得与这些数据上的操作同样重要。在大型结构化程序中，一个数据结构可能被多个过程处理，修改此数据结构将影响到所有这些过程。这给程序的修改和维护带来了困难，同时也给设计大型和复杂的系统带来了困难。

面向对象技术是当今比较流行的软件设计与开发技术，主要包括面向对象分析、面向对象设计、面向对象编程、面向对象测试以及面向对象软件维护等。面向对象程序设计技术的提出，主要是为了解决传统程序设计方法所不能解决的代码重用问题。

面向对象程序设计是建立在结构化程序设计基础之上的程序设计方法，其最重要的变化是程序设计围绕数据来设计，而不是围绕操作本身。在面向对象程序设计中，将程序设计为一组相互协作的对象，而不是一组相互协作的函数。

对象是面向对象技术中的一个重要概念。所谓“对象”就是对客观存在事物的一种表示，是包含现实世界物体特征的抽象实体，是物体属性和行为的一个组合体。从程序设计的角度看，对象是指将数据和使用这些数据的一组基本操作封装在一起的统一体，它是程序的基本运行单位，具有一定的独立性，其他对象的操作不能操作该对象隐藏起来的数据。