全国青少年信息学奥林匹克竞赛教程

Free Pascal 语言与基础算法

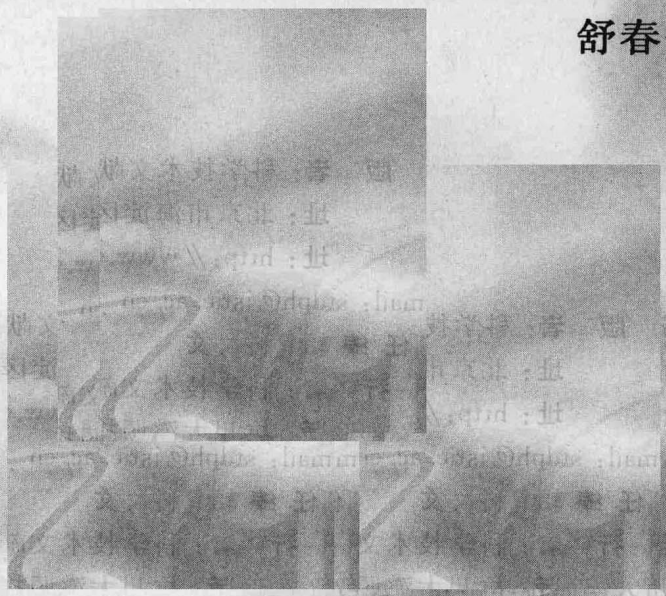
舒春平　董永建　著

科学技术文献出版社

• 全国青少年信息学奥林匹克竞赛教程

Free Pascal语言与基础算法

舒春平　董永建　编著



科学技术文献出版社

图书在版编目(CIP)数据

Free Pascal 语言与基础算法/舒春平，董永建主编. —
北京：科学技术文献出版社，2008.6
ISBN 978-7-5023-5560-9

Ⅰ. F... Ⅱ. ①舒...②董... Ⅲ. PASCAL
语言—程序设计
Ⅳ. TP312

中国版本图书馆 CIP 数据核字(2008)第 061002 号

出　版　者：科学技术文献出版社
地　　址：北京市海淀区西郊板井农林科学院农科大厦 A 座 8 层/100089
网　　址：http://www.stdph.com
E-mail：stdph@istic.ac.cn
责任编辑：科　文
发　行　者：科学技术文献出版社
排　　版：杭州大漠照排印刷有限公司
印　刷　者：杭州浙大同力教育彩印有限公司
版(印)次：2008 年 6 月第 1 版第 1 次印刷
开　　本：787×1092　16 开
字　　数：512 千
印　　张：20.5
定　　价：35.00 元(附光盘)

序　言

信息学奥林匹克竞赛是智力与应用计算机能力的比赛，选手除了要求具有扎实的基础知识、掌握计算机的程序设计语言、了解数据结构与算法外，还需要有较强的上机编程、调试程序的能力。

市面上关于程序设计与算法的书有很多，但针对信息学竞赛的书籍并不多，特别是适合入门者的书籍更少。作为教学第一线的资深教练，编者深深地体会到入门教学的不易。死板的填鸭式教学必然会让程序设计成为初学者心中“枯燥乏味”的代名词，进而导致大量的潜力选手在感受到信息学乐趣之前放弃。怎样让初学者轻松快速通过语言与基础算法关，是摆在选手和教练员面前的一道难题。

编者每年都会给高一新生上 Pascal 语言课，每年也都能看到一些同学感叹枯燥的语法与理论。“能不能让他们在模仿、实践中慢慢学会什么叫变量常量、什么叫分支循环呢?”多年的辅导经验和对这种问题的不断思考在编者心中萌发出一个想法：为广大的信息学参赛选手编一本有质量的入门教程。

现在，这本书终于问世了。在 Pascal 语言讲解方面，两位编者以坚持实用性为首的观点，绕过繁琐的语法规则，通过简单例题来解释 Pascal 语言中的概念、组成和用法，把抽象知识融合到实践中，降低学习门槛，提高学生兴趣。不仅如此，多年的教学积累还让两位编者深知入门的难点所在，从而作出更加有针对性的讲解。另外，编者还非常注重初学者的程序代码风格，强调培养选手的良好编程风格和习惯，这无疑是一个优秀选手所必备的素质。可以预见，本书对初学者学习 Pascal 语言能起到“立竿见影”的效果。

在算法讲解方面，本书的很多例题和练习题都是近几年的联赛试题，再加上附有第二部分所有练习题的测试数据的配套光盘，对于参赛选手来说是一份难得的学习和参考资料。

“万事开头难”，希望本书成为广大信息学初学者的良师益友，为进一步的学习和比赛打下坚实的基础。

信息学奥林匹克中国国家集训队教练

刘汝佳

2008 年 6 月

目 录 CONTENTS

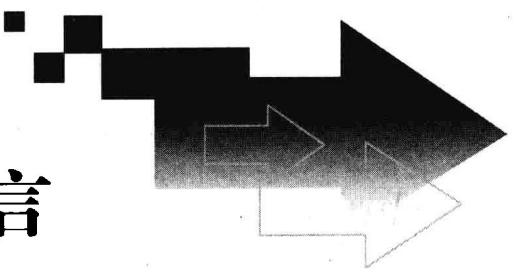
第一部分 Free Pascal 语言

第二部分 基 础 算 法

附录

第一部分

Free Pascal 语言



第一章　初识 Pascal 语言

1.1　Pascal 语言介绍

信息学奥林匹克竞赛是一项益智性的竞赛活动，其核心是考查选手的智力和使用计算机解题的能力。选手首先应针对竞赛题目的要求构建数学模型，进而构造出计算机可以接受的算法，最后编写出计算机能够执行的程序。程序设计是信息学竞赛的基本功，选手参与竞赛活动的第一步是熟练掌握一门程序设计语言，目前竞赛中允许使用的程序设计语言有 Pascal、C/C++语言。入门者以 Pascal 语言为最容易上手，选择 Pascal 语言可以节省中学阶段本来就捉襟见肘的时间，并且可以快速进入学习算法与数据结构的阶段。Pascal 语言指定的版本是 Free Pascal 2.0 以上，在讲 Free Pascal 之前，让我们先了解一下 Pascal 语言。

1.1.1　Pascal 语言概述

Pascal 语言是由瑞士苏黎世联邦工业大学的 N·沃思(Niklaus Wirth)教授于 1971 年正式完成，为纪念法国数学家 Pascal 而命名的。1975 年，对 Pascal 语言进行了修改，作为“标准 Pascal 语言”。Pascal 语言是在 ALGOL60 的基础上发展而成的，它是一种结构化的程序设计语言，可以用来编写应用程序；又是一种系统程序设计语言，可以用来编写顺序型的系统软件(如编译程序)。它功能强、编译程序简单，是 70 年代影响最大的一种算法语言。

在 Pascal 问世 30 多年来，产生了多种版本，其中影响最大的是前几年竞赛中还在使用的 Turbo Pascal。Turbo Pascal 是由美国 Borland 公司设计的一种适用于 16 位编译器的编译系统。目前竞赛中已经指定用的 Free Pascal 是一个 32 位、跨平台的专业编译器，几乎支持现有的所有操作系统，同时兼容 Turbo Pascal 中编写的程序。

1.1.2　Pascal 语言的特点

1. 它是世界上第一个结构化程序设计语言

结构化程序设计思想是程序设计发展史上的一个里程碑，结构化程序设计思想中主张去掉 goto 语句，所有程序都用三种基本结构(顺序、分支、循环)组成。Pascal 语言提供了三种基本结构的语句以及模块化(“过程”和“函数”)的功能。可以方便地书写出结构化程序。在编写程序时可以完全不使用 goto 语句和标号，这就易于保证程序的正确性和易读性。Pascal 语言强调的是可靠性、易于验证性、概念的清晰性和实现的简化。在结构化这一点上，比其他语言(如 BASIC，FORTRAN77)更好一些。

由于 Pascal 语言具有良好的结构化程序设计特性，所以它特别适合于教学，有利于培养学生良好的程序设计风格和严谨的思维。

2. 有丰富的数据类型

Pascal 提供了整型、实型、字符型、布尔型、枚举型、子界型以及由以上类型数据构成的数组类型、集合类型、记录类型和文件类型，此外，还提供了其他许多语言中所没有的指针类型。沃思的一个著名公式：“算法＋数据结构＝程序”，指出了在程序设计中研究数据结构的重要

性。丰富的数据结构和上述结构化性质，并且去掉了一些影响效率的因素（如动态数组），使得 Pascal 可以被方便地用来描述复杂的算法，使得程序编译和运行效率都很高。

3. 功能强，应用广

有些语言（如 FORTRAN66，ALGOL60）只适用于数值计算，有些语言（如 COBOL）则适用于商业数据处理和管理领域。Pascal 的功能较强，不仅是一门教学语言，而且广泛应用于编写各种系统软件和应用软件。Pascal 语言还可以用于辅助设计，实现计算机绘图功能。

4. 程序的书写格式自由

Pascal 不像有些编程语言那样对程序的书写格式有严格的规定，它允许一行内写多条语句或一条语句可以分开写在多行上，便于阅读。

5. 可移植、易推广

Pascal 是一个跨平台的专业编译器，不依赖于具体的机器。用 Pascal 编写的源程序可以在各种具有 Pascal 编译系统的机器上运行。

1.1.3　Pascal 语言程序的基本结构

Pascal 有着一组自己的记号和规则，并且对程序的结构有严格规定。我们先来看一个简单的例子：

例 1.1　如图 1-1，梯形中的阴影部分面积是 150 平方厘米，求梯形面积。

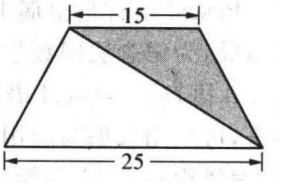


图 1-1

【分析】　已知梯形上、下底长为 15 和 25，令梯形的高为 h，则根据三角形面积为 150 平方厘米，有 150=(15 * h)/2，得 h 为 20，然后根据梯形面积公式算出梯形面积。

程序如下：

```
program ex1_1;                    //程序首部，可省略
var
  s,h,up,down : real;             //定义变量
begin
  up := 15;                       //已知上底
  down := 25;                     //已知下底
  h := 2 * 150/up;                //根据上底求出梯形的高
  s := (up+down) * h/2;           //求出梯形的面积
  writeln('s=',s );               //输出梯形的面积 s=400
end.                              //结束整个程序
```

以上程序的第一行称为程序首部，程序名称 ex1_1 可以自己命名，这一行可以省略不写。程序中“//”后面的内容称为注释，计算机不会理睬这些内容，注释可以增加程序的可读性。编写程序增加注释是一个好习惯，便于他人或自己日后阅读。注释内容还可以放在大括号中，如“{注释内容}”，一般单行内容用“//”，多行内容用“{ }”。

从第二行到最后一行为程序体（有的书本里也称分程序），由说明部分和语句部分构成。说明部分有变量（包括常量）的定义，Pascal 不允许变量未定义先使用。语句部分必须以 begin 开始，以 end. 结束，中间每条语句用分号隔开。

1.1.4　Free Pascal 语言系统的使用

目前，竞赛中指定的 Pascal 编译系统是 Free Pascal 2.0 以上的版本，了解了 Pascal 的特点及程序结构后，我们来学习 Free Pascal 2.x 软件的使用。

1. 系统的启动

运行安装目录 FPC\2.2.0\bin\i386 - win32 中的启动程序 fp.exe(默认安装时桌面上也有启动的快捷方式)启动 FP 系统，运行后屏幕上出现如图 1 - 2 所示的 FP 集成环境。

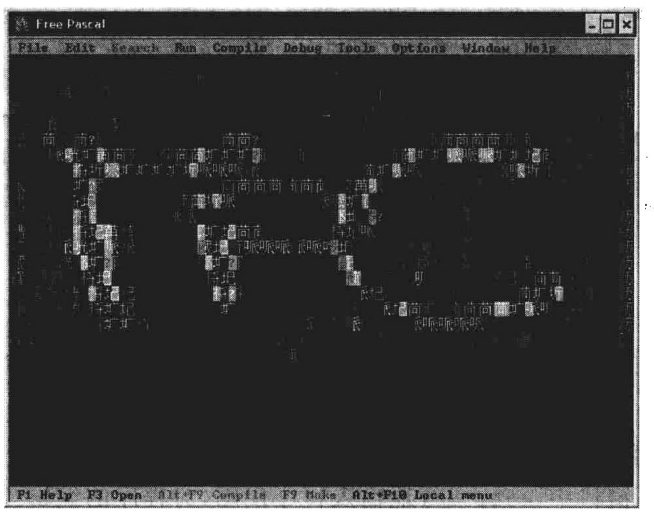


图 1 - 2

这样一片乱码，令很多入门者望而却步，导致的原因是我们操作系统“控制台窗口”中语言默认为中文。调整方法是在 FP 窗口最上面的标题栏上右键单击，在弹出的快捷菜单中选择“默认值”，接着会弹出如图 1 - 3 所示的控制台窗口属性框，在最下面的“默认代码页”的下拉菜单中选择“437 (OEM -美国)”即可，如图 1 - 4 所示。

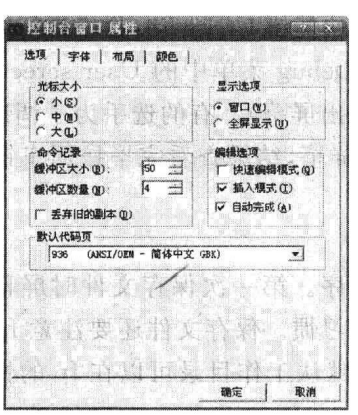


图 1 - 3

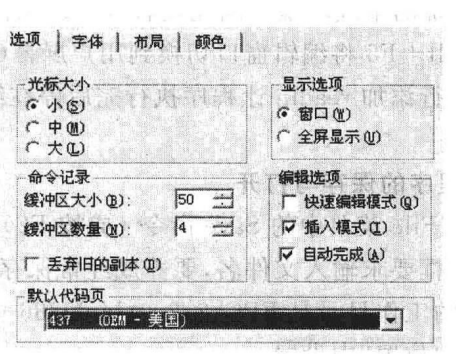


图 1 - 4

2. Free Pascal 系统集成环境简介

控制台窗口属性框中语言改成英文后，关闭并再次打开 fp 后出现如图 1 - 5 所示的界面。标题栏下方的一行为菜单栏。中间部分为编辑窗口，新建文件后，在它的编辑窗口内可以编写程序。最底部一行为提示栏，显示系统中常用命令的快捷键，如将当前正在编辑的文件存盘的

命令快捷键为 F2(平时要养成经常按这个键),编译程序的快捷键为 F9,等等。

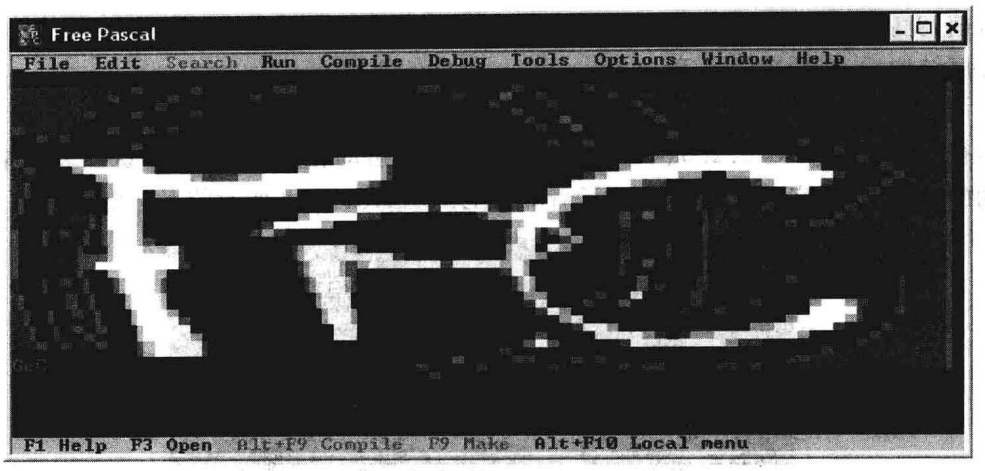


图 1-5

3. 新建程序

选择 File 菜单,执行其中的 New 命令,建立一个新的程序窗口(默认文件名为 Noname01.pas)。

4. 程序代码的输入、编辑与运行

在当前程序编写窗口中,一行一行地输入程序代码,对程序的编辑与其他文本编辑器的编辑方法类似,这里不再重复。

当程序输入完毕之后,选 Compile 菜单中的 Compile 命令(或按 F9)对程序进行编译(生成计算机能够执行的程序,扩展名为 exe)。如果程序有语法错误,会在窗口的第一行显示第一个红色错误信息;若无语法错误,窗口正中央会出现一个对话框,提示编译成功。程序编译成功后,接下来就可以运行程序了。

程序的运行可以选择 Run 菜单中的 Run 命令或按快捷键 Ctrl+F9,程序运行结束后回到 Pascal 系统的集成环境。因此查看运行结果需要选择 Debug 菜单中的 User screen 命令或按快捷键 Alt+F5 将编辑窗口切换到用户屏幕(即结果输出屏幕)。有的选手为了调试方便在程序最后一行添加 readln 让程序执行完后停在结果输出屏幕,结果查看完毕按回车键返回到编辑窗口。

5. 程序的保存与打开

选择 File 菜单中的 Save 命令(或按 F2)来保存程序。第一次保存文件时屏幕上会出现一个对话框要求输入文件名,要养成经常保存文件的好习惯。保存文件还要注意 fp 的工作目录(默认的工作目录是 FPC\2.2.0\bin\i386-win32),默认工作目录可以在 fp 的快捷方式属性的"起始位置"中设置。

打开已有源文件通过 File 菜单中的 Open 命令(或按 F3)来完成,也可以直接双击源文件来启动 fp 并加载该文件。

选手平时练习时要养成用快捷键代替鼠标操作的良好习惯,使用快捷键能大大提高编写代码的速度,从而有更多的时间来思考问题和调试程序。

1.2　简单程序设计

无论做什么事情，都要有一定的方式方法与处理步骤，所谓“没有规矩无以成方圆”。计算机程序设计比日常生活中的事务处理更具有严谨性、规范性和可行性。为了使计算机有效地解决实际问题，必须将处理步骤编排好，用计算机能理解的计算机语言编写成“序列”，让计算机自动识别并执行这个“序列”，达到解决实际问题的目的。将处理问题的步骤编排好，用计算机语言组成序列，就是常说的编写程序。在 Pascal 语言中，执行每条语句都是由计算机完成相应的具体操作，编写程序是利用 Pascal 语句的功能来实现预定的处理要求。“千里之行，始于足下”，我们从简单程序学起，逐步了解和掌握怎样编写程序。

在学习 Pascal 语言之前，让我们绕过那些繁琐的语法规则细节，通过一些简单的例题，来熟悉程序的基本组成和基本语句的用法，选手刚接触编程时，多动手模仿是一条捷径。

例 1.2　在屏幕上输出“Hello World!”。

程序如下：

```
program ex1_2;
begin
  write('Hello World!');
end.
```

通过这个简单程序的学习，希望大家在程序设计的学习中能有一个良好的开端。程序中的 write 是一条输出语句，它能命令计算机在屏幕上输出括号内的内容，其中单引号内的部分将被原样输出。大家可以试试没有引号的情况，会出现什么现象？

例 1.3　已知一位小朋友的电影票价是 10 元，计算 x 位小朋友的总票价是多少？

【分析】　假设总票价用 y 来表示，则这个问题可以用以下几个步骤来实现：

① 输入小朋友的数目 x；

② 用公式 y=10 * x 计算总票价；

③ 输出总票价 y 的值。

程序如下：

```
program ex1_3;                    //程序首部
var
  x,y : integer;                  //说明部分(定义变量)
begin
  readln(x);                      //小朋友的数目
  y := 10 * x;                    //计算总票价
  writeln('total=',y);            //输出总票价
end.
```

本题程序结构完整，从中可看出一个 Pascal 程序由三部分组成：

(1) 程序首部

由保留字 program 开头，后跟一个程序名(如 ex1_3)，其格式为：program 程序名。

程序名由选手自己取，它的第一个字符必须是英文字母，其后的字符只能是字母或数字和

下划线组成，程序名中不能出现运算符、标点符和空格等非法字符。

(2) 说明部分

说明部分以保留字 var 开头。程序中所用的常量、变量等必须先定义后使用。例 1.3 程序中 x,y : integer;是变量定义，x,y 被定义成整数类型的变量。只有被定义为某一类型的变量，在程序中才能将与变量类型所允许的值赋给该变量。

(3) 语句部分

语句部分指由保留字 begin (开始)至 end. (结尾)之间的语句系列，是解决问题的具体处理步骤，也是程序的执行部分。

不管是程序的哪个部分，每条语句末尾都必须以分号(;)结束，但允许最接近 end 的那条语句末尾的分号省略；程序结束语句 end 末尾必须有句号(.)，它是整个程序的结束标志。

程序中大括号"{…}"之间的部分或//后面的部分为注释部分，单行注释一般用//符号，多行语句注释用大括号更方便(不然每行语句前都要加//)。

程序的结构可用如下的示意图来表示：

```
program 程序名;          程序首部
常量说明; (const)      ┐
…                       ├ 说明部分
变量说明; (var)         ┘
begin
  语句 1;               ┐
  …                     ├ 程序体 (主程序)语句部分
  语句 n;               ┘
end.
```

图 1-6　程序的结构

把处理问题的步骤编成按从上到下顺序执行的程序，是简单程序的基本特征。再来分析下面两道例题的程序结构，同时继续学习基本语句。

例 1.4　有一牧场，牧场上的牧草每天都在匀速生长，这片牧场可供 15 头牛吃 20 天，或供 20 头牛吃 10 天，那么，这片牧场每天新生的草量可供几头牛吃 1 天？

【分析】　解决这类问题的关键是利用牛吃的草量，最终求出这片牧场每天新生的草量。设 1 单位的草量为 1 头牛 1 天所需的草量，于是 15 头牛 20 天所食的草量为 300 单位(包括这 20 天内的新生草量)，20 头牛 10 天所食的草量为 200 单位(包括这 10 天内的新生草量)，两者的差值即为 10 天内的新生草量。

程序如下：

```
program ex1_4;                    //程序首部
var
  s1, s2, s3 : integer;           //说明部分
begin
  s1 := 15 * 20;                  //15 头牛 20 天所食的草量
  s2 := 20 * 10;                  //20 头牛 10 天所食的草量
  s3 := (s1-s2) div (20-10);      //每天新生的草量单位数
```

```
  writeln(s3);                    //1 单位为 1 头牛 1 天所食的草量
end.
```

“ :＝ ”是赋值符号，赋值语句的格式为：变量 :＝ 表达式。

赋值语句的作用是将 :＝ 右边表达式的值赋给它左边的变量，也就说让变量的值等于表达式的值。

write 是输出语句，输出语句有三种格式：

(1) write(输出项 1,输出项 2,…);　　//执行输出后光标不换行

(2) writeln(输出项 1,输出项 2,…);　　//执行输出后光标换到下一行

(3) writeln。　　//什么都不输出立即换行

write 语句括号内的部分都是要输出的输出项，输出项是多项时各项之间要用逗号隔开。如果输出项被单引号括住时，输出项内容原样输出；如果输出项是表达式，则输出的是表达式的结果，而不是表达式本身。writeln 比 write 多一个后缀 ln，ln 是 line 的缩写，即输完内容后换到下一行。

例 1.5　已知两个自然数 a,b，输出 a 除以 b 的商和余数。

【分析】　设存储商和余数的变量名是 c 和 d，类型是整数类型。

① 输入两个自然数 a,b；

② 显示两数相除的数学表达式；

③ 求出 a 除以 b 的商 c；

④ 求出 a 除以 b 的余数 d；

⑤ 在表达式右边输出商和余数。

程序如下：

```
program ex1_5;
var
  a,b,c,d : integer;
begin
  readln(a,b);                    //输入 a,b
  write(a,'/',b,'=');             //输出 a/b=,不换行,a 和 b 是输入的数
  c :＝ a div b;                  //整除运算,取商的整数部分
  d :＝ a mod b;                  //mod 运算符是求两个数的余数
  writeln(c,'…',d);               //输出后换行
  readln;                         //运行后不立即返回编辑界面,等待按回车
end.
```

readln;是一个特殊的输入语句，要求输入一个回车(换行)才能往下执行。

read 是输入语句，输入语句的一般格式为：

(1) read(变量 1,变量 2,…);

(2) readln(变量 1,变量 2,…);

(3) readln。

前两种格式都是为括号内的变量读取数据，输入的多项数据之间以空格隔开，输出完毕后敲回车。若多输入了数据项(即数据个数超过变量个数)，read 语句读取完数据之后，能让后