高等院校信息技术规划教材

RAPTOR 程序设计案例教程

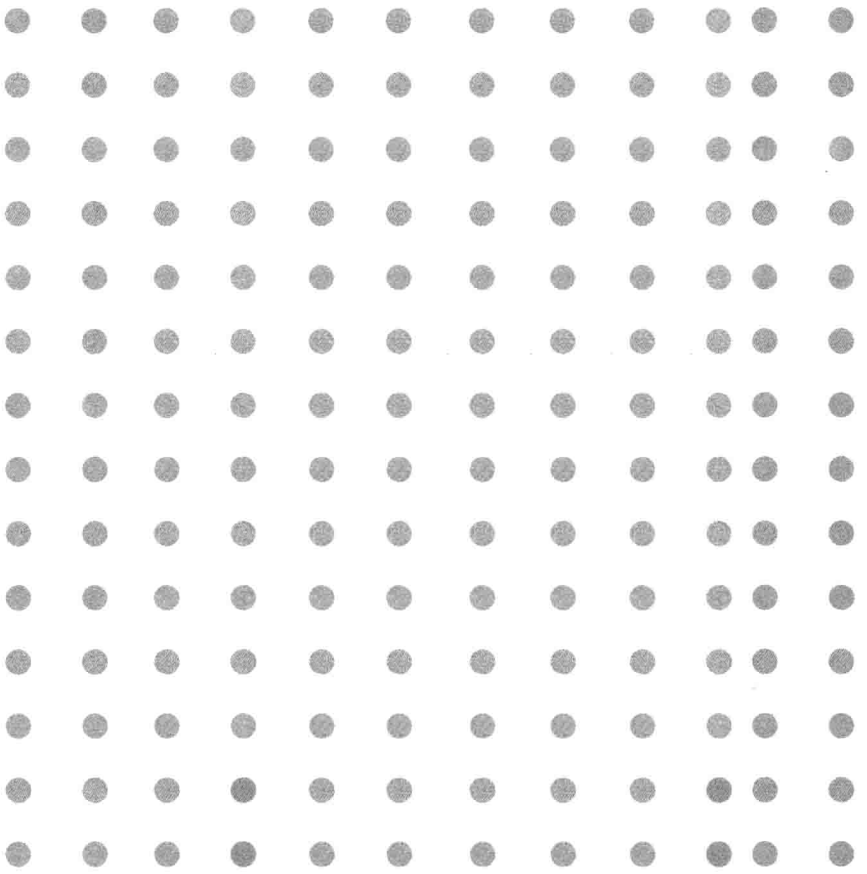
谢涛 程向前 杨金成 编著

清华大学出版社

高等院校信息技术规划教材

RAPTOR
程序设计案例教程

谢涛 程向前 杨金成 编著



清华大学出版社
北京

内 容 简 介

本书从技术角度出发介绍可视化编程工具 RAPTOR 的基础应用。全书由 8 章组成，主要内容包括 RAPTOR 简介、基本程序设计、数组、子图与子程序的应用、图形程序设计、视窗交互程序设计、I/O 程序设计和综合应用。书中案例大部分为可拓展的设计型实验，取材于学生自选并实现的可视化编程作业。本书着眼于以学生为学习主体精神指导下的实践与创新活动，充分体现现代大学生思想与表达方法的多样性、可贵的创新探索、旺盛的求知欲和好奇心。为读者跨入可视化计算的大门开辟了富有趣味、简便快捷的途径。

本书可以作为“大学计算机”、“计算思维导论”和“计算机科学导论”课程的配套实验教材，也可以独立设课，还可以供自学者学习参考。

图书在版编目(CIP)数据

RAPTOR 程序设计案例教程/谢涛，程向前，杨金成编著. --北京：清华大学出版社，2014
高等院校信息技术规划教材
ISBN 978-7-302-36394-1

Ⅰ. ①R…　Ⅱ. ①谢… ②程… ③杨…　Ⅲ. ①程序设计－高等学校－教材　Ⅳ. ①TP312

中国版本图书馆 CIP 数据核字(2014)第 099164 号

责任编辑：焦　虹　战晓雷
封面设计：常雪影
责任校对：焦丽丽
责任印制：刘海龙

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　**邮　　编**：100084
社 总 机：010-62770175　　**邮　　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
课件下载：http://www.tup.com.cn，010-62795954
印 装 者：三河市李旗庄少明印装厂
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张**：16　　**字　　数**：369 千字
版　　次：2014 年 9 月第 1 版　　**印　　次**：2014 年 9 月第1次印刷
印　　数：1～2000
定　　价：34.50 元

产品编号：054383-01

前言 foreword

RAPTOR作为一种可视化编程工具，在十年前就已经问世，而在此之前国外的诸多学校中的学者也设计过一些同类的教学工具。而本书作者是在进行大学计算机课程改革探索和研究的过程中，于2011年偶然注意到这一工具的。在将RAPTOR用于大学计算机课程教学的初期，作者认识到现在的大学新生，虽然大部分不具备熟练的程序设计基础，但对使用流程图仿真方式进行简单的程序设计并替代静态的流程图和伪代码进行基础算法训练，表现出很大的兴趣和高涨的热情。而更大的变化出现在学生了解和掌握了图形和视窗交互基础后，这种兴趣和热情转向了更高的自主算法学习和设计创新的层次。

在此过程中，作者注意到使用流程图仿真的方式进行面向过程的程序设计，很容易为大部分大学新生所掌握，学习的难点在于程序设计中的模块化问题；而图形程序设计也较容易为学生理解，大学新生会主动运用已有的解析几何的知识进行程序设计和问题求解；而具有挑战性和趣味性的问题是如何掌握视窗下的交互编程，因为只有掌握了交互编程，学生才有可能进行游戏和博弈类问题的视窗交互程序设计和问题求解。

必须承认，对程序设计的初学者进行如此庞杂的程序环境的教学和训练是一项严峻的挑战。这在国内现有的大学计算机教学体系中，也属于全新的教学尝试。

但是，在教学探索中作者惊讶地发现，现代大学生对掌握这样一个程序环境体系充满热情和兴趣，在学习的过程中，对教师所作的各种设定提出了疑问和挑战，并在完全自主的学习氛围下，取得了重要的进展和成果。

为解决学生在使用RAPTOR过程中所提出的问题，本书从技术角度出发介绍可视化编程工具RAPTOR应用基础。全书共8章，内容如下：RAPTOR简介(介绍RAPTOR的安装环境和简单应用方法)；基本程序设计(介绍基本程序概念，包括变量、常量、运算符及顺序、分支和循环结构)；数组(介绍一维和二维数组、字符串等

的基本应用);子图与子程序(介绍程序模块化的基本思想与原则);图形程序设计(进行简单矢量图绘制和位图的应用);视窗交互程序设计(了解和掌握鼠标、键盘的阻塞和非阻塞过程的应用方法);扩展 I/O(文件输入和输出,图形结果的输出,视窗环境下的文字输入等);综合应用(科赫雪花线绘制、可视化排序、图形界面下的无向图输入、RAPTOR 绘图程序等)。附录 A 是为将本书的设计案例转换成主流程序设计语言 C# 提供帮助信息;附录 B 是为 RAPTOR 程序编写 DLL 插件提供参考;附录 C 为术语对照表,可帮助读者查阅 RAPTOR 的联机帮助。

书中案例大部分为可拓展的设计型实验,取材于学生自选并实现的可视化编程作业。本书着眼于以学生为学习主体的实践与创新活动,充分体现现代大学生的思想与表达方法的多样性、可贵的创新探索、旺盛的求知欲和好奇心。为读者跨入可视化计算的大门开辟了简便快捷的途径。

本书由谢涛、程向前负责全书的构架设计与文稿编写,杨金成负责全书案例的设计、改进与调试。本书中的部分案例选取或参考了 2012—2013 年西安交通大学少年 103,104,111,112,113,114 班的"计算概论"课程和软件学院 2013 级新生"计算机应用技术基础"课程自选作业中所提交的程序。西安交通大学教务处拔尖办为少年班"计算概论"课程的教学改革提供了项目支持,作者在此表示衷心感谢。

由于时间原因,本书在文字和案例上难免存在瑕疵,恳请读者批评指正。

作　者

2014 年 5 月于西安交通大学

目录 contents

第1章 chapter 1

RAPTOR 简介

学习目标与设问：

- RAPTOR 是什么
- RAPTOR 有哪些特点
- RAPTOR 的安装步骤
- 如何编写一个最简单的 RAPTOR 程序

本书将介绍一门新的编程语言 RAPTOR。RAPTOR 是一种什么样的语言呢？和其他程序设计语言相比，它有什么特点呢？请跟随本书一起走进 RAPTOR 的世界吧。

1.1 RAPTOR 是什么

RAPTOR(the Rapid Algorithmic Prototyping Tool for Ordered Reasoning，用于有序推理的快速算法原型工具)是一款基于流程图的高级程序语言算法工具。它是一种可视化的程序设计环境，为程序和算法设计的基础课程的教学提供实验环境。使用 RAPTOR 设计的程序和算法可以直接转换成 C++ 、C# 和 Java 等高级程序语言，这就为程序和算法的初学者架起了一条平缓、自然的学习阶梯。

1.1.1 为什么要使用 RAPTOR

刚刚迈进大学校门的新生，大多数没有接触过程序设计，尤其对于以前很少用计算机的同学来说，程序设计更是遥不可及。已经学过程序设计相关课程的同学也往往对程序设计怀有不解和畏惧。为什么会出现这种普遍现象呢？笔者认为，有以下几个原因导致这一结果。

(1) 程序设计对相当一部分新生来讲是新生事物，其对学生有吸引力，但常见的编码化程序语言复杂的语法经常令初学者望而生畏。

(2) 进行程序设计开发所使用的传统开发工具相对来说比较复杂，而且不形象、不直观，成为初学者在使用上的瓶颈。

(3) 受外界环境和开发环境的影响，大多数学生会认为，程序设计只是专业的开发人员才能胜任的，所以主观上的积极性也极易受到影响，进而更加强化了“程序难学”这一

思维定势。

RAPTOR 的出现给程序设计的初学者带来了福音，它既没有复杂的语法，又提供了更直观的图形化界面，使没有任何程序设计基础的同学都能在很短的时间里迅速编写出简洁明了并且可以正常运行的程序，更能激发学生学习程序的热情。请先看一个非常简单的 RAPTOR 程序。

例 1-1 利用 RAPTOR 编写程序，输出“Hello, RAPTOR!”。

这是一个最简单的 RAPTOR 程序，只需要在开始符号 Start 和结束符号 End 之间添加输出语句，完成题目所要求的字符串“Hello, RAPTOR!”的输出即可。RAPTOR 中有专门的输出语句，并且配有输出提示，如图 1-1 所示，使学生很容易地就能完成输出工作。编辑以后的程序流程图如图 1-2 所示。

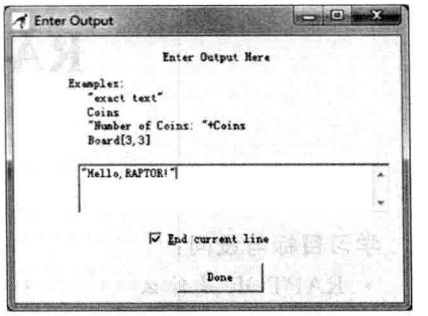


图 1-1 RAPTOR 的输出语句编辑对话

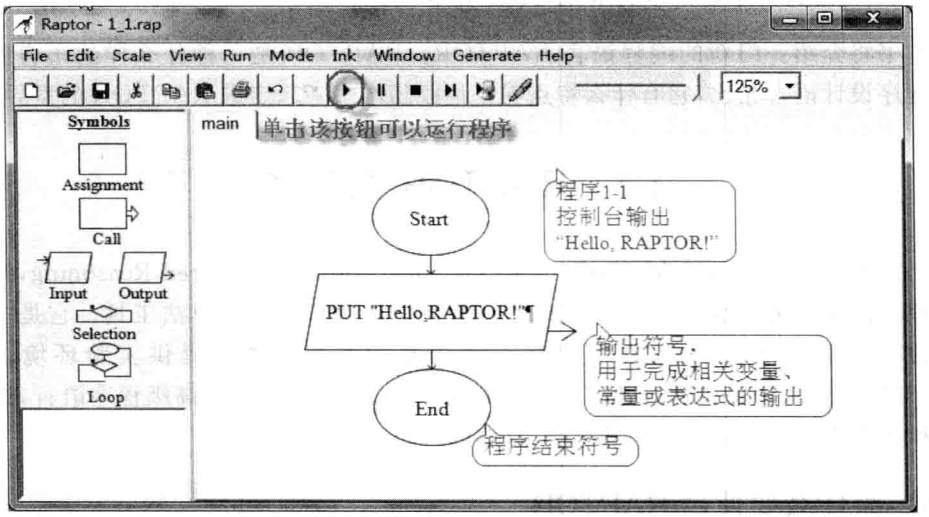


图 1-2 显示在 RAPTOR 工作区的程序流程图

图 1-2 所显示的程序运行结果如图 1-3 所示。

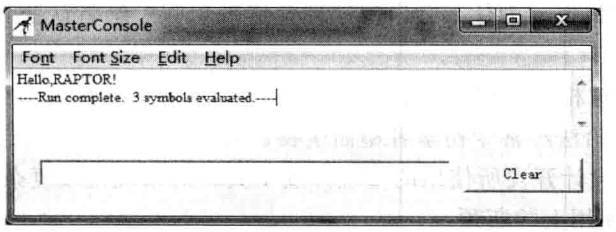


图 1-3 显示在主控制台(MasterConsole)的程序输出结果

在图 1-2 所显示的程序例子中，学生只需写一个语句，语句书写过程还有提示，而且整个程序设计过程是在图形界面中完成的，避免了复杂、枯燥的语法，即使没有任何程序设计基础的学生也能够轻易完成。相信各位读者看后一定感觉耳目一新。

RAPTOR 克服了非可视化环境的句法困难和缺点。它允许学生用连接基本流程图符号来创建算法，然后可以在其环境下直接调试和运行算法，包括运用单步执行或连续执行的模式。该环境能够直观地显示当前执行符号所在的位置，以及所有变量的变化过程。此外，RAPTOR 提供了简单图形库。学生不仅可以实现算法的可视化，而且可以实现要求解的问题本身的可视化。

RAPTOR 是一种基于流程图的可视化程序设计环境。流程图是一系列相互连接的图形符号的集合，其中每个符号代表要执行的特定类型的指令。符号之间的连接决定了指令的执行顺序。

使用 RAPTOR 主要基于以下几个原因：

- RAPTOR 开发环境可以在最大限度地减少语法要求的情形下，帮助用户编写出正确的程序。
- RAPTOR 开发环境是可视化的。RAPTOR 程序可以一次执行一个图形符号，以便帮助用户跟踪其指令流的执行过程。
- RAPTOR 是为提高易用性而设计的，用户可用它与其他任何编程开发环境进行复杂性比较。
- 使用 RAPTOR 所设计的程序的调试和报错消息更容易被初学者理解。
- 使用 RAPTOR 的目的是进行算法设计和运行验证，所以避免了重量级编程语言（如 C++ 或 Java）的过早引入给初学者带来的学习负担。

RAPTOR 让学生在其环境内执行算法，而无须单独编译和执行。这意味着，程序逻辑不必在传统程序设计的文字环境中进行设计和调试，可以直接在可视化表达环境中进行，从而也避免了使用多种工具时带来的学习负担。

对学生而言，RAPTOR 并没有强制要求对程序设计目标进行自上而下的分解，而是让学生逐步开发代码。此外，RAPTOR 中包含一维和二维数组、文件、字符串和一个允许用户交互的图形库。因此，与以前的工具相比，RAPTOR 能够让学生创造出更有趣的算法。

1.1.2 RAPTOR 的特点

RAPTOR 的特点如下：

① 语言简单、紧凑，灵活（6 个基本语句/符号）。使用流程图形式实现程序设计，使得初学者无须花费太多时间，就可以进入问题求解的实质性算法学习阶段。

② 具备基本运算功能，18 种运算符可以实现大部分基本运算。

③ 具备基本的数据类型与结构：数值、字符串和字符 3 种数据类型，一维及二维数组。组合以后，可以实现大部分算法所需要的数据结构，包括堆栈、队列、树和图。

④ 具有严格的结构化的控制语句。

⑤ 语法限制宽松，程序设计自由度大。例如，在一个数组中，可以存在不同的数据类

型，使得数据库类的记录实现有了可能。

⑥ 可移植性好，程序的设计结果可以直接执行，也可以转换成其他高级语言，如 C++、C#、Java 和 Ada 等。

⑦ 程序的设计结果可以编译成为可执行文件直接运行。

⑧ 支持图形库应用，可以实现计算问题的图形表达和图形结果输出。

⑨ 支持面向过程和面向对象的程序和算法设计。

⑩ 具备单步执行、断点设置等重要的调试手段，便于快速发现问题和解决问题。

1.2 RAPTOR 安装

RAPTOR 是一款免费的工具，可以从 RAPTOR 官方网站 http://raptor.martincarlisle.com 中获取。该网站共提供了两种版本的 RAPTOR，一种是安装版，当前最新版本是 2012 版；另一种是便携版本（也称为绿色版本），当前最新版本是 4.0 版。下面以安装版为例，介绍 RAPTOR 的安装。

从 RAPTOR 官方网站下载的 RAPTOR 安装版，文件名称为 raptor_2012.msi，将其存放于 D 盘根目录，如图 1-4 所示。双击运行该文件，出现安装界面，如图 1-5 所示，按提示选择默认选项完成安装。

图 1-4 RAPTOR 安装程序名称

安装完成后，在程序菜单中就会出现 RAPTOR，为了操作方便，建议在桌面建立 RAPTOR 的快捷方式。选择菜单“程序”→RAPTOR 命令，出现 RAPTOR 界面。一般应用界面主要包括两部分：程序设计界面（Raptor）和主控制台界面（Master Console），分别如图 1-6 和图 1-7 所示。主控制台界面用于显示程序的运行结果和错误信息等；程序设计界面主要用来进行程序设计。至此，RAPTOR 的安装操作完成。

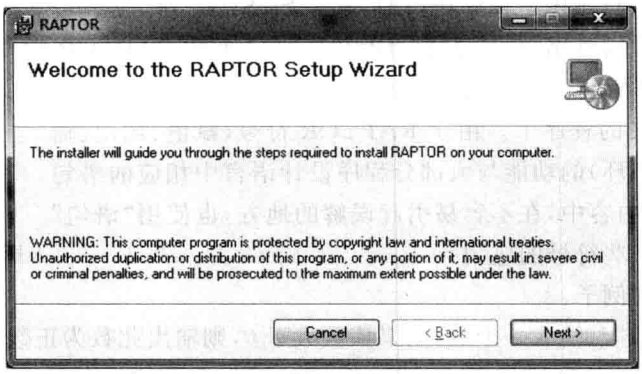


图 1-5 RAPTOR 安装主页面

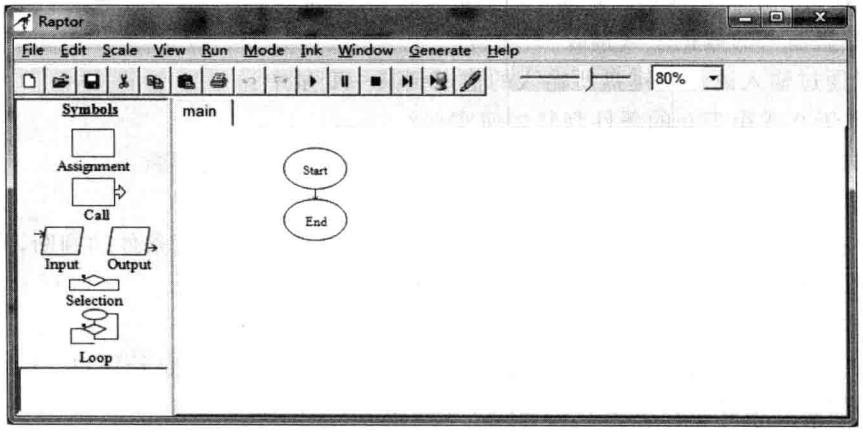


图 1-6 程序设计窗口

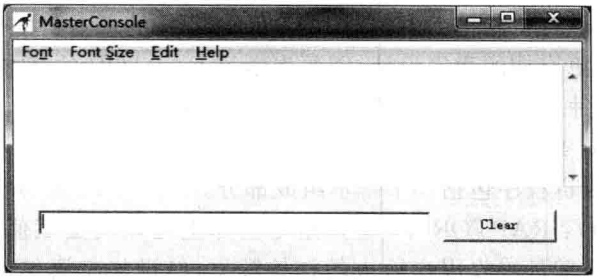


图 1-7 主控制台窗口

1.3 RAPTOR 基本程序环境与简单应用

本节介绍 RAPTOR 基本程序环境和简单应用。

RAPTOR 程序是一组连接的符号，表示要执行的一系列动作。符号间的连接箭头

确定所有操作的执行顺序。程序执行时，从开始(Start)符号起步，并按照箭头所指方向执行程序。程序执行到结束(End)符号时停止。最小的 RAPTOR 程序(什么也不做)如图 1-8 所示。在开始和结束符号之间插入一系列 RAPTOR 符号，就可以创建有意义的程序了。由于 RAPTOR 符号(赋值、调用、输入、输出、选择和循环)的功能与大部分程序设计语言中相应的语句或指令相当，后续内容中，在不容易引起误解的地方，也使用“语句”或“指令”来描述这些符号的应用。

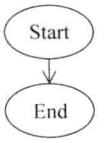


图 1-8 开始和结束符号

请先看下面的例子。

例 1-2 请从键盘输入一个数，如果该数大于 0，则输出此数为正数的信息；若该数小于 0，则输出此数为负数的信息。重复这样的过程，直到输入的数为 0，则结束程序的运行。

解：这是一个非常简单的判断正负数的程序，要完成该程序，需要考虑以下几个问题：

- 键盘输入的数据怎么接收？

可以通过输入语句将键盘所输入的值送到某一变量中。

- 大于 0 或小于 0 的条件判断如何实现？

可以通过选择结构 Selection 来实现大于 0 或小于 0 条件的判断。

- 循环如何实现，如何结束？

循环在 RAPTOR 中可通过 Loop 来完成，Loop 中有循环执行条件的判断，只要满足退出条件(本例中键盘输入值为 0)，即结束循环的执行。

- 如何输出信息？

RAPTOR 中专门提供了输出语句，可利用它输出相关提示信息给用户。

本例的流程图和运行结果分别如图 1-9 和图 1-10 所示。

1.3.1 基本符号

看过上面两个例子后，相信大家已对 RAPTOR 程序的基本结构有了些许感性认识，为了能实现更复杂、更有趣的程序设计，需要先学习 RAPTOR 的基本符号。

RAPTOR 有 6 种基本符号，每个符号代表一个独特的指令类型。基本符号如图 1-11 所示。这里首先对前 4 种符号作简单介绍，Selection 和 Loop 会在以后的章节中介绍。

一个典型的计算机程序包括 3 个基本组成部分。

(1) 输入(Input)：RAPTOR 中，输入指令实际上是以变量赋值的形式完成的。初学者使用输入语句从键盘接收用户输入的初始数据，经过加工语句处理后，再通过输出语句输出到计算机屏幕上。此外，还可以通过文件完成输入，这种输入方式提高了程序设计和运行的自动化程度，减少了用户与程序的交互，使程序可以真正解决工程和科学研究问题。在 RAPTOR 图形窗口中，输入还可以接收来自鼠标的操作指令。

(2) 处理(Process)：通过操作数据值来完成任务。处理语句主要包含了各种运算与赋值操作，负责将初始数据加工成为运算结果，或者产生运算过程中的各类中间结果。在加工过程中，需要关注各种数据的类型，因为加工操作与数据类型紧密相关。例如，可以将两个数值相乘得到乘积；但是，若将两个人名直接相乘，则得不到任何有意义的结果。

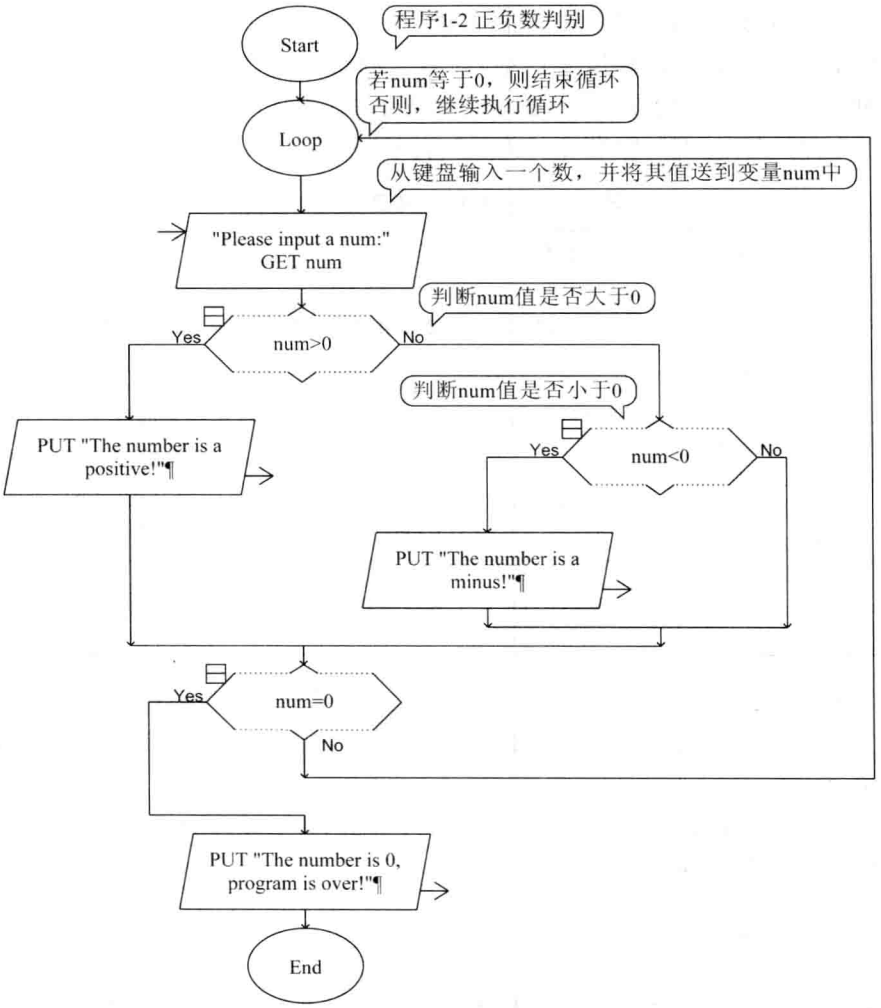


图 1-9 正负数判别程序流程图

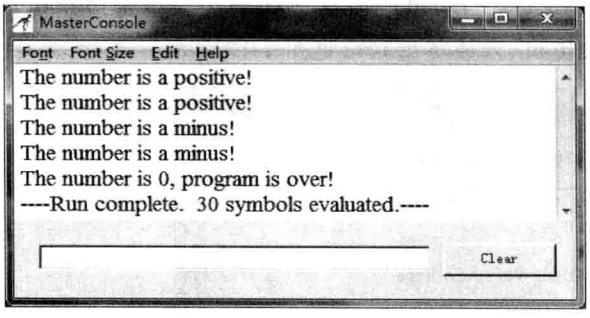


图 1-10 正负数判别主控制台显示结果

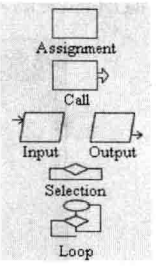


图 1-11 RAPTOR 的 6 种基本符号

(3) 输出(Output)：显示(或保存)加工处理后的结果。多数初学者会将计算结果直接在屏幕上显示输出。但输出语句也可以将计算结果输出到文件，保存在文件中的计算