

QBasic

游戏软件实例与开发

鲍居武 徐 新 编著
康振祯 潘玉保 审校



清华大学出版社

3.87424
C125

7387424

125

933254

6-17



阅览室清

10358369

QBasic 游戏软件实例与开发

鲍居武 徐新 编著
康振祯 潘玉保 审校

清华大学出版社

(京)新登字 158 号

内 容 摘 要

许多人认为计算机编程是件复杂和烦人的事情,但是本书将为你展示计算机程序娱乐性的一面。本书的程序是智慧与娱乐的统一体。通过编制游戏和其它图形程序,你不仅能够获得视觉上的巨大享受,而且还能学到一些编制游戏程序的知识。本书共有十二章,每章都详细介绍一个单独的游戏程序。书中的许多程序都可以作为模板去编制新的程序;许多子程序你都可以在自己的程序中直接调用。

本书内容丰富,资料新颖,是指导读者应用计算机技术设计计算机游戏软件的参考书。

版权所有,翻印必究。

本书封面贴有清华大学出版社激光防伪标签,无标签者不得销售。

图书在版编目(CIP)数据

QBasic 游戏软件实例与开发/鲍居武,徐新编著. —北京:清华大学出版社,1995.1
ISBN 7-302-01753-0

I. Q… II. ①鲍… ②徐… III. BASIC 语言-程序设计 IV. TP312

中国版本图书馆 CIP 数据核字(95)第 00868 号

出 版 者: 清华大学出版社(北京清华大学校内,邮编 100084)

责 任 编辑: 洪德忠

印 刷 者: 北京市海淀区清华园印刷厂

发 行 者: 新华书店总店北京科技发行所

开 本: 787×1092 1/16 印张: 16.25 字数: 404 千字

版 次: 1995 年 4 月第 1 版 1995 年 4 月第 1 次印刷

书 号: ISBN 7-302-01753-0/TP·767

印 数: 0001—2000

定 价: 18.00 元

前　　言

许多人认为计算机编程是件复杂和烦人的事情,但是本书将为你展示计算机程序娱乐性的一面。本书的程序是智慧与娱乐的统一体。通过编制游戏和其它图形程序,你不仅能够获得视觉上的巨大享受,而且还能学到一些编制游戏程序的知识。

本书共有十二章,每章都详细介绍一个单独的游戏程序。每章结构是这样的:首先介绍如何使用本程序,然后根据程序结构分段介绍,并附有完整的源程序。本书以上述12个游戏软件为实例,详细分析了游戏软件的实现过程和技巧。这一切对游戏软件的开发,特别是对用QBasic语言开发游戏软件具有参考价值和指导意义。

本书的许多程序都可以作为模板去编制新的程序,书中的许多子程序你都可以在自己的程序中直接调用。本书所使用的QBasic语言,在MS-DOS 5.0以上版本的系统中都附有它的运行环境。

总之,本书内容丰富,资料新颖,是指导读者应用计算机技术设计计算机游戏软件的参考书。

参加本书编写的还有徐劲祥、王天君、孙毅、王存山、陈阳泉、魏敏、闫广建、徐岭、王岩杰、王斌、杨欣、李东升、王强等同志,在此表示衷心地感谢。

由于我们水平有限,书中难免有缺点和错误,敬请广大读者批评和指正,以便进一步修改。

作者于北京
1994年9月

结 构 指 南

第一章 万花筒:本程序利用具有丰富色彩的屏幕存储技术,使许多连续变化、跳动的线条充满屏幕,给人以视觉上的充分享受。

第二章 对抗运动:本游戏快速而富有趣味,其主要内容是设法阻止球的运动,并且尽量使之停留在屏幕中央。

第三章 棒球:本游戏模拟真正的棒球比赛,当你在接、发球时,你将获得在球场打球的那种感觉。

第四章 跟我来:本游戏利用 3-D 颜色条来测试游戏者重复游戏次序的能力。

第五章 配对:本游戏通过相同图案配对来测试游戏者的记忆能力。

第六章 迷宫:本游戏包括 10 种已设计好的迷宫。此外,通过内存的迷宫编辑器还可修改或重建新的迷宫。

第七章 调色板编辑器:它是一个通用的调色板编辑器,利用它可轻而易举地获得其它游戏所需要的调色板文件。

第八章 图象编辑器:它是一个可产生 256 种颜色的通用图象编辑器,利用它你可以绘制并存储其它游戏所用到的各种图象。

第九章 财宝洞:本游戏刺激而富有趣味,它并不象拉斯维加斯的自动赌钱机那样会掏空你的钱包。

第十章 空中飞猪:本游戏 14 种不同形象的飞猪轮流布满屏幕给您美的享受。

第十一章 绝处逢生:本游戏富于幻想、冒险。你将同比利一道寻找海盗藏宝的地方,同时你必须帮助比利逃出海盗窟。

第十二章 太空防御:本游戏中游戏者必须设法保护自己的基地以免遭外来飞船的攻击。

附录 A 简介了本书各程序中所用到的 QBasic 扩展器 TSR 的各种函数功能。

目 录

第一章 万花筒	1
1.1 引言	1
1.2 程序操作	1
1.3 程序细节	1
1.3.1 启动	1
1.3.2 初始化	1
1.3.3 主程序	4
1.4 源程序清单	6
第二章 对抗运动	10
2.1 引言	10
2.2 程序操作	10
2.3 程序细节	11
2.3.1 启动	11
2.3.2 TSR 检测	11
2.3.3 初始化	11
2.3.4 主程序	14
2.3.5 再玩一次选择	17
2.4 源程序清单	17
第三章 棒球	23
3.1 引言	23
3.2 程序操作	23
3.3 程序细节	24
3.3.1 启动	24
3.3.2 TSR 检测	25
3.3.3 初始化	25
3.3.4 主程序	26
3.3.5 再玩一次选择	27
3.4 源程序清单	27
第四章 跟我来	34
4.1 引言	34
4.2 程序操作	34
4.3 程序细节	35
4.3.1 启动	36
4.3.2 TSR 及鼠标检测	36

4.3.3 初始化	37
4.3.4 主程序	39
4.3.5 再玩一次选择	40
4.4 源程序清单	40
第五章 配对	49
5.1 引言	49
5.2 程序操作	49
5.3 程序细节	51
5.3.1 启动	51
5.3.2 TSR 及鼠标检测	52
5.3.3 初始化	52
5.3.4 主程序	54
5.3.5 再玩一次选择	55
5.4 子程序 QBGMPUT	55
5.5 源程序清单	56
第六章 迷宫	66
6.1 引言	66
6.2 程序操作	66
6.2.1 游戏玩法	67
6.2.2 编制迷宫	68
6.3 程序细节	70
6.3.1 启动	71
6.3.2 TSR 及鼠标检测	71
6.3.3 程序初始化	72
6.3.4 主程序	73
6.4 源程序清单	81
第七章 调色板	96
7.1 引言	96
7.2 程序操作	96
7.2.1 Undo(废除)、Copy(拷贝)、Paste(粘贴)	97
7.2.2 Load(调入)、Save(存盘)、Exit(退出)	97
7.3 程序细节	97
7.3.1 启动	97
7.3.2 TSR 及鼠标检测	98
7.3.3 初始化	98
7.3.4 主程序	100
7.3.5 PullDown 函数	103
7.3.6 GetFileList 子程序	109
7.3.7 InPutFileName \$ 函数	110

7.4 源程序清单	110
第八章 图象编辑.....	126
8.1 引言	126
8.2 程序操作	126
8.2.1 主菜单.....	127
8.2.2 编辑菜单.....	128
8.3 程序细节	129
8.3.1 启动	129
8.3.2 TSR 及鼠标检测	130
8.3.3 初始化.....	131
8.3.4 主程序.....	132
8.4 源程序清单	147
第九章 财宝洞.....	177
9.1 引言	177
9.2 程序操作	177
9.3 程序细节	177
9.3.1 启动	178
9.3.2 TSR 检测	178
9.3.3 初始化.....	178
9.3.4 主程序.....	179
9.3.5 子程序 ScrollImages	181
9.4 源程序清单	182
第十章 空中飞猪.....	192
10.1 引言.....	192
10.2 程序操作.....	192
10.3 程序细节.....	192
10.3.1 启动	192
10.3.2 TSR 检测	193
10.3.3 初始化.....	193
10.3.4 主程序.....	194
10.4 源程序清单	195
第十一章 绝处逢生.....	200
11.1 引言.....	200
11.2 程序操作.....	200
11.3 程序细节.....	203
11.3.1 启动.....	203
11.3.2 TSR 检测	204
11.3.3 初始化.....	204
11.3.4 主程序.....	208

11.4 源程序清单.....	216
第十二章 太空防御.....	235
12.1 引言.....	235
12.2 程序操作.....	235
12.3 程序细节.....	236
12.3.1 启动.....	236
12.3.2 TSR 检测	236
12.3.3 初始化.....	236
12.3.4 主程序.....	239
12.3.5 胜利后飞船的轨迹.....	241
12.3.6 再玩一次选择.....	241
12.4 源程序清单.....	242
附录 A QBasic 扩展器 TSR	249
A.1 引言	249
A.2 操作	249
A.3 细节	249
A.3.1 TSR 函数 0——键盘地址	249
A.3.2 TSR 函数 1——鼠标检测	249
A.3.3 TSR 函数 2——按钮状态检测	250
A.3.4 TSR 函数 3/4——获得/设定鼠标位置	250
A.3.5 TSR 函数 5/6——鼠标器的开/关	250
A.3.6 TSR 函数 7/8——设定鼠标限制	250
A.3.7 TSR 函数 9——文件存在/大小检测	250
A.3.8 TSR 函数 10——文件统计	251
A.3.9 TSR 函数 11——列文件	251

第一章 万花筒

1.1 引言

屏幕存储技巧在当今许多程序中经常用到。通过屏幕存储得到许多美丽的图案(如万花筒游戏)，可以帮助你减轻现代生活方式的压力。屏幕存储操作简单但使用起来却又其乐无穷。因此，本书一开始便将万花筒游戏放在篇首介绍。正如名字所说，万花筒就是将屏幕布满美丽的图案。主程序每执行100次循环，程序就在四分屏(如图1.1)与十六分屏(如图1.2)间自动切换。

1.2 程序操作

本程序仅需很少的内存，将 KALDSCPE.BAS 装入 QBasic 环境后，选择“RUN”选项，然后便可坐下来观赏。如果要中止程序运行，按“Escape”键即可。

1.3 程序细节

本节包含产生万花筒游戏的全过程，并给出“KALDSCPE.BAS”文件的全部源程序。

1.3.1 启动

编译 QBasic 程序包含有几条启动语句，这些语句为非执行语句——即它们并非计算机要执行的真正操作，而只是建立 QBasic 环境不可缺少的一部分。万花筒程序的启动语句有以下两条：

```
DEFINT A-Z           ' set default variable type to integer  
DECLARE SUB RandPal ()      ' SUB to randomize the palette
```

当 QBasic 启动时，无符号的变量是浮点数。虽然浮点数有时非常有用甚至十分重要，但用整形数进行计算或比较可以大大提高运行速度，故大多数 QBasic 启动时，利用“DEFINT”语句将浮点数转换为整形数。

下一步就该声明使调色板随机化的子程序。声明子程序的重要性在于它告诉 QBasic 这是一个子程序而不是一个变量。程序中的“()”表明“RandPal”子程序不需要任何自变量。(有关子程序的详尽介绍见本章稍后)

1.3.2 初始化

编制 QBasic 程序的下一步工作是初始化各操作。本程序的初始化语句如下：

```
SCREEN 12          ' high resolution graphics mode  
RANDOMIZE TIMER    ' seed random number generator
```

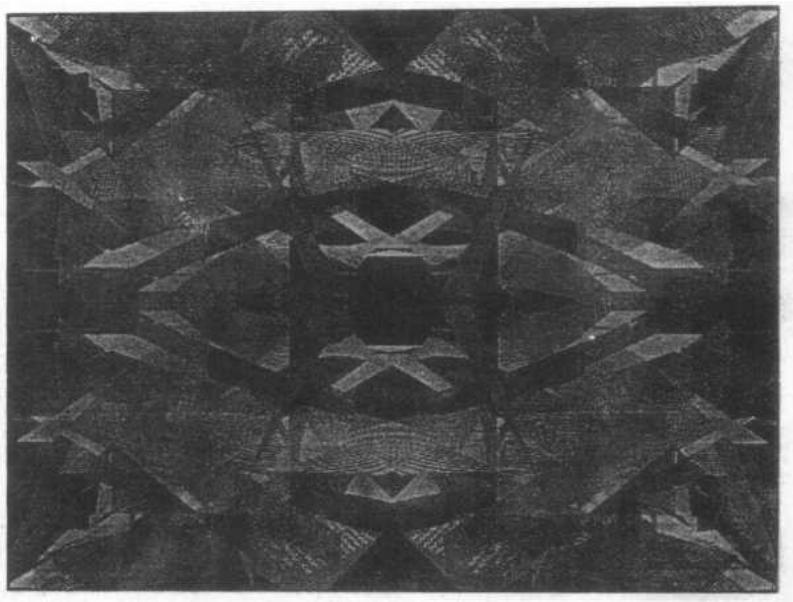


图 1.1 四分屏状态

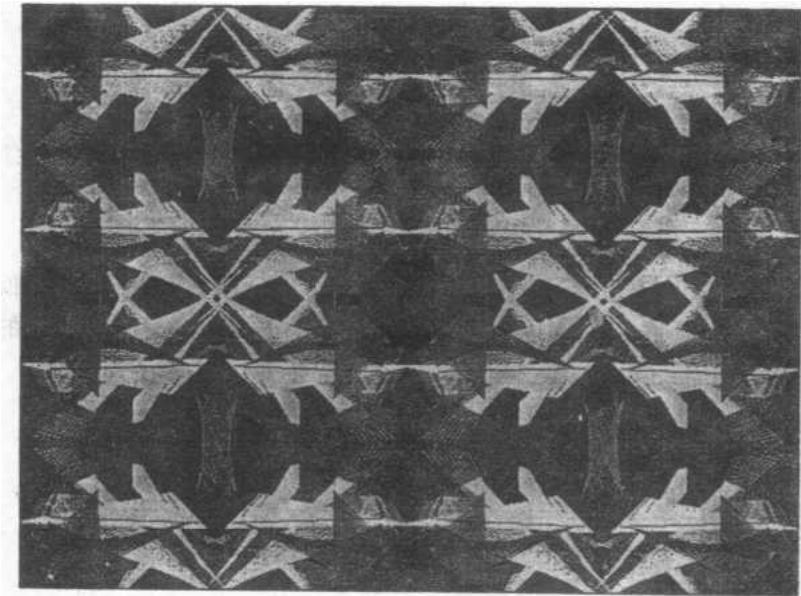


图 1.2 十六分屏状态

```
FourMode = NOT FourMode           ' set initial mode flag to -1
MaxRandomX = 319                  ' half width (starts at 0)
MaxRandomY = 239                  ' half height (starts at 0)
RandPal                           ' randomize the palette
```

屏幕控制是通过放在图形模式中的语句 SCREEN 来实现的。万花筒游戏的屏幕模式为 640×480 点阵，共有 16 种颜色。点阵是计算机屏幕上可以显现的最小点。在图形模式中，屏

幕如同一张坐标纸,坐标纸上的每一方格代表一个点阵。

首先,应将屏幕左上方的点定义为坐标原点(0,0)。由于0代表一个点,最后一个点的坐标值要小于屏幕分辨率的点数。在屏幕模式12中(图1.3)屏幕宽640(0到639),高480(0到479)。

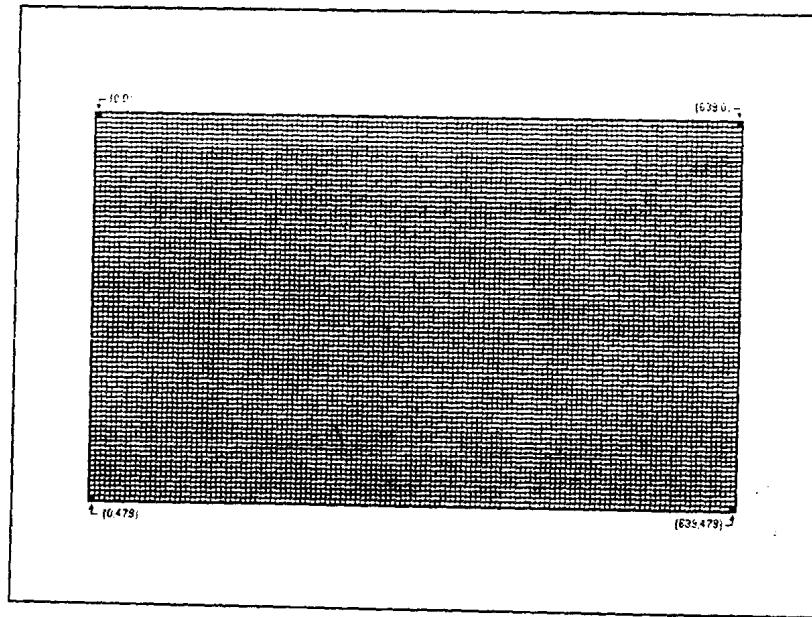


图1.3 屏幕模式12

接下来, RANDOMIZE语句初始化随机数发生器。QBasic的随机数发生器返回一个0—1的伪随机数。此数仅代表一个预定序列,而非真正的随机数。

由于万花筒游戏每次执行时开端不同,随机数发生器每次产生一个不同的序列。函数 TIMER从中途返回几秒,为确定开端提供方便之路。

其余的初始化语句是用来把原始数据分配给各有关变量。QBasic给各变量赋初值为0。变量 FourMode 是用来指明屏幕当前状态。逻辑 NOT语句将变量 FourMode 由假(0)转为真(非0),使四分屏有效。在四分屏中,屏幕分为4部分。X,Y坐标值必须填充至屏幕左上角。这样X的最大值为319,Y的最大值为239。

为了把颜色初始值分配给色彩变量,应调用子程序 RandPal。万花筒游戏在每次状态变化时都使调色板随机化。因完成这一任务的语句已放入子程序中,故可避免重复书写这些指令。子程序 RandPal 由以下语句组成:

```
SUB RandPal
    FOR WorkColor = 1 TO 15
        red = RND * 63
        green = RND * 63
        blue = RND * 63
        PALETTE WorkColor, 65536 * blue + 256 * green + red
    ' do colors 1 to 15
    ' random red intensity value (0-63)
    ' ditto for green
    ' ditto for blue
```

```

        ' set each color to random intensities
NEXT
END SUB

```

一个FOR—NEXT 结构用来把颜色值由 1 到 15 循环一遍。每种颜色的随机强化值经计算后,由 PALETTE 语句分配给各颜色。

1.3.3 主程序

程序主体由一个 DO—LOOP 结构所组成。循环不断执行直至游戏者按下“Escape”键为止。

1.3.3.1 计算第一条线

本部分用来计算万花筒游戏中第一条线的值,它由以下指令构成:

```

COLOR RND * 14 + 1           ' set current color to random (1 to 15)
X1 = RND * MaxRandomX       ' set X1 to a random (0 to max)
Y1 = RND * MaxRandomY       ' ditto for Y1
X2 = RND * MaxRandomX       ' ditto for X2
Y2 = RND * MaxRandomY       ' ditto for Y2

```

QBasic 当前颜色被 COLOR 语句设置为一随机数。通过计算第一条线的末端来保证它处于屏幕模式以内。画其它线时以此线作为参考。

1.3.3.2 步长计算

该游戏中各线发生变化这一动画效果,是通过把线的末端改变一小的步长来实现的。步长范围为 -2 到 2。这些值的计算由以下语句实现:

```

StepX1 = RND * 4 - 2          ' set StepX1 to random (-2 to 2)
StepY1 = RND * 4 - 2          ' ditto for StepY1
StepX2 = RND * 4 - 2          ' ditto for StepX2
StepY2 = RND * 4 - 2          ' ditto for StepY2

```

通过计算,一个独立的步长值赋给每条线两端点的 X 和 Y 值。

1.3.3.3 状态确定

程序中 IF—THEN—ELSE 结构被用来确定该游戏操作时的状态。用变量 FourMode 来代表屏幕的当前状态。若定它为逻辑真(非 0)则为四分屏;反之为十六分屏。

1. 四分屏状态

如前所述,四分屏时屏幕被分为四部分(如图 1.4),第一条线的末端处在屏幕左上部分。其它三条线的端点计算以第一条线为参考。从对应边减去第一条端点值,以所得值为另一条线的端点。该线与第一条线成镜象关系(如图 1.5)。

计算其余各条线端点的指令如下:

```

FOR Times = 1 TO 30           ' do 30 sets of lines
    X3 = 639 - X1             ' calculate points for
    X4 = 639 - X2             ' 4 lines
    Y3 = 479 - Y1             ' 4 X values
    Y4 = 479 - Y2             ' 4 Y values
    LINE (X1, Y1)-(X2, Y2)     ' draw the lines
    LINE (X3, Y1)-(X4, Y2)

```

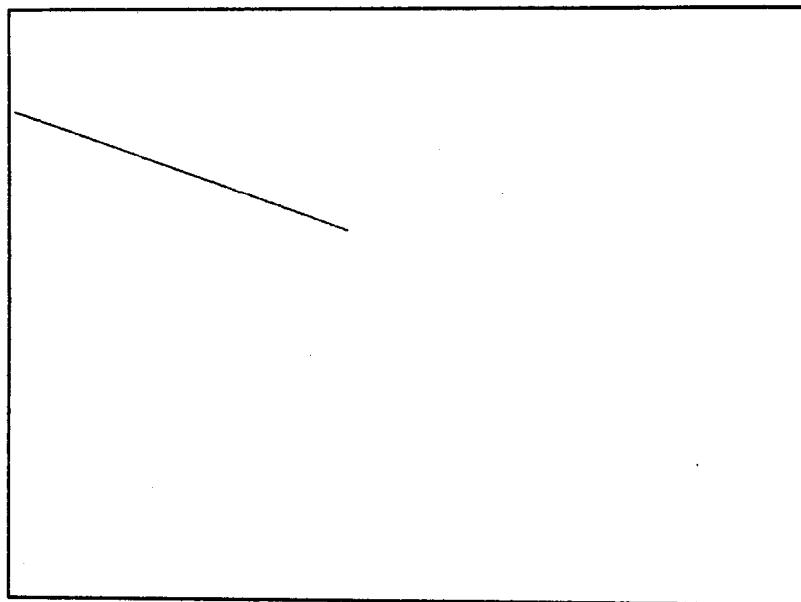


图 1.4 四分屏的第一条线

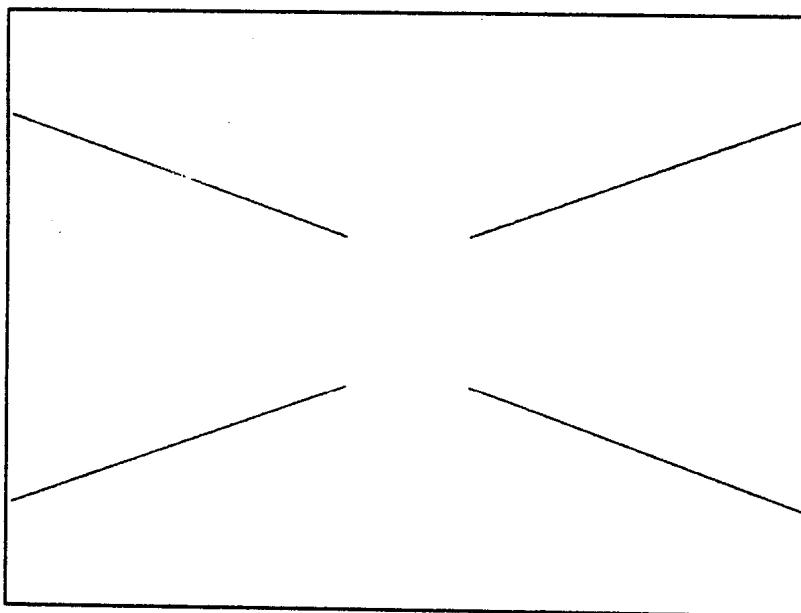


图 1.5 镜象作用

```
LINE (X1, Y3) - (X2, Y4)
LINE (X3, Y3) - (X4, Y4)
X1 = X1 + StepX1           ' adjust with step value
Y1 = Y1 + StepY1           ' ditto
X2 = X2 + StepX2           ' ditto
Y2 = Y2 + StepY2           ' ditto
NEXT
```

稍后，其它各条线同时画出。当第一组四条线画完时，第一条线的端点值由步长调节。整个过程由 FOR-NEXT 结构重复执行共画出 30 组线。

2. 十六分屏状态

十六分屏式与四分屏式基本相似，唯一不同的是屏幕分为十六部分。这样，将有更多的线被描绘出来。

1.3.3.4 模式自动开关

万花筒的模式开关是通过统计主循环的循环次数来完成的。每执行 100 次循环后屏幕模式便自动切换。

程序如下：

```

LoopCount = LoopCount + 1          ' count the number of loops
IF LoopCount = 100 THEN           ' after 100 loops
    LoopCount = 0                 ' reset loop count
    FourMode = NOT FourMode      ' toggle mode flag
    IF FourMode THEN             ' set maxes for next mode
        MaxRandomX = 319          ' half width (starts at 0)
        MaxRandomY = 239          ' half height (starts at 0)
    ELSE
        MaxRandomX = 159          ' quarter width (starts at 0)
        MaxRandomY = 119          ' quarter height (starts at 0)
    END IF
    CLS                          ' clear the screen
    RandPal                      ' randomize the palette
END IF

```

每次模式转换时,伴随着相应变量的设定,程序进行清屏并将调色板设定为新的随机值。

1.3.3.5 退出

退出是由 Escape 键是否被按下决定。程序如下：

```

KeyStroke $ = INKEY$           'get any keystrokes
IF LEN(KeyStroke $) THEN KeyVal = ASC(KeyStroke $) 'convert to an integer
LOOP WHILE KeyVal <> 27      'keep going until escape is pressed

```

若某键被按下,它的 ASCII 码被送给 KeyVal,这可以使 DO-LOOP 结构进行整形比较,其执行速度比字节比较快得多.

1.4 源程序清单

KALDSCPE.BAS 源程序清单如下。

```

RANDOMIZE TIMER           ' seed random number generator
FourMode = NOT FourMode   ' set initial mode flag to -1
MaxRandomX = 319           ' half width (starts at 0)
MaxRandomY = 239           ' half height (starts at 0)
RandPal                   ' randomize the palette
DO                         ' start of main loop
  COLOR RND * 14 + 1       ' set current color to random (1 to 15)
  X1 = RND * MaxRandomX   ' set X1 to a random (0 to max)
  Y1 = RND * MaxRandomY   ' ditto for Y1
  X2 = RND * MaxRandomX   ' ditto for X2
  Y2 = RND * MaxRandomY   ' ditto for Y2
  StepX1 = RND * 4 - 2     ' set StepX1 to random (-2 to 2)
  StepY1 = RND * 4 - 2     ' ditto for StepY1
  StepX2 = RND * 4 - 2     ' ditto for StepX2
  StepY2 = RND * 4 - 2     ' ditto for StepY2

IF FourMode THEN           ' do the appropriate mode
  FOR Times = 1 TO 30      ' four part mode
    X3 = 639 - X1           ' do 30 sets of lines
    X4 = 639 - X2           ' calculate points for
    Y3 = 479 - Y1           ' 4 lines
    Y4 = 479 - Y2           ' 4 X values
    LINE (X1, Y1) -- (X2, Y2) ' 4 Y values
    LINE (X3, Y1) -- (X4, Y2)
    LINE (X1, Y3) -- (X2, Y4)
    LINE (X3, Y3) -- (X4, Y4)
    X1 = X1 + StepX1        ' draw the lines
    Y1 = Y1 + StepY1
    X2 = X2 + StepX2
    Y2 = Y2 + StepY2
  NEXT
ELSE
  FOR Times = 1 TO 30      ' 16 part mode
    X3 = 319 - X1           ' do 30 sets of lines
    X4 = 319 - X2           ' calculate points for
    X5 = X1 + 320            ' 16 lines
    X6 = X2 + 320
    X7 = X3 + 320
    X8 = X4 + 320
    Y3 = 239 - Y1           ' 8 X values
    Y4 = 239 - Y2
    Y5 = Y1 + StepY1         ' 8 Y values
    Y6 = Y2 + StepY2
    Y7 = Y3 + StepY3
    Y8 = Y4 + StepY4
  NEXT

```

```

Y4 = 239 - Y2
Y5 = Y1 + 240
Y6 = Y2 + 240
Y7 = Y3 + 240
Y8 = Y4 + 240
LINE (X1, Y1)-(X2, Y2)      ' draw the lines
LINE (X3, Y1)-(X4, Y2)
LINE (X1, Y3)-(X2, Y4)
LINE (X3, Y3)-(X4, Y4)
LINE (X5, Y1)-(X6, Y2)
LINE (X7, Y1)-(X8, Y2)
LINE (X5, Y3)-(X6, Y4)
LINE (X7, Y3)-(X8, Y4)
LINE (X1, Y5)-(X2, Y6)
LINE (X3, Y5)-(X4, Y6)
LINE (X1, Y7)-(X2, Y8)
LINE (X3, Y7)-(X4, Y8)
LINE (X5, Y5)-(X6, Y6)
LINE (X7, Y5)-(X8, Y6)
LINE (X5, Y7)-(X6, Y8)
LINE (X7, Y7)-(X8, Y8)

X1 = X1 + StepX1           ' adjust with step value
Y1 = Y1 + StepY1           ' ditto
X2 = X2 + StepX2           ' ditto
Y2 = Y2 + StepY2           ' ditto

NEXT

END IF

LoopCount = LoopCount + 1    ' count the number of loops
IF LoopCount = 100 THEN     ' after 100 loops
  LoopCount = 0              ' reset loop count
  FourMode = NOT FourMode   ' toggle mode flag
  IF FourMode THEN          ' set maxes for next mode
    MaxRandomX = 319         ' half width (starts at 0)
    MaxRandomY = 239         ' half height (starts at 0)
  ELSE
    MaxRandomX = 159         ' quarter width (starts at 0)
    MaxRandomY = 119         ' quarter height (starts at 0)
  END IF
  CLS                      ' clear the screen
  RandPal                  ' randomize the palette
END IF

KeyStroke $ = INKEY $        ' get any keystrokes
IF LEN(KeyStroke $) THEN KeyVal = ASC(KeyStroke $)  ' convert to an integer

```