第8讲 程序设计

8－3 程序模块

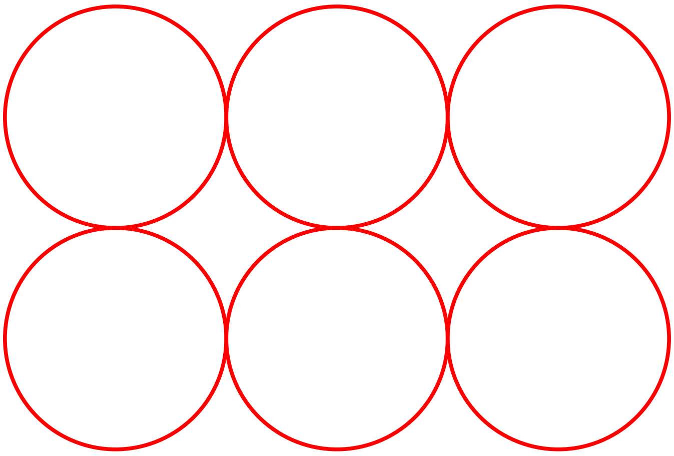
1. Block 块

Block[{x, y, ...}, expr]
Block[{x = x0, y = y0, ...}, expr]
对expr中的 {x, y, ...} 使用其局部值.Block语句的作用是保护模块外部的同名变量
{x, y, ...} 使得它们的值不受模块内部语句的影响。Block语句的效果相当于先备份
{x, y, ...} 的值, 接着赋以新值 {x0, y0, ...}, 再执行程序体expr,
最后还原 {x, y, ...} 的备份值.

例1：用 Block 定义函数输出图形

```
fa[m_, n_] := Block[{s = 2 m - 1, t = 2 n - 1},
  Graphics[{Red, Thick, Table[Circle[{i, j}], {i, 1, s, 2}, {j, 1, t, 2}]}]]
```

```
fa[3, 2]
```



2. Module 模块

Module[{x, y, ...}, expr]
Module[{x = x0, y = y0, ...}, expr]
对 expr中的 {x, y, ...} 创建局部变量.
Module语句的作用则是保护模块内部的局部变量 {x, y, ...}, 使得它们的值不受模块外部语句的影响。
每次执行Module语句之前，Mathematica都会自动创建新的变量来代替 {x, y, ...}.

例2：用 Module 定义函数计算最大公约数

```
gcd[m0_, n0_] :=
 Module[{m = m0, n = n0},
  While[n ≠ 0, {m, n} = {n, Mod[m, n]}];
  m]
```

```
gcd[105, 126]]
```

例3：观察 Block 和 Module 的区别

```
f[x_] := Block[{t}, Integrate[Exp[x * t], {t, 0, 1}]];
g[x_] := Module[{t}, Integrate[Exp[x * t], {t, 0, 1}]];
{f[x], g[x], f[t], g[t]}
```

3. With　　With 的速度比 Module 快

```
With[{x = x0, y = y0, ...}, expr]
```

将expr中的 {x, y, ...} 替换成 {x0, y0, ...}

例4：用 With 定义函数输出图形

```
fb[m_, n_] := With[{s = 2 m - 1, t = 2 n - 1},
  Graphics[{Purple, Thick, Table[Circle[{i, j}], {i, 1, s, 2}, {j, 1, t, 2}]}]]
```

```
fb[3, 2]
```

例5：Block 仅局部化值；它并不替代值.Module 创建新符号：

```
{Block[{x = 5}, Hold[x]], With[{x = 5}, Hold[x]], Module[{x = 5}, Hold[x]]}
```

```
{Hold[x], Hold[5], Hold[x$2311]}
```

例6：绘制复映射 $f(z) = z^2 + c$ 的Julia集和Mandelbrot (曼德尔布罗特) 集的图像，

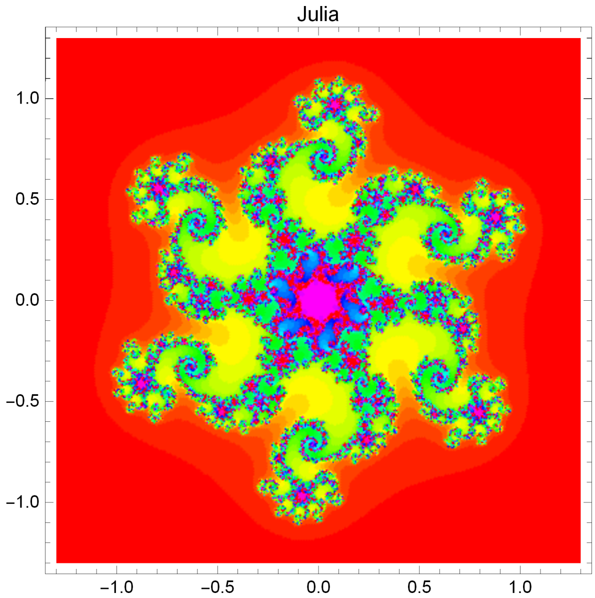
序列 $z_n = z_{n-1}^2 + c$，从 $z_0 = 0$ 开始迭代，其中Julia集是使迭代 不收敛 $z = f(z)$ 的初值 的集合，
Mandelbrot集是使迭代 不收敛的所有复数 c 的集合.

Julia：　朱莉娅　　　　　　Mandelbrot：曼德尔布罗特

```
F2[x_, y_, Cx_, Cy_, n_] := Block[{z, i = 0}, z = x + y * I;
  While[(Abs[z] < 2.0) && (i < 50), ++i; z = z^n + (Cx + Cy * I)]; Return[i]]
Julia[Cx_, Cy_, n_, xm_List, ym_List] :=
 DensityPlot[F2[x, y, Cx, Cy, n], {x, xm[[2]], xm[[3]]}, {y, ym[[2]], ym[[3]]},
  PlotPoints → 100, PlotLabel → "Julia", Mesh → False, ColorFunction → Hue]
```

```
J1 = Julia[0.27334, 0.00742, 2, {x, -1, 1}, {y, -1.2, 1.2}]
```

```
J2 = Julia[0.73251, -0.414193, 6, {x, -1.3, 1.3}, {y, -1.3, 1.3}];
Labeled[J2, Style["n=6, C=0.73251-0.414193*I", 18, Bold, Blue]]
```

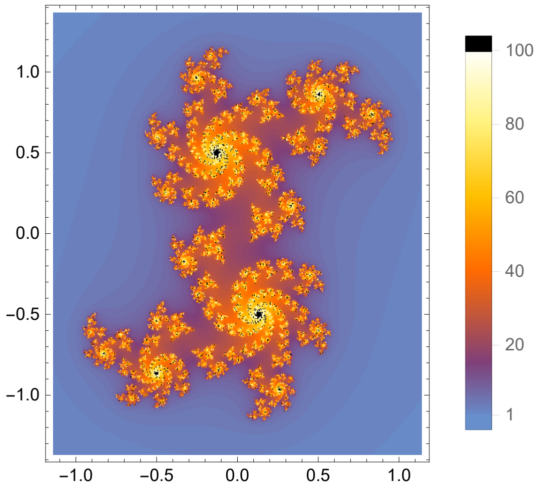


"n=6, C=0.73251-0.414193*I"

4. JuliaSetPlot 绘制朱莉娅集合，选项与Graphics相同
MandelbrotSetPlot 绘制曼德尔布罗特集，选项与Graphics相同

例7：用系统函数画分形图

```
JuliaSetPlot[0.365 - 0.37 ⅈ, PlotLegends → Automatic]
```




```
MandelbrotSetPlot[{-0.65 + 0.47 I, -0.4 + 0.72 I}, PlotLegends → Automatic]

MandelbrotSetPlot[ColorFunction → Hue]
```

5. 中止程序运行

单击 计算 (V) → 放弃计算 (A) 或 Alt + .

系统会退出全部表达式运算，返回 $Aborted

```
x = 1; While[x > 0, x]
```