第8讲 程序设计

语言概述

Wolfram 语言是一种高度发展的基于知识的语言，支持多种编程模式，
是一个可扩展的系统，可以方便的创建高效、模块化的程序包、
它的符号程序结构和界面体系结构，提供一个良好的软件开发环境.

过程式编程

Wolfram 语言从传统的计算机语言中脱颖而出.Wolfram 语言支持所有标准的过程式编程结构，通过集成把它们扩展到更为普遍的符号编程环境中。

例：条件 If，循环 While.

函数式编程

函数式编程是 Wolfram 语言的核心功能，它为符号功能提供了高效、简洁的特色.

例： Apply，Array，Nest，Select.

程序设计：输入输出 条件结构 循环结构 模块结构

8－1 条件结构

过程化条件：If，Which，Switch

1. If 语句

If[cond，expr1，expr2，expr3]

如果逻辑表达式 cond = True，返回expr1的值；如果cond = False，返回expr2的值；
否则返回expr3的值.expr2和expr3可缺省，所有表达式都是复合表达式.

If[cond，expr1]　如果cond = True，返回expr1的值.

If[cond，expr1，expr2]

如果cond = True，返回expr1的值；

如果cond = False，返回expr2的值.

例1： expr2 的默认值为Null，expr3的默认值为If表达式本身。

```
fa[x_] := If[x > 0, Green]
{fa[2], fa[-2], fa[t]}
```

例2：定义函数

$$f(x,y)=\begin{cases}x+y, x\cdot y\geq 0\\ x/y, x\cdot y<0\end{cases}$$

```
f[x_, y_] := If[x > 0 && y >= 0, x + y, x / y]
{f[12, 3], f[24, -12], f[2, u]}
```

2 Which 语句

Which[cond1, val1, cond2, val2, ..., condn, valn]
返回第一个满足condi = True的vali的值，
若每个condi都是False，返回Null，若某个condi既非True又非False，
返回表达式 Which[condi, vali, ..., condn, valn].

例3：用 If 嵌套和 Which 语句定义函数

$$g(x)=\begin{cases}\sin x, & x<-1\\ |x|, & -1\le x\le 1\\ \cos x, & x>1\end{cases}$$

```
ga[x_] := Which[x < -1, Sin[x], x ≥ -1 && x ≤ 1, Abs[x], x > 1, Cos[x]]
{ga[-2], ga[-0.5], ga[2]}
```

```
gb[x_] := If[x < -1, Sin[x], If[x <= 1, Abs[x], Cos[x]]]
{gb[-2], gb[-0.5], gb[2], gb[t]}
```

例4：用 Which 语句定义分段函数

$$h(x)=\begin{cases}-x, & x<0\\ \sin(x), & 0\le x<6\\ x/2, & 16\le x<20\\ 0, & \text{其它}\end{cases}$$

```
h[x_] := Which[x < 0, -x, x ≥ 0 && x < 6, Sin[x], x ≥ 16 && x < 20, x / 2, True, 0]
{h[-12], h[5], h[16.2], h[33]}
```

h[t]？课后练习

3. Switch 语句

Switch[expr, patt1, val1, ..., pattn, valn]
计算expr的值，与模式form1, form2...，依次做比较，
找出第一个与expr 匹配的模式formi计算并返回表达式valuei的值。
若无匹配，返回Switch表达式本身。

例5：Mod[x, 3]，余数为0，1，2 对应 sin x，cosx，在 {1, x} 作 Log 图.

```
g[x_] := Switch[Mod[x, 3], 0, Sin[x], 1, Cos[x], 2, Plot[Log[t], {t, 1, x}]]
{g[9], g[16], g[8]}
```

4. 函数式条件

(1) 数学函数条件

```
Piecewise[{{val1, cond1}, ..., {valn, condn}}, val]
```

(2) 条件表达式：ConditionalExpression

```
Simplify[Sqrt[x^2] + x^2, Assumptions → x < 0]
```

例6：定义分段线性函数并绘图.

$$f(x)=\begin{cases} x+4, & x<1 \\ 6-x, & -1\le x<2 \\ 2x, & 2\le x<3 \\ 9-x, & x\ge 3 \end{cases}$$

```
Plot[Which[x < 1, x + 4, x < 2, 6 - x, x < 3, 2 x, x ≥ 3, 9 - x], {x, -1, 4}]
```

```
f[x_] := Piecewise[{{x + 4, x < 1}, {6 - x, 1 ≤ x < 2}, {2 x, 2 ≤ x < 3}, {9 - x, x ≥ 3}}]
Plot[f[x], {x, -1, 4}]
```

(3) 基于列表条件

```
Cases[list, patt]
```
给出list中所有满足patt的元素

```
Select[list, cond]
```
选取list中所有满足cond的元素

例7：查找数列中的素数，Cases 和 Select 具有条件和循环的双重功能。

```
data = Range[20]; Cases[data, _?PrimeQ]
```

```
Select[data, PrimeQ]
```

```
Cases[{{a, b}, {1, 2, 3}, {{d, 6}, {d, 10}}}, {_, _}?VectorQ]
```

(4) 模式约束

`_h` 指定头部为*h*的任意表达式；

`/;` `pattern /; condition` 当条件满足时，模式才匹配；

`lhs:>rhs/;condition` 当条件满足时，才使用规则；

`p? test` 测试 p是否为 True .

```
f[x_Integer] := x + 1
```

```
f[x_?NumericQ] := NIntegrate[Sin[t^3], {t, 0, x}]
```

```
g[x_] := Sin[x] /; x > 0
```