

规格严格 功夫到家



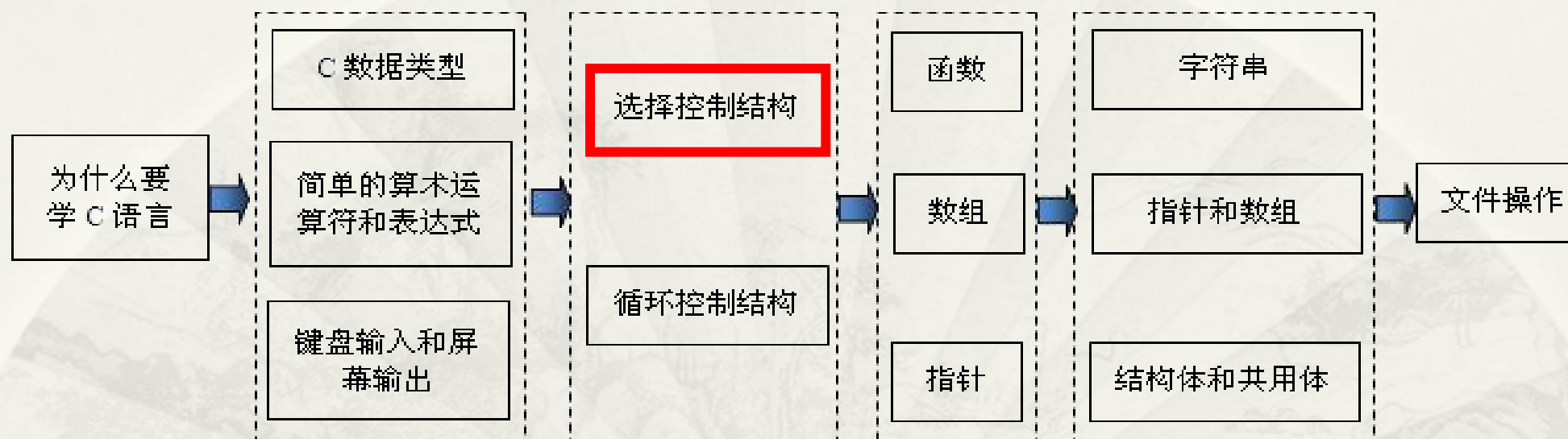
第5章 选择控制结构

哈尔滨工业大学（深圳）
计算机科学与技术学院
刘洋

liu.yang@hit.edu.cn

课件.版权：哈尔滨工业大学，苏小红，sxh@hit.edu.cn





第5章 学习内容

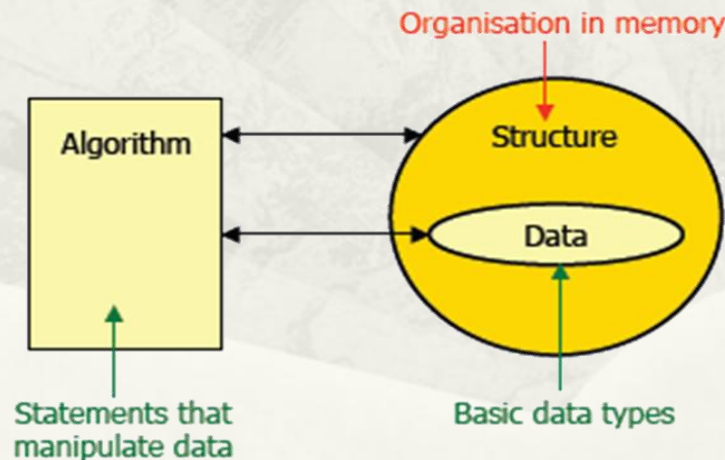
- 算法的描述方法
- 关系运算符，条件运算符，逻辑运算符
- 条件语句
- 开关语句
- 程序测试



5.1 计算机中的问题求解

■ 设计算法（Algorithm）

- 为解决一个具体问题而采取的、确定的、有限的操作步骤
- 仅指计算机能执行的算法



- **Data** to be manipulated
- **Structure**, defining how data is stored
- **Algorithm** – the operations on the data

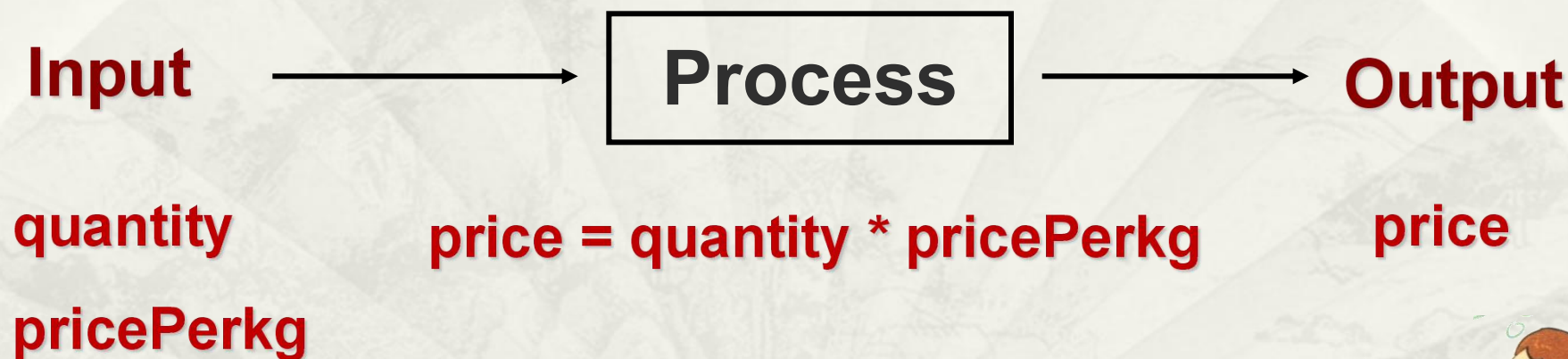
5.2 算法的概念及其描述方法

■ 算法的描述方法

- * 自然语言描述
- * 伪码（Pseudocode）描述
- * 传统流程图（Flowchart）
 - 在1966年，Bohra 与 Jacopini 提出
 - 算法的图形表示
- * N-S结构化流程图
 - 1973年，美国学者I.Nassi 和 B.Shneiderman 提出

5.2 算法的概念及其描述方法

例：已知苹果每公斤价格pricePerkg，买quantity 公斤的苹果，需要多少钱？



先确定问题的输入和输出。
再确定问题的求解方法。



5.2 算法的概念及其描述方法

■ 自然语言描述

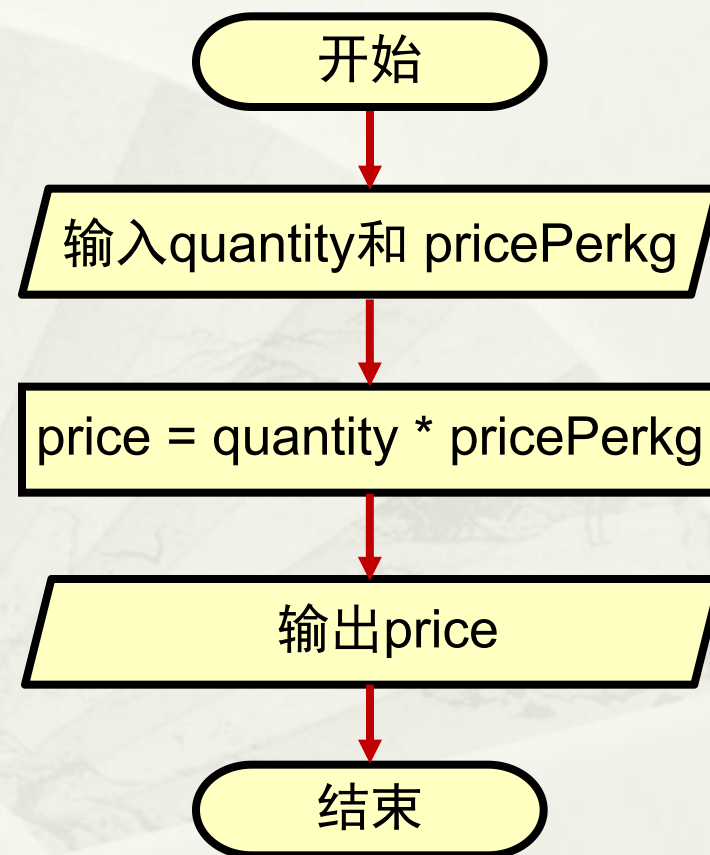
- * 输入 quantity 和 pricePerkg
- * 根据公式 $\text{price} = \text{quantity} * \text{pricePerkg}$ 计算 price
- * 输出 price

■ 伪码描述

- * Input quantity, pricePerkg
- * $\text{price} = \text{quantity} * \text{pricePerkg}$
- * Output price
- * End

5.2 算法的概念及其描述方法

- 传统流程图描述
- N-S结构化流程图描述



顺序结构 (Sequence Structure)

■ 计算和赋值

* 赋值表达式语句

赋值表达式 ;

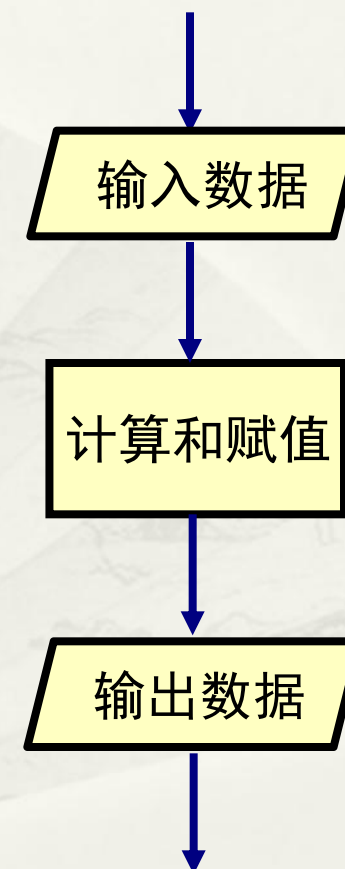
```
price = quantity * pricePerkg;
```

■ 数据的输入输出

* 标准输入输出函数调用语句

```
scanf("%d", &pricePerkg);
```

```
printf("%d", price);
```



5.2 算法的概念及其描述方法

■ 算法的特性

- * 有穷性

- 在合理的时间内完成

- * 确定性，无歧义

- * 如果 $x \geq 0$ ，则输出Yes；如果 $x \leq 0$ ，则输出No（歧义！）

- * 有效性

- 能有效执行

- * 负数开平方.....

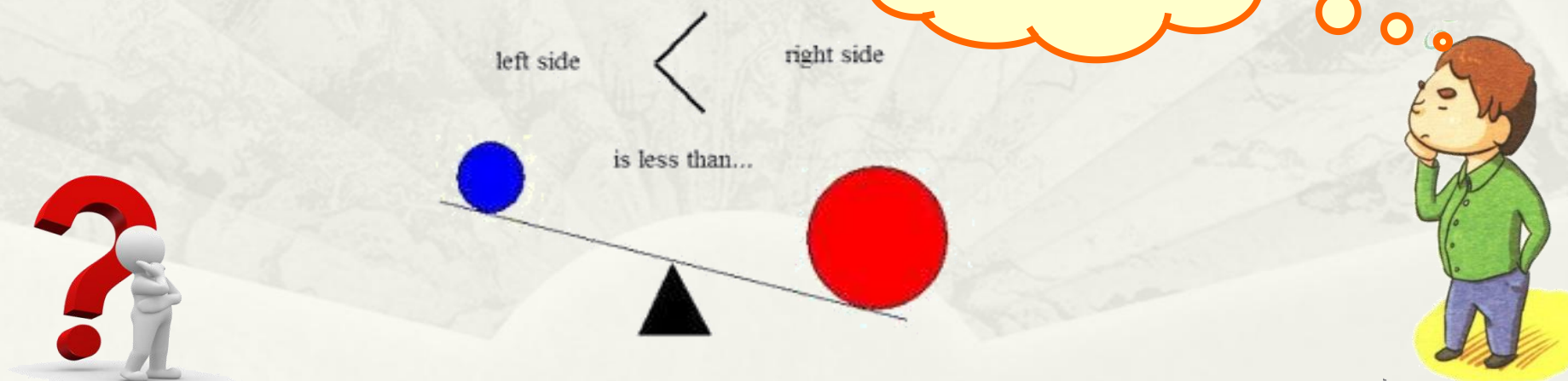
- * 没有输入或有多个输入

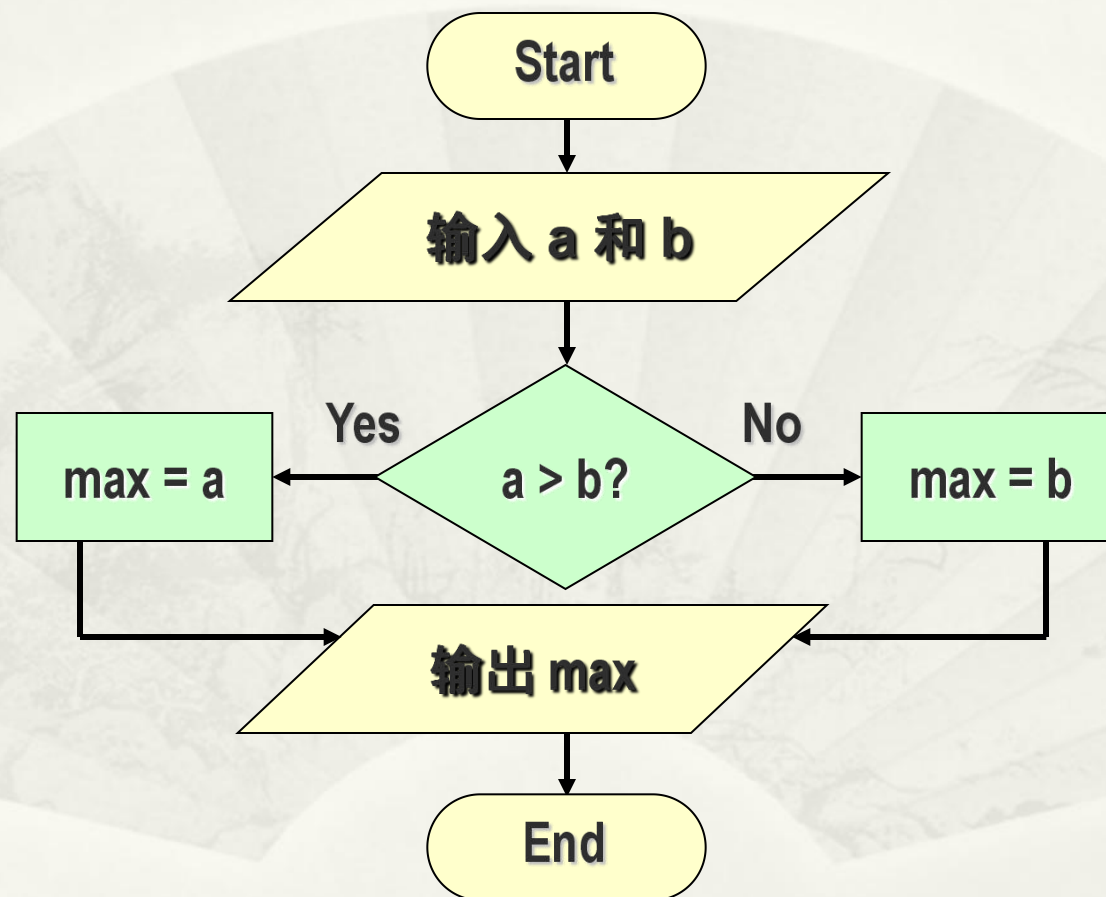
- * 有一个或多个输出

【例5.1】计算两整数的最大值



如何设计
算法呢?





5.3 关系运算符与关系表达式

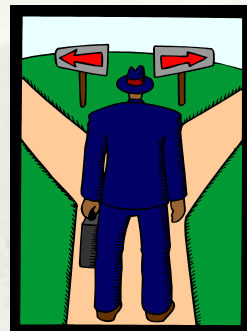
Relational Operation	Description	Examples of Expression	表达式的值
<	Less than	6 < 9	1 (true)
<=	Less than or equal to	5 <= 5	1 (true)
>	Greater than	2 > 6	0 (false)
>=	Greater than or equal to	9 >= 5	1 (true)
==	Equal to	7 == 5	0 (false)
!=	Not equal to	6 != 5	1 (true)

注意判等运算符的写法和表达式的值

更多需要使用选择结构的例子

- 计算n个数的最大值，最小值
- 判断三角形三边能否构成三角形？
- 判断输入的英文字母是大写还是小写？
- 判断某年是否是闰年？
-

选择结构（分支结构） (Selection Structure)



单分支 (Single Selection)



if



双分支 (Double Selection)



if - else

多分支 (Multiple Selection)

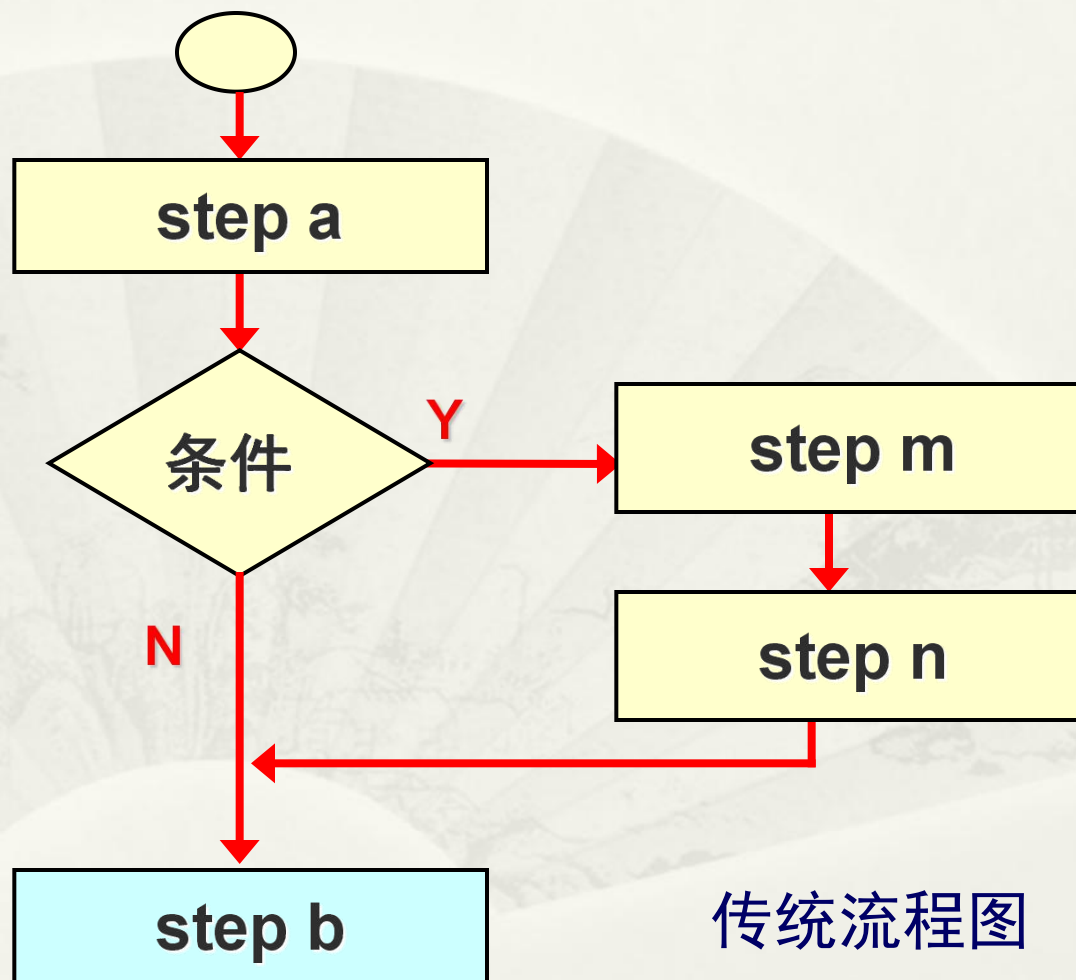


else - if

5.4用于单分支控制的条件语句

伪码表示

```
step a
if (条件为真)
{
    step m
    step n
}
step b
```



传统流程图

5.4用于单分支控制的条件语句

Syntax:

if (**表达式**)
语句;

用**表达式**表示**条件**

or

if (表达式)

{

语句1;
语句2;

}

复合语句

(Compound Statement)

被当作一条语句看待

5.4用于单分支控制的条件语句

Syntax:

```
if (表达式)  
    语句;
```

or

```
if (表达式)  
{  
    语句1;  
    语句2;  
}
```

不局限于关系表达式 (0或1)
也可数值表达式 (0或非0)
表达式的值非0时, 为真

如何表示条件的真和假呢?



5.5用于双分支控制的条件语句

伪码

Step a

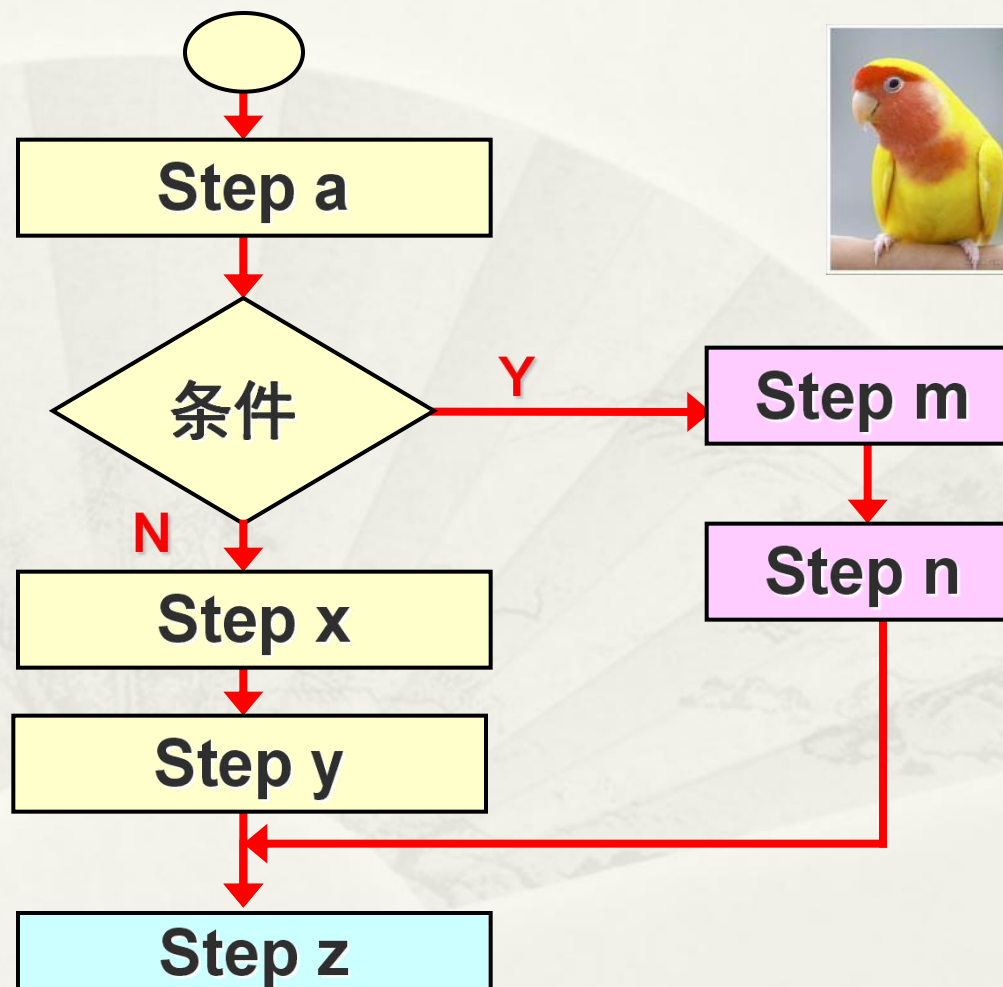
if (条件为真)

```
{  
    Step m  
    Step n  
}
```

else

```
{  
    Step x  
    Step y  
}
```

Step z



5.5用于双分支控制的条件语句

Syntax:

if (表达式)

语句1;

else

语句2;

即使分支中只有一条语句，大括号最好也不省略

or

if (表达式)

{

语句1;

语句3;

}

else

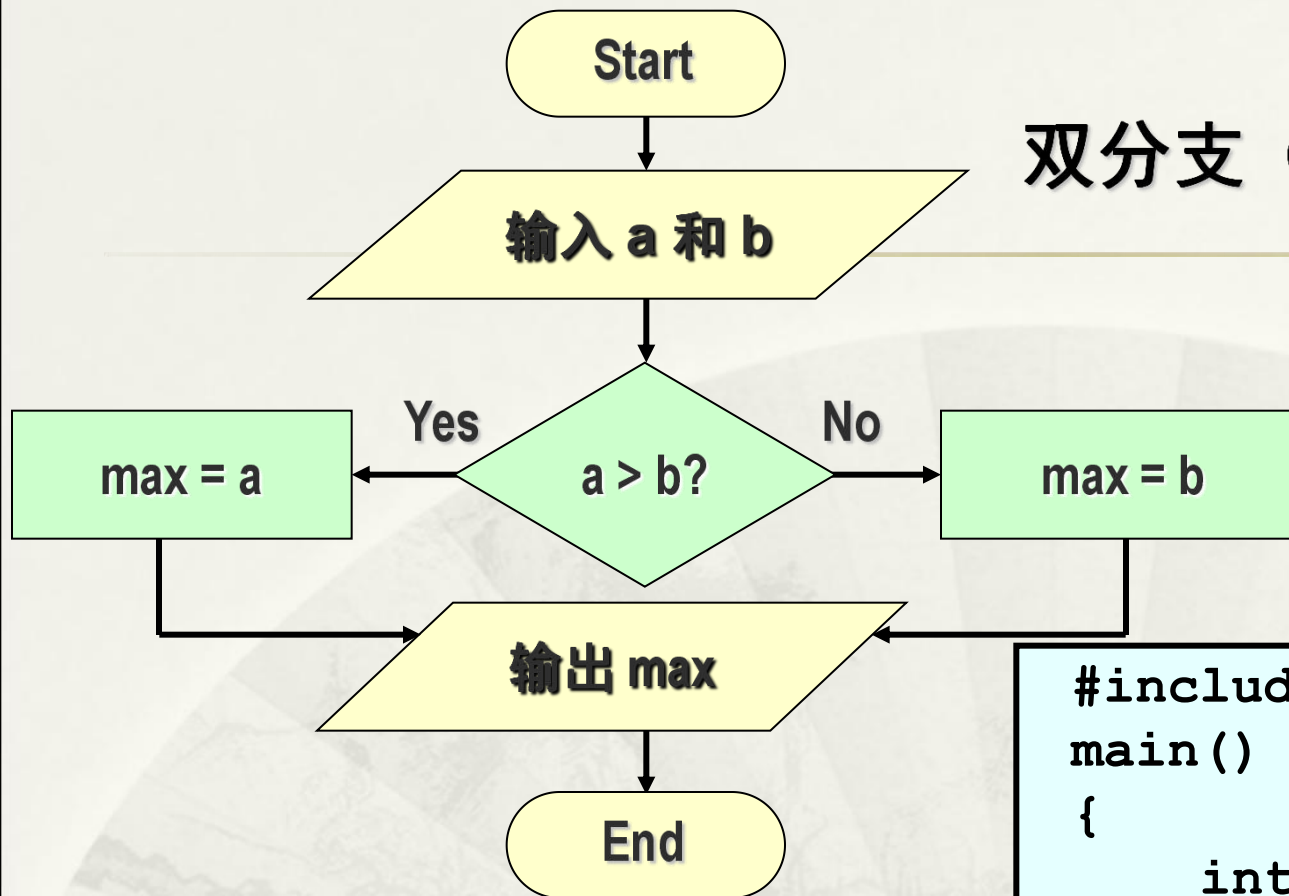
{

语句2;

语句4;

}

双分支 (Double Selection)



```
if (a > b)
    max = a;
else
    max = b;
```

```
if (a > b)
    max = a;
if (a <= b)
    max = b;
```

```
#include <stdio.h>
main()
{
    int a, b, max;
    printf("Input a,b:");
    scanf("%d,%d", &a, &b);
    if (a > b)    max = a;
    else        max = b;
    printf("max = %d\n", max);
}
```

5.6 条件运算符和条件表达式

```
#include <stdio.h>
```

```
main()
```

```
{
```

```
    int a, b, max;
```

```
    printf("Input a, b:");
```

```
    scanf("%d,%d", &a, &b);
```

```
    if (a > b)
        max = a;
```

```
    else
        max = b;
```

```
    printf("max = %d", max);
```

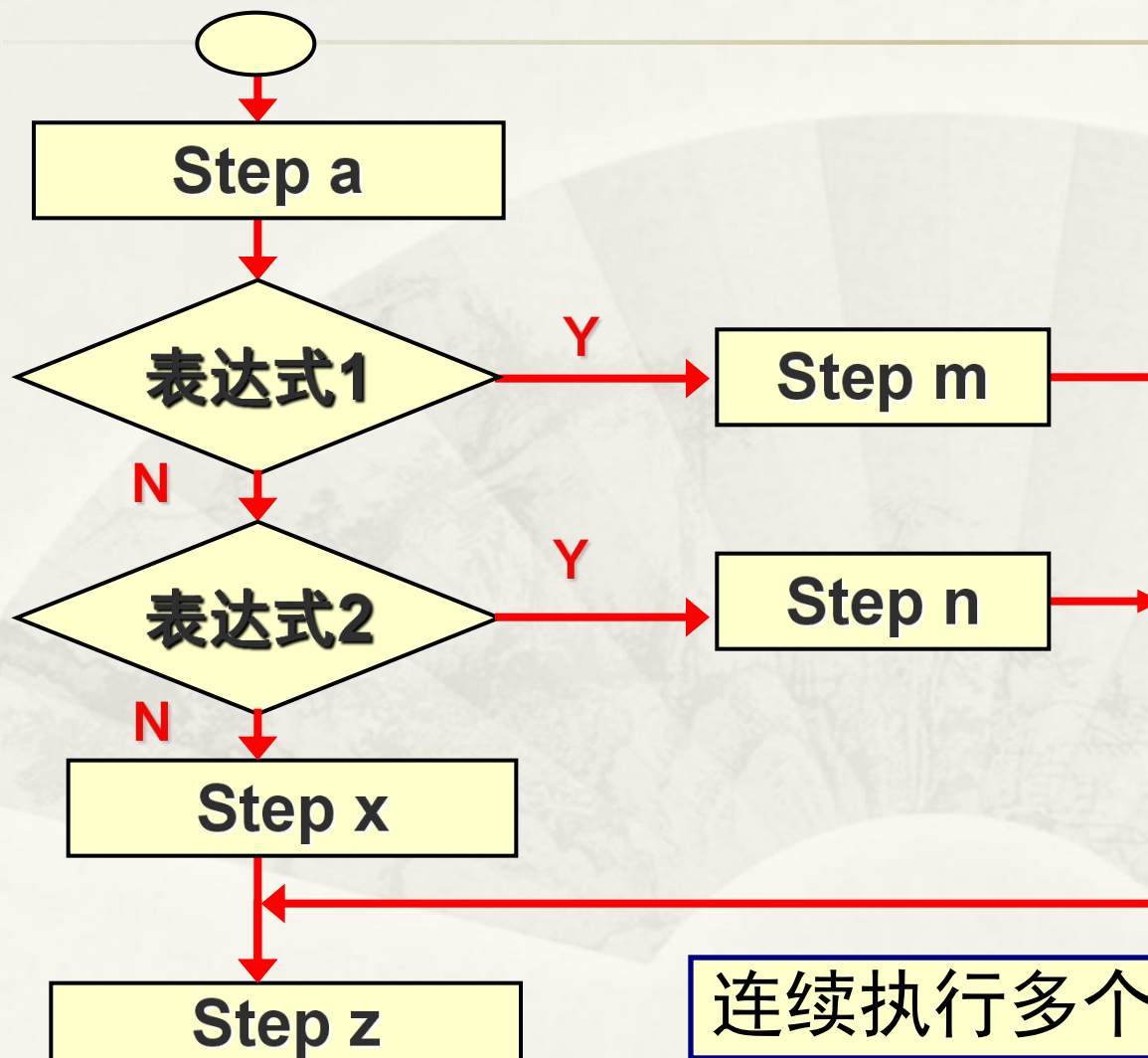
```
}
```

【例5.3】

表达式1 ? 表达式2 : 表达式3

max = a > b ? a : b;

5.7用于多分支控制的条件语句

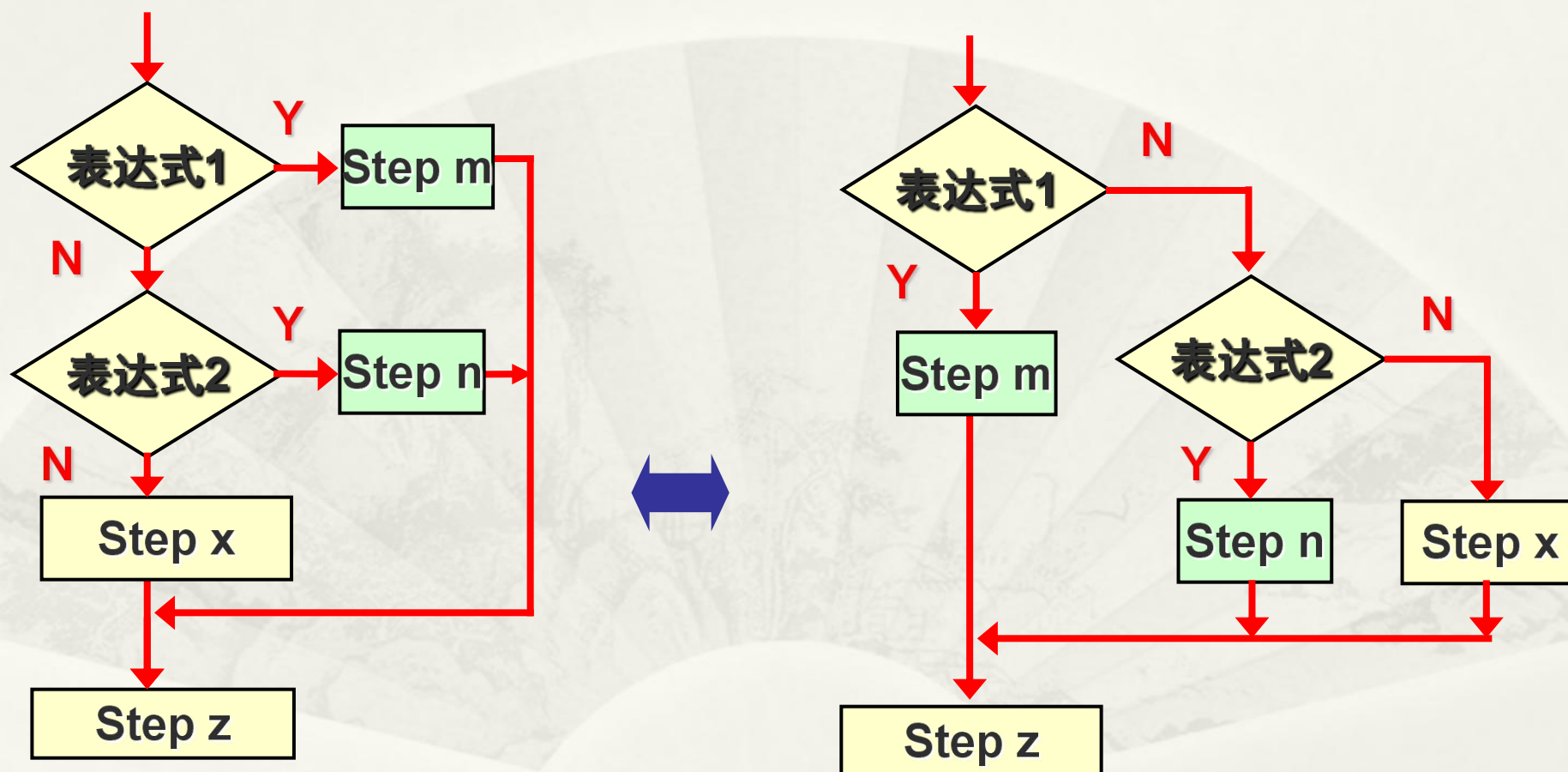


多分支如何控制呢？



连续执行多个条件判断

5.7 用于多分支控制的条件语句



5.7用于多分支控制的条件语句

级联式if语句: else-if

Step a

if (表达式1)

{

Step m

}

else if(表达式2)

{

Step n

}

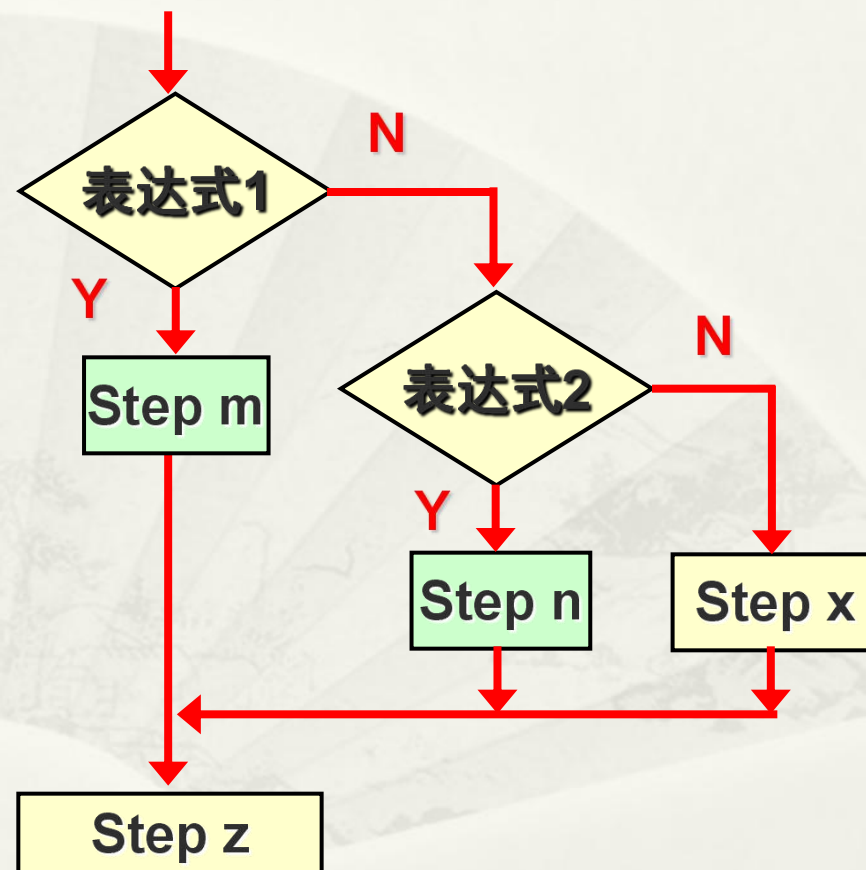
else

{

Step x

}

Step z



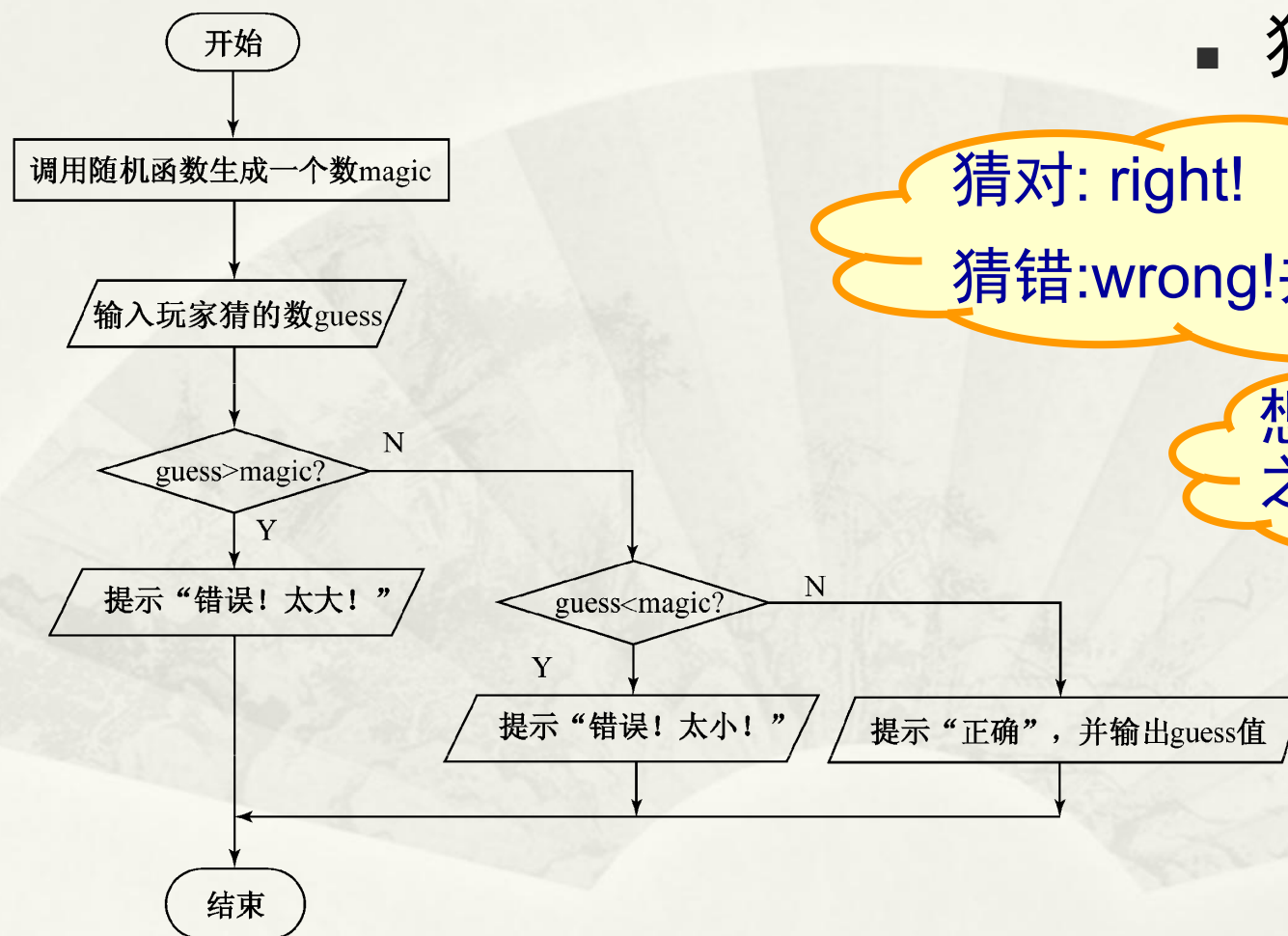
5.7用于多分支控制的语句

■ 猜数游戏

猜对: right!

猜错: wrong! 并提示大小

想一个1~100
之间的数



5.8用于多路选择的switch语句

开关语句

switch (表达式)

{

case value1 :

语句1;

break;

case value2 :

语句2;

break;

.....

default :

语句n;

break;


}

必须是int或char!

必须是常量!!

它是如何执行的
的呢?

必须有空格!!



5.8用于多路选择的switch语句

Example:

```
switch (month) {  
    case 1:  
        printf("January\n");  
        break;  
    case 2:  
        printf("February\n");  
        break;  
    case 3:  
        printf("March\n");  
        break;  
    default:  
        printf("Others\n");  
        break;  
}  
printf("End");
```

假设 **month = 1**

*this step will be ... **case** is terminated here. Jump to ...*

January
End _



5.8用于多路选择的switch语句

Example: `switch (month) {`

`case 1:`

`printf("January\n");`
`break;`

`case 2:`

`printf("February\n");`
`break;`

`case 3:`

`printf("March\n");`
`break;`

`default:`

`printf("Others\n");`
`break;`

`}`

`printf("End");`

March

End _

假设 **month = 3**

... *this step will be*
... **case** *is terminated*
here. Jump to ...



5.8用于多路选择的switch语句

Example: switch (month) {

case 1:

printf("January\n");
break;

case 2:

printf("February\n");
break;

case 3:

printf("March\n");
break;

default:

printf("Others\n");
break;

}

printf("End");

假设没有 *break*?



5.8用于多路选择的switch语句

Example

```
switch (month) {
```

```
case 1:
```

```
    printf("January\n");  
    break;
```

```
case 2:
```

```
    printf("February\n");
```

```
case 3:
```

```
    printf("March\n");  
    break;
```

```
default:
```

假设 **month = 2**

March

End _

*...this step will be
executed. Later ...*

...execution continues.

Thus, this step is executed .

So ...

*... **case** is*

terminated here.

Jump to ...



5.8用于多路选择的switch语句

```
Example: switch (month) {  
    case 1:  
        printf("January\n");  
        break;  
    case 2:  
        printf("February\n");  
    case 3:  
        printf("March\n");  
        break;  
    default:  
        printf("Others\n");  
        break;  
}  
printf("End");
```

month = 1 ?

假设这两个 **break** 也
移除?



5.8用于多路选择的switch语句

Example: switch (month) {
 case 1:
 printf("January\n");

 case 2:
 printf("February\n");

 case 3:
 printf("March\n");

 default:
 printf("Others\n");
 break;
}
printf("End");

month = 34 ?

default分支最好不省略!

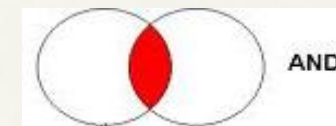
5.9逻辑运算符和逻辑表达式

Symbol

Description

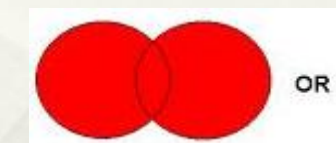
&&

与（**AND**）当且仅当两者都为真



||

或（**OR**）只要两者中有一个为真



!

非（**NOT**）

a	b	a && b	a b	!a	!b
0	0	0	0	1	1
0	1	0	1	1	0
1	0	0	1	0	1
1	1	1	1	0	0

高 **!** **&&** **||** 低

5.9逻辑运算符和逻辑表达式

■ `i > j > k`

- * 在C语言中是合法的，但它可能不是你期望表达的意思
- * 相当于 `(i > j) > k`，不测试j是否位于i和k之间
- * 不同于 `(i > j) && (j > k)`

■ 判断ch是大写英文字母

`'Z' >= ch >= 'A'` 是错误的

`(ch >= 'A') && (ch <= 'Z')`

■ 判断ch是数字字符

`(ch >= '0') && (ch <= '9')`

第6章循环控制结构与循环语句（自学内容部分, 1个自学学时）

自学以下内容的语法语义部分，在下一次课前，会有10分钟语法小测验，计入成绩

■程序流程图：

- * 要自行学会绘制标准、完整程序流程图

■循环结构与循环控制方法：

- * 计数控制、标记控制、条件控制

■循环语句：

- * for、while、do...while循环
- * for循环中3个表达式的执行顺序是什么样的？

第五章作业

- 5.4, 5.5, 5.6, 5.7, 5.10.

