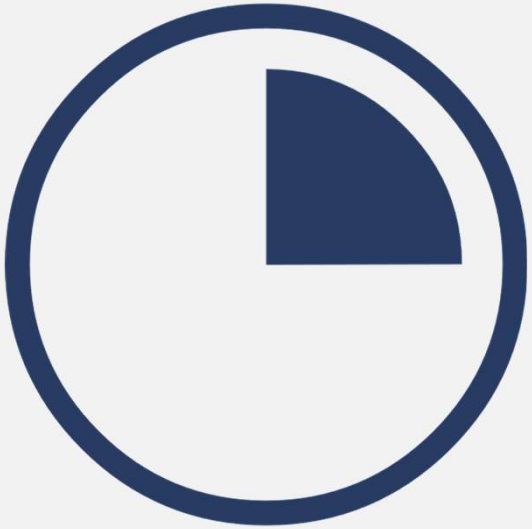
光电信息问题的MATLAB数学建模实验

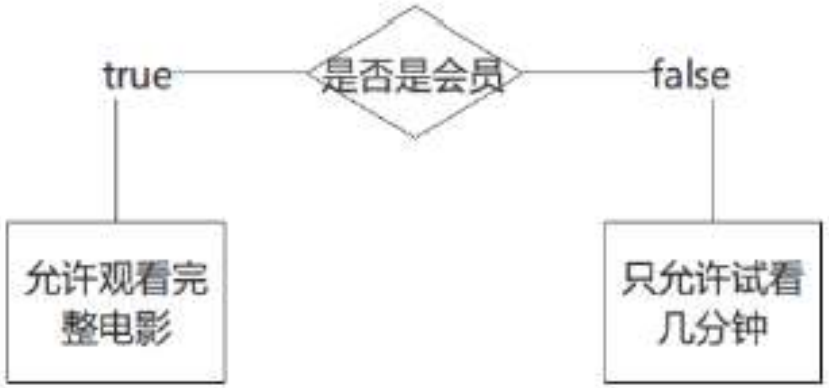
School of Precision Instrument and Opto-electronics Engineering
Tianjin University

2024.09.02



PART 04
MATLAB程序流程控制

一、条件语句



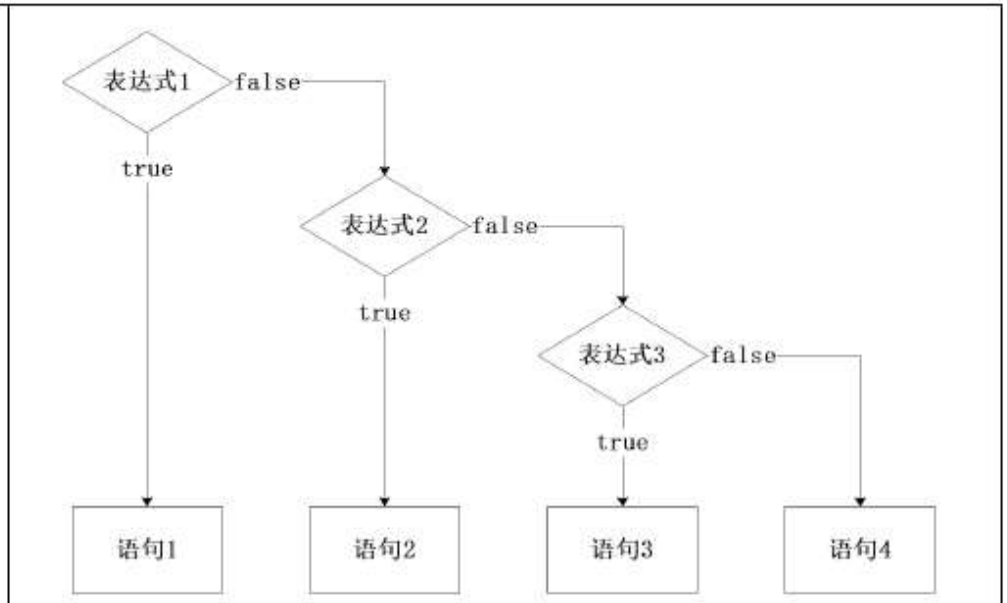
if-elseif-else-end语句和switch-case-otherwise-end语句

一、条件语句

1.if-elseif-else-end语句

elseif关键字中间不能加空格，不能写成**else if**

```
if 表达式1
    语句1
elseif 表达式2
    语句2
elseif 表达式3
    语句3
else
    语句4
end
```



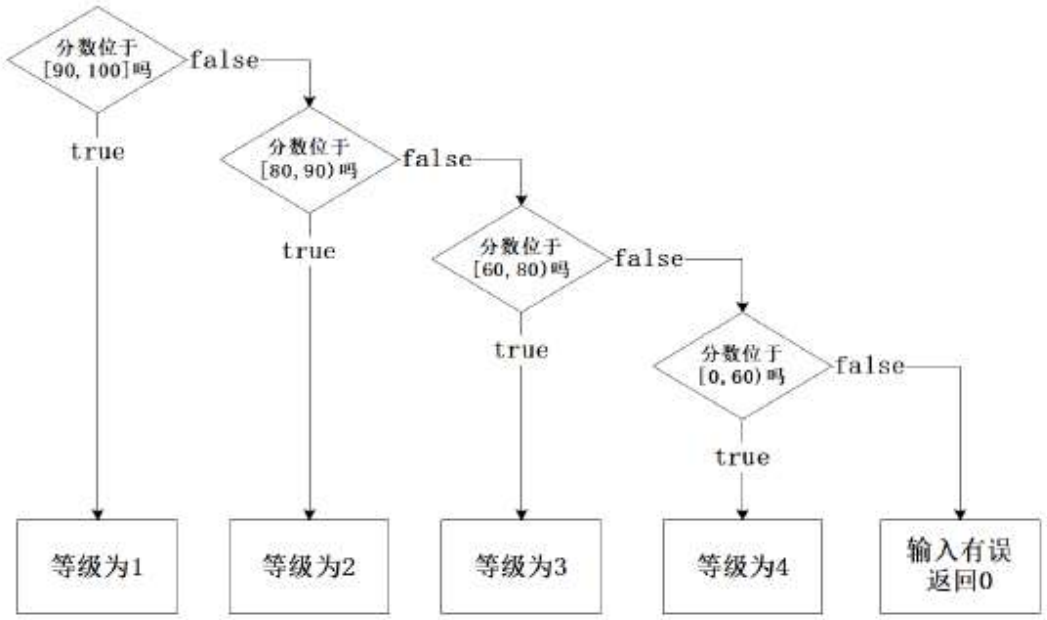
一、条件语句

1.if-elseif-else-end语句

我们举一个具体的例子：计算分段函数$y=\begin{cases}1, & 90\le x\le 100\\ 2, & 80\le x<90\\ 3, & 60\le x<80\\ 4, & 0\le x<60\\ 0, & x\text{取其他值}\end{cases}$，你可以将这个分段函数想象成如下场景：给定一个同学的成绩（假设为整数），输出这个同学的等级。等级规则如下：90 至 100 分为 1 级、80 至 89 分为 2 级、60 至 79 分为 3 级、低于 60 分为 4 级；如果成绩小于 0 分或者大于 100 分，则代表成绩输入有误，此时等级为 0。

一、条件语句

1.if-elseif-else-end语句




```
x = 88;  % x表示成绩
if x>=90 && x <= 100
    dj = 1;  % 等级为1级
elseif x>=80 && x < 90
    dj = 2;  % 等级为2级
elseif x>=60 && x < 80
    dj = 3;  % 等级为3级
elseif x>=0 && x < 60
    dj = 4;  % 等级为4级
else
    dj = 0;  % 输入有误
end
dj
```

```
dj =
     2
```

一、条件语句

1.if-elseif-else-end语句

（1） 判断 x 是否位于[90,100]这个区间时，不能写成 90<=x<=100，MATLAB 不支持这种连续的判断。因此我们将其拆分成两个条件：x>=90 && x<=100。这里用的是具有短路功能的逻辑与运算符&&，这样判断效率会更高。

（2） if 只能有一个，但 elseif 可以有多个，它们后面都需要跟上相应的判断条件。只有在 if 后面的条件不满足时，才会判断 elseif 后面的条件是否成立。

（3） else 后面不能加上条件，当 if 和 elseif 后面的条件全部都不满足时，才会执行 else 对应的语句。

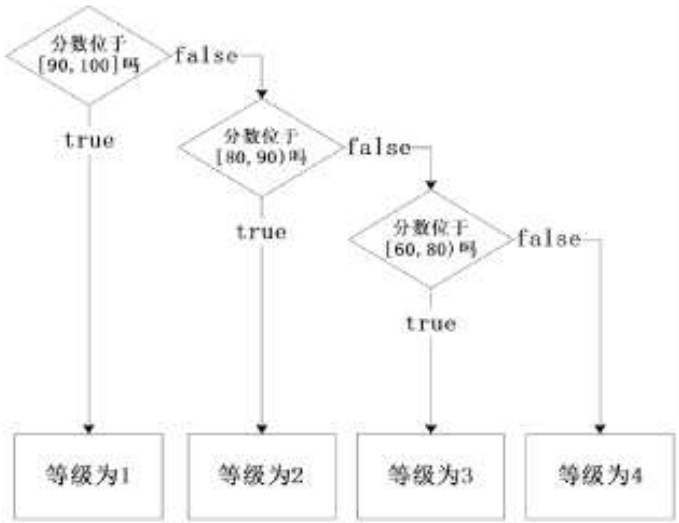
（4） 不需要在 if、elseif、else 和 end 所在的行的最后面添加冒号或者分号。

```
x = 88;  % x表示成绩
if x>=90 && x <= 100
    dj = 1;  % 等级为1级
elseif x>=80 && x < 90
    dj = 2;  % 等级为2级
elseif x>=60 && x < 80
    dj = 3;  % 等级为3级
elseif x>=0 && x < 60
    dj = 4;  % 等级为4级
else
    dj = 0;  % 输入有误
end
dj
```

```
dj =
     2
```

一、条件语句

1.if-elseif-else-end语句




```
x = 68;  % x表示成绩
if x>=90 && x <= 100
    dj = 1;  % 等级为1级
elseif x>=80 && x < 90
    dj = 2;  % 等级为2级
elseif x>=60 && x < 80
    dj = 3;  % 等级为3级
else
    dj = 4;  % 等级为4级
end
dj
```

```
dj = 3
```

```
x = 68;  % x表示成绩
if x>=90
    dj = 1;  % 等级为1级
elseif x>=80
    dj = 2;  % 等级为2级
elseif x>=60
    dj = 3;  % 等级为3级
else
    dj = 4;  % 等级为4级
end
dj
```

```
dj = 3
```

使用if语句时，if和end这两个关键字是无论如何都不能省略的，而elseif和else可以根据自己的编程需要来决定是否添加。

一、条件语句

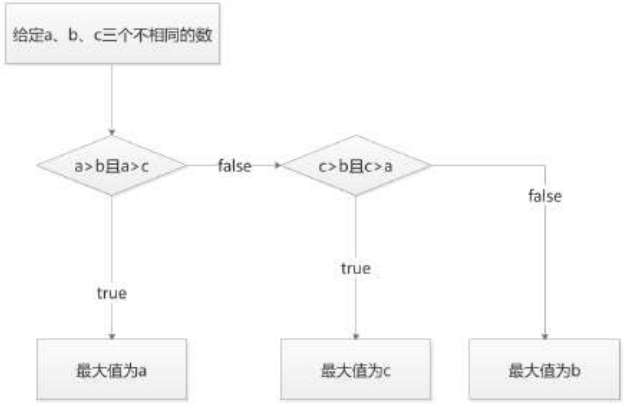
1.if-elseif-else-end语句

例题：已知a、b和c是三个互不相等的常数，请使用if语句找出a、b和c三个数的最大值。（注意，这里是练习条件语句，请不要使用max函数直接求最大值）

一、条件语句

1.if-elseif-else-end语句

6种可能的排序:		
b	c	a
c	b	a
a	b	c
b	a	c
c	a	b
a	c	b

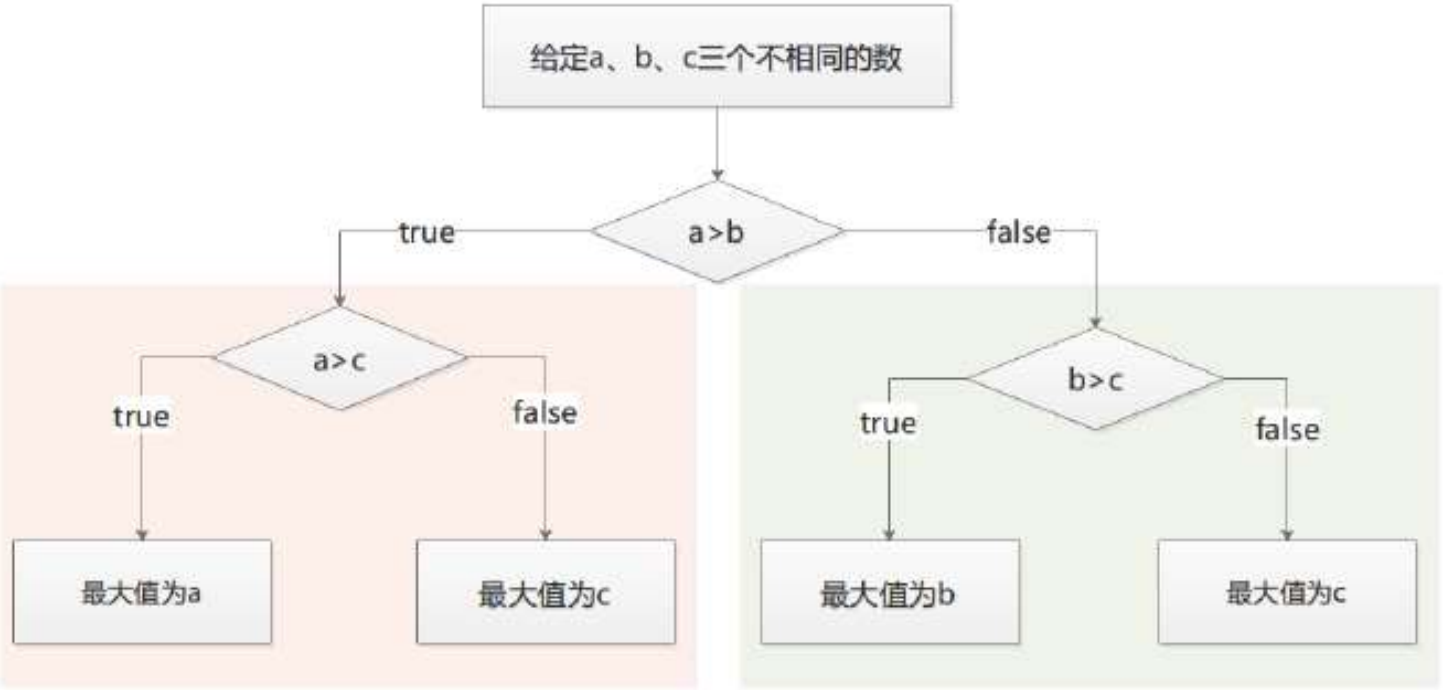


注意：a>b 且 a>c 涵盖了表中前两种排序的情况；c>b 且 c>a 则涵盖了表中第三种和第四种排序的情况；如果都不满足则只剩下表中最后两种排序的情况，此时最大值为 b.

```
a = 5;  b = 8;  c = 3;  % 随便编一组数据测试
if a > b && a > c
    Max = a;  % 不要命名为小写的 max，否则和内置函数重名了
elseif c > b && c > a
    Max = c;
else
    Max = b;
end
Max
```

一、条件语句

1.if-elseif-else-end语句



一、条件语句

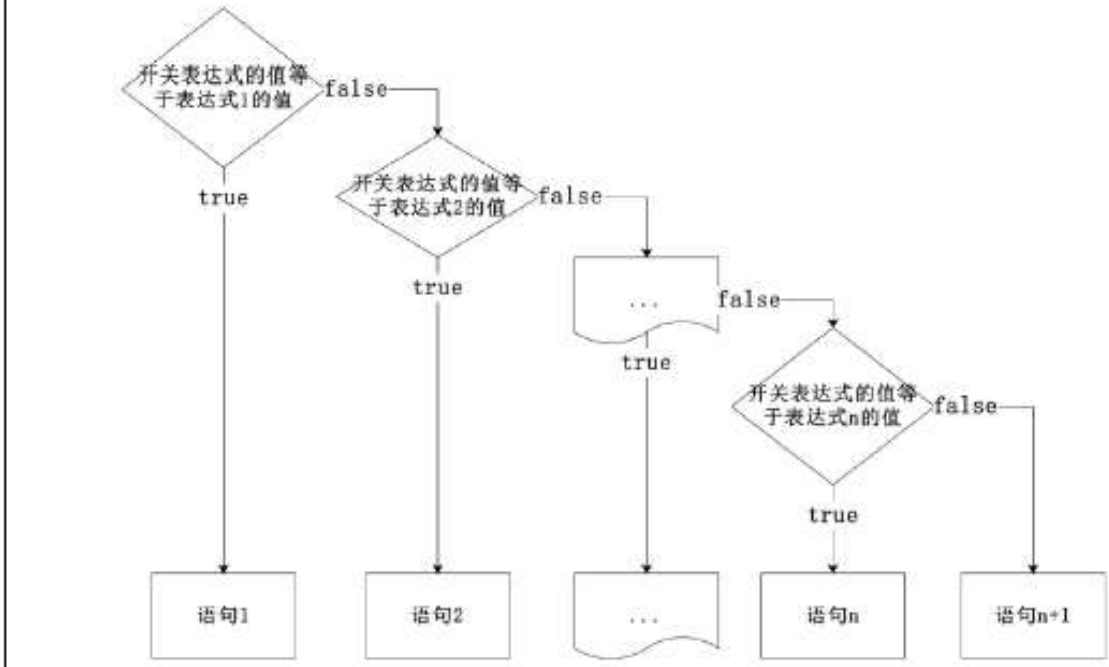
1.if-elseif-else-end语句

```
if a>b
    if a>c    % 此时 a>b 且 a>c
        Max=a;
    else  % 此时 a>b 且 c>a
        Max=c;
    end  % a>c 前面的那个 if 配套的 end，通过缩进可以看出
else   % 此时 a<b
    if b>c    % 此时 a<b 且 b>c
        Max=b;
    else    % 此时 a<b 且 b<c
        Max=c;
    end    % b>c 前面的那个 if 配套的 end，通过缩进可以看出
end  % 最上方的 if 配套的 end
Max
```

一、条件语句

2. switch-case-otherwise-end语句

```
switch 开关表达式
    case 表达式 1
        语句 1
    case 表达式 2
        语句 2
    …
    case 表达式 n
        语句 n
    otherwise
        语句 n+1
end
```



开关表达式的计算结果必须是一个数值标量或者是一个字符向量/字符串，不能是向量或者矩阵。

一、条件语句

2. switch-case-otherwise-end语句

```
season = randi([1,4])    % 季节
switch season
    case 1
        disp("第一季度为春季");
    case 2
        disp("第二季度为夏季");
    case 3
        disp("第三季度为秋季")
    otherwise
        disp("第四季度为冬季")
end
```

season 是在区间[1,4]上随机生成的一个整数，用来表示季节。switch 后面的开关表达式就是 season 这个数值标量，程序会按照从上到下的顺序依次判断 season 和 case 后面的数值是否相等，若相等则执行对应的语句。若 season 取值为 4，则和 case 后面的数值均不相等，此时程序会执行 otherwise 后面的语句。

注意：MATLAB 中的单引号和双引号表示的文本有一定的区别，使用单引号引起来被称为字符向量，例如'abc'，而双引号引起来的被称为字符串，例如"abc"。

一、条件语句

2. switch-case-otherwise-end语句

```
a = randi(10)
b = randi(10)
way = "乘法";
switch way
    case "加法"
        disp(a+b)
    case "减法"
        disp(a-b)
    case "乘法"
        disp(a*b)
    case "除法"
        disp(a/b)
    otherwise
        disp("你的输入有误")
end
```

a 和 b 是在区间[1,10]上随机生成的两个整数，我们希望根据变量 way 表示的字符串决定对 a 和 b 的计算方式。

switch 后面的开关表达式就是 way 表示的字符串，程序会按照从上到下的顺序依次判断 way 和 case 后面的字符串是否相同，若相同则执行对应的语句。例如 way 等于“乘法”，则程序会先判断第一个 case，第一个 case 为“加法”，字符串不相同，则会继续判断第二个 case，第二个 case 的字符串和 way 的字符串也不同，则会判断第三个 case，此时字符串一致，程序会执行 disp(a*b)这一行语句，假设 a 为 5、b 为 3，则会在窗口输出 15。

和if语句一样，**switch语句必须以end结束，千万不能漏写！**

在**if语句中，有时候我们会不写elseif和else关键字，而switch语句也可以不写case和otherwise这两个关键字。**

二、循环语句

1. for-end语句

for-end 语句（简称 for 循环）用于事先已知循环次数的情形，其语法如下：

```
for 循环变量 = 向量或者矩阵
    循环体
end    % 循环体以 end 结束，千万不能漏写了！
```

在该语法中，循环变量是用于迭代的变量名，它会在每次循环迭代中从向量或矩阵中取出一列的值。数值向量或者矩阵则表示了循环变量可以取值的范围，通常根据实际需要事先给定。一旦循环变量遍历完数值向量或者矩阵中的所有值，循环就会结束。

二、循环语句

1. for-end语句

```
x = 1:5;
for ii = x
    ii
end
% 第二行可以直接写成：
% for ii = 1:5
% for 循环一定以 end 结束，别漏写了！
```

```
% x = [1, 2, 3, 4, 5]是一个行向量，可以看成一个 1 行 5 列的矩阵，因此每次取出一列的元素，共循环五次。
ii = 1
ii = 2
ii = 3
ii = 4
ii = 5
```

```
x = [3;5;9;8];
for ii = x
    ii
end
% for 和 end 所在的行的最后面什么都不用写，不需要添加冒号或者分号
```

```
% 注意：和上一个例子不同，此时的 x 是一个列向量，只有一列，因此只会循环一次。这种情况几乎用不到。
ii = 4×1
     3
     5
     9
     8
```

```
% 循环遍历矩阵的每一列，这种情况在实际编程中用的较少
x = [1 5 8;
     3 6 9;
     4 7 2;
     6 5 3];
for ii = x
    ii
end
%通常我们将循环体的开头增加缩进，这样看起来更美观。你可以选中代码，然后智能缩进，快捷键“ctrl+i”。
```

```
ii = 4×1
     1
     3
     4
     6

ii = 4×1
     5
     6
     7
     5

ii = 4×1
     8
     9
     2
     3
```

二、循环语句

1. for-end语句

（1）不使用 sum 函数，计算行向量 x 中所有元素的和。

```
x = [5 8 9 1 4 3 7];
s = 0;  % 初始化最终的求和结果为0
for ii = x
    s = s + ii;
end
disp(s)
```

在这个示例中，for 循环遍历了向量 x 中的每个元素，将它们逐个加到变量 s 中，最终得到了所有元素的和。

思考：如果 x 是一个列向量，左侧的代码输出的 s 是什么，应该如何修改代码？

二、循环语句

1. for-end语句

（2）计算当 n 等于 100 时，下面式子的结果：

$$y(n) = 1 + \frac{1}{2^2} + \frac{1}{3^2} + \cdots + \frac{1}{n^2}$$

```
n = 100;   % 设置 n 的值
y = 0;     % 初始化 y 的值为 0
for k = 1:n
    % 计算每一项并累加到 y 中
    y = y + 1 / (k^2);
end
disp(y)
```

在这个示例中，for 循环从 1 到 n 遍历每个整数 k，并计算每一项 1 / (k^2)，然后将它们累加到变量 y 中。最终，y 的值将是整个表达式的结果。

思考：使用上一章的知识点，如何通过一行代码直接计算 y 的值？

参考答案： y = sum(1./(1:100).^2)

二、循环语句

1. for-end语句

（3）计算当 n 从 1 一直取到 100 时，上一小问式子的计算结果，并将计算结果保存到一个长度为 100 的行向量 S 中（S 中第 i 个元素表示 y(i)的结果）。

```
S = zeros(1, 100);
for n = 1:100
    y = 0;      % 初始化 y 的值为 0
    for k = 1:n
        y = y + 1 / (k^2);
    end
    S(n) = y;
end
disp(S)
```

这里使用了循环的嵌套，上一问的代码可用来求出任意一个具体的 n 对应的 y。因此，这一问我们只需要使用循环让 n 从 1 遍历到 100，并将每次的计算结果保存到向量 S 中。请大家思考：（1）y = 0;这行代码能否放在循环的外面？（2）能否优化左侧的代码，使得通过一次循环就得到 S。

二、循环语句

1. for-end语句

（4）计算从公元 1 年到公元 9999 年间，有多少个闰年。闰年的判读条件是年份能够被 4 整除，但不能被 100 整除，或者年份能够被 400 整除。

```
% 初始化闰年计数器
leap_year_count = 0;
% 循环遍历从公元 1 年到公元 9999 年的每一年
for year = 1:9999
    % 检查是否为闰年的条件
    if (mod(year, 4) == 0 && mod(year, 100) ~= 0) || (mod(year, 400) == 0)
        % 如果是闰年，增加计数器
        leap_year_count = leap_year_count + 1;
    end
end
disp(leap_year_count)
```

二、循环语句

1. for-end语句

（5）一个三位正整数各位数字的立方和等于该数本身则称该数为水仙花数，例如：1^3 + 5^3+ 3^3 = 153，则 153 是水仙花数。请你找出所有的水仙花数并将其保存到向量 S 中。

```
% 初始化存储水仙花数的向量 S 为空
S = [];
% 循环遍历所有的三位整数
for num = 100:999
    % 拆解数字
    digit1 = floor(num / 100);      % 百位
    digit2 = floor(mod(num, 100) / 10);  % 十位
    digit3 = mod(num, 10);           % 个位
    % 检查是否为水仙花数的条件
    if num == digit1^3 + digit2^3 + digit3^3
        S = [S, num];  % 若是水仙花数，则添加到向量 S 中
    end
end
% 显示所有的水仙花数
disp(S)
```

二、循环语句

1. for-end语句

（1）若 for 语句后面的向量或者矩阵为空，则循环一次也不会被执行。

```
for ii = 2:1
    x = 10;
    disp(x)
end
% MATLAB 什么都不会输出
```

```
for ii = []
    x = 10;
    disp(x)
end
% MATLAB 什么都不会输出
```

（2）for 语句后面的向量或者矩阵只会在循环开始时使用一次，向量或者矩阵元素一旦确定将不会再改变。即使你在循环体中改变向量或者矩阵的值，循环变量的值也不改变。

```
x = 1:4;
for ii = x
    x = [0 0 0 0];
    disp(ii)
end
disp(x)
```

```
     1

     2

     3

     4

     0     0     0     0
```

（3）可以在循环体中修改循环变量的值，但当程序执行流程再次回到循环开始时，循环变量会自动恢复成向量或者矩阵的下一列元素。

```
for ii = 1:3
   disp(ii)
   ii = 10;
   disp(ii)
end
```

```
     1
    10
     2
    10
     3
    10
```

二、循环语句

2. while-end语句

与for循环不同，while循环的特点在于它允许我们在不知道具体循环次数的情况下执行循环体，

```
while 表达式
      循环体
end   % 循环体以 end 结束，千万不能漏写了！
```

二、循环语句

2. while-end语句

下面我们来看一个具体的例子：已知$y(n) = 1+\frac{1}{2}+\frac{1}{3}+\cdots+\frac{1}{n}$，当$n$最小取多少时，$y$的值大于10?

```
y = 1;  n = 1;
while y <= 10
    n = n + 1;
    y = y + 1/n;
end
disp(n)
```

y和n的初始值都是1，while后面的表达式是y<=10，即只要当前的y值小于等于10，循环就会继续执行。在循环体内，会更新n和y的值，直到满足y>10才会退出循环，此时的n就是最小的满足y>10的n.

二、循环语句

2. while-end语句

在使用 while 循环时，需要注意以下事项：

（1） 如果不小心执行了一个无限循环（即永远不会自行结束的循环，又称死循环），可以在脚本编辑器或命令行窗口中按下快捷键 Ctrl+C 来中断程序的运行。

MATLAB 的左下角会一直显示正忙： 


```
x = 1;
while x < 10
    disp(x)
end
```

二、循环语句

2. while-end语句

（2）while 后面表达式的计算结果不一定非得是逻辑值 1 或 0。如果表达式的计算结果是一个数值常数，则只有当这个常数为非零值时循环才会进行；若表达式的计算结果是一个数值向量或者矩阵，则仅当该向量或矩阵中的所有元素都是非零数时循环才会进行。

```
ii = 5;
while ii
    disp(ii)
    ii = ii - 1;
end
```

```
     5

     4

     3

     2

     1
```

```
x = [1 2; 3 4];
while x
    x(1) = 0;
    disp(x)
end
```

```
     0     2
     3     4
```

二、循环语句

3. break和continue

break 和 continue 也是 MATLAB 中的关键字，它们可以更加灵活地控制循环过程的执行。在 MATLAB 中，**break 和 continue 只能与 for 循环或 while 循环一同使用**，不能用于其他场合。下面我们来简要介绍一下 break 和 continue 的用法：

- ✧ **break 关键字**用于终止执行 for 或 while 循环。实际使用中，当满足某个条件时，我们会使用 break 立即退出循环。这在找到所需结果后立即退出循环的场景非常有用。
- ✧ **continue 关键字**用于跳过循环的当前迭代，然后继续下一次迭代。实际使用中，当满足某个条件时，continue 将跳过当前循环迭代的剩余部分，然后继续进行下一次迭代。这对于在某些情况下跳过特定的迭代非常有用，而不必完全退出循环。

二、循环语句

3. break和continue

（1）已知$y(n)=1+\frac{1}{2}+\frac{1}{3}+\cdots+\frac{1}{n}$，当$n$最小取多少时，$y$的计算结果大于10？这个例子在while函数中出现过，下面我们尝试使用for循环求解。

```
y = 0;
for n = 1:1e8
    y = y + 1/n;
    if y > 10
        disp(n)
        break  % 退出 for 循环
    end  % if 配套的 end
end  % for 配套的 end
```

使用 for 循环需要通过向量或者矩阵给出循环的次数，由于我们这个问题的循环次数是未知的，因此可以预先给一个很大的循环范围，我们这里给定 n 为 1 至 1×10^8构成的向量。
如果在循环体中找到了我们所需的结果（即 y 大于 10），就可以通过 break 关键字退出循环。通常情况下，判断条件是否成立需要用到 if 语句。

3. break和continue

（2）使用循环输出 1 至 10 中所有的奇数。

```
for i = 1:10
    if mod(i, 2) == 0
        continue
    end
    disp(i)
end
```

在每次迭代中，使用 mod(i, 2)来检查 i 是否为偶数。如果 i 是偶数，那么 mod(i, 2) 的结果将为 0，这时会执行 continue，直接跳到下一次迭代，因此当 i 为偶数时，代码后面的 disp(i)不会被执行；只有 i 为奇数时才会被输出。

思考：如果不使用 continue 关键字，代码应该如何修改？

注意，如果存在循环的嵌套，**break**和**continue** 仅在调用它的循环的主体中起作用。即**break** 仅从它所发生的循环中退出，**continue** 仅跳过它所发生的循环体内的剩余语句。