# 示例

|  |
| --- |
| @echo off    echo \*\*\*\*\* Replace "abcd" as "ABCD" \*\*\*\*\*  set strFilePath=./CBSD\_PROC\_RULE\_TEST1.dlv  set strOld=abcd  set strNew=ABCD  echo %strFilePath%  setlocal enabledelayedexpansion  for /f "tokens=\*" %%i in (%strFilePath%) do (  set "var=%%i"  echo %%i  if not !var!.==. (  set "var=!var:%strOld%=%strNew%!"  echo !var!!>>%strFilePath%.bk  )  )    move /y %strFilePath%.bk %strFilePath%    pause |

# IF

if用于条件判断，适用于以下情形  
1、判断驱动器、文件或文件夹是否存在，用 if exist 语句；  
2、判断某两个字符串是否相等，用 if "字符串1"=="字符串2" 语句；  
3、判断某两个数值是否相等，用 if 数值1 equ 数值2 语句；  
4、判断某个变量是否已经被赋值，用 if defined str 语句；  
[if语句](http://www.baidu.com/s?wd=if%E8%AF%AD%E5%8F%A5&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao)的完整格式是这样的：if [条件表达式](http://www.baidu.com/s?wd=%E6%9D%A1%E4%BB%B6%E8%A1%A8%E8%BE%BE%E5%BC%8F&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao) (语句1) else (语句2)，它的含义是：如果[条件表达式](http://www.baidu.com/s?wd=%E6%9D%A1%E4%BB%B6%E8%A1%A8%E8%BE%BE%E5%BC%8F&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "_blank)成立，那么，就执行语句1，否则，将执行语句2。  
对于以上四种情形，可以分别使用如下代码：  
1、if exist d:\test.txt (echo D盘下有test.txt存在) else (echo D盘下不存在test.txt)  
2、if "abc"=="xyz" (echo 字符串abc等于字符串xyz) else (echo 字符串abc不等于字符串xyz)  
3、if 1 equ 2 (echo 1等于2) else (echo 1不等于2)  
4、if defined str (echo 变量str已经被赋值，其值为%str%) else (echo 变量str的值为空)  
判断字符串是否相等的时候，if会区分大小写，比如，单纯的[if语句](http://www.baidu.com/s?wd=if%E8%AF%AD%E5%8F%A5&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "_blank)会认为字符串abc和字符串Abc不相同，若不想区分大小写，则需要添加 /i 开关，使用 if /i "字符串1"=="字符串2" 的格式；另外，等于符号是连续的"=="而非单独的"="。  
判断两个数值之间的大小关系，除了等于用equ之外，还有其他的关系符号，所有适用于[if语句](http://www.baidu.com/s?wd=if%E8%AF%AD%E5%8F%A5&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "_blank)的关系符号见下表：  
中文含义 关系符 英文解释  
等于 equ equal  
大于 gtr greater than  
大于或等于 geq greater than or equal  
小于 lss less than

# for

## 基本格式

(这里写的是在命令行里用的格式，如果是在批处理中,需要把其中%再多加个%形成%%)：

for /参数 %变量 in (集) do 命令

（注：上面除中文的以外，其余的是按它的格式要求书写的,大小写都行）  
参数：FOR分四种参数 D L R F，并且有的参数还可附加另外的选项下面会分别介绍  
变量：(记住如果是在批处理中使用for命令时，变量前的%需改为%%)这个变量名是由单个字母组成且区分大小写（原帮助是这么说的，实际运用中用单个数字作为变量名试过证明也可行），如%B和%b代表的是不同的变量。  
     FOR命令会在每次循环中，把in (集)中读取到的值赋于这个变量，以便其后的命令中引用。  
集：由系列文件、字符串或由命令产生的内容形成的集合（当然可用通配符 \* ？，还可引用环境变量），FOR命令是按一定顺序和规律分次读取集中内容，赋值给变量，并执行do后的命令，进行循环下一轮，直至集中内容读取完毕，而括号是格式必须的(in到后面括号之间要有空格)。  
命令：可以是任何合格的DOS命令或外部可被DOS调用的程序，且可采用括号把多条命令括起来，在一次循环中执行。  
附注：由于一些目录或文件名可能会有空格，所以很多时候集里和命令里往往需要用英文引号括起来(但有时引号里的内容可能会被认为是字符串)表示是一整体，下面开始的有些例中为简捷起见，忽略文件名或目录名带空格这种情况。  
现在按参数分类举例解释其用法：

## 一、参数/d

for /d %%变量 in (集) do 命令  
/d 参数是指定仅对目录而不是文件执行的for命令。  
例1:  
在命令行输入（不是在批处理，之后不再解释）  
for /d %a in (c:\\*.\*) do echo %a  
运行会把C盘根目录下的全部目录分次显示出来,而不显示文件名  
看起来有点乱，如果把命令提示回显关闭就清晰了:  
for /d %a in (c:\\*.\*) do @echo %a

## 二、参数/R

/R参数之后还可带盘符及路径  
for /r 此处可以带有路径 %变量 in (集) do 命令  
在/r 之后的那个路径，指包含它之下的整个目录树（相当于DOS命令tree里的范围）中的所有目录，如果仅为一个英文句点 . ，是指当前路径下的目录树，如果省略了路径则特指当前目录，而之后的in (集)则相当于与前面每个目录相配的文件集  
这里按in(集)中有无通配符分两种情况  
1) in(集)中没有通配符  
指定的是单个文件或列举的具体文件(多个文件名之间用分隔符分隔，如空格、逗号等)  
例2

[?](https://www.jb51.net/article/93171.htm)

|  |  |
| --- | --- |
| 1  2  3 | @echo of  for /r . %i in (abc.txt) do echo. > %i  echo on |

注：这里for /r 后的路径仅有一个 . 而后面每个循环中echo. > %i相当于创建一个仅有一空行的文本文件，整体效果是在当前目录下包括子录，每个目录中建一个abc.txt。  
例3 （放入批处理中）

[?](https://www.jb51.net/article/93171.htm)

|  |  |
| --- | --- |
| 1  2  3  4 | @echo off  rem 显示d:盘中所有文件名为file1和file2的列表  for /r d:\ %%h in (file1,file2) do if exist %%h echo %%h  pause |

2) in(集)中含有通配符\*或？  
这种里面的do命令将处理前面 /r指定的目录系列里每个含有in(集)中文件的项，而不去理会不含有相配文件的那些目录  
例4：

[?](https://www.jb51.net/article/93171.htm)

|  |  |
| --- | --- |
| 1  2  3  4 | @echo off  rem 删除C盘中所有\*.chk的文件  for /r c:\ %%h in (\*.chk) do del /q %%h  pause |

注：del /q 表示用安静模式删除（不需确认）

## 三、参数/L

for /L %%变量 in (起始值，每次增值，结束时的比较值) do 命令  
（上面L也可用小写，主要为了视觉上不与数字1混淆而没用小写）  
(起始值，每次增值，结束时的比较值)相当于一个等差数字序列，从“起始值”的数字开始，每次增加多少（也可设定为负数）为“每次增值”，并与“结束时的比较值”比较，超出则退出for循环（也不执行本轮后面的do 命令）  
例如 (1,1,3) 将产生序列 （1 2 3）；（1,2,9）将产生序列（1 3 5 7 9）；(5,-1,1) 将产生序列 (5 4 3 2 1)；（1,3,18）将产生序列（1 7 10 13 16）  
例5

[?](https://www.jb51.net/article/93171.htm)

|  |  |
| --- | --- |
| 1  2  3  4 | @echo off  ::在D盘建立aa1~ aa5五个文件夹  for /L %%i in (1,1,5) do md d:\aa %%i  pause |

注：在行首，单个冒号:接一名称，是标号行，对应于批处理中go后指向的位置，而双冒号::一般是用来作注释用，注释在批处理中可以用rem加空格来表达，二者稍有不同，rem注释在未关闭命令回显时会在屏幕显示出来，而::则什么情况下都不会显示。

## 四、参数/f

这个参数/f将会打开（集）里的文件，使for命令能处理文本文件的读取和添加删除替换等编辑性的操作，可谓功能强大，因此也相对复杂一些。  
                                                 文件名-集       
for /f “选项” %变量 in （ “字符串”-集 ） do 命令  
                                              ‘命令'-集  
/f 后可以带有几种选项，不带选项当然也是合格的格式，而带有参数则必须以引号整体括起来，后面的集里主要由三种形式形成的，最终在for循环中的每一轮中会形成读取一行字符串，来给指定的%变量、以及给由于选项中派生出附加变量赋值后，执行do后面的命令  
下面以例子来具体说明和逐步理解各分项的用法  
例6  
假定d:\abc.txt内容如下：  
姓名 性别 年龄 等-级  
张三 男 36   A-1  
李四 男 29   B-2  
赵六 女 31   A-2

执行如下命令：  
for /f %c in (d:\abc.txt) do @echo %c  
则屏幕上显示：  
姓名  
张三  
李四  
赵六

解释：这是for /r 在“%变量”前缺省参数选项时的情况，循环中每轮会默认以空格为分隔，在打开的文件中逐行给字符串分段，又因为没给增添附加变量（即仅一个变量%c）则仅把第一段的字符赋给%c，再执行 do后的命令，然后进行循环的下一轮，并且默认忽略空行  
改一下：  
for /f “skip=1 tokens=1,4 delims= ” %c in (d:\abc.txt) do @echo %c %d  
显示为：  
张三 A-1  
李四 B-2  
赵六 A-2

解：  
skip=1 表示文本开始忽略的行数为1 ——忽略几行  
    delims= 在一行中，用什么单个符号（可以有多字符组合，之间也不能加空格，被理解为多项单个字符，如要空格符须放最后）来分隔字符串作为读取赋值的单元(形成一段)，本例中等号后是空的表示仅用空格来分隔。——用什么刀来切分  
    tokens=1,4 这个等号后的数字表示依次取第几个被分隔的字符串段，来分别赋给%变量及顺序附加的变量，本例取第1个段赋给%c，第4个段赋给c后的一个变量也就是赋给%d，并且，可以写成tokens=1,2,5-7 或tokens=1,2,3\* 或tokens=1,2,5,7 分别表示取第1,2,5,6,7（依次赋给%c, %d, %e, %,f, %g共5个变量）、1,2,3及3后的所有段（要赋给3个变量）、1,2,5,7(要赋给4个变量)，tokens=后的数字号可以不按顺序，但书写的顺序与分配给变量的顺序是对应的，这是赋值，至于之后do命令中用不用是另一回事。换句话 --——最多只需取哪几段  
in (变量) 中的那个变量，代表起始的一个变量名，按tokens中定义的总个数来扩充附加变量名，如总个数为3，则%c 就附加%d和%e ，要是%C就附加%D%E… 本例中tokens=1,4仅需两个，起始的是in () 括号中的%c 则每行中第一段赋给%c,第4段赋给变量%d  
以第二行（第一行被skip=1跳过了）为例，在 “张三 男 36 A-1 ” 中（正好也是用的空格分隔）共被空格之刀切为五段，只要第1、4，即张三赋给%c, A-1赋给%d，执行@echo %c %d然后下一轮…而空行照旧被省去了。

再稍改一下：

for /f “skip=1 tokens=4,1 delims=- “ %c in (d:\abc.txt) do @echo %c %d

则显示为：  
A 张三  
B 李四  
A 赵六

例7  
假定d:\aa.txt内容如下：

Volume in drive D is MYDA  
Volume Serial Number is C35D-8998

Directory of D:tmp

09/25/2001 10:40 AM 11,235 yg0925.txt  
11/12/2001 04:29 pM 795 buple.txt  
04/11/2002 04:18 AM 2,043 vitn.txt  
3File(s) 12,673 bytes  
0 Dir(s) 5,020,200,655 bytes free

在命令行输入：  
for /f "skip=5 tokens=5" %a in (d:\aa.txt) do @echo %a  
会显示：  
yg0925.txt  
buple.txt  
vitn.txt  
free  
    本意想把文件里列出的文件显示出来（当然也可以换成对文件进行其他命令操作）  
通过skip=5 忽略掉前5行，默认以空格分隔后tokens=5取每行第五段字符就顺利地把文件名赋给变量%a,美中不足最后一行取了个不是文件名的（当然可用其他方法处理这个多余的只是for/f中没提供忽略最后几行的格式），而倒数第二行则无第五段。  
显然例中aa.txt里的内容是某次执行dir命令后的内容。它可用类似命令：  
dir > d:\aa.txt来建立  
   题外话，如果在dir中加入合适的参数/b，就可以回避多余的部分,还可加入/ad只显示目录，加入/a-d只显示文件等  
那么，我们完全可以直接书写命令放入in后的（‘命令'-集）中  
for /f "skip=5 tokens=5 " %a in ('dir') do @echo %a

效果一样。  
注：命令集需用单引号括起来以表示不是文件集，如用双引号括起来则表示是字符串集，本例是为了说明for命令的用法，真正有这种用途也愿意用前面“题外话”的方法。如果你在执行本例后什么也没显示，你需要先用集里的命令先执行一次，看它显示的格式，也许需要把tokens=5 改成tokens=4 或许还应当给dir加上参数 /a-d以回避显示出目录。  
如果集里是由多个文件组成，那么处理完一个文件后又处理完又去处理另一个文件，每个文件行数不同循环次数（do命令的次数）也将因此不同。  
如果集里是由命令产生的系统，那么你必须首先熟悉该命令执行后会产生怎样效果的字符系统，才能正确安排后面的do命令

画龙点睛：无论in后的集是哪种形式，for/f 都最终分解为字符串，按需要是否“忽略几行”（skip=）、“用什么刀来切分”（delims= ）、“最多只需取哪几段”（tokens=）将集里形成的字符串,逐行地分段赋给%或%%后的变量及可能顺延扩展出的变量，以执行do后的命令，每一行即为一轮循环。这里没完整说明全部参数，请在命令行用for/?查看。(下面的斜体字是复制的帮助里的内容)  
例如：  
对于带有空格的文件名，您需要用双引号将文件名括起来。为了用这种方式来使用双引号，您还需要使用 usebackq 选项，否则，双引号会被理解成是用作定义某个要分析的字符串的。——换句话说,带有usebackq（放在for /f 之后的引号里）参数时 in ()里用双引号表示的仍是文件名。  
还有一个选项eol= ：前面所说skip=是表示忽略开始的几行，其实默认状况还忽略所有分号“ ; ”开始的行，如果你想不忽略分号开始的行，或者想忽略自己指定一字符开始的行就可以在for /f 之后那引号参数里使用eol=你自己定义的字符，但它不像delims=的那样可定义多个，只允许定义一个。

另一花样：可以用 %~ 操作符将文件名分离成文件名、扩展名、盘符等独立部分 ，请看for/?中的解释（其中示例的变量为%I）：  
另外，FOR 变量参照的替换已被增强。您现在可以使用下列选项语法:

~I - 删除任何引号(")，扩充 %I  
%~fI - 将 %I 扩充到一个完全合格的路径名  
%~dI - 仅将 %I 扩充到一个驱动器号  
%~pI - 仅将 %I 扩充到一个路径  
%~nI - 仅将 %I 扩充到一个文件名  
%~xI - 仅将 %I 扩充到一个文件扩展名  
%~sI - 扩充的路径只含有短名  
%~aI - 将 %I 扩充到文件的文件属性  
%~tI - 将 %I 扩充到文件的日期/时间  
%~zI - 将 %I 扩充到文件的大小  
%~$PATH:I - 查找列在路径环境变量的目录，并将 %I 扩充  
到找到的第一个完全合格的名称。如果环境变量名  
未被定义，或者没有找到文件，此组合键会扩充到空字符串

可以组合修饰符来得到多重结果:

%~dpI - 仅将 %I 扩充到一个驱动器号和路径  
%~nxI - 仅将 %I 扩充到一个文件名和扩展名  
%~fsI - 仅将 %I 扩充到一个带有短名的完整路径名  
%~dp$PATH:I - 查找列在路径环境变量的目录，并将 %I 扩充  
到找到的第一个驱动器号和路径。  
%~ftzaI - 将 %I 扩充到类似输出线路的 DIR

简记：凡是 %~ 打头的操作符，都是文件名或环境变量的分离操作。而每项要想运用自如，则需要付出辛勤的练习。

## 练习

遍历C、D盘，查找已知文件名(接收键盘输入)，把其存放位置、时间，记录到D:\mynote.txt 记录格式如：

xx年xx月xx日 经查找在C盘、D盘的xx文件情况如下：  
   时间             位置  
。。。。。 。。。。。。  
。。。。。 。。。。。。  
。。。。。 。。。。。。  
。  
。  
。

提示：可能用到的DOS命令、变量、参数： echo、set 、set/p 、%date%、%~    >、>>

总结及提示：  
for 命令的实际用法基本上已终结，但是仅此是不能写出强大功能的批处理的，它只是一条DOS命令，需要熟练一些其他的DOS命令和Windows系统提供的命令，组合运用，才能充分发挥其强大、实用的功能，使得一些复杂事情，处理起来意想不到的简洁方便。

附：常见在批处理for命令中需要的一个的命令或者叫环境设置:  
for命令实际上是会作循环，如果在每轮的命令中改变某环境变量值，在默认状态，一条for命令用%环境变量%只取一次值的,那么下轮循环中再用时还是改变前的值（包括do后面带有括号里的多条命令的执行期间），就没达到预期目的，为此，引入下面命令：  
setlocal enabledelayedexpansion  
开始批处理文件中环境改动的本地化操作,并启动延缓环境变量扩展。在执行SETLOCAL 达到批处理文件结尾时，对于该批处理文件的每个尚未执行的 setlocal 命令，都会有一个隐含的 endlocal 被执行。  
在取变量值时，用!变量名!可以动态取值，延迟环境变量扩充允许您使用一个不同的字符(惊叹号)在执行时间扩充环境变量。这个用法实际是属于在批处理中所有复合型命令都需要注意的。如果批处理结束后不希望将改变的环境保留，建议总是加上setlocal 。  
要是结合一些其他复杂些的有关系统的、网络的命令（如wmic、net）进来，那才是方显FOR英雄本色，比如遍历本地磁盘可以用命令：wmic logicaldisk where "drivetype=3" get name 显然要在所有磁盘里查找某文件并作相应操作就很容易了，用好for命令也是需要其他命令和计算机基础配合的。呵呵，本人水平有限，写的只是低级层次的...但愿能对有缘来这里看的初学DOS的FOR命令者有所帮助。