NAME

for, foreach, repeat – shell built-in functions to repeatedly execute action(s) for a selected number of times

SYNOPSIS

sh

for *word* [**in** *wordlist*. . .] **;** **do** *actions* **; done**

csh

foreach *word* **(***wordlist***)**
. . .
end

repeat *count command*

ksh

for *word* [**in** *wordlist* . . .] **;** **do** *actions* **; done**

DESCRIPTION

sh

Each time a **for** command is executed, *word* is set to the next item taken from the **in** *wordlist*. If **in** *wordlist* **. . .** is omitted, then the **for** command executes the **do** *actions* once for each positional parameter that is set. Execution ends when there are no more words in the list.

csh

The variable *word* is successively set to each member of *wordlist*. The sequence of commands between this command and the matching **end** is executed for each new value of *word*. Both **foreach** and **end** must appear alone on separate lines.

repeat executes *command* repeatedly *count* times. *count* must be a number. *command* is restricted to a one-line statement.

ksh

Each time a **for** command is executed, *word* is set to the next item taken from the **in** *wordlist*. If **in** *wordlist* **. . .** is omitted, then the **for** command executes the **do** *actions* once for each positional parameter that is set. Execution ends when there are no more words in the list.

loop interrupts

The built-in command **continue** may be used to terminate the execution of the current iteration of a **for** or **foreach** loop, and the built-in command **break** may be used to terminate execution of a **for** or **foreach** command.

EXAMPLES

In the examples using **for**/**foreach**, the code counts the number of lines for each file in the current directory whose name ends with a ".c" extension. The **repeat** example prints "I will not chew gum in class" 500 times.

sh

```
for file in *.c ; do wc -l $file ; done
```

csh

```
foreach file ( *.c )
        wc -l $file
end
```

ksh

```
for file in *.c ; do wc -l $file ; done
```

csh

The **repeat** command re-executes the single subsequent *command* for *count* number of times.

```
@ repetition = 500
repeat $repetition echo "I will not chew gum in class."
```

SEE ALSO

break(1), **csh**(1), **ksh**(1), **sh**(1)

NOTES

Both the Bourne shell, **sh**, and the Korn shell, **ksh**, can use the semicolon and the carriage return interchangeably in their syntax of the **if**, **for**, and **while** built-in commands.