

Chapter 7

Loops

Overview

- The loop with counter
- The conditional loop
- break and continue
- Reading Lines From Files
- Using Arrays with Loops

Lesson: The for loop

Repeat the loop for each member of the list.

■ Syntax:

```
for var in word ...  
do  
    list_A  
done
```

■ Example:

```
$ cat test_for  
for num in 1 2 3 4 5 6 7 8 9  
do  
    echo "The number is $num"  
done  
  
$ ./test_for  
The number is 1  
The number is 2  
...  
The number is 9
```



Lesson: The conditional loop

- while loop
- until loop

The while construct

Repeat the loop while the condition is true.

Syntax:

```
while
  list_A
do
  list_B
done
```



Example:

```
$ cat test_while
X=1
while [ $X -le 10 ]
do
  echo hello x is $X
  let X=X+1
done

$ ./test_while
hello x is 1
hello x is 2
...
hello x is 10
```

The until construct

Repeat the loop as long as the condition is false.

Syntax:

```
until
  list_A
do
  list_B
done
```



Example:

```
$ cat test_until
X=1
until [ $X -gt 10 ]
do
  echo hello x is $X
  let X=X+1
done

$ ./test_until
hello x is 1
hello x is 2
...
hello x is 10
```

Lesson: break and continue

- Breaking Out of a Loop
- The continue Command

Breaking Out of a Loop

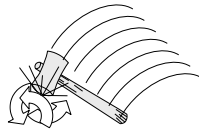
■ Syntax

```
break [n]
```

- When the `break` command is used, the shell ceases processing the loop commands and executes the next command line that follows `done`.

■ Example

```
...  
do  
...  
break  
...  
done  
next_command
```



The continue Command

■ Syntax

```
continue [n]
```

- When the `continue` command is used, the shell starts next iteration of the loop.

■ Example

```
...  
do  
...  
continue  
...  
done  
next_command
```



Lesson: Reading Lines From Files

- Using while
- Using for

Reading file content with loop

- while loop can be part of pipe
- Redirecting input for while loop

```
ls -al | while read perms links owner group size
mon day time file
do
  if [ "$perms" != "total" ] && [ $size -gt 100 ]
  then
    echo "$file $size"
  fi
done | sort -rn -k 2
```

```
while read animal name
do
  echo "Animal is $animal, name is $name."
done < pets
```

Reading file contents

- For loop can get file contents by words

```
for file in $(cat config_files)
do
  perms=$(ls -l "$file" | cut -c2-10)
  echo "$file \t$perms"
done
```

```
cat file | while read line
do
  for word in $line
  do
    echo "the next word: $word"
  done
done
```

Lesson: Using Arrays with Loops

- Array Variables
- Substituting and Counting All Elements
- Using Integer Variables as Element Numbers

Array Variables

- An array variable
 - Can be used to store multiple values
 - Is one-dimensional
 - Can hold up to 1024 elements (in the range 0 - 1023)

```
varname[integer]=value
```
- Array can be assigned values using set command

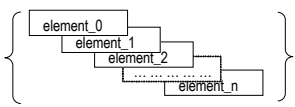

```
$ set -A array value ...
```
- The contents of an array variable can be viewed using the syntax:


```
$ echo ${varname[integer]}
```

Substituting and Counting All Elements

- The entire contents of an array variable can be viewed using the syntax:


```
$ echo ${varname[*]}
```


- The number of elements being used can be displayed using the syntax:


```
$ echo ${#varname[*]}
```

Using Integer Variables as Element Numbers


- An integer variable can be used instead of a literal number.

```
$ echo ${varname[x]}
```

- There is no requirement to use `$x` because an element identifier would always be an integer value. The variable, `x`, must contain an integer variable.


```
typeset -i x=0
while [ x -lt ${#varname[*]} ]
do
  echo ${varname[x]}
  (( x = x + 1 ))
done
```

Review Exercises



- Complete the exercises from the Learning Guide

Topics for Review



- 1 Read the review topics
- 2 Think about what you learned in this Session in the context of your own work environment
- 3 Discuss your answers as a class
