

Shell

Deci\$ion Making and Loops

Divya Kumar

CSED, MNNIT

Decision Making in Shell

if [expression]

then

Statement(s) to be executed if expression
is true

fi

Decision Making in Shell (cntd...)

if [expression]

then

Statement(s) to be executed if expression is
true

else

Statement(s) to be executed if expression is
not true

fi

Decision Making in Shell (cntd...)

if [expression 1]

then

Statement(s) to be executed if expression 1 is
true

elif [expression 2]

then

Statement(s) to be executed if expression 2 is
true

else

Statement(s) to be executed if no expression is

Decision Making in Shell (cntd...)

Try it!!

```
#!/bin/sh
```

```
a=10
```

```
b=20
```

```
if [ $a == $b ]
```

```
then
```

```
    echo "a is equal to b"
```

```
elif [ $a -gt $b ]
```

```
then
```

```
    echo "a is greater than b"
```

```
elif [ $a -lt $b ]
```

```
then
```

```
    echo "a is less than b"
```

```
else
```

Decision Making in Shell (cntd...)

Switch Case Statements

case word in

pattern1)

Statement(s) to be executed if pattern1 matches

;;

pattern2)

Statement(s) to be executed if pattern2 matches

;;

pattern3)

Statement(s) to be executed if pattern3 matches

;;

Decision Making in Shell (cntd...)

Switch Case Statements

```
#!/bin/sh
```

```
FRUIT="kiwi"
```

```
case "$FRUIT" in
```

```
    "apple") echo "Apple pie is quite tasty."
```

```
;;
```

```
    "banana") echo "I like banana nut bread."
```

```
;;
```

```
    "kiwi") echo "New Zealand is famous for kiwi."
```

Decision Making in Shell (cntd...)

Switch Case Statements most usage!

```
#!/bin/sh
```

```
option="${1}"
```

```
case ${option} in
```

```
  -f) FILE="${2}"
```

```
    echo "File name is $FILE"
```

```
    ;;
```

```
  -d) DIR="${2}"
```

```
    echo "Dir name is $DIR"
```

```
    ;;
```

```
  *)
```

```
    echo "`basename ${0}`:usage: [-f file] | [-d directory]"
```

```
    exit 1 # Command to come out of the program with status 1
```


Loops in Shell(cntd...)

For loops

```
for var in word1 word2 ... wordN
```

```
do
```

```
    Statement(s) to be executed for every word.
```

```
done
```

```
#!/bin/sh
```

```
for var in 0 1 2 3 4 5 6 7 8 9
```

```
do
```

Loops in Shell(cntd...)

For loops

```
#!/bin/sh
```

```
for FILE in $HOME/.bash*  
do  
    echo $FILE  
done
```

This will produce following result:

```
/root/.bash_history  
/root/.bash_logout
```

Loops in Shell(cntd...)

For loops more examples:

```
$ cat for1.sh
```

```
i=1
```

```
for day in Mon Tue Wed Thu Fri
```

```
do
```

```
    echo "Weekday $((i++)) : $day"
```

```
done
```

```
$ ./for1.sh
```

```
Weekday 1 : Mon
```

```
Weekday 2 : Tue
```

Loops in Shell(cntd...)

For loops more examples (in-list):

```
$ cat for2.sh
```

```
i=1
```

```
weekdays="Mon Tue Wed Thu Fri"
```

```
for day in $weekdays      # if "$weekdays" then single iteration
```

```
do
```

```
    echo "Weekday $((i++)) : $day"
```

```
done
```

```
$ ./for2.sh
```

```
Weekday 1 : Mon
```

```
Weekday 2 : Tue
```

Loops in Shell(cntd...)

For loops more examples (using positional parameters):

```
$ cat for3.sh
```

```
i=1
```

```
for day
```

```
do
```

```
    echo "Weekday $((i++)) : $day"
```

```
done
```

```
$ ./for3.sh Mon Tue Wed Thu Fri
```

```
Weekday 1 : Mon
```

```
Weekday 2 : Tue
```

```
Weekday 3 : Wed
```

Loops in Shell(cntd...)

For loops more examples (getting files and directories):

```
$ cat for5.sh
```

```
i=1
```

```
cd ~
```

```
for item in *                               ##(for all *.c, a*.....)
```

```
do
```

```
    echo "Item $((i++)) : $item"
```

```
done
```

```
$ ./for5.sh
```

```
Item 1 : positional-parameters.sh
```

Loops in Shell(cntd...)

For loops more examples (c style):

```
$ cat for8.sh
```

```
for (( i=1; i <= 5; i++ ))
```

```
do
```

```
    echo "Random number $i: $RANDOM"
```

```
done
```

```
$ ./for8.sh
```

```
Random number 1: 23320
```

```
Random number 2: 5070
```

```
Random number 3: 15202
```

```
Random number 4: 23861
```

Loops in Shell(cntd...)

For loops more examples (c style):

```
$ cat for9.sh
```

```
i=1;
```

```
for (( ;; ))
```

```
do
```

```
    echo "Number: $((i++))"
```

```
done
```

>>Infinite loop

>>use ctrl c

Loops in Shell(cntd...)

For loops more examples (c style):

```
$ cat for10.sh
```

```
for ((i=1, j=10; i <= 5 ; i++, j=j+5))
```

```
do
```

```
    echo "Number $i: $j"
```

```
done
```

```
$ ./for10.sh
```

```
Number 1: 10
```

```
Number 2: 15
```

Loops in Shell(cntd...)

For loops more examples (c style):

```
$ cat for11.sh  
for num in {1..10}  
do  
    echo "Number: $num"  
done
```

```
$ ./for11.sh
```

```
Number: 1
```

```
Number: 2
```

Loops in Shell(cntd...)

For loops more examples (c style):

```
$ cat for12.sh  
for num in {1..10..2}  
do  
    echo "Number: $num"  
done
```

```
$ ./for12.sh
```

```
Number: 1
```

```
Number: 3
```

Loops in Shell(cntd...)

While loop:

while command

do

Statement(s) to be executed if command is true

done

Loops in Shell(cntd...)

While loop:

```
#!/bin/sh
a=0
while [ $a -lt 10 ]
do
    echo $a
    a=`expr $a + 1`
done
```

This will produce following result:

0

1

Loops in Shell(cntd...)

Until loop:

until command

do

Statement(s) to be executed until command is
true

done

Loops in Shell(cntd...)

Until loop:

```
#!/bin/sh
```

```
a=0
```

```
until [ ! $a -lt 10 ] # as long as this fails the loop continues
```

```
do
```

```
    echo $a
```

```
    a=`expr $a + 1`
```

```
done
```

Loops in Shell(cntd...)

- Shell Loop Controls

continue

continue n

break

break n

Loops in Shell(cntd...)

select loop:

The *select* loop provides an easy way to create a numbered menu from which users can select options. It is useful when you need to ask the user to choose one or more items from a list of choices.

This loop was introduced in ksh and has been adapted into bash. It is not available in sh.

Loops in Shell(cntd...)

select loop:

Syntax:

```
select var in word1 word2 ... wordN
```

```
do
```

```
    Statement(s) to be executed for every word.
```

```
done
```

Loops in Shell(cntd...)

select loop:

```
select DRINK in tea cofee water juice appe all none
do
```

```
case $DRINK in
```

```
tea|cofee|water|all)
```

```
    echo "Go to canteen"
```

```
    ;;
```

```
juice|appe)
```

```
    echo "Available at home"
```

```
    ;;
```

```
none)
```

```
    break
```

```
    ;;
```

```
*) echo "ERROR: Invalid selection"
```

Loops in Shell(cntd...)

select loop:

```
$/test.sh
```

```
1) tea
```

```
2) cofee
```

```
3) water
```

```
4) juice
```

```
5) appe
```

```
6) all
```

```
7) none
```

```
#? juice
```

```
Available at home
```

```
#? None
```

```
$
```

Loops in Shell(cntd...)

For loops more examples (break outs and continue):

1) Even numbers from 2 to 20.

2) Even numbers from a file num.txt

```
>cat num.txt
```

```
44 67 95 32 89 2 98
```

Using:

```
break;
```

```
continue;
```

Loops in Shell(cntd...)

Selection Sort:

```
echo "Enter Numbers to be Sorted : "
```

```
read -a ARRAY
```

```
count=${#ARRAY[@]}
```

```
echo "-----"
```

```
echo "Numbers Before Sort:"
```

```
printnumbers
```

```
sortnumbers
```

```
echo "Numbers After Sort: "
```

Loops in Shell(cntd...)

Selection Sort:

```
printnumbers()  
{  
    echo ${ARRAY[*]}  
}
```

```
swap()  
{  
    temp=${ARRAY[$1]}  
    ARRAY[$1]=${ARRAY[$2]}  
    ARRAY[$2]=$temp  
}
```

Loops in Shell(cntd...)

Selection Sort:

```
sortnumbers()
{
for ((i=0;i<count;i++))
do
    min=$i
    for ((j=i+1;j<count;j++))
    do
        if [ ${ARRAY[j]} -lt ${ARRAY[min]} ]
        then
            min=$j
        fi
    done
done
```


Loops in Shell(cntd...)

Selection Sort:

```
]# sh selectionsort.sh
```

Enter Numbers to be Sorted :

34 76 -8 12 23 5 9 -2 88 41 62

Numbers Before Sort:

34 76 -8 12 23 5 9 -2 88 41 62

Numbers After Sort:

-8 -2 5 9 12 23 34 41 62 76 88

Shell Wildcards (cntd...)

<i>Wildcard</i>	<i>Matches</i>
*	zero or more characters
?	exactly one character
[abcde]	exactly one character listed
[a-e]	exactly one character in the given range
[!abcde]	any character that is not listed
[!a-e]	any character that is not in the given range
{debian,linux}	exactly one entire word in the options

Shell Wildcards (cntd...)

\$ rm *

Removes every file from the current directory

\$ mv *linux*.html dir1

Moves all the HTML files, that have the word "linux" in their names, from the working directory into a directory named dir1

Shell Wildcards (cntd...)

\$ rm junk.???

Removes all files whose names begin with junk., followed by exactly three characters

\$ ls hda[0-9]

List all files or directories whose names begin with hda, followed by exactly one numeral

Shell Wildcards (cntd...)

\$ ls hda[0-9][0-9]

Lists all files or directories beginning with hda, followed by exactly two numerals

\$ ls {hd,sd}[a-c]

Lists all files or directories whose name starts with either hd or sd, followed by any single character between a and c

Shell Wildcards (cntd...)

```
$ cp [A-Z]* dir2
```

Copies all files, that begin with an uppercase letter, to directory dir2

```
$ rm *[^cehg]
```

Deletes all files that don't end with c, e, h or g.