Shell Information

#!/bin/dash
: `first line of file`

command `>` *filename*
: `write output to` *filename*

command `>>` *file*
: `append output to` *filename*

command `2>` *filename*
: `write stderr to` *filename*

command `2>&1`
: `write stderr to stdout`

command `>` *file* `2>&1`
: `write stdout and stderr to` *filename*

command `<` *filename*
: `input from` *filename*

command `<<` *EOF*
: `heredoc until` *EOF*

$command_1$ `|` $command_2$
: `pipe output from` $command_1$
`as input to` $command_2$

$command_1$ `&&` $command_2$
: `execute` $command_2$ `if` $command_1$
`has exit status zero`

$command_1$ `||` $command_2$
: `execute` $command_2$ `if` $command_1$
`does not have exit status zero`

`$((`*`expression`*`))`

  `expression` `evaluated as arithmetic`

`$#` = count of command-line arguments
`$0` = name of currently executing command
`$1,$2,$3,...,$9,${10},...,${255}` = command-line arguments
`$@` = list of all command-line arguments
`$*` = list of all command-line arguments
`$?` = exit status of previous command

`read` *`varName`*

  sets value of variable *`varName`* to
  next line read from `stdin`

`'`*`str`*`'` = *`str`*
`"`*`str`*`"` = *`str`* `with variables interpolated`
`` ` ``*`command`*`` ` `` = `output of` *`command`* `as string`
`$(`*`command`*`)` = `output of` *`command`* `as string`

Zero exit status means true/successful
Non-zero exit status means false/failure

`test` *`expression`*
`[` *`expression`* `]`

  returns *`expression`* result as exit status
  `integer operators: -lt,-gt,-eq,-ne,-ge,-le`
  `string operators: =, -z, -n`
  `file operators: -d, -e, -f, -s, -nt`

`exit` *`Number`*

  terminate script with exit status *`Number`*

```
if Command_a ; then
   Commands_1
elif Command_b ; then
   Commands_2
else
   Commands_3;
```

```
fi

case Word in
Pattern_1) Commands_1 ;;
Pattern_2) Commands_2 ;;
...
*)         Commands_n ;;
esac

while Command ; do
   Commands
done

for var in Word_1 Word_2 ...
do
   Commands
done

# Display lines from file
count=0
while read line
do
   count=$((count + 1))
   echo "Line $count: $line"
done <file

# Interactively rm files in current dir
for f in *
do
   echo -n "Remove $f? "
   read answer
   if test $answer = y
   then
        echo $f
   fi
done
```

Regular Expressions

Atomic Patterns:

letters, digits, punctuation (except those below)
    match any occurrence of themselves
`\. \* \+ \? \| \^ \$ \[ \]`
    match any occurrence of the second character
`.` (dot)
    matches any single character
(*pattern*)
    matches *pattern*

Anchors:

^*pattern*
    matches *pattern* at the start of a line
pattern$
    matches *pattern* at the end of a line

Selection:

[*charList*]
    matches any single character in *charList*
[^*charList*]
    matches any single character not in *charList*
$pattern_1$|$pattern_2$|$pattern_3$|...
    matches any of the $pattern_i$s

*charList*s use c_1-c_2 to denote char ranges, and
meta-characters lose their special meaning inside *charList*s

Repetition:

pattern?
    zero or one occurrences of *pattern*
*pattern**
    zero or more occurrences of *pattern*
pattern+
    one or more occurrences of *pattern*

`\w` matches alphanumeric, including `'_'`
`\s` matches whitespace
`\d` matches numeric
`\b` word boundary

pattern{*N*,*M*}
matches *N* to *M* occurrences of *pattern*