Filters & Text Manipulation Tools

Filters

Printing: pr-sh

% ls /home/cs476 | pr -h "-cs476-" -l 20 -2 -n -d | more -20

-h header, -l page length, -2 columns, -n add line numbers, -d double space

File Comparison: compare-sh

% paste f1 f2

|   |   
|---|---
| a | a |
| b | b |
| c | d |
|   | e |

% cmp f1 f2

f1 f2 differ: char 5, line 3

% comm f1 f2

<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
</tr>
<tr>
<td>b</td>
</tr>
<tr>
<td>c</td>
</tr>
<tr>
<td>d</td>
</tr>
<tr>
<td>e</td>
</tr>
</tbody>
</table>

% comm -12 f1 f2

-12 suppress columns 1 and 2.

% diff f1 f2

3c3,4
< c
---
> d
> e
**Head & Tail:**  

```bash
% cat fl
  a
  b
  c
% head -n 1 fl
  a
% tail -n 1 -f fl
  c
```

**usr/bin/tail options:**

- `-n` no of lines,  
- `-f` force wait  
- `-r` reverse lines

**Exercise:** getting the middle  

```bash
echo "Usage: middle-sh <head> <tail> <file>"
length=`wc -l < $3`
top=`expr $length - $2`
bottom=`expr $top - $1`
cat $3 | head -n $top | tail -n $bottom
```

**Cut & Paste:**  

```bash
% cat fields
  Wahab, Hussein
  Maly, Kurt
% cut -d, -f2 fields > F
  Hussein
  Kurt
-d delimiter,  
-f field number
% cut -d, -f1 fields > L
% cat L
  Wahab
  Maly
% paste -d " " F L > names
% cat names
  Hussein Wahab
  Kurt Maly
% cat names emails
  Hussein Wahab
```
-u unique, -t field separator, -k field number -m merge

**Uniq:** uniq-sh

```bash
% cat grades
1: A
2: B
3: C
4: D
5: F
6: A
7: A
8: B
9: F
```

```bash
% cut -d: -f2 grades | sort | uniq -c
3  A
2  B
1  C
1  D
2  F
```

-c count the occurrence of each value

**Translate:** translate-sh

```bash
% cat fields | tee /dev/tty | tr 'a-z' 'A-Z' | tee /dev/tty | tr 'A-Z' 'a-z'
```

```
Wahab, Hussein
Maly, Kurt

WAHAB, HUSSEIN
MALY, KURT

wahab, hussein
maly, kurt
```
Egrep:  egrep-sh

% ls -l /home/cs476 | egrep "^d.*[M|m]ail.*"

```
drwx------ 2 cs476 cs476 512 Oct 22 1999 Mail
drwx------ 2 cs476 cs476 512 Dec 9 2003 mail
drwx------ 2 cs476 cs476 512 Nov 3 1998 nsmail
```

% ypcat passwd | egrep "^cs[5-9][0-9]+:" | cut -d: -f1 | sort -u | tee /dev/tty | wc -l

```
cs554
cs555
cs558
cs656
cs695
cs745
cs772
cs775
cs778
cs779
cs845
12
```

Sed:  sed-sh

% cat grades

```
1: A
2: B
3: C
4: D
5: F
6: A
7: A
8: B
9: F
```

% sed "s/^/000/; s/A/Excellent/; s/B/Very Good/; s/C/Good/; s/D/Pass/; s/F/Fail/" grades

```
0001: Excellent
```
**Note:**

You may put sed commands in file, e.g. `sedscript`, and use:

```
% sed -f sedscript grades.
```

**Exercise:** getting the middle `midsed-sh`

```bash
echo "Usage: midsed-sh <head> <tail> <file>"
length=`wc -l < $3`
bottom=`expr $length - $2 + 1`
sed "1,$1 d; $bottom,$length d" $3
Example 1: Selecting and printing lines: ex1-sh

Courses accounts:

- print login and name:

  ```
  % ypcat passwd |
  awk -F: '/^cs[5-9][0-9]+/ {print $1,$5}'
  ...
  cs779 cs779 grader
  ...
  ```

  -F Field separator

Faculty accounts:

- print login and name:

  ```
  % ypcat passwd |
  awk -F: '$4 == 13 {print $1 "-->" $5 }'
  ...
  Wahab --> Dr. wahab
  ...
  ```

- print all fields except the password field:

  ```
  % ypcat passwd |
  awk -F: '$4 == 13 { $2= "" ; {print}}'
  ...
  wahab 51 13 Dr.wahab /home/wahab /usr/local/bin/tcsh
  ...
  ```

- print line number, login and name:

  ```
  % ypcat passwd |
  awk -F: '$4 == 13
  {print ++count "": "$1"-> " $5"}
  ...
  11: Wahab  --> Dr. Wahab
  ...
  ```

Exercise: getting file name from a path pathtofile-sh

  ```
  % echo $1 | awk -F/ '{print $NF}'
  ```

  E.g.:

  ```
  % pathtofile-sh /home/wahab/public_html
  public_html
  ```

Example 2: awk file: ex2-sh & ex2-awk
**Example 3: Begin and END:**

```
% ex3-sh & ex3-awk
```

```
ex3-sh:
ypcat passwd | awk -F: -f ex3-awk
```

```
ex3-awk:
BEGIN {
    system ("date");
    printf " HOME = %s \n", ENVIRON["HOME"]
}

$4 == 13 {print ++count, $1 "-->", $5}

END {
    printf "Total number is %d\n", count
    system ("who");
    printf " PATH = %s \n", ENVIRON["PATH"]
}
```

**Usage:**

```
% ex3-sh
```
**Scalers:**

`string1.string2`: catenate `string1` & `string2`

`string x n`: repeat `string` `n` times.

**Example: strRepeat**

```perl
print "String: "; $a = <STDIN>;
print "Number of times: "; chomp($b = <STDIN>);
$c = $a x $b; print "The result is: \n$c"
```

**Example Usage:**

```bash
% strRepeat
String: Hussein
Number of times: 2
The result is:
Hussein
Hussein
```

**Arrays:**

```perl
@array = (1,2,"three");
$array[0] is 1
$array[2] is three
```

**Example: randArray**

```perl
srand;
print "List of strings: "; @b = <STDIN>;
print "Answer: \$b[rand(@b)]"
```

**Example Usage:**

```bash
% randArray
List of strings:
a
b
```
Flow control:

If  elseif  else
while (1) {}
until (0) {}
for (i=0, i<n, i++) {}
foreach $i (list) {}

Example: squareForeach

foreach $number (0..32) {
    $square = $number * $number;
    printf "%5g %8g\n", $number, $square;
}

Associative Arrays:

Example: wordcountAssocaitive

chomp(@words = <STDIN>);
foreach $word (@words) {
    $count{$word}++
}
foreach $word (keys %count) {
    print "$word was seen $count{$word} times\n";
}

Example usage:
% wordcountsAssociative
Hussein
Omar
Hussein
Hussein was seen 2 times
Omar was seen 1 times
Basic I/O:

Input from STDIN
@strings = <STDIN>

Input from the Diamond Operator
@strings = <>
if @ARGV is empty it uses STDIN
otherwise it uses the files specified by:
$ARGV[0], $ARGV[1], etc.

Example: basicIO

    print "List of strings:\n";
    #chomp(@strings = <STDIN>);
    chomp(@strings = <>);
    foreach (@strings) {
        printf "%s\n", $_;
    }

Example Usage:
% basicIO
aaa
bbb
^D
List of strings:
aaa
bbb

% basicIO basicIO
List of strings:
print "List of strings:\n";
#chomp(@strings = <STDIN>);
chomp(@strings = <>);
foreach (@strings) {
    printf "%s\n", $_;
}

Regular Exressions:

Example: vowelsAnyOrder

/i : ignore case.
To find all words that have all 5 vowels (a,e,i,o,u) in any order:

```perl
while (<>) {
    if (/a/i && /e/i && /i/i && /o/i && /u/i) {
        print;
    }
}
```

**Example Usage:**

```
% vowelsAnyOrder /usr/dict/words
adventitious
aeronautic
ambidextrous
argillaceous
argumentation
auctioneer
audiotape
.....
```

**Example:** **vowelsInOrder**

To find all words that have all 5 vowels (a,e,i,o,u) in order:

```perl
while (<>) {
    if (/a.*e.*i.*o.*u/i) {
        print;
    }
}
```

**Example Usage:**

```
% vowelsInOrder /usr/dict/words
adventitious
facetious
sacrilegious
```

```

date
if (`date` =~ /^S/) {
    print "Go play!\n";
} else {
    print "Get to work!\n";
}
```

**Example Usage:**

```
% backqoute
```

```
"system ("date");
if (`date` =~ /^S/) {
    print "Go play!\n";
} else {
    print "Get to work!\n";
}
```

**Example Usage:**

```
% backqoute
```
Functions:

Example: subHello

&say("hello", "world");
&say("goodbye", "cruel world");
sub say {
    print "$_[0],$_[1]!\n";
}

Example Usage:
% subHello
hello, world!
goodbye, cruel world!

Example: subAdd

@_ : function arguments

print &add(@ARGV), "\n";
print &add(1..5), "\n";
print &add(1,3,5), "\n";

sub add {
    local ($sum);
    $sum = 0;
    foreach $_ (@_) {
        $sum += $_;
    }
    return $sum;
}

Example Usage:
% subAdd 2 4 6 8
  20
  15
  9

File IO:

Example: fileIO
print "Input file name: ";
chomp($infilename = <STDIN>);

print "Output file name: ";
chomp($outfilename = <STDIN>);

print "Search string: ";
chomp($search = <STDIN>);

print "Replacement string: ";
chomp($replace = <STDIN>);

open(IN,$infilename) ||
die "cannot open $infilename for reading: $!";
die "will not overwrite $outfilename"
    if -e $outfilename;

open(OUT,">$outfilename") ||
die "cannot create $outfilename: $!";

while (<IN>) {#read a line from file IN into $_
    s/$search/$replace/g; # change the lines
    print OUT $_; # print that line to file OUT
}
close(IN);
close(OUT);

Example usage:
% more testfile
   Hussein Abdel-Wahab
% fileIO
   Input file name: testfile
   Output file name: testfile2
   Search string: Hussein
   Replacement string: Omar
% more testfile2
   Omar Abdel-Wahab

Format STDOUT:

Example: formatSTDOUT
open(PW,"ypcat passwd|") ||
die "How did you get logged in?";
while (<PW>) {
    ($user,$gid,$gcos) = (split /:/)[0,3,4];
    if ($gid == $ARGV[0]) {
        ($real) = split /,/, $gcos;
        write;
    }
}

format STDOUT =

@<<<<<<<<< @<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
$user, $real
.

format STDOUT_TOP =
Page @<<<<
$%
Username  Real Name
========  =========

Usage examples:
% formatSTDOUT 13
...list of faculty

Page 1
Username  Real Name
========  =========
zeil      Steven J. Zeil
nadeem    Tamer Nadeem
wahab     Hussein Abdel-Wahab

% formatSTDOUT 22
...list of grad students.
Example 1: Selecting and printing lines: sel-print-pl

```perl
#!/usr/bin/perl
system ("date");
print ("BEGIN - finding cs accounts \n");
open (PW, "ypcat passwd | ");
while (<PW>) {
    if ( /^cs[4-9][0-9]+:/ ) {
        split (/:/);
        print ("$. --> $_[0] $_[4] \n ");
        $count++;
    }
}
print ("END - total: $count \n");

$.       Current line number,
$_[i]   Content of the ith field
$_        Content of entire current line.

Usage:
% sel-print-pl
```

Example 2: Selecting fields: pathtofile-pl

```perl
$path = $ARGV[0];
$nf = @fields = split (/\//, $path);
print (@fields[$nf-1]) ;
```

E.g.:
% pathtofile-sh /home/wahab/public_html
public_html

Example 3: Translate & substitute: tr-pl & substitute-pl
```
tr-pl:

#! /usr/bin/perl
open(INFILE, "fields");
print("BEGIN - translate \n");
while (<INFILE>) {
    tr /a-z/A-Z/;
    print;
}
print ("END - list: \n");

% more fields
Wahab, Hussein
Maly, Kurt

% tr-pl
BEGIN - list
   WAHAB, HUSSEIN
   MALY, KURT
END - list:

substitute-pl:

#! /usr/bin/perl
open(INFILE, "grades");
print("BEGIN - substitute \n");
while (<INFILE>) {
    s/^/000/;s/F/Fail/;s/A/Excellet/;
    s/B/Very Good/;s/C/Good/; s/D/Pass/;
    print;
}
print ("END - list: \n");

% more grades
1: A
2: B
3: C
4: D
5: F
6: A
7: A
8: B
9: F```
% substitute-pl
BEGIN - list
0001: Excellet
0002: Very Good
0003: Good
0004: Pass
0005: Fail
0006: Excellet
0007: Excellet
0008: Very Good
0009: Fail
END - list:

Example 4: Grade count: grade_count

open(INFILE, "grades");
while (<INFILE>) {
    chomp();
    split (/:/);
    $grade = $_[1] ;
    $gradelist{$grade}++ ;
}
foreach $grade (sort (keys %gradelist)){
    print("$grade-->$gradelist{$grade}\n");
}

Example 5: Comperhensive Example: login-pl

Each user has name and password saved in a file.
The program allows the user 3 times to enter the correct password
and then send email to administrator of the violation.

#!/usr/bin/perl
$ADMIN_EMAIL = cs476 ;
$MAX_TRIALS = 3;
init_words();
print "what is your name? ";
$name = <STDIN>;
chomp($name);
print "Hello, $name!\n";
print "What is the secret word? ";
chomp ($guess = <STDIN>);
while (1) {
    if ($words{$name} eq $guess) {
        print "Welcome $name\n";
        last;
    }
    elsif ( ++$trial < $MAX_TRIALS ) {
        print "Wrong, try again. What is the secret word? ";
        chomp ($guess = <STDIN>);
    }
    else {
        print "$name, you tried $MAX_TRIALS times, mail sent to admin.\n";
        open (MAIL, " | Mail -s \"login violation\"$ADMIN_EMAIL")||
        print MAIL "bad news: $name guessed $MAX_TRIALS times\n";
        close MAIL;
        last;
    }
}

sub init_words {
    $filename = <passwd.file>;
    open (WORDSLIST, $filename)||
        die "can't open $filename: $!";
    while ($name = <WORDSLIST>) {
        chomp ($name);
        $word = <WORDSLIST>;
        chomp ($word);
        $words{$name} = $word;
    }
    close WORDSLIST;
}

Usage:
% login-pl