Open Computing ‘‘Hands-On’’: ‘‘Wizard’s Grabbag’’ Column: April 1994: Listings

Listing 1. A Perl program is suitable to summarize system data. The `uuhistory` program shown here summarizes the logs written by HoneyDanBer UUCP programs.

```
#!/usr/bin/perl
# @(#) uuhistory  History log file summary
# Author: Michael P. Brininstool
# Adapted by Becca Thomas, December 1993
#
# Define default log-file path names:
$logdir="/var/uucp/.Log";    # Log files under here
$logcico="$logdir/uucico";   # uucico logs
$loguux="$logdir/uux";       # uux logs
$loguucp="$logdir/uucp";     # uucp logs
$loguuxqt="$logdir/uuxqt";   # uuxqt logs

$fflag = 0; $logfile = "";

# Get host names from command line or by running uuname:
if (@ARGV) {                 # If command-line arguments
  if ($ARGV[0] eq "-f") {    # and if user specifies log file
    $fflag++; shift;         # note it, then discard -f flag
    $logfile = shift;        # and save file name
    die "Can't read $logfile: $!: stopped" unless -r $logfile;
    @hosts = ("dummyhost"); # only once around outermost loop
  } else {
    @hosts=@ARGV;            # The rest should be host names
  }
} else {
  open(UUNAME,"uuname|")  ||  # Read uuname output?
    die "Can't run uuname: $!: stopped";
  @hosts=;          # Read them all at once
  chop(@hosts);             # Remove newline from each name
  close(UUNAME);
}

# So all output can be sorted by date and time:
open(SORT, "|sort") || die "Can't run sort: $!: stopped";

# Examine log files:
foreach $host (@hosts) {     # For each host

  # Examine UUCICO log for host:
  $logfile = "$logcico/$host" unless $fflag;
  if (open(LOG,"$logfile")) {   # Failure isn't fatal
    @lines=(grep(/OK|SUCCEEDED|FAILED|CAUGHT|INTREXIT/,));
    close(LOG) unless $fflag;

    $sdate="";
    # Process each line selected from the uucico log:
    foreach $line (@lines) {
      chop($line);
      ($user, $sys, $date, $files, $statmsg) = $line =~
        /^(\S+)\s+(\S+)\s+\((.*),\d+,(\d+)\)(.*)$/;
      $date = &formatdate($date);
```

```
52        $_ = $statmsg; # Using $_ simplifies matching test syntax
53
54        # Process entries depending on status messages:
55        if (/SUCCEEDED/) {  # Outbound call reached target system
56          $sdate=$date; $direction="->";
57        } else {        #
58          if (/startup/) {  # Start of transaction
59            $direction = "<-" unless $direction eq "->";
60            $sdate=$date;
61          } else {      # End of transaction
62            if (/complete|CAUGHT|INTREXIT/) {
63              if (/OK/) { # Successful transaction
64                # Compute transaction time as hh:mm:ss
65                ($tty, $sec) = /.*\s(\S+)\s(\d+)\)$/;
66                $hr = int($sec / 3600); $sec -= ($hr * 3600);
67                 $min = int($sec / 60);  $sec -= ($min * 60);
68                $time = sprintf("in %2d:%2d:%2d",
69                  $hr, $min, $sec);
70                $time =~ tr/ /0/; $time =~ s/in0/in /;
71              } else {  # FAILED, CAUGHT, INTREXIT, or ???
72                $tty=""; $time = $statmsg;
73              }
74              printf SORT "%s to %s %s %s %s %s files %s\n",
75                $sdate, $date, $tty, $direction,
76                $sys, $files, $time;
77            } else {  # Not "complete", CAUGHT, or INTREXIT
78              printf SORT "%s %s %s\n", $date, $sys, $statmsg;
79            }       # End of "complete" or "CAUGHT" section
80          $direction = ""; $sdate = ""; # Reset before next transaction
81          }         # End of else "startup" section
82        }           # End of else /SUCCEEDED/ section
83      }             # End of foreach $line (@lines)
84    }               # End of if open $logfile
85
86    # Examine UUCP and UUX logs for host:
87    if ($fflag) {   # User-specified log file
88      &report("");  # Log file already open
89    } else {
90      $logfile = "$loguux/$host";
91      &report("$logfile");  # The uux cmd report
92      $logfile = "$loguucp/$host";
93      &report("$logfile");  # The uucp cmd report
94    }
95
96    # Examine uuxqt[RIK: CAP?] log for host:
97    if ($fflag) {
98      seek(LOG, 0, 0) ||    # Begin again
99        die "Can't seek: $!: stopped";
100   } else {
101     $logfile = "$loguuxqt/$host";
102     open(LOG,"$logfile") || next;   # done with this host
103   }
104   @lines=(grep(/XQT/,)); # "XQT" entries only
105   close(LOG);             # Done
106
107   foreach $line (@lines) {
108     chop($line);
109     ($date, $user, $command) = $line =~
110       /^.*\((.*),\d+,\d+\)\s+(\S+)\s+XQT\s+\(.*;(.*)\)$/;
111     $date = &formatdate($date);
112     printf SORT "%s uuxqt for %s: %s\n",
```

```
113         $date, $user, $command;
114     }
115   }   # End of foreach $host (@hosts)
116   close(SORT);              # Flush
117   exit 0;
118
119   # Generate report from logs written by uucp and/or uux
120   sub report {
121     local($line, $user, $sys, $date, $command);
122     local($file) = @_;
123
124     if ($file) {            # Log file specified
125       open(LOG,"$file") || return;
126     } else {                # Log opened earlier
127       seek(LOG, 0, 0) ||    # Back to beginning
128         die "Can't seek: $!: stopped\n";
129     }
130     @lines=(grep(/QUEUED/,));  # "QUEUED" entries only
131     close(LOG) if $file;    # Done with named log
132
133     foreach $line (@lines) {
134       chop($line);
135       ($user, $sys, $date, $command) = $line =~
136         /^(\S+)\s(\S+)\s.*\((.*),\d+,\d+\)\sQUEUED\s\((.*)\)$/;
137       $date = &formatdate($date);
138       printf SORT "%s %s queued for %s: %s\n",
139         $date, $user, $sys, $command;
140     }
141   }
142
143   # Reformat the date for easier sorting:
144   sub formatdate {
145     local($mon, $day, $hr, $min, $sec);
146     local($date) = @_;
147
148     ($mon, $day, $hr, $min, $sec) = $date =~
149       /^(\d+)\/(\d+)-(\d+):(\d\d):(\d\d)$/;
150     $fdate = sprintf("%2s/%2s-%2s:%2s:%2s",
151       $mon, $day, $hr, $min, $sec);
152     $fdate =~ tr/ /0/;  # pad with zeros, not spaces
153     $fdate;             # Return this value
154   }
```

Listing 2. The `krand` Korn shell script generates a random number in a specified range with an optionally specified ‘‘seed’’ value.

```
 1  #!/bin/ksh
 2  # @(#) krand  Produces a random number within integer limits
 3  # Author: Peter Turnbull, May 1993
 4  # Modified by: Becca Thomas, January 1994
 5  $DBG_SH         # dormant debugging directive (Apr. 92)
 6  Usage="Usage: `basename $0` lower-limit upper-limit [seed]"
 7
 8  Seed=$$         # Initialize random-number seed value with PID
 9  # Process command-line arguments:
10  case $# in
11      2)  Lower=$1; Upper=$2 ;;
12      3)  Lower=$1; Upper=$2; Seed=$3 ;;
13      *)  echo $Usage >&2 ; exit 1 ;;
14  esac
```

```
15
16  # Check that specified values are integers:
17  expr "$Lower" + 0 >/dev/null 2>&1
18  [ $? -eq 2 ] && { echo "Lower ($Lower) not an integer" >&2; exit 2;}
19
20  expr "$Upper" + 0 >/dev/null 2>&1
21  [ $? -eq 2 ] && { echo "Upper ($Upper) not an integer" >&2; exit 2;}
22
23  expr "$Seed" + 0 >/dev/null 2>&1
24  [ $? -eq 2 ] && { echo "Seed ($Seed) not an integer" >&2; exit 2;}
25
26  # Check that values are in the correct range:
27  [ "$Lower" -lt 0 -o `expr "$Lower" : '.*'` -gt 5 ] &&
28      { echo "Lower limit ($Lower) less than zero" >&2; exit 2;}
29
30  [ "$Upper" -gt 32767 -o `expr "$Upper" : '.*'` -gt 5 ] &&
31      { echo "Upper limit ($Upper) greater than 32767" >&2; exit 2;}
32
33  [ "$Seed" -lt 0 -o "$Seed" -gt 32767 -o `expr "$Seed" : '.*'` -gt 5 ] &&
34      { echo "Seed value ($Seed) out of range (0 to 32767)" >&2; exit 2;}
35
36  [ "$Upper" -le "$Lower" ] &&
37      { echo "Upper ($Upper) <= lower value ($Lower)" >&2; exit 2;}
38
39  # Seed the random-number generator:
40  RANDOM=$Seed
41
42  # Compute value, scaled within range:
43  let rand="$RANDOM % ($Upper - $Lower + 1) + $Lower"
44
45  # Report result:
46  echo $rand
```

Copyright © 1995 The McGraw-Hill Companies, Inc. All Rights Reserved.
Edited by Becca Thomas / Online Editor / UnixWorld Online / beccat@wcmh.com

Last Modified: Tuesday, 22-Aug-95 16:23:41 PDT