```perl
#!/usr/bin/perl
# $Id: graph.perl,v 1.1 2018-11-06 18:50:43-08 - - $

use strict;
use warnings;
$0 =~ s|.*/||;

# Example setting up a directed graph.

my @inputs = (
   "all : hello",
   "hello : main.o hello.o",
   "main.o : main.c hello.h",
   "hello.o : hello.c hello.h",
   "ci : Makefile main.c hello.c hello.h",
   "test : hello",
   "clean : ",
   "spotless : clean",
);

sub parse_dep ($) {
   my ($line) = @_;
   return undef unless $line =~ m/^(\S+)\s*:\s*(.*?)\s*$/;
   my ($target, $dependency) = ($1, $2);
   my @dependencies = split m/\s+/, $dependency;
   return $target, \@dependencies;
}

my %graph;
for my $input (@inputs) {
   my ($target, $deps) = parse_dep $input;
   print "$0: syntax error: $input\n" and next unless defined $target;
   $graph{$target} = $deps;
}

for my $target (keys %graph) {
   print "\"$target\"";
   my $deps = $graph{$target};
   if (not @$deps) {
      print " has no dependencies";
   }else {
      print " depends on";
      print " \"$_\"" for @$deps;
   }
   print "\n";
}

```

```
"test" depends on "hello"
"clean" has no dependencies
"all" depends on "hello"
"main.o" depends on "main.c" "hello.h"
"ci" depends on "Makefile" "main.c" "hello.c" "hello.h"
"hello.o" depends on "hello.c" "hello.h"
"spotless" depends on "clean"
"hello" depends on "main.o" "hello.o"
```

```perl
#!/usr/bin/perl
# $Id: mtime.perl,v 1.1 2018-11-06 18:50:44-08 - - $
#
# NAME
#    older.perl - check whether a pair of files are older or newer
#
# SYNOPSIS
#    older.perl filename...
#
# DESCRIPTION
#    The two files' modification times are compared and a
#    relationship is printed.
#

use strict;
use warnings;
use POSIX qw(strftime);
$0 =~ s|.*/||;

sub mtime ($) {
   my ($filename) = @_;
   my @stat = stat $filename;
   return @stat ? $stat[9] : undef;
}

sub fileinfo ($) {
   my ($filename) = @_;
   my $mtime = mtime $filename;
   print "$filename: ";
   if (defined $mtime) {print strftime "%c\n", localtime $mtime}
                  else {print "$!\n"}
   return $mtime;
}

for my $filename (@ARGV) {
   unless (-e $filename) {
      printf STDERR "$0: $filename: $!\n";
   }else {
      my $mtime = mtime $filename;
      my $ctime = strftime "%c", localtime $mtime;
      printf "%-20s %12d %s\n", $filename, $mtime, $ctime;
   }
}

```

```
HEADER.html            1549578416 Thu 07 Feb 2019 02:26:56 PM PST
Listing.pdf            1541559137 Tue 06 Nov 2018 06:52:17 PM PST
Listing.ps             1541559137 Tue 06 Nov 2018 06:52:17 PM PST
RCS                    1549578440 Thu 07 Feb 2019 02:27:20 PM PST
graph.output           1549578440 Thu 07 Feb 2019 02:27:20 PM PST
graph.perl             1549578440 Thu 07 Feb 2019 02:27:20 PM PST
mk                     1549578440 Thu 07 Feb 2019 02:27:20 PM PST
mtime.output           1549578440 Thu 07 Feb 2019 02:27:20 PM PST
mtime.perl             1549578440 Thu 07 Feb 2019 02:27:20 PM PST
sigtoperl              1541559137 Tue 06 Nov 2018 06:52:17 PM PST
sigtoperl.cpp          1541559136 Tue 06 Nov 2018 06:52:16 PM PST
sigtoperl.output       1541559137 Tue 06 Nov 2018 06:52:17 PM PST
zerotime               1541559107 Tue 06 Nov 2018 06:51:47 PM PST
zerotime.cpp           1541559107 Tue 06 Nov 2018 06:51:47 PM PST
zerotime.txt           1549578440 Thu 07 Feb 2019 02:27:20 PM PST
.                      1549578440 Thu 07 Feb 2019 02:27:20 PM PST
/dev/null              1549328910 Mon 04 Feb 2019 05:08:30 PM PST
```

```
// $Id: sigtoperl.cpp,v 1.1 2018-11-06 18:50:44-08 - - $

#include <cstdlib>
#include <cstring>
#include <ctime>
#include <iomanip>
#include <iostream>
#include <string>
#include <sys/utsname.h>
using namespace std;

int main (int, char** argv) {

   // Print UTS system information.
   struct utsname utsbuf {};
   uname (&utsbuf);
   cout << "# " << basename (argv[0]) << ": " << utsbuf.machine
        << " " << utsbuf.sysname << " " << utsbuf.nodename << endl;

   // Print current date/time.
   struct tm tm_local;
   time_t tm_now = time (nullptr);
   localtime_r (&tm_now, &tm_local);
   char tm_buffer[256];
   strftime (tm_buffer, sizeof tm_buffer, "%c", &tm_local);
   cout << "# " << basename (argv[0]) << ": " << tm_buffer << endl;

   // Print strsignal information.
   constexpr int MAXSIG = 255;
   cout << "my %strsignal = (" << endl;
   for (int sig = 0; sig < MAXSIG; ++sig) {
      const char* strsig = strsignal (sig);
      if (strsig == nullptr) continue;
      string stringsig = strsig;
      if (stringsig.find_first_of ("Unknown signal ") == 0) continue;
      if (stringsig.find_first_of ("Real-time signal ") == 0) continue;
      cout << setw(5) << sig << " => \"" << stringsig << "\"," << endl;
   }
   printf (");\n");
   return EXIT_SUCCESS;
}

```

```
# sigtoperl: x86_64 Linux unix2.lt.ucsc.edu
# sigtoperl: Thu Feb  7 14:27:20 2019
my %strsignal = (
    1 => "Hangup",
    2 => "Interrupt",
    3 => "Quit",
    4 => "Illegal instruction",
    5 => "Trace/breakpoint trap",
    6 => "Aborted",
    7 => "Bus error",
    8 => "Floating point exception",
    9 => "Killed",
   11 => "Segmentation fault",
   13 => "Broken pipe",
   14 => "Alarm clock",
   15 => "Terminated",
   16 => "Stack fault",
   17 => "Child exited",
   18 => "Continued",
   19 => "Stopped (signal)",
   20 => "Stopped",
   21 => "Stopped (tty input)",
   22 => "Stopped (tty output)",
   24 => "CPU time limit exceeded",
   25 => "File size limit exceeded",
   26 => "Virtual timer expired",
   27 => "Profiling timer expired",
   28 => "Window changed",
   29 => "I/O possible",
   30 => "Power failure",
   31 => "Bad system call",
);
```