```perl
#!/usr/bin/perl
# $Id: haversine.perl,v 1.1 2018-05-24 10:39:07-07 - - $

# Find distance between two airports using the haversine formula.
# http://andrew.hedges.name/experiments/haversine/
# Airport database is in prolog syntax.

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

my $PI = 3.141592653589793238462643383279502884;
my $EARTH_RADIUS_MILES = 3961;

my $database_name = ".score/database.pl";

my %database;
open DATABASE, "<$database_name" or die "$0: $database_name: $!";
while (<DATABASE>) {
   next unless m/airport\(\s*(.*?),\s*’(.*?)’,\s*
                 degmin\(\s*(\d+),\s*(\d+)\s*\),\s*
                 degmin\(\s*(\d+),\s*(\d+)\s*\)\s*\)/x;
   my ($airport, $name, $nlatdeg, $nlatmin, $wlondeg, $wlonmin)
         = ($1, $2, $3, $4, $5, $6);
   $airport = uc $airport;
   $database{$airport} = [$name, $nlatdeg, $nlatmin,
                                 $wlondeg, $wlonmin];
}
close DATABASE;

sub radians ($$) {
   # Convert degrees and minutes of arc to radians.
   my ($degrees, $minutes) = @_;
   return ($degrees + $minutes / 60) * $PI / 180;
}

sub print_location(@) {
   my ($deg, $min, $dir) = @_;
   printf " %3d°%2d’%s(%6.2f°,%6.4f)",
          $deg, $min, $dir, $deg + $min / 60, radians ($deg, $min);
}

sub print_airport($$) {
   my ($airport, $data) = @_;
   printf "%-3s (%-16s)", $airport, $$data[0];
   print_location @$data[1,2], "N";
   print_location @$data[3,4], "W";
   printf "\n";
}

for my $airport (sort keys %database) {
   print_airport $airport, $database{$airport};
}

```

```perl

my $circumference = 2 * $PI * $EARTH_RADIUS_MILES;
printf "\n";
printf "Earth radius:        %7.1f miles\n", $EARTH_RADIUS_MILES;
printf "Earth circumference: %7.1f miles\n", $circumference;
printf "Earth 1 degree arc:  %7.1f miles\n", $circumference / 360;
printf "Earth 1 minute arc:  %7.1f miles\n", $circumference / 360 / 60;
printf "Earth 1 radian arc:  %7.1f miles\n", $circumference / $PI / 2;

sub haversine_distance ($$$$) {
   # Latitude1, longitude1 in radians.
   # Latitude2, longitude2 in radians.
   my ($lat1, $lon1, $lat2, $lon2) = @_;
   my $dlon = $lon2 - $lon1;
   my $dlat = $lat2 - $lat1;
   my $tmpa = (sin ($dlat / 2)) ** 2
            + cos ($lat1) * cos ($lat2) * (sin ($dlon / 2)) ** 2;
   my $unit_distance = 2 * atan2 (sqrt ($tmpa), sqrt (1 - $tmpa));
   my $distance_miles = $EARTH_RADIUS_MILES * $unit_distance;
   return $distance_miles;
}

while (@ARGV >= 2) {
   my $airport1 = shift; $airport1 = uc $airport1;
   my $airport2 = shift; $airport2 = uc $airport2;
   my $data1 = $database{$airport1};
   my $data2 = $database{$airport2};
   warn "$0: $airport1, $airport2: invalid airport\n" and next
          unless $data1 && $data2;
   my $lat1 = radians ($data1->[1], $data1->[2]);
   my $lon1 = radians ($data1->[3], $data1->[4]);
   my $lat2 = radians ($data2->[1], $data2->[2]);
   my $lon2 = radians ($data2->[3], $data2->[4]);
   my $distance = haversine_distance ($lat1, $lon1, $lat2, $lon2);
   print "\nDistance:\n";
   print_airport $airport1, $data1;
   print_airport $airport2, $data2;
   printf "%.0f miles\n", $distance;
}
```

```
COMMAND: haversine.perl lax sfo sjc nyc sfo sea

```