These data are not to be used without permission from owner.

This is a set of exercises I made for Tom Parchman's upper division Bioinformatics with Perl & Unix course (University of Nevada, Reno). These exercises are designed to highlight different skills in Perl. The included scripts correspond to the solutions to the prompts below.

The files used here are from automated visitation data of mountain chickadees (small songbirds) equipped with passive integrated transponder (PIT)-tags to bird feeders that have radio-frequency identification (RFID) readers. Each bird's PIT-tag has a unique 10-digit hexadecimal identification number, allowing us to identify which bird visited which feeder and when.

I used these scripts or similar scripts to look for patterns in visits during my tenure as a Graduate Research Assistant as well.

Assignment description:

1. Concatenating files:

-Here you will create a script that concatenates in all files in our "InFiles/" directory into a single file. Some of our IDs do not correspond to real birds, so you must use the "CaughtIndex.txt" file to exclude fake IDs.

-You may have noticed that the line endings are wonky in these files, making it especially useful to concatenate all data into a single file (so we only have to deal with these once). You will want to fix these using $/ = "\r" when you open each file

-Each file name has "GPR0DATA", followed by either "H" or "L" (indicates either high or low elevation sites), a digit (1,3, or 8) indicating the array number, a decimal followed by another digit (1-8) indicating the feeder number, and ends in ".TXT". We will add some location information to our concatenated file, specifically the elevation combined with the array number and the feeder number so that each line in our out file will have a total of 5 elements ($lin[0-4]).

Here is an example of what the first two lines should look like (here tab separated, "\t"):

ID	Date	Time	Array	Feeder
0700ED985B	11/30/17	08:32:41	H1	1

Hint: It will be useful to make a hash of the identified individuals ("CaughtIndex.txt" file), the 'exists' function for hashes, and regular expressions/split function.

2. Total number of visits per array for each bird:

Here we will simply count up all the visits each bird makes to any feeder at individual arrays using the concatenated file we made in our first script. Because birds might move between arrays, we will separate visits not only by bird, but also by array.

-I would suggest making a hash with unique identifiers of ID_Array keys. The outfile will include bird ID, Array, and number of visits of that bird to that array.

Hint: You might want to use some type of counter (+=) to count the number of matches ($randomscalar = () = $scalar1 =~ m/($scalar2)/gi;)

Here is an example of what the first 3 lines should look like:

ID	Array	TotalVisits
0700ED8BD1	H3	15
0700ED9084	H3	137

3. Number of birds coming to each array:

We want to know how many birds are visiting each array (we'll ignore the fact that we might have birds visiting multiple arrays for now). We will again use the concatenated file from our first script to create multiple hashes to determine the number of birds at each array.

Instead of making an outfile, you can just have the numbers print to the screen as such:

There are ## birds visiting array H3
There are ## birds visiting array H1

4. Visits to unique feeders within an array:

Here we want to know if birds have favorite feeders within an array. We will count up the number of visits each bird makes to each feeder at each array it visits using our concatenated file from our first script.

This should be similar to how you counted visits in the second script except visits will be broken down by feeder number (but still separated by array so birds moving between arrays will have multiple lines of data in our outfile).

We can write two different subroutines that might be particularly helpful for this. One will allow us to count up visits to each feeder and we can write another that calculates the total number of visits. Even though we counted up the total number of visits in our second script, this one will calculate it in a different way.

The outfile should look a little something like this (where F# is the feeder number):

ID	Array	TotalVisits	F1	F2	F3	F4	F5	F6	F7	F8

010799F978	H1	105	15	7	4	3	16	41	12	7
01101719DA	H1	415	85	70	22	36	51	55	30	66

5. Bonus: Birds moving between locations:

We will use our outfile from the first script to determine which birds have moved between arrays. I would recommend making two hashes: one with bird ID keys and one that has a unique ID_array keys. You can loop through the first hash to find matches between keys of the two hashes and push the arrays each bird visits to an @array.

We can simply have this print to the terminal.

Hint: you should find three birds that have moved between arrays