Finding MatchCode probabilities

In the first part of the WordleBot problem, we created a 14k x 14k matrix of matchcodes where a given entry in the matrix contained `matchcodes(solutionRow, guessCol)`. Any matchcode where `guessCol` and `solutionRow` were the same gave the matchcode 242 (all green) and others give some other number between 242 and 0.

Given that there are only 242 possible matchcodes and there are 14000+ solutions we see that any guess will have multiple solutions that match the matchcode. This means that each of the 242 possible matchcodes has a probability of being found for each guess. The WordleBot uses these probabilities to find its next guess.

In this assignment, we will create a 14k x 14k array that shadows the `matchcodes` matrix in that there are solutions on the columns and guesses in the rows. But instead of containing the matchcodes, this matrix contains the probability that that matchcode will be found for that guess.

A small example

Consider a small matrix containing more rows and columns than numbers:

```
smallMatrix

3    1    1    1    2

3    1    3    2    1

1    2    3    3    3

3    3    2    3    3

2    3    3    3    3
```

We can see that different columns contain different probabilities for each value. For example column 1 `[3; 3; 1; 3; 2]` contains a 60% chance of finding a 1 and a 20% chance of finding a 3 and a 2. Replacing the numbers with the probability of seeing the numbers we get the following matrix:

```
probMatrix

.6    .4    .2    .2    .2

.6      .4    .6    .2    .2

.2    .2    .6    .6    .6

.6    .4    .2    .6    .6

.2    .4      .6     .6    .6
```

Now we see that `smallMatrix(3,3)` contains a 3 and `probMatrix(3,3)` contains a .6, which is the proability of finding a 3 in the third row.

We could write a program to figure out this probability, but it is easier to use the MATLAB `groupcounts()` function.

Using `groupcounts()`

The `groupcounts()` function takes an array of numbers and finds the the number of elements in each group of unique values, the list of unique values, and the probability of choosing any unique value. When you call the function it returns three column vectors:

`grpSize`–The number of elements in each group.

`grpVals`–The value in each group.

`grpProb`–The percentage chance of seeing each value as a number from 0 to 100. (You need to divide this by 100 to get probabilities)

Here is an example using the top row above:

```
dm = [3 ;1 ;1 ;1 ;2];
[grpSize, grpVals, grpProb] = groupcounts(dm);
grpSize
```

```
grpSize = 3×1
     3
     1
     1
```

`grpSize` tells us that there are three different values and that the first value has 3 entries while the other two values have 2 entries.

```
grpVals
```

```
grpVals = 3×1
     1
     2
     3
```

`grpVals` tells us the values in each group. The indices of the values match the indices of `grpSize`. There are three 1s, one 2, and one 3.

```
grpProb
```

```
grpProb = 3×1
    60
    20
    20
```

`grpProb` tells us there is a 60% chance of finding a 1 in the first column and a 20% chance of finding a 2 or 3.

Gathering data from groupcount()

We can gather data from groupcount() by indexing `grpProb` and `grpSize` using a conditional selection wiht `grpVals`. Actually, we don't care about `grpSize`, so we can ignore it.

```
disp(sprintf("The probability of choosing a 1 is %2.2f", grpProb(grpVals ==
1) ./ 100 ))
```

```
The probability of choosing a 1 is 0.60
```

The assignment

This example showed groupcount() working on a single column array. Your job is to create a function named findprobabilities() that takes a matrix and returns a matrix of the same size that contains the probability of seeing every element across every column.

Then we will use your function to create a probability matrix for matchcodes.