Arrays and files in Visual Basic II

Topics

- Arrays and parallel arrays
- Reading from CSV files into parallel arrays
- More practice with formatting strings
- Finding the minimum value in an array
- Using the `Arrays.sort` function in Visual Basic

Instructions

5. Copy and paste the folder `RatingPredictions_Part4`. Rename the copy `RatingPredictions_Part5`. Launch the VS Express 2013 software and open the project `RatingPredictions_Part5`.

   Work on `btnPredictRatings`.

   TODO Better description with reference to book on how to find minimum. Using a `For`-loop, find the minimum value in the `distances` array. Be sure to keep track of the index where the minimum value appeared. This index represents the id of the viewer that is the least different from the user. Now make a prediction as to the rating for the user for movie 5 by using the rating for movie 5 of the viewer that is the least different from the user. Output the predicted rating in a message with the following format:

   ```
   The most similar viewer was viewer #39 and the distance calulcated was 1.0.
   The predicated rating for movie 5 is 3.0.
   ```

   After you get your program working, fill out `Test Table 3` in the file `TestTables.docx`, making sure that your *expected output* and *actual output* are consistent.

Checkpoint 5 (80/100)

- ☐ project produces correct output for all test cases in Test Table 3
- ☐ project produces correctly formatted output according to the example above

6. Copy and paste the folder `RatingPredictions_Part5`. Rename the copy `RatingPredictions_Part6`. Launch the VS Express 2013 software and open the project `RatingPredictions_Part6`.

   As you may be able to tell, the simple model of finding the viewer from the dataset that is the least different from the user and using that viewer's rating for movie 5 to predict the user's rating for movie 5 is not very robust, i.e., it is likely to produce incorrect predictions.

TODO: Give some examples, what are the predictions for user XXX (find a user later in the dataset that is identical to one of the first three users, but has different predictions for movie 5

However, your work up to this point is not for naught. Rather than using just the viewer that is least different from the user, we can greatly improve the accuracy of our predictions by looking at the *k* least different dataset viewers, where the value of *k* can be adjusted to tune the system's prediction accuracy.

Unfortunately, finding the *k* least different viewers is more complicated than find just one, so as before, we will work through it in steps. The basic idea is to sort the `distances` array. By doing this, it will order the *k* minimum distances at the beginning of the array.

Start by using the Visual Basic built-in function, conveniently named `Array.Sort`, in the action for `btnPredictRatings` to sort the `distances` array. To do this, you would say:

```
Array.Sort(distances)
```

Next, output the sorted `distances` array to `outResults` to see that it is in fact sorted. For the input values 3.5, 4.5, 4.0, and 3.5, you should have output that looks like:

```
0.0
0.474
0.233
...
```

Now, can you tell which viewer is the least different from the user? Is it viewer 0? The problem that you are facing is the sorting the `distances` array does just that, but nothing else. Recall that the key to the previous part, i.e., using the minimum value to predict, was to keep track of the index in the `distances` array where the minimum value was found. Unfortunately `Array.Sort`, at least as we have used it above, does not keep track of the indices that each of the distances in their new sorted order, correspond to.

TODO Have students work on student name problem to see how the two parameter variant of `Array.Sort` can help them.

Fortunately, with a small modification, we can have `Array.Sort` produce a second array which stores the index from the unsorted array for each value in the sorted `distances` array. To do this, we must supply a second array to the `Array.Sort` function. We will call this array `viewerId`. For this to work, we must first populate the array `viewerId` with the values or the index for each value in the unsorted `distances` array and then use `viewerId` as the second argument to `Array.Sort`. For example:

```
For i = 0 To totalViewers - 1
    viewerId(i) = i
Next i

Array.Sort(distance, viewerID)
```

Now we can find out which viewer corresponds to a particular distance in the `distances` array like so:

```
outResults.AppendText("The distance between viewer #" & viewerID(0) & _
    " and the user is " & distances(0))
```

Now, reproduce the prediction from the previous part, i.e., predict based on the rating of the viewer that is the least different from the user. This time however, use the sorted array instead of computing the minimum distance in a loop. Output your prediction using the same statements as above.

Test the project by checking that it is still producing correct results for all of the test cases in Test Table 3.

Checkpoint 6 (90/100)

- ☐ project uses sorted parallel arrays, `distances` and `viewerId` to make predictions
- ☐ project produces correct output for all test cases in Test Table 3
- ☐ project produces correctly formatted output according to the example from Part5

7. Copy and paste the folder `RatingPredictions_Part6`. Rename the copy `RatingPredictions_Part7`. Launch the VS Express 2013 software and open the project `RatingPredictions_Part7`.

   Ask the user to input an integer, k, that will be used to choose the k closest matches to the user's sample. Using the sorted `dataset` array and the `sampleNumber` array, output the k least different dataset samples in a table formatted as follows:

```
Viewer ID   Movie five   Distance
---------------------------------
       39          3.0        1.0
       75          5.0        2.0
       76          4.5        2.0
        8          0.5        2.0
        6          3.0        2.0
---------------------------------
```

Checkpoint 7 (95/100)

- ☐ project uses sorted parallel arrays, `distances` and `userId` to make predictions
- ☐ project produces correctly formatted output according to the example above

8. Copy and paste the folder `RatingPredictions_Part7`. Rename the copy `RatingPredictions_Part8`. Launch the VS Express 2013 software and open the project `RatingPredictions_Part8`.

   Find the average of the top k values and use this as a prediction.

Checkpoint 8 (100/100)

- ☐ project uses the average of the top k most similar viewers from sorted parallel arrays, `distances` and `userId` to make predictions
- ☐ project produces correctly formatted output according to the example above

Submission Instructions

Your `M:\CS130\Labs\Lab06_YourLastName_YourFirstName` folder should contain your solutions to this and Tuesday's lab.

To submit your work, copy this folder and paste it to `N:/Handins/CS130/Lab06_Arrays_files` PRIOR to midnight tonight. Submissions received after 11:59pm tonight will be considered late and will receive a grade of 0.

You are not allowed to seek help from TAs on this lab outside lab time. TAs have been specifically instructed NOT to provide any help so please refrain from this activity.