Black Bear Sightings: You are helping the forest service in Alaska to track the number of black bear sightings. You have a collection data from seven different regions. This data spans 12 months and you are given the totals for each month, from each region. You need to compute the following

1. Average black bear sightings per month
2. Average black bear sightings per region

The data has been exported to a comma separated file of the following structure. Each line in the file represents the data from a region. Each region has 12 data points, one for each month.

black_bear_sightings.txt

```
41,3,4,47,70,69,56,88,40,34,5,76
1,92,35,30,88,9,49,40,100,32,59,82
86,78,48,77,93,60,24,9,85,81,61,77
47,38,3,43,35,42,39,13,26,22,4,85
55,21,21,41,2,52,16,93,43,16,74,28
12,91,93,31,10,83,20,15,78,21,35,84
67,23,12,14,66,78,59,7,48,38,66,72
```

Design Choices

1. When the program launches read the data from the file into a 2 dimensional array and then close the file. All calculations will happen on the array.
2. Regions will be numbered as ***Region 1, Region 2, . . . Region N***
3. Months will be labeled by their name.

Operations

We now have a two dimensional table (matrix) and we need to process this table by column and by row. To get the average per month we need to sum each column. To get the average per region we need to sum each row.

Average for Region One is: 44.41

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
0	41	3	4	47	70	69	56	88	40	34	5	76
1	1	92	35	30	88	9	49	40	100	32	59	82
2	86	78	48	77	93	60	24	9	85	81	61	77
3	47	38	3	43	35	42	39	13	26	22	4	85
4	55	21	21	41	2	52	16	93	43	16	74	28
5	12	91	93	31	10	83	20	15	78	21	35	84
6	67	23	12	14	66	78	59	7	48	38	66	72

Average for January is: 44.14

Program Setup

- Lines 9 and 10 define constants used for dimensioning the array. This allows for easy change of table size in the future.
- Line 13 declares the 2D array using the constants
- Line 14 declares an array of Strings to hold the month names. This way we can easily extract the correct month with a loop index without any conditional logic.

```
5   public class BlackBears{
6
7       public static void main(String[] args){
8
9           final int NUM_REGIONS   = 7;
10          final int NUM_MONTHS    = 12;
11
12          String      fileName    = "";
13          int[][]     sightings   = new int[NUM_REGIONS][NUM_MONTHS];
14          String[]    months      = { "January", "February", "March", "April", "May",
15                                      "June", "July", "August", "September", "October",
16                                      "November", "December"};
17
```

- Line 18 makes sure the file name was provided when the program was launched
- Lines 22 and 23 create instances of the classes necessary to setup the Scanner and connect it to a file.
- The nested loops on lines 26 through 32 load the data from the file line by line. Each line is split into an array, which is then iterated through. The individual data items are placed into table cells using the **[row][column]** indexing idiom. Notice that the constants are used again to control the loop variables. Simply changing those constant values would allow the loops to expand or contract as needed.

```
18          if(args.length > 0){
19              fileName = args[0];
20
21              try{
22                  FileReader fr   = new FileReader(fileName);
23                  Scanner scanner = new Scanner(fr);
24
25                  // load the table with the sightings data
26                  for(int i = 0; i < NUM_REGIONS; i++){
27                      String line     = scanner.nextLine();
28                      String[] values = line.split(",");
29                      for(int j = 0; j < NUM_MONTHS; ++j){
30                          sightings[i][j] = Integer.parseInt(values[j]);
31                      } // end inner for
32                  } // end outer for
33
```

Average Sightings Per Month

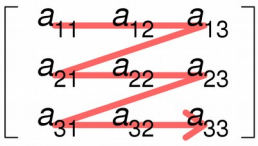
```
kenneth@dragonborn:~/OneDrive/Spring2020/165/examples/flow_control$ java BlackBears black_bear_sightings.txt
Average for January:     44.14
Average for February:    49.43
Average for March:       30.86
Average for April:       40.43
Average for May:         52.00
Average for June:        56.14
Average for July:        37.57
Average for August:      37.86
Average for September:   60.00
Average for October:     34.86
Average for November:    43.43
Average for December:    72.00
```

Months

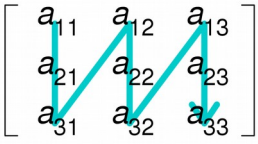
	0	1	2	3	4	5	6	7	8	9	10	11
0	41	3	4	47	70	69	56	88	40	34	5	76
1	1	92	35	30	88	9	49	40	100	32	59	82
2	86	78	48	77	93	60	24	9	85	81	61	77
3	47	38	3	43	35	42	39	13	26	22	4	85
4	55	21	21	41	2	52	16	93	43	16	74	28
5	12	91	93	31	10	83	20	15	78	21	35	84
6	67	23	12	14	66	78	59	7	48	38	66	72

January is month 0, February is month 1 etc. In order to accomplish this task we need to iterate through the array in **column major order;** processing the table one column at a time. This is different from how we loaded the table, which was in row **major order**. The difference between these two table iterations lies in the way we deal with our loop index variables.

Row-major order



Column-major order



Here is a review of the row major order code to fill the table. Notice that the outer loop is based on the number of regions. In the data set that is the row. One row for each region. Controlling the loop in this fashion means that we implement the outer loop with the major order dimension. The outer loop in this scenario will iterate 7 times. For each one of those steps, the inner loop will execute NUM_MONTHS times . . . or 12.

```
// load the table with the sightings data
for(int i = 0; i < NUM_REGIONS; i++){
    String line     = scanner.nextLine();
    String[] values = line.split(",");
    for(int j = 0; j < NUM_MONTHS; ++j){
        sightings[i][j] = Integer.parseInt(values[j]);
    } // end inner for
} // end outer for
```

To refactor this into column major order we simply reverse the structure of the for loops, placing the loop that goes NUM_MONTHS times with the loop that goes NUM_REGIONS times.

```
// compute average per month
for(int i = 0; i < NUM_MONTHS; ++i){
    double sum = 0.0;
    for(int j = 0; j < NUM_REGIONS; ++j)
        sum += sightings[j][i];
    System.out.printf("Average for %s:\t %.2f \n", months[i], sum / NUM_REGIONS);
}
```

Look carefully at this code because you also have to exchange the locations of loop control variables **i and j.**

- **Row Major Order:** [i][j]
- **Column Major Order:** [j][i]

Average Sightings Per Region

This is handled in simple row major order like the loading of the table. The outer loop control for the rows (regions) and the inner loop controls for the columns (months).

```
44              // compute average per region
45              for(int i = 0; i < NUM_REGIONS; ++i){
46                  double sum = 0;
47                  for(int j = 0; j < NUM_MONTHS; ++j)
48                      sum += sightings[i][j];
49                  System.out.printf("Average for Region %d:\t %.2f \n", i+1, sum / NUM_MONTHS);
50              }
51
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL

kenneth@dragonborn:~/OneDrive/Spring2020/165/examples/flow_control$ javac BlackBears.java
kenneth@dragonborn:~/OneDrive/Spring2020/165/examples/flow_control$ java BlackBears black_bear_sightings.txt
Average for Region 1:    44.42
Average for Region 2:    51.42
Average for Region 3:    64.92
Average for Region 4:    33.08
Average for Region 5:    38.50
Average for Region 6:    47.75
Average for Region 7:    45.83
```