CSC120 Week 11 Lab.: Precipitation Query

Instructor: Mitsu Ogihara

The goal of this assignment is to write a program, `Precipitation`, that reads a historical precipitation record of London from 1932 to 2013 and then compute statistics interactively with the user.

You can use a template `Precipitation-Template.java` for your work (make sure that you change the file name to `Precipitation.java` after downloading).

Let us see some execution examples first, to get the sense of how the program works.

```
% java Precipitation
1. Day value
2. Month stat
3. Annual stat
4. All data stat
0. Quit
Enter your choice: 1
Enter year month day: 1981 12 12
No.      1
Ave.     0.420
Max.     0.420
Min.     0.420
1. Day value
2. Month stat
3. Annual stat
4. All data stat
0. Quit
Enter your choice: 2
Enter year month: 1981 12
No.      31
Ave.     2.201
Max.     10.060
Min.     0.000
1. Day value
2. Month stat
3. Annual stat
4. All data stat
0. Quit
Enter your choice: 3
Enter year: 1981
```

```
No.      365
Ave.     3.835
Max.     41.890
Min.     0.000
1. Day value
2. Month stat
3. Annual stat
4. All data stat
0. Quit
Enter your choice: 4
No.      29951
Ave.     3.792
Max.     42.220
Min.     0.000
1. Day value
2. Month stat
3. Annual stat
4. All data stat
0. Quit
Enter your choice: 0
%
```

The data

The data is provided in a file `precipitation-data.txt` attached to this assignment. You must download it into the directory in which you are writing the code. Each line of the data file is in the following format:

YEAR MONTH VALUE1 ... VALUE31

VALUE1 through VALUE31 represent the values for the 31 days of the month of the year. For months with fewer than 31 days, -99.0 as the values for the non-existing dates (since precipitations cannot be negative). The lines appear in the increasing order of year and month. Because of this format, each year has $12 * 31 = 372$ entries. We will read values from the file and store them in a three-dimensional array of `double` ignoring the year and month values. The first dimension of the array is the year, where the index 0 represents year 1932, and the last index 82 represents 1983. The second dimension has length 12. The indices represent 0 .. 11. The last dimension has length 31. The indices represent 0 .. 30. For example, the index triple `[22][10][4]` is the data for November 5th, 1954 (that is, 1932 + 22 for the year, 10 + 1 for the month, and 4 + 1 for the day). Here are the first 12 lines with the first 4 and the last 4 for the days entries:

```
1932    1  10.69  26.46  13.09   ...   0.27   0.25   1.77
1932    2   0.14   0.90   0.30   ...   0.05 -99.99 -99.99
1932    3   0.14   0.03   0.08   ...   7.41  11.65   2.14
1932    4   6.35   9.11   1.66   ...   2.12   1.05 -99.99
1932    5   0.17   0.08   0.05   ...   0.75   0.98   0.00
```

```
1932    6   0.03   0.00   0.08   ...   7.22  15.37 -99.99
1932    7   1.80   6.01  13.47   ...   9.29   9.30   6.52
1932    8   0.08   1.55   1.58   ...   2.55   4.45  11.97
1932    9  14.82   5.91   7.67   ...   0.03   5.39 -99.99
1932   10   4.19   3.22   0.65   ...  15.85   7.44   2.80
1932   11  11.56   8.29  22.57   ...  17.49   4.35 -99.99
1932   12  15.50  15.10   3.42   ...   4.15   0.15   6.50
```

The Program Components

Constants

The program uses two constants

```
public static final int FIRST_YEAR = 1932;
public static final int LAST_YEAR = 2013;
```

and three non-main methods, `read`, `get`, and `stat`. The first is for reading the data from the data file for the program. The second is to obtain the value corresponding to any combination of year, month, and day. The third is for computing the average, maximum, and minimum of the data over a period.

The method read

The header is

```
public static double[][][] read()
```

You need some throws declaration. The method opens a `Scanner` object with the file name `"precipitation-data.txt"` by

```
Scanner sc = new Scanner( new File( "precipitation-data.txt" ) );
```

For instantiation, the dimension specification of the array of `double`, which we name `data`, is `[LAST - FIRST_YEAR +1][12][31]`. We then use a triple for-loop. The external loop is for the year makes an iteration from 0 to `LAST - FIRST_YEAR`, and the middle loop makes an iteration from 0 to 11. Inside the middle loop, the program makes two calls to `sc.nextInt()` to skip the year and month values, and then executes the innermost loop with index from 0 to 30, to read the data for the index combination.

After completion, the program closes the scanner and returns `data`.

The method get

The header is

```
public static double get( double[][][] data, int year, int month, in day )
```

The method "tries" to return `data[year - FIRST_YEAR][month-1][day-1]`. If the combination is invalid, the run-time error of `ArrayIndexOutOfBoundsException` occurs. We use `try-catch` to capture the error. If the error occurs, the method returns a negative value, say `-1`.

stat

This is a method for computing the stats over a range of years, a range of months, and a range of days. The method receives seven parameters

```
public static void stat( double[][][] data,
      int y1, int y2, int m1, int m2, int d1, int d2 )
```

We use a triple for-loop to access the data the ranges specify. We use an `int` variable `count` to keep track of how many value data values we find. We use three `double` variables `sum`, `max`, and `min` to keep track of the sum, the maximum, and the minimum of the valid values we find. The initial value of `count` is 0, and the initial values of the other three are 0, -1, and 100000. (The last two can be different.) To access data, we use a triple loop and obtain the value at the index combination using `get`. If the `value` the method returns is negative, the value is invalid. Otherwise, we add 1 to `count`, add the value to `sum`, and update the maximum and the minimum. After completing the triple loop, we report `count`, and then if `count > 0`, we report the values of `sum / count`, `max`, and `min` as the average, the maximum, and the minimum, respectively.

main

In the main method, we begin by reading the data with the method `read`. We will store the data in a variable `data`. We then declare variables, including the one that stores the choice of action that the user makes (0..4). Then we start a do-while loop. In the loop, we print the instruction and receive the choice from the user. Then, depending on the choice, we receive the year, the month, and the day from the user. Choice 1 requires all three, Choice 2 requires the first two, Choice 3 requires the first one, and Choice 4 requires none. If the value of year is absent, we call `stat` with `y1 = 1932` and `y2 = 2013`; if the value of month is absent, we call `stat` with `m1 = 1` and `m2 = 12`; and if the value of day is absent, we call `stat` with `d1 = 1` and `d2 = 31`. In other words, we use the same method `stat` for all four choices by appropriately setting the ranges.