Psuedocode for TwoDimRaggedArrayUtility.java

* Static getAverage(double[][]) method
  + Variables:
    - sum to hold the sum of the array
    - count to get the count of items in the array
  + Loop through the length of the rows of the passed 2d array
    - Loop through the columns of the array
      * Add 1 to count
      * Add the current value into the sum
  + Return the average by doing the sum divided by the count
* Static getColumnTotal(double[][], int) method
  + Variables:
    - sum to hold the sum of the array
  + Loop through the length of the rows of the passed 2d array
    - Check if the column is in bounds of the array
      * If it is, add the data at that column and row into the sum
      * If not, add 0 or just do nothing
  + Return the sum
* Static getHighestInArray(double[][]) method
  + Variables:
    - highest to hold the highest item in the array
  + Loop through all the rows of the array
    - Loop through all columns of the array
      * Check if the current column and row of the data is higher than the highest found.
        + If It is, set that as the highest
  + Return the highest in the array
* Static getHighestInColumn(double[][], int) method
  + Variables:
    - highest to hold the highest data in the array
  + Loop through all the rows
    - Check if the column requested is in bounds
      * If it is, check if the current row of the array are greater than the current highest
        + If it is, set that as the highest of the column
  + Return the highest in the column
* Static getHighestInColumnIndex(double[][], int) method
  + Variables:
    - highest to hold the highest data in the array
    - highestIndex to hold the index of the highest of the array
  + Loop through all the rows
    - Check if the column requested is in bounds
      * If it is, check if the current row of the array are greater than the current highest
        + If it is, set that as the highest of the column
        + Set the highestIndex to the current iteration found
  + Return the column index with the highest in its column
* Static getHighestInRow(double[][], int) method
  + Variables:
    - highest to hold the highest in the row
  + Loop through the columns of the selected row
    - Check if the current column is greater than the current highest
      * If it is, set that as the highest
  + Return the highest in the row
* Static getHighestInRowIndex(double[][], int) method
  + Variables:
    - highest to hold the highest in the row
    - highestIndex to hold the index of the highest column
  + Loop through the columns of the selected row
    - Check if the current column is greater than the current highest
      * If it is, set that as the highest
      * Set the highestIndex to the current iteration
  + Return the row index with the highest in the row
* Static getLowestInArray(double[][]) method
  + Variables:
    - lowest to hold the lowest in the array
  + Loop through every row
    - Loop through every column
      * Check if the current row and column is less than the lowest
        + If it is, set that as the lowest in the array
  + Return the lowest in the array
* Static getLowestInColumn(double[][], int) method
  + Variables:
    - lowest to hold the lowest in the column
  + Loop through all the rows
    - Check if the column requested is in bounds
      * If it is, check if the current row and requested column is less than the lowest
        + If it is, set that as the lowest
  + Return the lowest in the column
* Static getLowestInColumnIndex(double[][], int) method
  + Variables:
    - lowest to hold the lowest in the column
    - lowestIndex to hold the index of the lowest in the column
  + Loop through all the rows
    - Check if the column requested is in bounds
      * If it is, check if the current row and requested column is less than the lowest
        + If it is, set that as the lowest
        + Set lowestIndex to the current iteration of the loop
  + Return the index with the lowest number in the columns
* Static getLowestInRow(double[][], int) method
  + Variables:
    - lowest to hold the lowest in the row
  + Loop through the columns in the row selected
    - Check if the current column is less than the lowest
      * If it is, set that to the lowest
  + Return the lowest in the row
* Static getLowestInRowIndex(double[][], int) method
  + Variables:
    - lowest to hold the lowest in the row
    - lowestIndex to hold the index for the lowest in the row
  + Loop through the columns in the row selected
    - Check if the current column is less than the lowest
      * If it is, set that to the lowest
      * Set the lowestIndex to the current iteration of the loop
  + Return the row index with the lowest in the row
* Static getRowTotal(double[][], int) method
  + Variables:
    - sum to hold the sum of the row
  + Loop through the columns in the row
    - Add the current column in the selected row to the sum
  + Return the sum of the row
* Static getTotal(double[][]) method
  + Variables:
    - sum to hold the total of all items
  + Loop through all rows in the array
    - Loop through all columns of the array
      * Add the current row and column item to the sum
  + Return the sum
* Static readFile(File) method
  + Variables
    - numRows to hold the number of rows in the file
    - nextLine to get the data of the next line in the file
    - dataReader to read the file
    - data 2d array where to put the data
  + Try to read the file, if it gets an error, say that the file was not found
    - Set the dataReader to read the file
    - Check if the file has more lines
      * If it does, increase numRows by one and read again, do this until there are no more lines
    - Close the dataReader before reading the file again to preserve space
    - Read the file again
    - Loop through the number of rows in the file
      * Create an array that will hold the numbers in each row
      * Put the numbers split by a space into the array created before
      * Set the current array in data to a new array of the size of the array created before
      * Loop through the length of the array created before
        + Set the current item at row and column from the loops to the number found in the earlier array
  + Return the data 2d array
* Static writeToFile(double[][], File) method
  + Try to open a new file, and if it fails, tell the user it did
    - Set a file writer to where the user wants the file to be outputted at
    - Loop through the rows of the data passed
      * Loop through the columns of the data passed
        + Write the current data separated by a space
      * Put a new line at the end of the row
    - Close the fileWriter to prevent memory leakage

Pseudocode for HolidayBonus.java

* Static calcualteHolidayBonus(double[][], double, double, double) method
  + Variables:
    - bonus to hold the holiday bonus for all categories
  + Loop through every row of the data passed through
    - Check if the data is greater than 0
      * If it is, check if the current data is the highest in the column,
        + If it is, add the high bonus to the current iteration of bonus
      * If it’s not, check if it’s the lowest in the column
        + If it’s not, add the lowest bonus to the current iteration of bonus
      * If it’s not the lowest or the highest, then add the other bonus to the current iteration of the bonus
  + Return bonus
* Static calculateTotalHolidayBonus(double[][], double, double, double) method
  + Variables:
    - bonus to hold the holiday bonus for all categories
    - sum to hold the sum of all bonuses
  + Loop through every row of the data passed through
    - Check if the data is greater than 0
      * If it is, check if the current data is the highest in the column,
        + If it is, add the high bonus to the current iteration of bonus
      * If it’s not, check if it’s the lowest in the column
        + If it’s not, add the lowest bonus to the current iteration of bonus
      * If it’s not the lowest or the highest, then add the other bonus to the current iteration of the bonus
  + Loop through the length of the bonus array
    - Add the current iteration of bonus to the sum
  + Return the sum of all the holiday bonuses.

Test plan:

I will test random values and manually calculate their average. I will compile a unit test to test 3 cases and make sure that the class is working properly

Test cases:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case #** | **Input** | **Expected Output** | **Actual Output** | **Did the test pass?** |
| 1 | 1.4 54.5 7.77, 300 43.4 775.4,  78 4664.5 765 8, 400 56 643 | Total: 7796.97  Average: 599.77  Total of row 0:  63.67  Total of column 0:  779.4  Highest in array:  4664.5 | Same as expected | Yes, unit test passed, look at screenshot |
| 2 | 3 55 884 564,  634 66 34 6,  34 33 885 4,  42 75 32 6 | Total: 3357.0  Average: 209.81  Total of row 0: 1506.0  Total of column 0: 713.0  Highest in array: 885 | Same as expected | Yes, unit test passed, look at screenshot |
| 3 | 3 70 22 1 9,  80 69 50 32,  50 92 83 40,  17 10 21 63 | Total: 712.0  Average: 41.88  Total of row 0: 105.0  Total of column 0: 150.0  Highest in array: 92 | Same as expected | Yes, unit test passed, look at screenshot |

Screenshot:

A screenshot of a cell phone

Description automatically generated

UML Class Diagram:

A screenshot of a cell phone

Description automatically generated

Lessons learned:

In this assignment, I learned that is not very easy to loop through arrays. I had some trouble when the columns were over the bounds of the array so I had to make a special check every time I wanted to check columns so that I would not run into the same issue. I also had some trouble when writing the code for HolidayBonus.java.

When I was working through the Javadoc, there was no mention that a holiday bonus should not be added if the data is below 0 and I didn’t realize that I had issues with negative numbers until I talked to Priscila. Once I made the change to only include when the data was over 0, the issue was fixed and my code worked as intended.