Arrays, Methods and File Processing

This lab requires you to write methods to read and manipulate data that is stored in a text file. It is designed to give you more hands on practice with arrays and methods before Classes and Objects are reintroduced. You will also read and write data to a file using exception handling (try/catch block).

```
try {
      //perform some operation that could cause an exception
}
catch ( SOME_EXCEPTION_TYPE exception ) {
      //Actions to perform if there is an exception
}
```

Use your programming IDE to create a new project. Name your project PlayerStats. In the main method, declare and instantiate two arrays of size 80. One array to hold the player's name (names) and another array to hold the player's integer high game (scores). These arrays are considered parallel because corresponding elements in the arrays refer to the same player. For example, the high score in scores[15] belongs to the player in names[15]. So, if you ever move a player name to another location in the names array (for example, during sorting), you must also move the score to the same location in the scores array to keep them synchronized.

names

Before	After
Kia	Kia
Adam	Caleb
Caleb	Adam

scores

Before	After
568	568
239	450
450	239

For each of the following problems, you should modify the main method to invoke the method and show output that proves that your method is working correctly.

1. Write a method that will read player names and high scores from the input file into the parallel arrays. The parameters for the method are the two arrays and the input file name (String). The method has a void return type. For this problem, set a breakpoint on the method call and then issue a step-over debugger command. After the step-over command, click on the "Variables" tag and inspect the names and scores arrays to see if they contain the expected 80 values.
2. Write a method that will output the parallel arrays to the console. Your output should have two columns: Player Name and High Score. Research the String.format() method in order to get your output in aligned columns.

3. Write a method that will return the average (double) high score of all the players. You will need to pass the method the scores array.
4. Write a method that will use Selection Sort to sort the arrays in decreasing high score order. Remember, you need keep the names and scores matched up during the array swap operation.
5. Write a method that will output the top X players and their scores. So, if topX=10, it will output the top 10 scores. The parameters for the method are the two arrays and topX.