```java
/* Gross Pay, Savings, and Investment Calculator read from a file
   and output to a file.
   This program purpose is to calculate gross pay, savings, and investment based
   off of data stored in a txt file and then write the data to a new file.
   Zachary Stall
   Program #4, CS 1050, Section 2
   jGRASP, Custom PC, Windows 10
   Assiduoud - Constant in application or attention; dilligen.
   "Great things are done by a series of small things brought together."
   -Vincent Van Gogh (1853 - 1890)
*/


import java.util.Scanner;           // Access the Scanner class
import java.io.*;                   // Access PrintWriter and related classes


public class ZacharyStall_2_04 {

   static Toolkit tools = new Toolkit();

   public static void main(String[] args) throws IOException {


      // Files to get inputs and write outputs
      // All variable declared and described below

      final String INPUT_FILE  = "ZacharyStall_2_04_Input.txt";
      final String OUTPUT_FILE = "ZacharyStall_2_04_Output.txt";

      int numInputLines = 0;        // Number of lines in the input file
      int numValidLines = 0;        // Number of valid lines in the input file
      double grossPay = 0.0;        // Input file's gross pay
      double savingsRate = 0.0;     // Input file's savings rate
      double iraRate = 0.0;         // Input file's IRA investment rate
      double savingsAmount = 0.0;   // Calculated percentage saved
      double iraAmount = 0.0;       // Calculated percentage in ira
      double sumGrossPay = 0.0;     // Sum of all valid gross pay amounts
      double sumSavings = 0.0;      // Sum of all valid savings amounts
      double sumIra = 0.0;          // Sum of all valid IRA investment amounts
      double grossAverage = 0.0;    // Average of all valid gross salary
      double savingsAverage = 0.0;  // Average of all valid savings
      double iraAverage = 0.0;      // Average of all valid ira amounts

      String lineHeaders;           // Stores string for headers
      String tableData;             // Stores string to create table
      String lineOutput;            // Stores string to output data
      String sumGrossPayStr;        // String var for formating and output
      String sumSavingStr;          // String var for formating and output
      String sumIraStr;             // String var for formating and output
      String grossAverageStr;       // String var for formating and output
      String savingsAverageStr;     // String var for formating and output
      String iraAverageStr;         // String var for formating and output

      // Access the input/output files

      File inputDataFile = new File(INPUT_FILE);
      Scanner inputFile  = new Scanner(inputDataFile);

      FileWriter outputDataFile = new FileWriter(OUTPUT_FILE);
      PrintWriter outputFile = new PrintWriter(outputDataFile);

      // Begin program execution

      System.out.println("Reading  file " + INPUT_FILE  + "\r\n" +
                         "Creating file " + OUTPUT_FILE + "\r\n");


      // Headers for the table of values for console and output file
      lineHeaders = tools.padString("Grosspay", 12, " ", "") +
         tools.padString("Savings Rate", 15, " ", "") +
         tools.padString("Savings", 15, " ", "") +
         tools.padString("IRA Rate", 12, " ", "") +
         tools.padString("IRA", 15, " ", "") +
         "\r\n";

      outputFile.print(lineHeaders);
      System.out.print(lineHeaders);

      // Read the input file and sum the numbers.
      while (inputFile.hasNext()) {
         numInputLines++;                       // Adds the number of lines in input file

         grossPay = inputFile.nextDouble();     // Gets grossPay from input file

         savingsRate = inputFile.nextDouble();  // Gets savingRate from input file

         iraRate = inputFile.nextDouble();      // Gets iraRate from input file



         // Calculates the amount of money in savings and IRA

         savingsAmount = (savingsRate / 100.0) * grossPay;
         iraAmount = (iraRate / 100.0) * grossPay;

         /*
         If statement checks that the line input is a valid line,
         meaning that all values in the line are positive.If the
         line is valid, it adds it to valid lines, and each
         variable respectively.
         */
         if(grossPay > 0 && savingsRate > 0 && iraRate > 0) {
            numValidLines ++;
            sumGrossPay += grossPay;
            sumSavings += savingsAmount;
            sumIra += iraAmount;
         }

         // Formats numbers and stores them in tableData
         tableData =
            tools.leftPad(grossPay, 12, "##,##0.00") +
            tools.leftPad(savingsRate, 15, "#0.0") +
            tools.leftPad(savingsAmount, 15, "#,##0.00") +
            tools.leftPad(iraRate, 12, "#0.0") +
            tools.leftPad(iraAmount, 15, "#,##0.00") +
            "\r\n";
         // Prints tableData to file and console
         outputFile.print(tableData);
         System.out.print(tableData);
      } // End while

      /*
      If there is data then the averages will be calculated and stored
      into their respective variables. If there is not data then to avoid
      dividing by zero, the else statement will assign zero to the averages
      and warn the user that the input file is empty.
      */
      if(numValidLines > 0) {
      grossAverage = sumGrossPay / numValidLines;
      savingsAverage = sumSavings / numValidLines;
      iraAverage = sumIra / numValidLines;
      }
      else {
      System.out.println("ERROR: FILE CONTAINS NO DATA!");
      outputFile.println("ERROR: FILE CONTAINS NO DATA!");
      grossAverage = 0.0;
      savingsAverage = 0.0;
      iraAverage = 0.0;
      }

      /*
      formats all the data collected, and outputs all the data to the file
      and the console
      */
      sumGrossPayStr = tools.leftPad(sumGrossPay, 17, "$###,##0.00");
      sumSavingStr = tools.leftPad(sumSavings, 19, "$##,##0.00");
      sumIraStr = tools.leftPad(sumIra, 11, "$##,##0.00");
      grossAverageStr = tools.leftPad(grossAverage, 16, "$##,##0.00");
      savingsAverageStr = tools.leftPad(savingsAverage, 11, "$##,##0.00");
      iraAverageStr = tools.leftPad(iraAverage, 15, "$##,##0.00");

      // stores out put string to lineOutput
      lineOutput =
        "\r\n" + "The number of input lines read: " + numInputLines
      + "\r\n" + "The number of valid input lines read: " + numValidLines + "\r\n" + "\r\n"
      + "The sum of gross pay: " + sumGrossPayStr + "\r\n"
      + "The sum of savings: " + sumSavingStr + "\r\n"
      + "The sum of IRA investments: " + sumIraStr + "\r\n" + "\r\n"
      + "The average gross pay: " + grossAverageStr + "\r\n"
      + "The average savings amount: " + savingsAverageStr + "\r\n"
      + "The average ira amount: " + iraAverageStr + "\r\n";

      // prints lineOutput to the file and console
      outputFile.println(lineOutput);
      System.out.println(lineOutput);

      inputFile.close();
      outputFile.close();

      System.exit(0);

   } // End main
} // End class
```