Test cases

|  |  |
| --- | --- |
| District sets | Implementation |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| Pseudocode | Java code |
| Initializes array  Creates new scanner object  Checks if file has next line  Scanner reads next line  Counts number of rows and initializing them to data  Closes file  Creats new scanner  Checks if scanner reads another line  Initializes columns and a temporary string  Initializing cols scanner object  Checks if col has next line  Increments columns  Initializes data[rowIndex] tor cols  Closes col  Increments rowIndex  Closes scanner object  Creates new scanner object that reads from file  Sets row index to 0  Checks if scanner has next line  Stores data from scanner into temp  Reads line from temp  Checks if col has next element  Converts from string to double and stores in the data array  Increments colIndex  Closes col  Sets colIndex to 0  Closes scanner object  Returns data array  Prints to a file with a space between the variables  Adds all elements in the data array  Calculates highest in the array  Returns highest variable  Finds lowest element in array  Finds highest in column  Returns max variable  Finds Lowest variable in the column  Returns min variable  Calculates column Index  Returns row;  Calculates lowest in column index  Returns row  Calculates column total  Returns column total  Finds highest element in row  Returns max variable  Gets the highest in row index  Finds lowest element in the data array  Returns the minimum value in the data array  Finds lowest in row index from the data array  Returns the minimum index  Gets Average of the entire data array  Returns sum divided by counter | public static double[][] readFile(java.io.File file)  throws java.io.FileNotFoundException {  int cols = 0,rows = 0, rowIndex = 0, colIndex = 0;  double data[][] = null;  if (file!=null)  {  Scanner scanner = new Scanner(file);  while (scanner.hasNextLine())  {  scanner.nextLine();  rows++;  }  data = new double [rows][];  scanner.close();  scanner = new Scanner (file);  while (scanner.hasNextLine())  {  cols = 0;  String temp = scanner.nextLine();  Scanner col =new Scanner (temp);  while(col.hasNext())  {  col.next() ;  cols++;  }  data [rowIndex]= new double [cols];  col.close();  rowIndex++;  }  scanner.close();  scanner = new Scanner (file);  rowIndex = 0;  while (scanner.hasNextLine())  {  String temp = scanner.nextLine();  Scanner col =new Scanner (temp);  while(col.hasNext())  {  data [rowIndex][colIndex] = Double.parseDouble(col.next()) ;  colIndex++;  }  col.close();  colIndex = 0;  rowIndex++;  }  scanner.close();  }  return data;  }  public static void writeToFile(double[][] data,java.io.File outputFile)throws java.io.FileNotFoundException  {  PrintWriter printwriter = new PrintWriter(outputFile);  for (int row = 0; row < data.length; row++) {  for (int col = 0; col < data[row].length; col++) {  printwriter.print(data[row][col] +" ");  }  printwriter.println();  }  }  public static double getTotal(double[][] data) {  double sum = 0;  for (int row = 0; row < data.length; row++) {  for (int col = 0; col < data[row].length; col++) {  sum = sum + data[row][col];  }  }  return sum;  }  public static double getHighestInArray(double[][] data) {  double max = 0;  for (int row = 0; row < data.length; row++) {  for (int col = 0; col < data[row].length; col++) {  if (max < data[row][col])  max = data[row][col];  }  }  return max;  }  public static double getLowestInArray(double[][] data) {  double min = data[0][0];  for (int row = 0; row < data.length; row++) {  for (int col = 0; col < data[row].length; col++) {  if (min > data[row][col])  min = data[row][col];  }  }  return min;  }  public static double getHighestInColumn (double[][] data, int col)  {  double max = 0;  double maxm = 0;  for (int j = 0; j < data.length; j++)  {  if(col < data[j].length)  {  if (data[j][col] > maxm)  maxm = data[j][col];  }  }  max = maxm;  return max;  }  public static double getLowestInColumn (double[][] data, int col)  {  double min = 0;  for (int j = 0; j < data.length; j++)  {  for ( int i = 0; i < data[j].length; i++ )  {  if(col < data[j].length){  double minm = data[j][col];  if (data[j][col] < minm)  {  minm = data[j][col];  min = minm;  }  }  }  }  return min;  }  public static int getHighestInColumnIndex (double[][] data, int col)  {  int row= 0;  double maxm = 0;  for (int j = 0; j < data[col].length; j++)  {  if(col < data[j].length)  {  if (data[j][col] > maxm)  row = j;  }  }  return row;  }  public static int getLowestInColumnIndex (double[][] data, int col)  {  int row= 0;  for (int j = 0; j < data.length; j++)  {  for (int i = 0; i < data.length; i++)  {  if(col < data[j].length)  {  double maxm = data[j][col];  if (data[j][col] <maxm)  row = j;  }  }  }  return row;  }  public static double getColumnTotal (double[][] data,int col)  {  double total = 0;  for (int j = 0; j < data.length; j++){  if(col < data[j].length)  {  total += data[j][col];  }  }  return total;  }  public static double getRowTotal(double[][] data, int row) {  double total = 0;  for (int i = 0; i < data[row].length; i++) {  total += data[row][i];  }  return total;  }  public static double getHighestInRow(double[][] data, int row) {  double max = 0;  for (int j = 0; j < data[row].length; j++)  {  if (data[row][j] > max) {  max = data[row][j];  }  }  return max;  }  public static int getHighestInRowIndex(double[][] data, int row) {  int max = 0;  for (int j = 0; j < data[row].length; j++)  {  if (data[row][j] > max) {  max = j;  }  }  return max;  }  public static double getLowestInRow(double[][] data, int row) {  double min = 0;  for (int j = 0; j < data[row].length; j++)  {  if (data[row][j] < min) {  min = data[row][j];  }  }  return min;  }  public static double getLowestInRowIndex(double[][] data, int row) {  double min = 0;  for (int j = 0; j < data[row].length; j++)  {  if (data[row][j] < min) {  min = j;  }  }  return min;  }  public static double getAverage(double data[][]){  int counter=0;  double sum = 0;  for(int i=0;i<data.length;i++){  for(int j=0;j<data[i].length;j++){  sum = sum+data[i][j];  counter++;  }  }  return sum / counter;  }  } |

Holiday Bonus

|  |  |
| --- | --- |
| Pseudocode | Java code |
| Creates a single array  Stores variables from the data array into single array  Validates if variable is more that zero  Calculates lowest holiday bonus  Calculates highest holiday bonus  Returns single  Calculates the lowest holiday bonus and adds it to the total  Calculates the highest holiday bonus and adds it to the total  Calculates the other holiday bonuses and adds them to the total  Returns total | public static double []calculateHolidayBonus(double[][] data, double high,double low, double other)  {  double single[] = new double[data.length];    for (int i =0; i< data.length;i++)  {  for(int j=0;j < data[i].length;j++)  {  if (data[i][j] > 0) {  if(i == TwoDimRaggedArrayUtility.getLowestInColumnIndex(data, j))  {  single[i] +=low;  }  else if (i == TwoDimRaggedArrayUtility.getHighestInColumnIndex(data, j))  {  single [i] +=high;  }  else  {  single[i] += other;  }  }  }  }  return single;    }    public static double calculateTotalHolidayBonus(double[][] data,double high,double low,double other)  {  double total = 0;    for (int i =0; i< data.length;i++)  {  for(int j=0;j < data[i].length;j++)  {  if (data[i][j] > 0) {  if(i == TwoDimRaggedArrayUtility.getLowestInColumnIndex(data, j))  {  total +=low;  }  else if (i == TwoDimRaggedArrayUtility.getHighestInColumnIndex(data, j))  {  total +=high;  }  else  {  total += other;  }  }  }  }  return total;  }  } |

Learning Experience

This project was challenging because of the aspect of reading from and writing to a file and finding the lowest array index I will go over the concepts so that I am more familiar with them and learning how to get help from the tutoring center.The most challenging aspect of this project was reading from a file and the indexes it was hard of the array out of bounds errors and getting the right information on to the data array.