**[CSE 1310](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/index.html) -** [**Assignments**](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/index.html) **- Programming Assignment 9**

The assignment will be graded out of 100 points.

Some tasks ask you to write code, and specify what name to use for the file where you save that code. You need to use exactly the name that is given (do not change the case, or make any other modification). Remember, the name of the main class must match the filename.

For some tasks you need to answer questions. Create a text document entitled answers.txt, or answers.docx, or answers.pdf, and put all your answers there. Acceptable file formats are plain text, Word document, OpenOffice document, and PDF. Put your name and UTA ID in the file on the first line.

Each task below will instruct you where to put your answers.

**Task 1 (10 pts.)**

File [task1.java](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/task1.java) contains an incomplete program. The goal of the program is to:

1. Read a file that contains integers.
2. Determine if each of those integers is a prime or not.

Complete that program, by defining a check\_primes function, that satisfies the following specs:

* Function check\_primes takes two arguments, called in\_file, out\_file. Argument in\_file specifies the name of the input file, that contains the integers to be processed. Argument out\_file specifies the name of the output file, where the results will be saved.
* For each integer X stored in in\_file, the function writes to out\_file a line stating either "X is prime" or "X is not prime".
* For example, for input file [in1.txt](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/in1.txt), the program should produce an output file that looks EXACTLY like [out1.txt](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/out1.txt).

**IMPORTANT: you are only allowed to modify the provided code by writing the requested function, and possibly additional auxiliary functions. You are NOT allowed to modify in any way the main function, or any other function that is already provided.** You are free to use any piece of code from the slides, or the course website, or from anywhere else, that you find useful for solving this task.

Hint: the slides on functions contain a function called is\_prime that can be useful here.

**Task 2 (20 pts.)**

File [task2.java](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/task2.java) contains an incomplete program. The goal of the program is to count the number of times 'a', 'e', 'i', 'o', 'u' (and their upper case equivalents) appear in a file.

Complete that program, by defining a count\_vowels function, that satisfies the following specs:

* Function count\_vowels takes one argument, called in\_file, that specifies the name of the input file
* The function counts and prints the number of times 'a', 'e', 'i', 'o', 'u' (and their upper case equivalents) appear in a file. Counting should be case-insensitive, so an 'A' should be counted as an 'a' and so on.

**IMPORTANT: you are only allowed to modify the provided code by writing the requested function, and possibly additional auxiliary functions. You are NOT allowed to modify in any way the main function, or any other function that is already provided.** You are free to use any piece of code from the slides, or the course website, or from anywhere else, that you find useful for solving this task.

For example, for input file [in2.txt](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/in2.txt), the complete program should produce EXACTLY this output:

a: 6 times

e: 5 times

i: 4 times

o: 2 times

u: 5 times

**Task 3 (10 pts.)**

File [task3.java](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/task3.java) contains an incomplete program, that prints the name of a column in a CSV file. In this program, we make the assumption that the names of all columns are shown at the top line of the CSV file.

Complete that program, by defining a column\_name function, that satisfies the following specs:

* The function takes two arguments, called filename, column. Argument filename is a string specifying the name of a CSV file. Argument column is an integer, specifying a column in the CSV file. Columns are counted starting at 0, so the leftmost column is column 0.
* The function returns the name of the specified column in the specified CSV file. If the column is invalid (less than 0 or too high) the function should return null.

**IMPORTANT: you are only allowed to modify the provided code by writing the requested function, and possibly additional auxiliary functions. You are NOT allowed to modify in any way the main function, or any other function that is already provided.** You are free to use any piece of code from the slides, or the course website, or from anywhere else, that you find useful for solving this task.

Example CSV files you can test your program with are [nba.txt](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/nba.txt) and [weather1.txt](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/weather1.txt).

An example run of the complete program is shown below.

run:

Enter a filename (or q to quit): nba.txt

Enter a column: -1

Failed to extract a valid column name for column -1 of nba.txt

Enter a filename (or q to quit): nba.txt

Enter a column: 0

In file nba.txt, column 0 is "player"

Enter a filename (or q to quit): nba.txt

Enter a column: 5

In file nba.txt, column 5 is "field goal attempts"

Enter a filename (or q to quit): nba.txt

Enter a column: 10

In file nba.txt, column 10 is "free throws made"

Enter a filename (or q to quit): nba.txt

Enter a column: 15

In file nba.txt, column 15 is "steals"

Enter a filename (or q to quit): nba.txt

Enter a column: 16

In file nba.txt, column 16 is "blocks"

Enter a filename (or q to quit): nba.txt

Enter a column: 17

Failed to extract a valid column name for column 17 of nba.txt

Enter a filename (or q to quit): weather1.txt

Enter a column: 0

In file weather1.txt, column 0 is "Date"

Enter a filename (or q to quit): weather1.txt

Enter a column: 3

In file weather1.txt, column 3 is "Rain"

Enter a filename (or q to quit): weather1.txt

Enter a column: 4

Failed to extract a valid column name for column 4 of weather1.txt

Enter a filename (or q to quit): q

Exiting...

Your program's output should match EXACTLY the format shown above. There should be no deviations, no extra spaces or lines, no extra punctuation in your output. What you see above as uppercase letters should remain uppercase in your output, what you see as lowercase letters should remain as lowercase in your output, what you see as spaces and punctuation should remain exactly as spaces and punctuation in your output.

**Task 4 (10 pts.)**

File [task4.java](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/task4.java) contains an incomplete program, that prints the sum of all values in a specific column of a CSV file. In this program, we make the assumption that the names of all columns are shown at the top line of the CSV file, so when we sum up the values in a column we should exclude the top line.

Complete that program, by defining a column\_sum function, that satisfies the following specs:

* The function takes two arguments, called filename, column. Argument filename is a string specifying the name of a CSV file. Argument column is an integer, specifying a column in the CSV file. Columns are counted starting at 0, so the leftmost column is column 0.
* The function returns (as a double number) the sum of all numbers found in the specified column in the specified CSV file.
* In this task, it is OK to assume that, for every row of the file except the top row, the specified column exists and contains a valid double number.

**IMPORTANT: you are only allowed to modify the provided code by writing the requested function, and possibly additional auxiliary functions. You are NOT allowed to modify in any way the main function, or any other function that is already provided.** You are free to use any piece of code from the slides, or the course website, or from anywhere else, that you find useful for solving this task.

Example CSV files you can test your program with are [nba.txt](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/nba.txt) and [weather1.txt](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/weather1.txt).

An example run of the complete program is shown below.

Enter a filename (or q to quit): weather1.txt

Enter a column: 1

In file weather1.txt, column 1 has sum 1177.00

Enter a filename (or q to quit): weather1.txt

Enter a column: 2

In file weather1.txt, column 2 has sum 906.00

Enter a filename (or q to quit): weather1.txt

Enter a column: 3

In file weather1.txt, column 3 has sum 0.60

Enter a filename (or q to quit): nba.txt

Enter a column: 5

In file nba.txt, column 5 has sum 2346.90

Enter a filename (or q to quit): nba.txt

Enter a column: 12

In file nba.txt, column 12 has sum 19767.10

Enter a filename (or q to quit): q

Exiting...

Your program's output should match EXACTLY the format shown above. There should be no deviations, no extra spaces or lines, no extra punctuation in your output. What you see above as uppercase letters should remain uppercase in your output, what you see as lowercase letters should remain as lowercase in your output, what you see as spaces and punctuation should remain exactly as spaces and punctuation in your output.

**Task 5 (10 pts.)**

This task is almost identical to task 4. File [task5.java](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/task5.java) contains an incomplete program, that prints the sum of all values in a CSV file.

Complete that program, by defining a column\_sum2 function, that can handle rows where the specified column is not present, or does not contain a valid number. Here are the specs for column\_sum2:

* Same as in Task 4, the function takes two arguments, called filename, column. Argument filename is a string specifying the name of a CSV file. Argument column is an integer, specifying a column in the CSV file. Columns are counted starting at 0, so the leftmost column is column 0.
* Same as in Task 4, the function returns (as a double number) the sum of all numbers found in the specified column in the specified CSV file.
  + Unlike Task 4, if, for any row, the specified column is invalid (less than 0 or too high) the function should skip that row and continue to the next.
  + Unlike Task 4, if, for any row, the specified column does not contain a valid double number (i.e., if Double.parseDouble fails to convert the text in that column into a number) the function should skip that row and continue to the next one.

**IMPORTANT: you are only allowed to modify the provided code by writing the requested function, and possibly additional auxiliary functions. You are NOT allowed to modify in any way the main function, or any other function that is already provided.** You are free to use any piece of code from the slides, or the course website, or from anywhere else, that you find useful for solving this task.

If you solve this task correctly, the exact same solution also works for Task 4 (just change the function name to column\_sum).

Example CSV files you can test your program with are [nba.txt](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/nba.txt), [weather1.txt](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/weather1.txt), and [crazy\_weather.txt](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/crazy_weather.txt).

An example run of the complete program is shown below.

Enter a filename (or q to quit): crazy\_weather.txt

Enter a column: -1

In file crazy\_weather.txt, column -1 has sum 0.00

Enter a filename (or q to quit): crazy\_weather.txt

Enter a column: 0

In file crazy\_weather.txt, column 0 has sum 0.00

Enter a filename (or q to quit): crazy\_weather.txt

Enter a column: 1

In file crazy\_weather.txt, column 1 has sum 1177.00

Enter a filename (or q to quit): crazy\_weather.txt

Enter a column: 2

In file crazy\_weather.txt, column 2 has sum 906.00

Enter a filename (or q to quit): crazy\_weather.txt

Enter a column: 3

In file crazy\_weather.txt, column 3 has sum 0.60

Enter a filename (or q to quit): crazy\_weather.txt

Enter a column: 4

In file crazy\_weather.txt, column 4 has sum 0.00

Enter a filename (or q to quit): q

Exiting...

Your program's output should match EXACTLY the format shown above. There should be no deviations, no extra spaces or lines, no extra punctuation in your output. What you see above as uppercase letters should remain uppercase in your output, what you see as lowercase letters should remain as lowercase in your output, what you see as spaces and punctuation should remain exactly as spaces and punctuation in your output.

**Task 6 (20 pts.)**

File [task6.java](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/task6.java) contains an incomplete program, that processes CSV files containing weather data, like [weather1.txt](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/weather1.txt). For each line in the file (except the top one), the program prints out the month that that line corresponds to.

In this program, we make the assumption that the names of all columns are shown at the top line of the CSV file. We also make the assumption that the CSV file contains four columns, and the leftmost column specifies the date, in the format "day\_name month/day/year".

Complete that program, by defining a get\_month function, that satisfies the following specs:

* The function takes one argument, called date. Argument filename is a string specifying the date, in the format used in the [weather1.txt](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/weather1.txt) file.
* The function returns (as an int) the month from that date.

Note that the code provides a print\_months function, which is called from main. The print\_months function calls the get\_month function which you are asked to write.

**IMPORTANT: you are only allowed to modify the provided code by writing the requested function, and possibly additional auxiliary functions. You are NOT allowed to modify in any way the main function, or any other function that is already provided.** You are free to use any piece of code from the slides, or the course website, or from anywhere else, that you find useful for solving this task.

Example CSV files you can test your program with are [weather1.txt](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/weather1.txt), [weather2.txt](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/weather2.txt) and [weather3.txt](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/weather3.txt).

An example run of the complete program is shown below.

Enter a filename (or q to quit): weather1.txt

row 1, month = 3

row 2, month = 3

row 3, month = 3

row 4, month = 3

row 5, month = 3

row 6, month = 3

row 7, month = 4

row 8, month = 4

row 9, month = 4

row 10, month = 4

row 11, month = 4

row 12, month = 4

row 13, month = 7

row 14, month = 7

row 15, month = 7

Enter a filename (or q to quit): weather3.txt

row 1, month = 3

row 2, month = 3

row 3, month = 7

row 4, month = 3

row 5, month = 3

row 6, month = 4

row 7, month = 4

row 8, month = 3

row 9, month = 3

row 10, month = 4

row 11, month = 4

row 12, month = 7

row 13, month = 7

row 14, month = 4

row 15, month = 4

Enter a filename (or q to quit): q

Exiting...

Your program's output should match EXACTLY the format shown above. There should be no deviations, no extra spaces or lines, no extra punctuation in your output. What you see above as uppercase letters should remain uppercase in your output, what you see as lowercase letters should remain as lowercase in your output, what you see as spaces and punctuation should remain exactly as spaces and punctuation in your output.

**Task 7 (20 pts.)**

File [task7.java](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/task7.java) contains an incomplete program, that processes CSV files containing weather data, like [weather3.txt](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/weather3.txt). that prints the average of all values in a specific column of the file.

In this program, we make the assumption that the names of all columns are shown at the top line of the CSV file. We also make the assumption that:

* As in task 6, the CSV file contains four columns, and the leftmost column specifies the date, in the format "day\_name month/day/year".
* The other columns (except for the leftmost one) contain numbers.

Complete that program, by inserting your column\_name function from Task 3, and by defining a monthly\_average function, that satisfies the following specs:

* The function takes three arguments, called filename, column. Argument filename is a string specifying the name of a CSV file. Argument column is an integer, specifying a column in the CSV file. Argument month is an integer, specifying the month for which we want to compute the average. Columns are counted starting at 0, so the leftmost column is column 0.
* The function returns (as a double number) the average of all numbers found in the specified column in the specified CSV file. If no values were found that match the specified month, the function should return -1.
* In this task, it is OK to assume that, for every row of the file except the top row, the specified column exists and contains a valid double number.

**IMPORTANT: you are only allowed to modify the provided code by writing the requested function, and possibly additional auxiliary functions. You are NOT allowed to modify in any way the main function, or any other function that is already provided.** You are free to use any piece of code from the slides, or the course website, or from anywhere else, that you find useful for solving this task.

Example CSV files you can test your program with are [weather1.txt](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/weather1.txt), [weather2.txt](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/weather2.txt) and [weather3.txt](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment9/weather3.txt).

An example run of the complete program is shown below.

Enter a filename (or q to quit): weather3.txt

Enter a column: 1

Enter a month: 1

In file weather3.txt, there is no data for Temperature High for month 1.

Enter a filename (or q to quit): weather3.txt

Enter a column: 1

Enter a month: 4

In file weather3.txt, the average Temperature High for month 4 is 82.33.

Enter a filename (or q to quit): weather2.txt

Enter a column: 1

Enter a month: 4

In file weather2.txt, the average Temperature High for month 4 is 76.57.

Enter a filename (or q to quit): weather2.txt

Enter a column: 2

Enter a month: 4

In file weather2.txt, the average Temperature Low for month 4 is 57.80.

Enter a filename (or q to quit): weather2.txt

Enter a column: 3

Enter a month: 5

In file weather2.txt, the average Rain for month 5 is 0.61.

Enter a filename (or q to quit): q

Exiting...

Your program's output should match EXACTLY the format shown above. There should be no deviations, no extra spaces or lines, no extra punctuation in your output. What you see above as uppercase letters should remain uppercase in your output, what you see as lowercase letters should remain as lowercase in your output, what you see as spaces and punctuation should remain exactly as spaces and punctuation in your output.

**Suggestions**

Pay close attention to all specifications on this page, including file names and submission format. Even in cases where the program works correctly, points will be taken off for non-compliance with the instructions given on this page (such as wrong file names, wrong compression format for the submitted code, and so on). The reason is that non-compliance with the instructions makes the grading process significantly (and unnecessarily) more time consuming. Contact the instructor or TA if you have any questions.

**How to submit**

The assignment should be submitted via [Blackboard](http://elearn.uta.edu). Submit a ZIPPED directory called assignment9.zip (no other forms of compression accepted, contact the instructor or TA if you do not know how to produce .zip files).

To create a zipped directory called assignment9.zip, follow these steps:

1. Create a folder called assignment9.
2. Copy to that folder all your Java files.
3. Zip that folder. On Windows, you can zip a folder by right-clicking on the folder, and then selecting Send to->Compressed (zipped) folder.

Your zip file should contain only 7 files: task1.java, task2.java, task3.java, task4.java, task5.java, task6.java, task7.java.