SAP Coding Masters 2013

Introduction

The following document describes the challenges we have set up to determine this year's SAP Coding Masters candidates.

We have installed an online judge system where you can login with your credentials and load up coding in either C, C++, Java or Python. This coding will then be automatically executed and the results will be checked. You get feedback weather or not your code solved the problem.

Here are some things to keep in mind when working on the challenges:

- The scoring takes into account:
  - Time of upload (sooner is better)
  - Penalty for wrong uploads (fewer attempts is better)
- However, you are free to submit each task as often as you wish.
- The judge system will use the standard in / standard out streams to communicate with your program. You get all inputs via standard in and have to write all outputs into standard out. The inputs will end with an EoF (End of File).
- All decimal numbers that are put out by your program should be formatted with 2 decimal figures. For example 1.2999 needs to be rounded to 1.30.
- For all computation within your programs, please use double precision.
- For each task, you will find example input and output data. You can use this data to test your programs before submission.
- At the end of each output-line, a linebreak is expected ("\n").
- The CSV – formatted inputs use "," as a field-delimiter and "\n" as a line-delimiter.
- Although it is recommended to start with task one and then go through the challenges, you are free to solve the problems in whatever order suites you best.
- For each task there is a specified maximum execution time, so runtime performance should be taken into consideration when submitting your solutions.
- C99 is not supported.

Problem 1: Hello world!

Description

You are to write the most basic program; it should just output “Hello world!" with the number of exclamation marks “!” being defined by the input. For each line of input, there should be one line of output.

Sample input/output

Sample input and output for this problem:

Input
1
2
3
0
10

Output
Hello world!
Hello world!!
Hello world!!!
Hello world
Hello world!!!!!!!!!!

Problem 2: Get Stock quote

Description

You are to write a program that parses csv formatted input data and returns the value range for a specified stock symbol within a given minute. This minute will be specified in the first line of the input data in the format HH:MMAM . The data schema of the csv formatted input is as follows:

<SYMBOL>, <TIMESTAMP>,<OPEN>,<HIGH>,<LOW>,<CLOSE>,<VOLUME>

The timestamp field is formatted in “MM/DD/YYYY HH:MMAM”, for example “7/1/2013 9:30am” would be the first of July 2013 09:30am.

Keep in mind that your code should be robust enough to accept leading zeros for timestamps (for example 9:31am = 09:31am) as well as ignore case sensitivity for the am/pm part of the timestamp (am=AM=Am=aM).

Sample input/output

Sample input and output for this problem:

Input
09:31am,AAPL
AAPL,7/1/2013 9:31am,396.53,402.90,401.6001,403.65,627681
AMAT,7/1/2013 9:31am,14.92,15.05,15.02,15.08,142587
ADSK,7/1/2013 9:31am,33.94,34.01,33.80,34.10,42399
ADP,7/1/2013 9:31am,68.86,68.85,68.75,69.29,70529

Output
1.30

Problem 3a: Get Stock quote (part one)

Description

You are to write a program that parses csv formatted input data and returns the average closing quote for each stock symbol sorted by symbol alphabetically. The data schema of the csv formatted input is as follows:

<SYMBOL>, <TIMESTAMP>,<OPEN>,<HIGH>,<LOW>,<CLOSE>,<VOLUME>

The timestamp field is formatted in "MM/DD/YYYY HH:MMAM", for example "7/1/2013 9:30am" would be the first of July 2013 09:30am.

Keep in mind that your code should be robust enough to accept leading zeros for timestamps (for example 9:31am = 09:31am) as well as ignore case sensitivity for the am/pm part of the timestamp (am=AM=Am=aM).

Sample input/output

Sample input and output for this problem:

Input
AAPL,7/1/2013 9:31am,396.53,402.90,401.6001,403.65,627681
AMAT,7/1/2013 9:31am,14.92,15.05,15.02,15.08,142587
ADSK,7/1/2013 9:31am,33.94,34.01,33.80,34.10,42399
ADP,7/1/2013 9:31am,68.86,68.85,68.75,69.29,70529
AAPL,7/1/2013 9:32am,405.3,406.01,404.10,405.73,630876

Output
AAPL,404.69
ADP,69.29
ADSK,34.10
AMAT,15.08

Problem 3b: Get Stock quote (part two)

Description

You now need to enhance your solution in order to work with a slightly different input format including noise and fraud data. The data schema stays the same:

<SYMBOL>, <TIMESTAMP>,<OPEN>,<HIGH>,<LOW>,<CLOSE>,<VOLUME>

Every line of the input that does not match this schema should be ignored!

The timestamp field is formatted in “MM/DD/YYYY HH:MMAM”, for example “7/1/2013 9:30am” would be the first of July 2013 09:30am.

Keep in mind that your code should be robust enough to accept leading zeros for timestamps (for example 9:31am = 09:31am) as well as ignore case senility for the am/pm part of the timestamp (am=AM=Am=aM).

Changes in Logic

For the computation of the average you now need to make sure that each minute in the dataset is only taken into account once. If there are multiple lines for the same minute and the same symbol, only the latest of those should be considered for the average. (see input for an example)

Sample input/output

Sample input and output for this problem:

Input
AAPL,7/1/2013 9:31am,396.53,402.90,401.6001,403.65,627681
AMAT,7/1/2013 9:31am,14.92,15.05,15.02,15.08,142587
ADSK,7/1/2013 9:31am,33.94,34.01,33.80,34.10,42399
ADP,7/1/2013 9:31am,68.86,68.85,68.75,69.29,70529
AAPL,7/1/2013 9:32am,401.3,410.01,399.28,403.33,630876
AAPL,7/1/2013 9:32am,405.3,406.01,404.10,405.73,630876

Output
AAPL,404.69
ADP,69.29
ADSK,34.10
AMAT,15.08

Problem 4: Form groups of candidates

Description

You are to write a program that calculates the number of possible combinations to form groups of candidates regarding certain restrictions:

- The summed skill-level of all candidates in the group must be exactly 100
- Each group must contain exactly 6 candidates

The skill levels will be given as numeric inputs. Each input line represents one unique candidate with his/her given skillset.

Sample input/output

Sample input and output for this problem:

Example1

Input
10
10
10
10
10
50

Output
1

Example 2

Input
10
10
10
10
10
50
10

Output
6