C++ Fundamentals – Retake Exam – 17 February 2024

Please submit your source code to all below-described problem in [Judge](#).

1. Cash Joint

You're tasked to create the software for exchange office. Your software must have the following functionalities:

1. Support a number of currencies
2. Support the following commands:

     3.1. Adding currency ballance to an account
     3.2. Subtract currency ballance from an account
     3.3. Exchange currency ballance between accounts
     3.4. Print all balances
     3.5. Quit

Input

The **first part** of the input is your data initialization. You will currency balance initialization commands, each on a separate row. The commands will have the following format:

CURRENCY_TICKER CURRENCY_AMOUNT

Where:

- **CURRENCY_TICKER** is a **three-letter abbreviation** (e.g., the "ticker") of the currency, for example "**USD**", "**BGN**", "**EUR**" or "**GBP**"

- **CURRENCY_AMOUNT** is a **double** value, which instructs the program to initialize the respective "currency drawer" with this amount.
    - If **CURRENCY_TICKER** is **END**, this means that all initial amounts are loaded, so now follows the second part of the initialization.

The **second part** of the input are your "commands". Each command comes in a separate row. The list of the commands is as follows:

Command	Example	Explanation
Add amount to account	+ USD 20	Adds a specified amount to specified account. In the example, the account with ticker USD receives 20. Bear in mind: the numbers are double, not int, although there might be no . in the examaples.
Subtract amount from an account	- EUR 20.0	The same as above, but subtracts the given amount from the account with the corresponding

Command	Example	Explanation
		ticker. The example subtracts 20 from the account EUR.
Exchange from account to another account	X USD 25.34 EUR 1.05	This command exchanges from the currency (USD) the amount of 25.34 and adds EUR amount 26.607 (25.34 * 1.05) the correct value. Please note: we **do not do** any rounding to the second digit!
Prints out all accounts with their corresponding amounts.	P	Prints all accounts in the order of their initialization. Each account and its amount are on a separate row and must have the following format: CURRENCY_TICKER : CURRENCY_AMOUNT. For example: USD: 26.61. Please note: we **do round** to the second digit.
Exits the program	E	The program exits. This is your the command in your sequence.

Limitations:

1. You will always get correct input and output.
2. In the operations part, you'll never get a currency, which was not initialized
3. It's OK the amounts to become negative, do not add verification if you have enough money in the source currency when you exchange or subtract.

Example 1

Input	Output	Explanation
USD 1000.0 EUR 500.5 GBP 300 END + USD 200 - EUR 150.5 X GBP 100 EUR 1.2 P END	USD: 1200.00 EUR: 470.00 GBP: 200.00	Three currencies are initialized: USD (1000), EUR (500.50) and GBP (300). The amounts are: EUR: 500.50 GBP: 300.00 USD: 1000.00 Then we add USD 200 with the '+' command, and subtract EUR 150.50 with the '-' command. The amounts at this moment are: EUR: 350.00 GBP: 300.00 USD: 1200.00

		Then we exchange GBP 100 to EUR using rate of 1.2: GBP amount gets decreased with 100, and EUR amount gets increased by 120. Then we print the final amounts, keeping the same order as they were specified in the beginning, during the initialization.

Example 2

Input	Output
JPY 10000 EUR 200 USD 150.25 BGN 500 END + JPY 500 - USD 50.25 + BGN 200 X EUR 100 JPY 130.5 X BGN 200 EUR 0.5114147778 P END	JPY: 23550.00 EUR: 202.28 USD: 100.00 BGN: 500.00

Example 3

Input	Output
USD 1000.50 EUR 750.25 JPY 50000 GBP 320.75 END + USD 150.00 - EUR 200.50 + JPY 3000 - GBP 20.75 X USD 250.25 JPY 110.5 + EUR 100 X GBP 100.00 USD 1.37 P - JPY 1500 + GBP 50.00 P END	USD: 1037.25 EUR: 649.75 JPY: 80652.62 GBP: 200.00 USD: 1037.25 EUR: 649.75 JPY: 79152.62 GBP: 250.00