



## Programming Exercise – FX Calculator

Your task is to create a console-based currency converter application.

### Requirements

The application allows a user to convert an amount in a specific currency to the equivalent amount in another currency.

Your calculator should allow a user to enter an amount in any of the known currencies, and provide the equivalent amount in another currency.

Your calculator should parse console input like "<ccy1> <amount1> in <ccy2>", and provide a suitable response.

For example:

```
%> AUD 100.00 in USD
AUD 100.00 = USD 83.71

%> AUD 100.00 in AUD
AUD 100.00 = AUD 100.00

%> AUD 100.00 in DKK
AUD 100.00 = DKK 505.76

%> JPY 100 in USD
JPY 100 = USD 0.83
```

If the rate cannot be calculated, the program should alert the user:

```
%> KRW 1000.00 in FJD
Unable to find rate for KRW/FJD
```

When displaying amounts back to the user, they should be formatted with the precision given below:

AUD=2 decimal places



CAD=2 decimal places  
CNY=2 decimal places  
CZK=2 decimal places  
DKK=2 decimal places  
EUR=2 decimal places  
GBP=2 decimal places  
JPY=0 decimal places  
NOK=2 decimal places  
NZD=2 decimal places  
USD=2 decimal places

The list of currency pairs and rates that you are given is:

AUDUSD=0.8371  
CADUSD=0.8711  
USDCNY=6.1715  
EURUSD=1.2315  
GBPUSD=1.5683  
NZDUSD=0.7750  
USDJPY=119.95  
EURCZK=27.6028  
EURDKK=7.4405  
EURNOK=8.6651

In order to perform the required calculations, the program needs the amount of decimal places, currency pairs and rates above, and also a table to show how to derive non-quoted/crossed rates.

Currency pairs are specified as <<BASE>> <<TERMS>> (sometimes written as BASE/TERMS - eg. AUD/USD).

Rates are always specified as "1 BASE = x TERMS", where x is the rate.

For example "AUDUSD=0.8371" means that 1 Australian Dollar is equivalent to 0.8371 United States Dollars.

The rates specified above are directly quoted, but they can also be used to create non quoted rates.

For example, in the simplest case, the AUDUSD rate can be used to derive USDAUD by simply inverting the rate. "AUDUSD=0.8371" (1 AUD = 0.8371 USD) is the same as saying "USDAUD=1.1946" (1 USD = 1.1946 AUD).

A slightly more complicated example involves deriving a particular rate by "crossing" via another currency.



For example: you are asked for the AUDJPY rate - but there is no direct feed for this rate. Instead, you can "cross via USD" and use the AUDUSD and USDJPY rates. This works by first converting AUD to USD, and then USD to JPY.

For example

$$\begin{array}{l}
 1 \text{ AUD} = 0.8371 \text{ USD} \\
 0.8371 \text{ USD} = 100.41 \text{ JPY} \\
 \hline
 1 \text{ AUD} = 100.41 \text{ JPY}
 \end{array}$$

In order to help with the conversion, the following "cross-via" matrix may help. "Base" currencies are down the left, "terms" currencies across the top.

| /   | AUD | CAD | CNY | CZK | DKK | EUR | GBP | JPY | NOK | NZD | USD |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| AUD | 1:1 | USD | USD | USD | USD | USD | USD | USD | USD | USD | D   |
| CAD | USD | 1:1 | USD | USD | USD | USD | USD | USD | USD | USD | D   |
| CNY | USD | USD | 1:1 | USD | USD | USD | USD | USD | USD | USD | D   |
| CZK | USD | USD | USD | 1:1 | EUR | Inv | USD | USD | EUR | USD | EUR |
| DKK | USD | USD | USD | EUR | 1:1 | Inv | USD | USD | EUR | USD | EUR |
| EUR | USD | USD | USD | D   | D   | 1:1 | USD | USD | D   | USD | D   |
| GBP | USD | USD | USD | USD | USD | USD | 1:1 | USD | USD | USD | D   |
| JPY | USD | USD | USD | USD | USD | USD | USD | 1:1 | USD | USD | Inv |
| NOK | USD | USD | USD | EUR | EUR | Inv | USD | USD | 1:1 | USD | EUR |
| NZD | USD | USD | USD | USD | USD | USD | USD | USD | USD | 1:1 | D   |
| USD | Inv | Inv | Inv | EUR | EUR | Inv | Inv | D   | EUR | Inv | 1:1 |

D = direct feed - eg. the currency pair can be looked up directly.

Inv = inverted - eg. the currency pair can be looked up if base and terms are flipped (and rate = 1/rate)

1:1 = unity - the rate is always 1.

CCY = in order to calculate this rate, you need to cross via this currency.

You can choose to represent the above table however you choose. You may, for example, be able to use the inherent symmetry of the table to derive a more compact representation.

As an example of using the table above, consider that you need to calculate an AUD/JPY rate:

- first, look up AUD/JPY in the above table.
- the resulting entry shows that you need to cross via USD
- next look up AUD/USD - the rate for this is fed direct
- now look up USD/JPY - also fed direct
- fetch the rates for AUD/USD and USD/JPY and generate the cross rate from there.



Similarly, to calculate NOK/USD

- look up NOK/USD in the table
- the resulting entry says to cross via EUR
- look up EUR/USD and EUR/NOK, and cross the two to find the NOK/USD rate.

Please use the above rates, and the above cross-via table to build your solution.

### **Deliverables**

- The solution submitted should include structure, source code, configuration and any tests or test code you deem necessary - no need to package class files.
- Solve the problem in Java, C# or in a specific language that you may have been directed to use.
- Solve the problem as though it were "production level" code.
- It is not required to provide any graphical interface.

In order to get around firewall issues we recommend the solution be packaged as a password protected zip file.