```
import numpy as np
import cvxpy as cp


# Exchange rate data.
tickers = ["USD", "EUR", "GBP", "CAD", "JPY", "CNY", "RUB", "MXN", "INR", "BRL"]

# Number of currecies
n = len(tickers)

F = np.zeros((n, n))
# USD
data = ([1.0, 0.87, 0.76, 1.31, 108.90, 6.72, 65.45, 19.11, 71.13, 3.69],
# EUR
[1.0, 0.88, 1.51, 125.15, 7.72, 75.23, 21.96, 81.85, 4.24],
# GBP
[1.0, 1.72, 142.94, 8.82, 85.90, 25.08, 93.50, 4.84],
# CAD
[1.0, 82.93, 5.11, 49.82, 14.54, 54.23, 2.81],
# JPY
[1.0, 0.062, 0.60, 0.18, 0.65, 0.034],
# CNY
[1.0, 9.74, 2.85, 10.61, 0.55],
# RUB
[1.0, 0.29, 1.09, 0.056],
# MXN
[1.0, 3.73, 0.19],
# INR
[1.0, 0.052],
# BRL
[1.0])
for i in range(n):
    F[i,i:] = data[i]
for j in range(n):
    for i in range(j+1,n):
        F[i,j] = 1.035/F[j,i]

# Initial and final portfolios.
c_req = np.arange(1,n+1)
c_req = 1e4*c_req/c_req.sum()
c_init = c_req[::-1]


# Matrix of currencies exchange where X(i,j) denote exchange of currency j for currency i
X = cp.Variable((n, n))



# c_final is the vector containing final number of units of each currency where
# (X.T @ np.ones) is a column vector where ith entry denote total exchange of currency i to o
# (X/F @ np.ones(n)) is a culumn vector where ith entry denote total exchange of other curren
c final = c init - (X.T @ np.ones(n)) + (X/F @ np.ones(n))
```

```
# First three constraints are direct whereas fourth constraint
# (X.T @ np.ones(n)) <= c_init  denote that our total exchange of currency cannot exceed our
constraints = [c_final >= c_req, X >= 0, cp.diag(X) == 0, (X.T @ np.ones(n)) <= c_init]


# Our objective is to minimize the currency exchange cost (given in terms of USD)
objective = cp.Minimize((c_init - c_final) @ np.sqrt(F[:,0]/F[0,:]))


prob = cp.Problem(objective, constraints)
optimal_cost = prob.solve()
print("\n\nAnswers: ")
print(" Optimal cost: ", optimal_cost)
print("\n Optimal final units of each currency: ", c_final.value)
print("\n Optimal currency exchanges is given by matrix:\n", X.value)
```

```
Answers:
 Optimal cost:  7.720059340057957

 Optimal final units of each currency:  [1257.49577907 1636.36363639 1771.6701903   727
 1090.90909094 1272.72727321 1454.54545464 1636.3636367  1818.1818182 ]

 Optimal currency exchanges is given by matrix:
 [[6.23500805e-14 4.26378962e-09 3.69571711e-09 5.64615472e-09
  3.28710650e-07 1.61321086e-08 1.52124644e-07 4.45680500e-08
  1.28905492e-07 5.88965811e-09]
 [5.05940761e-09 6.23513554e-14 8.66968379e-09 2.18217089e-08
  1.25513932e-06 3.31083355e-08 3.09318779e-07 8.83503561e-08
  1.95695253e-07 8.91695364e-09]
 [5.06144792e-09 2.78991627e-09 6.23537860e-14 5.45454545e+02
  2.19926086e-06 3.75451490e-08 3.67347691e-07 1.01892604e-07
  2.11540956e-07 9.74007895e-09]
 [5.55175378e-09 2.51244215e-09 1.88832907e-09 6.23477978e-14
  6.78131186e-06 4.93100738e-08 4.50037900e-07 1.25444863e-07
  2.37110316e-07 1.01946294e-08]
 [1.01153910e-08 2.71050628e-09 2.17947362e-09 3.48820899e-09
  6.23430103e-14 4.06564748e-08 8.83995727e-07 4.38317197e-08
  9.49168016e-07 1.01810185e-08]
 [1.65029401e+01 4.03105383e-09 3.17630883e-09 4.65764454e-09
  4.05623583e-07 6.23420970e-14 7.27272724e+02 6.53362004e-07
  3.74312655e-06 1.61201962e-08]
 [1.17316464e-08 3.04283792e-09 2.40179088e-09 3.77586488e-09
  2.48497737e-07 1.35185805e-08 6.21954452e-14 3.69090908e+02
  3.25939264e-07 1.80371420e-08]
 [1.56860433e+01 4.06057113e-09 3.13684416e-09 4.73007004e-09
  1.81818175e+02 1.70055618e-08 1.27016042e-07 6.23455535e-14
  5.64062708e-06 1.81818182e+02]
 [1.85192287e+01 4.36530077e-09 3.38120485e-09 5.14133496e-09
  2.75898083e-07 1.69385277e-08 1.65479656e-07 4.66467495e-08
```

```
 6.23494362e-14 1.53692384e-08]
[5.09977827e+02 4.12289510e-09 3.12535598e-09 5.02687645e-09
 3.79957363e-07 1.69783405e-08 1.27518074e-07 3.07748862e-08
 1.35518058e-07 6.25190693e-14]]
```