```
import numpy as np
import cvxpy as cp

m = 29
n = 36

P = np.matrix([
 [0.166667,  0.111111,  0.083333,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
 [0.166667,  0.111111,  0.083333,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
 [0.166667,  0.111111,  0.083333,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
 [0.166667,  0.111111,  0.083333,  0.173437,  0.115439,  0.086163,  0.000000,  0.000000,
 [0.166667,  0.111111,  0.083333,  0.173437,  0.115439,  0.086163,  0.000000,  0.000000,
 [0.166667,  0.111111,  0.083333,  0.173437,  0.115439,  0.086163,  0.000000,  0.000000,
 [0.166667,  0.111111,  0.083333,  0.173437,  0.115439,  0.086163,  0.187216,  0.125382,
 [0.000000,  0.111111,  0.083333,  0.173437,  0.115439,  0.086163,  0.187216,  0.125382,
 [0.000000,  0.111111,  0.083333,  0.173437,  0.115439,  0.086163,  0.187216,  0.125382,
 [0.000000,  0.111111,  0.083333,  0.173437,  0.115439,  0.086163,  0.187216,  0.125382,
 [0.000000,  0.000000,  0.083333,  0.000000,  0.115439,  0.086163,  0.187216,  0.125382,
 [0.000000,  0.000000,  0.083333,  0.000000,  0.115439,  0.086163,  0.187216,  0.125382,
 [0.000000,  0.000000,  0.083333,  0.000000,  0.115439,  0.086163,  0.187216,  0.125382,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.086163,  0.000000,  0.125382,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.086163,  0.000000,  0.125382,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.086163,  0.000000,  0.125382,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
 [0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,  0.000000,
])

e = np.matrix([
  0.000000,
  1.000000,
  2.000000,
  3.000000,
  4.000000,
  5.000000,
  6.000000,
```

```
    7.000000,
    8.000000,
    16.000000,
    16.600000,
    17.200000,
    17.800000,
    18.400000,
    19.000000,
    19.600000,
    20.200000,
    18.000000,
    17.000000,
    16.000000,
    15.000000,
    14.000000,
    13.000000,
    12.000000,
    11.000000,
    10.000000,
    9.000000,
    8.000000,
    7.000000,
    6.000000,
    5.000000,
    4.000000,
    3.000000,
    2.000000,
    1.000000,
    0.000000,
]).T

rho = 0.002500

x = cp.Variable((m,1))
b = cp.Variable((n,1))
w = cp.Variable((n,1))

def totalCost(x, b):
  return b[0] + np.ones(m).T@x

objective = cp.Minimize(totalCost(x, b))
constraints = [x >= 0, b >= 0, w + P@x >= e]

for i in range(n-1) :
  constraints.append(b[i+1] == (1+rho)*b[i] - w[i])

prob = cp.Problem(objective, constraints)
prob.solve()

print(f"Expenses Matrix: {x.value}\n")
print(f"Bank balance matrix: {b.value} \n")
print(f"Withdrawal amount Matrix: {w.value} \n")

print("Final expense cost: ", totalCost(x, b).value, "USD")
```

```
[3.73064878e+00]
```

```
 [1.41059001e-10]
 [1.27383514e+00]
 [1.38336945e-01]
 [9.10867920e-10]
 [6.22430364e+00]
 [2.61666137e+00]
 [5.71528268e-10]
 [1.67419623e+00]
 [3.38767401e-01]
 [1.89583710e-10]
 [3.14863491e+00]
 [1.94215498e+00]
 [1.73265886e+00]
 [2.52263899e+00]
 [1.01719482e+00]
 [5.07987036e-01]
 [9.97506235e-01]
 [2.42618152e-10]]

Withdrawal amount Matrix: [[-9.98335416e-01]
 [ 1.66458436e-03]
 [ 1.00166458e+00]
 [-2.82383551e+00]
 [-1.82383551e+00]
 [-8.23835505e-01]
 [ 1.76164493e-01]
 [ 2.17449991e+00]
 [ 3.17449991e+00]
 [-3.81198341e+00]
 [ 1.61351668e+00]
 [ 2.21351668e+00]
 [-5.99001251e-01]
 [ 9.98750928e-04]
 [ 6.00998751e-01]
 [-9.64650791e+00]
 [ 5.93997540e+00]
 [ 3.73997540e+00]
 [-1.27383514e+00]
 [ 1.13868279e+00]
 [ 1.38682787e-01]
 [-6.22430364e+00]
 [ 3.62320303e+00]
 [ 2.62320303e+00]
 [-1.67419623e+00]
 [ 1.33961432e+00]
 [ 3.39614319e-01]
 [-3.14863491e+00]
 [ 1.21435151e+00]
 [ 2.14351514e-01]
 [-7.85648486e-01]
 [ 1.51175077e+00]
 [ 5.11750769e-01]
 [-4.88249231e-01]
 [ 1.00000000e+00]
 [ 3.60192610e+00]]

Final expense cost:  [177.51278715] USD
```

✓ 0s completed at 6:29 PM