```
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, w>=0]

  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")
```

```
[5.16968746e+00]
```

```
 [1.98261199e+00]
 [1.98756852e+00]
 [9.92537455e-01]
 [9.95018798e-01]
 [9.97506346e-01]
 [1.36441216e-07]
 [1.36516141e-07]
 [1.37754333e-07]
 [9.64305516e-08]
 [9.70636150e-08]
 [9.75663921e-08]
 [8.24787733e-08]
 [8.21515428e-08]
 [8.14788428e-08]
 [7.10169425e-08]
 [6.90637342e-08]
 [6.63915121e-08]
 [3.75326450e-09]]

Withdrawal amount Matrix: [[ 1.89242935e-10]
 [ 9.99999988e-01]
 [ 1.99999999e+00]
 [ 2.74701857e-09]
 [ 9.99999998e-01]
 [ 2.00000000e+00]
 [ 1.00818556e-08]
 [ 1.00000001e+00]
 [ 2.00000001e+00]
 [ 6.36286591e-09]
 [ 3.59999999e+00]
 [ 4.19999999e+00]
 [-9.22693034e-10]
 [ 3.59999963e+00]
 [ 4.19999963e+00]
 [-1.16795307e-09]
 [ 5.39999969e+00]
 [ 3.19999969e+00]
 [-1.14928623e-09]
 [ 9.99999990e-01]
 [ 1.00690906e-09]
 [-1.18848207e-09]
 [ 9.99999975e-01]
 [ 2.66007618e-10]
 [-8.97066359e-10]
 [ 4.16680264e-08]
 [-3.92116776e-10]
 [-2.60244796e-10]
 [ 1.53314319e-08]
 [ 5.33354064e-10]
 [ 8.78030133e-10]
 [ 1.06655790e-08]
 [ 2.13074481e-09]
 [ 2.84487527e-09]
 [ 6.28042079e-08]
 [ 6.18978532e+00]]

Final expense cost:  [197.91922193] USD
```

0s completed at 7:36 PM