Content

OR-541

COURSE PROJECT

Chanyoung Park(G01408089),Saketh Kallepalli(G01425732)

Current and optimal policy

The report outlines the recommendations for Super Chip, a Virginia-based computer chip manufacturer, regarding their production and distribution operations for the upcoming fiscal year. The company has five manufacturing facilities with varying production capacity levels, and each facility has different equipment and setup costs that impact the cost of manufacturing each of the 30 types of computer chips. Additionally, there are variations in shipping costs for distributing the computer chips to 23 sales regions across the US. The report evaluates an alternative production policy that could potentially reduce costs for Super Chip.

With its current production policy, Super Chip follows a production policy where each facility produces each of the 30 types of chips at levels that are proportional to the facility's total portion of production capacity. This means that if a facility has y% of the total production capacity across all facilities, then it currently produces y% of every chip's total demand.

After analyzing the data provided by Super Chip, it is recommended that the company should adopt a new production policy where the share of each facility for each chip is different from the current policy. The new policy is based on the optimization model that Super Chip can use to minimize their total costs. The optimization model calculates the minimum cost of producing and shipping the computer chips to each sales region by assigning the production of each chip to the facility that has the lowest cost of production for that specific chip.

The formulation of the optimization model is started by defining the decision variables. If we denote with x_{ijk} the number of chips produced at facility i (i = Alexandria(1), Richmond(2), Norfolk(3), Roanoke(4), Charlottesville(5)), for sales region j (j = 1,..23) and type of chip k (k = 1,..30), we are able to write the following constraints:

$$1)\ \ \sum_{j=1}^{23} \sum_{k}^{30} x_{ijk} \leq C_i$$

Here, the total number of chips produced at facility i for all sale regions and types needs to be lower or equal to the production capacity of the very same facility (C_i).

$$2)\ \ \sum_{i=1}^{5} x_{ijk} = D_{jk}$$

Here, the total number of chips produced at all facilities for a sale region j and type k needs to be equal to the demand of the very same region and type of the chip (D_{jk}).

$$3)\ \ x_{ijk} \geq 0$$

In the end, the production needs to be larger or equal to zero. Let's also define the objective function as:

$$f = \vec{c} * \vec{x}$$

The objective function is defined in an aggregated way, combining the cost for production and distribution in one element for each facility and sales region.

The results obtained from the optimization model show that the recommended policy can lead to cost savings for Super Chip. The cost of the current policy is $ 5.6133+07 while the cost after optimization is $4.9083e+07. This means that Super Chip could potentially save $7.05+06 in costs by adopting the recommended policy.

Investment Policy

Based on the recommended policy, Super Chip should invest in expanding the production capacity at the facility that has the lowest cost of production for the chips that would be produced with the additional equipment. According to the optimization model, the facility that should obtain the additional equipment is the one in Roanoke. This facility has the lowest cost associated with the recommended policy, with a total cost of $9.172e+06.

Expanding the production capacity at the Roanoke facility would likely decrease the cost of production due to the acquisition of more efficient equipment. However, it is important to note that the cost of distribution would increase due to the higher number of manufactured chips. Therefore, the total impact on costs would depend on the magnitude of the cost reductions in production and the cost increases in distribution. It is important to note that the total impact on costs would depend on the specific details of the expansion and how it affects production and distribution costs.

Increase in demand

The third strategic-level question was whether Super Chip had sufficient capacity to handle the estimated increase in demand of 10% across all sales regions for the next year. This new constraint thus can be formulated as (compared to the original optimization formulation).

$$\sum_{i=1}^{5} x_{ijk} = 1.10 * D_{jk}$$

After reformulating the optimization problem to increase the demand by 10%, the results showed that Super Chip had enough capacity to adapt to this new demand plan. The associated costs for filling new demand were $ 4.9410e+06.

Decrease in Production Cost

It is important to note that the implementation of this new manufacturing technology will have a significant impact on the overall production and distribution costs for Super Chip.

While the production costs will decrease by 15%, there will be additional costs associated with the implementation of the new technology such as training and equipment upgrades. Furthermore, the distribution costs may also be impacted due to changes in production output and shipping logistics. Therefore, it is important for Super Chip to conduct a thorough cost-benefit analysis before making a decision on the implementation of this new technology in any of its facilities. We need to formulate five different optimization problems (each associated with a 15% decrease in production cost at the respective facility). The constraints of the optimization problem are not the subject of change, so only the objective function is changed:

1) $c_{1jkpost} = 0.85 * c_{1jkpre}, c_{ijkpost} = c_{ijkpre}, i = [2,3,4,5]$
2) $c_{2jkpost} = 0.85 * c_{2jkpre}, c_{ijkpost} = c_{ijkpre}, i = [1,3,4,5]$
3) $c_{3jkpost} = 0.85 * c_{3jkpre}, c_{ijkpost} = c_{ijkpre}, i = [1,2,4,5]$
4) $c_{4jkpost} = 0.85 * c_{4jkpre}, c_{ijkpost} = c_{ijkpre}, i = [1,2,3,5]$
5) $c_{5jkpost} = 0.85 * c_{5jkpre}, c_{ijkpost} = c_{ijkpre}, i = [1,2,3,5]$

Based on the results of the optimization, it is recommended that Super Chip should evaluate the new manufacturing technology in the Alexandria facility. This is because implementing the new technology at this facility results in the lowest overall cost of production. The estimated 15% reduction in production costs for all of the chips translates to significant cost savings for Super Chip, which could lead to increased profits and competitiveness in the market. However, it is important to note that there may be additional costs associated with implementing the new technology, such as training and equipment costs. These factors should be taken into consideration when making the final decision. The total cost associated with this change at Alexandria is $4.6682e+07.

In [1]:
```
import numpy as np
import gurobipy as gp
import pandas as pd
```

In [2]:
```
# Read the data from Excel files
data1 = np.array(pd.read_excel("SuperChipData.xlsx",sheet_name=0))
data2 = np.array(pd.read_excel("SuperChipData.xlsx",sheet_name=1))
data3 = np.array(pd.read_excel("SuperChipData.xlsx",sheet_name=2))
data4 = np.array(pd.read_excel("SuperChipData.xlsx",sheet_name=3))
```

In [3]:
```
# Current policy shares of each facility
share1 = data1[0,1]/sum(data1[:,1]);
share2 = data1[1,1]/sum(data1[:,1]);
share3 = data1[2,1]/sum(data1[:,1]);
share4 = data1[3,1]/sum(data1[:,1]);
share5 = data1[4,1]/sum(data1[:,1]);
```

In [4]:
```
# Define the problem data
n = 30*23
m = 5*len(data2)
```

In [5]:
```
# Left-hand side inequalities
Aineq = np.concatenate((np.hstack((np.ones((1,n)), np.zeros((1,4*n)))),
                        np.hstack((np.zeros((1,n)), np.ones((1,n)), np.zeros((1
                        np.hstack((np.zeros((1,2*n)), np.ones((1,n)), np.zeros(
                        np.hstack((np.zeros((1,3*n)), np.ones((1,n)), np.zeros(
                        np.hstack((np.zeros((1,4*n)), np.ones((1,n)))),
                        -np.eye(m)), axis=0)
```

In [6]:
```
# Right-hand side inequalities
bineq = np.concatenate((data1[:,1]*1000, np.zeros((m,))), axis=0)
```

In [7]:
```
# Left-hand side equalities
Aeq = np.zeros((len(data2), m))
```

In [8]:
```
for i in range(len(data2)):
    for j in range(5):
        Aeq[i, i+j*n] = 1
```

In [9]:
```
# Right-hand side equalities
beq = data2[:,2]*1000
```

In [10]:
```
# Cost vector
c = np.zeros((m,))

# Iterate through the cost vectors
for k in range(1,6):
    for i in range(23):
        for j in range(30):
            c[(k-1)*n + i*30 + j] = c[(k-1)*n + i*30 + j] + data4[(k-1)*30 + j,
```

In [11]:
```
# Cost with current policy
COST1 = 0
COST2 = 0
```

```
COST3 = 0
COST4 = 0
COST5 = 0

# Sum up all of the elements to obtain the cost with current policy
for k in range(1, 6):
    for i in range(1, 24):
        for j in range(1, 31):
            if k == 1:
                COST1 += c[30*23*(k-1) + (i-1)*30 + j-1]*share1*data2[(i-1)*30+
            elif k == 2:
                COST2 += c[30*23*(k-1) + (i-1)*30 + j-1]*share2*data2[(i-1)*30+
            elif k == 3:
                COST3 += c[30*23*(k-1) + (i-1)*30 + j-1]*share3*data2[(i-1)*30+
            elif k == 4:
                COST4 += c[30*23*(k-1) + (i-1)*30 + j-1]*share4*data2[(i-1)*30+
            elif k == 5:
                COST5 += c[30*23*(k-1) + (i-1)*30 + j-1]*share5*data2[(i-1)*30+
```

In [12]:

```
for i in range(5):
    if COST1 == min(COST1,COST2,COST3,COST4,COST5):
        minindex = i
        print("Facility with lowest cost with current policy is Alexandria, wit
    elif COST2 == min(COST1,COST2,COST3,COST4,COST5):
        minindex = i
        print("Facility with lowest cost with current policy is Richmond, with
    elif COST3 == min(COST1,COST2,COST3,COST4,COST5):
        minindex = i
        print("Facility with lowest cost with current policy is Norfolk, with t
    elif COST4 == min(COST1,COST2,COST3,COST4,COST5):
        minindex = i
        print("Facility with lowest cost with current policy is Roanoke, with t
    elif COST5 == min(COST1,COST2,COST3,COST4,COST5):
        minindex = i
        print("Facility with lowest cost with current policy is Charolottesvill
```

```
Facility with lowest cost with current policy is Roanoke, with the cost of:917
2090.517030565
Facility with lowest cost with current policy is Roanoke, with the cost of:917
2090.517030565
Facility with lowest cost with current policy is Roanoke, with the cost of:917
2090.517030565
Facility with lowest cost with current policy is Roanoke, with the cost of:917
2090.517030565
Facility with lowest cost with current policy is Roanoke, with the cost of:917
2090.517030565
```

In [13]:

```
# Total cost of all facilities
COST = COST1 + COST2 + COST3 + COST4 + COST5

print("Cost with current policy is:"+str(COST))
```

```
Cost with current policy is:56133859.957641914
```

In [14]:

```
# Create a Gurobi model
model = gp.Model()
```

```
Set parameter Username
Academic license - for non-commercial use only - expires 2024-01-18
```

In [15]:
```
# Define the decision variables
x = model.addMVar(shape=m, lb=-1e30, ub=gp.GRB.INFINITY, name="x")
```

In [16]:
```
# Set the objective function
model.setObjective(c @ x, sense=gp.GRB.MINIMIZE)
```

In [17]:
```
# Add the constraints
model.addConstr(Aeq @ x == beq, name="eq")
model.addConstr(Aineq @ x <= bineq, name="ineq")
```

Out[17]:
```
<MConstr (3455,) *awaiting model update*>
```

In [18]:
```
# Solve the model
model.optimize()
```

```
Gurobi Optimizer version 10.0.1 build v10.0.1rc0 (win64)

CPU model: 12th Gen Intel(R) Core(TM) i5-1240P, instruction set [SSE2|AVX|AVX
2]
Thread count: 12 physical cores, 16 logical processors, using up to 16 threads

Optimize a model with 4145 rows, 3450 columns and 10350 nonzeros
Model fingerprint: 0xf6941b08
Coefficient statistics:
  Matrix range     [1e+00, 1e+00]
  Objective range  [4e+01, 7e+01]
  Bounds range     [0e+00, 0e+00]
  RHS range        [1e+01, 3e+05]
Presolve removed 3451 rows and 5 columns
Presolve time: 0.02s
Presolved: 694 rows, 3445 columns, 6890 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    4.9059636e+07   7.737500e+03   0.000000e+00      0s
      45    4.9083430e+07   0.000000e+00   0.000000e+00      0s

Solved in 45 iterations and 0.03 seconds (0.00 work units)
Optimal objective  4.908343040e+07
```

In [19]:
```
# Get the optimal solution
x_opt1 = model.getAttr('x')
```

In [20]:
```
print("Optimal solution:")
print(x_opt1)
```

```
Optimal solution:
[0.0, 0.0, 0.0, 0.0, 0.0, 460.0, 0.0, 1750.0, 2920.0, 0.0, 2340.0, 1500.0, 131
0.0, 0.0, 730.0, 0.0, 0.0, 0.0, 280.0, 0.0, 0.0, 0.0, 0.0, 1630.0, 0.0, 0.0, 1
580.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2940.0, 0.0, 1210.0, 1230.0, 0.
0, 2040.0, 2520.0, 1330.0, 0.0, 140.0, 0.0, 0.0, 0.0, 1700.0, 2970.0, 0.0, 0.
0, 0.0, 260.0, 0.0, 0.0, 1810.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 890.
0, 0.0, 2840.0, 730.0, 0.0, 180.0, 1360.0, 1620.0, 0.0, 1220.0, 0.0, 0.0, 0.0,
1040.0, 0.0, 0.0, 0.0, 0.0, 1270.0, 0.0, 0.0, 440.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 470.0, 1660.0, 0.0, 2300.0, 930.0, 540.0, 0.0, 1450.
0, 0.0, 0.0, 0.0, 610.0, 0.0, 0.0, 0.0, 0.0, 1470.0, 0.0, 0.0, 1860.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1590.0, 0.0, 1720.0, 2029.9999999999998, 0.0,
490.0, 1660.0, 2580.0, 0.0, 260.0, 0.0, 0.0, 0.0, 2040.0, 0.0, 0.0, 0.0, 0.0,
760.0, 0.0, 0.0, 940.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1310.0, 0.0, 2
50.0, 780.0, 0.0, 380.0, 520.0, 140.0, 0.0, 2400.0, 0.0, 0.0, 0.0, 1680.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2420.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 930.0, 0.0, 2870.0, 1910.0, 0.0, 2650.0, 450.0, 1190.0, 0.0, 1060.0, 0.0,
0.0, 0.0, 140.0, 1390.0, 0.0, 0.0, 0.0, 2800.0, 0.0, 0.0, 2160.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2760.0, 1710.0, 0.0, 2810.0, 290.0, 295
0.0, 0.0, 1740.0, 0.0, 0.0, 0.0, 800.0, 0.0, 0.0, 0.0, 0.0, 2850.0, 0.0, 0.0,
1690.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1890.0, 0.0, 70.0, 2540.0, 0.
0, 2020.0, 660.0, 1380.0, 0.0, 1040.0, 0.0, 0.0, 0.0, 2060.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 1630.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 940.0, 0.
0, 1350.0, 1940.0, 0.0, 2490.0, 1610.0, 1150.0, 0.0, 2750.0, 0.0, 0.0, 0.0, 26
70.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1870.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 790.0, 0.0, 2630.0, 2520.0, 0.0, 570.0, 70.0, 890.0, 0.0, 1720.0, 0.
0, 0.0, 0.0, 550.0, 0.0, 0.0, 0.0, 0.0, 670.0, 0.0, 0.0, 2930.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1130.0, 300.0, 0.0, 1530.0, 1910.0, 155
0.0, 0.0, 950.0, 0.0, 0.0, 0.0, 1680.0, 0.0, 0.0, 0.0, 0.0, 1650.0, 0.0, 0.0,
420.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 480.0, 0.0, 2910.0, 1430.0, 0.
0, 2110.0, 1090.0, 1750.0, 0.0, 920.0, 0.0, 0.0, 0.0, 2830.0, 0.0, 0.0, 0.0,
0.0, 190.0, 0.0, 0.0, 2450.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 480.0,
0.0, 2730.0, 1770.0, 0.0, 1600.0, 2390.0, 2080.0, 0.0, 2170.0, 0.0, 0.0, 0.0,
1050.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 110.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 2400.0, 0.0, 290.0, 1800.0, 2980.0, 0.0, 1650.0,
0.0, 0.0, 0.0, 650.0, 0.0, 0.0, 0.0, 0.0, 1520.0, 0.0, 0.0, 2230.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 590.0, 0.0, 360.0, 940.0, 0.0, 620.0, 1410.0, 17
60.0, 0.0, 590.0, 0.0, 0.0, 0.0, 310.0, 0.0, 0.0, 0.0, 0.0, 2850.0, 0.0, 0.0,
1430.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2510.0, 0.0, 0.0, 1480.0, 0.0,
1050.0, 1350.0, 800.0, 0.0, 220.0, 0.0, 0.0, 0.0, 1670.0, 0.0, 0.0, 0.0, 0.0,
1980.0, 0.0, 0.0, 2550.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1690.0, 0.0,
2160.0, 710.0, 0.0, 910.0, 2800.0, 130.0, 0.0, 1020.0, 0.0, 0.0, 0.0, 2710.0,
2270.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1150.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 2870.0, 0.0, 110.0, 2600.0, 0.0, 790.0, 320.0, 1320.0, 0.0, 630.0, 0.
0, 0.0, 0.0, 2040.0, 2250.0, 0.0, 0.0, 0.0, 2750.0, 0.0, 0.0, 1570.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2130.0, 0.0, 0.0, 530.0, 0.0, 1010.0, 20.0, 6
50.0, 0.0, 2410.0, 0.0, 0.0, 0.0, 1260.0, 0.0, 0.0, 0.0, 0.0, 20.0, 0.0, 0.0,
2380.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1580.0, 0.0, 1320.0, 2480.0,
0.0, 1160.0, 1810.0, 1090.0, 0.0, 2550.0, 0.0, 0.0, 0.0, 390.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 2460.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
2000.0, 210.0, 0.0, 2580.0, 2850.0, 1710.0, 0.0, 750.0, 0.0, 0.0, 0.0, 630.0,
0.0, 0.0, 0.0, 0.0, 150.0, 0.0, 0.0, 2740.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 550.0, 0.0, 0.0, 390.0, 0.0, 1350.0, 2040.0, 20.0, 0.0, 2230.0, 0.0,
0.0, 0.0, 1780.0, 0.0, 0.0, 0.0, 0.0, 1730.0, 0.0, 0.0, 2530.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 2630.0, 0.0, 0.0, 0.0, 0.0, 300.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 940.0, 660.0, 0.0, 0.0, 1370.0, 2620.0, 560.0, 0.0, 0.0, 0.0, 0.0, 292
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1060.0, 0.0, 0.0, 0.0, 0.0, 2920.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 1660.0, 2970.0, 0.0, 0.0, 210.0, 410.0, 2720.0, 0.0, 83
0.0, 0.0, 0.0, 2620.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 740.0, 0.0, 0.0, 0.0, 0.
0, 2250.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1610.0, 240.0, 0.0, 0.0, 2360.0, 296
0.0, 260.0, 0.0, 0.0, 0.0, 0.0, 80.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2950.0, 11
```

```
50.0, 0.0, 0.0, 0.0, 1480.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1050.0, 1830.0, 0.
0, 0.0, 2280.0, 1000.0, 1640.0, 0.0, 0.0, 0.0, 0.0, 940.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 800.0, 0.0, 0.0, 0.0, 0.0, 1480.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 257
0.0, 790.0, 0.0, 0.0, 2690.0, 1180.0, 1910.0, 0.0, 0.0, 0.0, 0.0, 2270.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 980.0, 0.0, 0.0, 0.0, 0.0, 810.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 2430.0, 1090.0, 0.0, 0.0, 1190.0, 1310.0, 2870.0, 470.0, 0.0, 0.0,
0.0, 1170.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1250.0, 0.0, 0.0, 0.0, 0.0, 1020.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2920.0, 160.0, 0.0, 0.0, 2180.0, 2430.0, 2790.0,
0.0, 0.0, 0.0, 0.0, 1320.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1990.0, 1070.0, 0.0,
0.0, 0.0, 2330.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 550.0, 1580.0, 0.0, 0.0, 1030.
0, 2060.0, 470.0, 0.0, 630.0, 0.0, 0.0, 1400.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
50.0, 0.0, 0.0, 0.0, 0.0, 1300.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 590.0, 1310.0,
0.0, 0.0, 180.0, 560.0, 1430.0, 2430.0, 310.0, 0.0, 0.0, 1750.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 2720.0, 0.0, 0.0, 0.0, 0.0, 320.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 640.0, 2960.0, 0.0, 0.0, 2480.0, 810.0, 810.0, 2950.0, 0.0, 0.0, 0.0, 20
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2200.0, 0.0, 0.0, 0.0, 0.0, 1360.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 1470.0, 2029.9999999999998, 0.0, 0.0, 500.0, 1910.0, 20
90.0, 0.0, 0.0, 0.0, 0.0, 2950.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 740.0, 2000.0,
0.0, 0.0, 0.0, 150.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2780.0, 2230.0, 0.0, 0.0,
890.0, 2860.0, 2870.0, 0.0, 0.0, 0.0, 0.0, 510.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 2340.0, 0.0, 0.0, 0.0, 0.0, 1070.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1580.0, 2
550.0, 0.0, 0.0, 2330.0, 1030.0, 450.0, 0.0, 0.0, 0.0, 0.0, 520.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 2420.0, 0.0, 0.0, 0.0, 0.0, 2140.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 1290.0, 2029.9999999999998, 0.0, 0.0, 2490.0, 550.0, 600.0, 370.0, 0.
0, 0.0, 0.0, 2790.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2170.0, 2160.0, 0.0, 0.0,
0.0, 750.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1520.0, 1980.0, 0.0, 0.0, 1920.0, 29
00.0, 840.0, 0.0, 1040.0, 0.0, 0.0, 2580.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 940.
0, 0.0, 0.0, 0.0, 0.0, 2640.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 680.0, 1090.0, 0.
0, 0.0, 2990.0, 290.0, 340.0, 0.0, 1860.0, 0.0, 0.0, 1930.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 2780.0, 0.0, 0.0, 0.0, 0.0, 2140.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
2510.0, 1810.0, 0.0, 0.0, 1400.0, 2430.0, 990.0, 0.0, 0.0, 0.0, 0.0, 1730.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1920.0, 0.0, 0.0, 0.0, 0.0, 2610.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 1210.0, 780.0, 0.0, 0.0, 840.0, 2740.0, 1360.0, 720.0, 0.0,
0.0, 0.0, 280.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 580.0, 0.0, 0.0, 0.0, 0.0, 138
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1380.0, 370.0, 0.0, 0.0, 2090.0, 2980.0, 17
90.0, 0.0, 1200.0, 0.0, 0.0, 2170.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 350.0, 0.0,
0.0, 0.0, 0.0, 790.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2970.0, 2760.0, 0.0, 0.0,
1400.0, 1540.0, 2780.0, 0.0, 2830.0, 0.0, 0.0, 220.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 1910.0, 0.0, 0.0, 0.0, 0.0, 500.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1880.0,
960.0, 0.0, 0.0, 280.0, 880.0, 1540.0, 1290.0, 170.0, 0.0, 0.0, 420.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 1470.0, 1690.0, 0.0, 0.0, 0.0, 590.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 620.0, 1980.0, 0.0, 0.0, 1570.0, 1280.0, 2100.0, 0.0, 0.0, 0.0,
0.0, 2210.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1810.0, 0.0, 0.0, 0.0, 0.0, 130.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1340.0, 1860.0, 0.0, 0.0, 2330.0, 1770.0, 230.0,
0.0, 2890.0, 0.0, 0.0, 2550.0, 0.0, 0.0, 0.0, 2170.0, 2820.0, 870.0, 0.0, 0.0,
760.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2670.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 690.0, 700.0, 0.0, 1030.0, 2050.0, 80.0,
0.0, 0.0, 120.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1120.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1420.0, 1830.0, 0.0, 1320.0, 1
510.0, 2920.0, 0.0, 0.0, 2890.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2990.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1400.0, 90.0,
0.0, 2930.0, 190.0, 1660.0, 0.0, 0.0, 1710.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 680.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 165
0.0, 1780.0, 0.0, 900.0, 2140.0, 1370.0, 0.0, 0.0, 180.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 2910.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 1610.0, 2190.0, 0.0, 1430.0, 1730.0, 2990.0, 0.0, 0.0, 1390.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 970.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 1520.0, 700.0, 0.0, 2670.0, 1130.0, 550.0, 0.0, 0.0,
2540.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2540.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2720.0, 1010.0, 0.0, 2930.0, 2280.0, 119
```

```
0.0, 0.0, 0.0, 2710.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2720.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 820.0, 410.0, 0.0, 1460.
0, 2940.0, 2700.0, 0.0, 0.0, 930.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 20
80.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2130.0, 220
0.0, 0.0, 1310.0, 670.0, 160.0, 0.0, 0.0, 40.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 2060.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
1670.0, 2910.0, 0.0, 1440.0, 660.0, 430.0, 0.0, 0.0, 430.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 970.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 1660.0, 1820.0, 0.0, 880.0, 330.0, 1880.0, 0.0, 0.0, 430.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1490.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 2670.0, 2820.0, 0.0, 1710.0, 1130.0, 2050.0, 0.0, 0.
0, 600.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 40.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2310.0, 230.0, 0.0, 2920.0, 2690.0, 246
0.0, 0.0, 0.0, 1550.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 820.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2020.0, 2330.0, 0.0, 131
0.0, 2520.0, 860.0, 0.0, 0.0, 2470.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
1010.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2690.0, 89
0.0, 0.0, 1140.0, 2880.0, 1060.0, 0.0, 0.0, 1750.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 1430.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 2490.0, 1850.0, 0.0, 2310.0, 2370.0, 1240.0, 0.0, 0.0, 2100.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 750.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 650.0, 870.0, 0.0, 2280.0, 2780.0, 1630.0, 0.0, 0.0, 1850.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2120.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 90.0, 1270.0, 0.0, 1510.0, 520.0, 100.0, 0.0,
0.0, 1780.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2560.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2920.0, 1840.0, 0.0, 1280.0, 2150.
0, 1680.0, 0.0, 0.0, 2940.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 340.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1640.0, 710.0, 0.0,
2670.0, 2720.0, 2730.0, 0.0, 0.0, 2610.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 1530.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 250.
0, 1550.0, 0.0, 1130.0, 2600.0, 2150.0, 0.0, 0.0, 330.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 2830.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 1520.0, 320.0, 0.0, 2600.0, 1780.0, 1440.0, 0.0, 0.0, 1730.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 810.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 1570.0, 2870.0, 1180.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1580.0, 0.0, 0.0,
0.0, 0.0, 520.0, 660.0, 0.0, 0.0, 0.0, 0.0, 660.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 2000.0, 0.0, 0.0, 0.0, 0.0, 740.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1670.0, 0.0, 0.
0, 0.0, 0.0, 1860.0, 1270.0, 0.0, 0.0, 0.0, 0.0, 90.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1760.0, 0.
0, 0.0, 0.0, 0.0, 210.0, 2250.0, 0.0, 0.0, 0.0, 0.0, 760.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1860.
0, 0.0, 0.0, 0.0, 0.0, 150.0, 1970.0, 0.0, 0.0, 0.0, 0.0, 2460.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
70.0, 0.0, 0.0, 0.0, 0.0, 1780.0, 1350.0, 0.0, 0.0, 0.0, 0.0, 2370.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1210.0, 1670.0, 0.0, 0.0, 0.0, 0.0, 1720.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 2510.0, 0.0, 0.0, 0.0, 0.0, 0.0, 900.0, 0.0, 0.0, 0.0, 0.0, 1470.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 630.0, 0.0, 0.0, 0.0, 0.0, 0.0, 360.0, 0.0, 0.0, 0.0, 0.0, 1370.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 500.0, 0.0, 0.0, 0.0, 0.0, 2670.0, 1640.0, 0.0, 0.0, 0.0, 0.0, 13
00.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 890.0, 0.0, 0.0, 0.0, 0.0, 1350.0, 1280.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 2500.0, 0.0, 0.0, 0.0, 0.0, 1190.0, 660.0, 0.0, 0.0, 0.0,
0.0, 1160.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
```

```
0.0, 0.0, 0.0, 0.0, 0.0, 290.0, 0.0, 0.0, 0.0, 0.0, 1290.0, 1510.0, 0.0, 0.0,
0.0, 0.0, 140.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1300.0, 0.0, 0.0, 0.0, 0.0, 240.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 1270.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 540.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 910.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1710.0, 0.
0, 0.0, 0.0, 0.0, 2760.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 440.0, 0.0, 0.0, 0.0, 0.0, 0.0, 210.0,
0.0, 0.0, 0.0, 0.0, 610.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2200.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 850.0, 0.0, 0.0, 0.0, 0.0, 1220.0,
400.0, 0.0, 0.0, 0.0, 0.0, 1830.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1720.
0, 1170.0, 0.0, 0.0, 0.0, 0.0, 2410.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 2780.0, 0.0, 0.0, 0.0, 0.0, 1470.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1280.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2320.0, 0.0, 0.0, 0.0,
0.0, 0.0, 870.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 910.0, 0.0, 0.0, 0.0,
0.0, 0.0, 2730.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1850.0, 0.0, 0.0, 0.0,
0.0, 1790.0, 1900.0, 0.0, 0.0, 0.0, 0.0, 2930.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
1800.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2080.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 780.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 10.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 2029.9999999999998, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 750.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1030.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1860.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1420.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 650.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 1830.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 1590.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 1160.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 660.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 410.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 90.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 2930.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 2180.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 980.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 2980.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 610.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
2870.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 256
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2370.
```

```
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 1360.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1570.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 130.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2090.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1470.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0]
```

In [21]:
```
# Preallocation
aux = 0
share = np.zeros((5, 30))
```

In [22]:
```
# Cost of the new policy
print('Cost of the new policy is:')
print(np.dot(c, x_opt1))
```

```
Cost of the new policy is:
49083430.4
```

In [23]:
```
print("The potential save if adopting new policy:")
print(COST-np.matmul(np.transpose(c),x_opt1))
```

```
The potential save if adopting new policy:
7050429.557641916
```

In [24]:
```
# Finding the share of each facility for each type of chip
for k in range(1, 6):
    for i in range(1, 31):
        for j in range(i, len(data2)+1, 30):
            share[k-1, i-1] += x_opt1[(k-1)*30*23 + j-1]
            aux += data2[j-1, 2]*1000
        share[k-1, i-1] /= aux
        aux = 0
```

In [25]:
```
# Cost per facility with optimized policy
costperfac = np.zeros(5)
```

In [26]:
```
for i in range(1, 6):
    costperfac[i-1] = np.dot(c[(i-1)*30*23:i*30*23], x_opt1[(i-1)*30*23:i*30*23
```

In [27]:
```
# Solve for 10% increase in production
# Right-hand side equalities
beq = data2[:,2]*1000*1.10
```

In [28]:
```
# Cost vector
c = np.zeros((m,))
```

In [29]:
```
# Iterate through the cost vectors
for k in range(1,6):
    for i in range(23):
        for j in range(30):
            c[(k-1)*n + i*30 + j] = c[(k-1)*n + i*30 + j] + data4[(k-1)*30 + j,
```

In [30]:
```
# Create a Gurobi model
model = gp.Model()
```

In [31]:
```
# Define the decision variables
x = model.addMVar(shape=m, lb=-1e30, ub=gp.GRB.INFINITY, name="x")
```

In [32]:
```
# Set the objective function
model.setObjective(c @ x, sense=gp.GRB.MINIMIZE)
```

In [33]:
```
model.addConstr(Aeq @ x == beq, name="eq")
model.addConstr(Aineq @ x <= bineq, name="ineq")
```

Out[33]:
```
<MConstr (3455,) *awaiting model update*>
```

In [34]:
```
# Solve the model
model.optimize()
```

```
Gurobi Optimizer version 10.0.1 build v10.0.1rc0 (win64)

CPU model: 12th Gen Intel(R) Core(TM) i5-1240P, instruction set [SSE2|AVX|AVX
2]
Thread count: 12 physical cores, 16 logical processors, using up to 16 threads

Optimize a model with 4145 rows, 3450 columns and 10350 nonzeros
Model fingerprint: 0x93ba71f1
Coefficient statistics:
  Matrix range     [1e+00, 1e+00]
  Objective range  [4e+01, 7e+01]
  Bounds range     [0e+00, 0e+00]
  RHS range        [1e+01, 3e+05]
Presolve removed 3451 rows and 5 columns
Presolve time: 0.01s
Presolved: 694 rows, 3445 columns, 6890 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    5.3965600e+07   1.241125e+04   0.000000e+00      0s
      81    5.4024420e+07   0.000000e+00   0.000000e+00      0s

Solved in 81 iterations and 0.02 seconds (0.00 work units)
Optimal objective  5.402442027e+07
```

In [35]:
```
# Get the optimal solution for 10% increase
x_opt2 = model.getAttr('x')
```

In [36]:
```
print("Optimal solution for 10% increase in production:")
print(x_opt2)
```

```
Optimal solution for 10% increase in production:
[0.0, 0.0, 0.0, 0.0, 0.0, 506.00000000000006, 0.0, 1925.0000000000002, 3212.00
00000000005, 0.0, 2574.0, 1650.0000000000002, 1441.0000000000002, 0.0, 803.000
0000000001, 0.0, 0.0, 0.0, 308.0, 0.0, 0.0, 0.0, 0.0, 1793.0000000000002, 0.0,
0.0, 1738.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 3234.00000000
00005, 0.0, 1331.0, 1353.0, 0.0, 2244.0, 2772.0, 1463.0000000000002, 0.0, 154.
0, 0.0, 0.0, 0.0, 1870.0000000000002, 3267.0000000000005, 0.0, 0.0, 0.0, 286.
0, 0.0, 0.0, 1991.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 979.0
000000000001, 0.0, 0.0, 803.0000000000001, 0.0, 198.00000000000003, 1496.00000
00000002, 1782.0000000000002, 0.0, 1342.0, 0.0, 0.0, 0.0, 1144.0, 0.0, 0.0, 0.
0, 0.0, 1397.0, 0.0, 0.0, 484.00000000000006, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 517.0, 1826.0000000000002, 0.0, 2530.0, 1023.0000000000001,
594.0, 0.0, 1595.0000000000002, 0.0, 0.0, 0.0, 671.0, 0.0, 0.0, 0.0, 0.0, 161
7.0000000000002, 0.0, 0.0, 2046.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 1749.0000000000002, 0.0, 1892.0000000000002, 2233.0, 0.0, 539.0, 182
6.0000000000002, 2838.0000000000005, 0.0, 286.0, 0.0, 0.0, 0.0, 2244.0, 0.0,
0.0, 0.0, 0.0, 836.0000000000001, 0.0, 0.0, 1034.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 1441.0000000000002, 0.0, 275.0, 858.0000000000001, 0.0, 418.000
00000000006, 572.0, 154.0, 0.0, 2640.0, 0.0, 0.0, 0.0, 1848.0000000000002, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2662.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 1023.0000000000001, 0.0, 3157.0000000000005, 2101.0, 0.0, 2915.000000000000
5, 495.00000000000006, 1309.0, 0.0, 1166.0, 0.0, 0.0, 0.0, 154.0, 0.0, 0.0, 0.
0, 0.0, 3080.0000000000005, 0.0, 0.0, 2376.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 1177.0, 0.0, 3036.0000000000005, 1881.0000000000002, 0.0, 3091.0000000
000005, 319.0, 3245.0000000000005, 0.0, 1914.0000000000002, 0.0, 0.0, 0.0, 88
0.0000000000001, 0.0, 0.0, 0.0, 0.0, 3135.0000000000005, 0.0, 0.0, 1859.000000
0000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2079.0, 0.0, 0.0, 2794.0, 0.
0, 2222.0, 726.0000000000001, 1518.0000000000002, 0.0, 1144.0, 0.0, 0.0, 0.0,
2266.0, 0.0, 0.0, 0.0, 0.0, 2673.0, 0.0, 0.0, 1793.0000000000002, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 1034.0, 0.0, 0.0, 2134.0, 0.0, 2739.0, 1771.000000
0000002, 1265.0, 0.0, 3025.0000000000005, 0.0, 0.0, 0.0, 2937.0000000000005,
0.0, 0.0, 0.0, 0.0, 3245.0000000000005, 0.0, 0.0, 2057.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 869.0000000000001, 0.0, 2893.0000000000005, 2772.0, 0.0, 6
27.0, 77.0, 979.0000000000001, 0.0, 1892.0000000000002, 0.0, 0.0, 0.0, 605.0,
0.0, 0.0, 0.0, 0.0, 737.0000000000001, 0.0, 0.0, 3223.0000000000005, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2200.0, 0.0, 1243.0, 330.0, 0.0, 1683.0000000000
002, 2101.0, 1705.0000000000002, 0.0, 1045.0, 0.0, 0.0, 0.0, 1848.000000000000
2, 0.0, 0.0, 0.0, 0.0, 1815.0000000000002, 0.0, 0.0, 462.00000000000006, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 528.0, 0.0, 0.0, 1573.0000000000002, 0.0, 2
321.0, 1199.0, 1925.0000000000002, 0.0, 1012.0000000000001, 0.0, 0.0, 0.0, 311
3.0000000000005, 0.0, 0.0, 0.0, 0.0, 209.00000000000003, 0.0, 0.0, 2695.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 528.0, 0.0, 3003.0000000000005, 1947.000
0000000002, 0.0, 1760.0000000000002, 2629.0, 2288.0, 0.0, 2387.0, 0.0, 0.0, 0.
0, 1155.0, 0.0, 0.0, 0.0, 0.0, 407.00000000000006, 0.0, 0.0, 121.0000000000000
1, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 443.99999999999926, 0.0, 0.0, 2640.
0, 0.0, 319.0, 1980.0000000000002, 3278.0000000000005, 0.0, 1815.000000000000
2, 0.0, 0.0, 0.0, 715.0000000000001, 0.0, 0.0, 0.0, 0.0, 1672.0000000000002,
0.0, 0.0, 2453.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 649.0, 0.0, 396.0000
0000000006, 1034.0, 0.0, 682.0, 1551.0000000000002, 1936.0000000000002, 0.0, 6
49.0, 0.0, 0.0, 0.0, 341.0, 0.0, 0.0, 0.0, 0.0, 3135.0000000000005, 0.0, 0.0,
1573.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2761.0, 0.0, 0.0,
1628.0000000000002, 0.0, 1155.0, 1485.0000000000002, 880.0000000000001, 0.0, 2
42.00000000000003, 0.0, 0.0, 0.0, 1837.0000000000002, 0.0, 0.0, 0.0, 0.0, 217
8.0, 0.0, 0.0, 2805.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1859.0000000000
002, 0.0, 0.0, 781.0000000000001, 0.0, 1001.0000000000001, 3080.0000000000005,
143.0, 0.0, 1122.0, 0.0, 0.0, 0.0, 2981.0000000000005, 0.0, 0.0, 0.0, 0.0, 79
2.0000000000001, 0.0, 0.0, 1265.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 315
7.0000000000005, 0.0, 121.00000000000001, 2860.0000000000005, 0.0, 869.0000000
000001, 352.0, 1452.0000000000002, 0.0, 693.0, 0.0, 0.0, 0.0, 2244.0, 0.0, 0.
0, 0.0, 0.0, 3025.0000000000005, 0.0, 0.0, 1727.0000000000002, 0.0, 0.0, 0.0,
```

0.0, 0.0, 0.0, 0.0, 0.0, 2343.0, 0.0, 0.0, 583.0, 0.0, 1111.0, 22.0, 715.00000
00000001, 0.0, 2651.0, 0.0, 0.0, 0.0, 1386.0, 0.0, 0.0, 0.0, 0.0, 22.0, 0.0,
0.0, 2618.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1738.0000000000002, 0.0,
1452.0000000000002, 2728.0, 0.0, 1276.0, 1991.0000000000002, 1199.0, 0.0, 280
5.0, 0.0, 0.0, 0.0, 429.00000000000006, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 270
6.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1859.0000000000002, 0.0, 0.0, 23
1.00000000000003, 0.0, 2838.0000000000005, 3135.0000000000005, 1881.0000000000
002, 0.0, 825.0000000000001, 0.0, 0.0, 0.0, 693.0, 0.0, 0.0, 0.0, 0.0, 165.0,
0.0, 0.0, 3014.0000000000005, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 605.0,
0.0, 0.0, 429.00000000000006, 0.0, 1485.0000000000002, 2244.0, 22.0, 0.0, 245
3.0, 0.0, 0.0, 0.0, 1958.0000000000002, 0.0, 0.0, 0.0, 0.0, 1903.000000000000
2, 0.0, 0.0, 2783.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2893.0000000000005, 0.
0, 0.0, 0.0, 0.0, 330.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1034.0, 726.00000000000
01, 0.0, 0.0, 1507.0000000000002, 2882.0000000000005, 616.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1166.0, 0.0, 0.0, 0.0, 0.0, 3212.0000000
000005, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1826.0000000000002, 3267.0000000000005,
0.0, 0.0, 231.00000000000003, 451.00000000000006, 2992.0000000000005, 0.0, 91
3.0000000000001, 0.0, 0.0, 2882.0000000000005, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 8
14.0000000000001, 0.0, 0.0, 0.0, 0.0, 2475.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 17
71.0000000000002, 264.0, 0.0, 0.0, 2596.0, 3256.0000000000005, 286.0, 0.0, 0.
0, 0.0, 0.0, 88.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 3245.0000000000005, 1265.0,
0.0, 0.0, 0.0, 1628.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1155.0, 2013.
0000000000002, 0.0, 0.0, 2508.0, 1100.0, 1804.0000000000002, 0.0, 0.0, 0.0, 0.
0, 1034.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 880.0000000000001, 0.0, 0.0, 0.0, 0.
0, 1628.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2827.0000000000005, 869.0
000000000001, 0.0, 0.0, 2959.0000000000005, 1298.0, 2101.0, 0.0, 0.0, 0.0, 0.
0, 2497.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1078.0, 0.0, 0.0, 0.0, 0.0, 891.00000
00000001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2673.0, 1199.0, 0.0, 0.0, 1309.0, 144
1.0000000000002, 3157.0000000000005, 517.0, 0.0, 0.0, 0.0, 1287.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 1375.0, 0.0, 0.0, 0.0, 0.0, 1122.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 3212.0000000000005, 176.0, 0.0, 0.0, 2398.0, 2673.0, 3069.000000000000
5, 0.0, 0.0, 0.0, 0.0, 1452.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2189.
0, 0.0, 0.0, 0.0, 0.0, 2563.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 605.0, 1738.00000
00000002, 0.0, 0.0, 1133.0, 2266.0, 517.0, 0.0, 0.0, 0.0, 0.0, 1394.0000000000
007, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 55.00000000000001, 0.0, 0.0, 0.0, 0.0, 143
0.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 649.0, 1441.0000000000002, 0.0,
0.0, 198.00000000000003, 616.0, 1573.0000000000002, 0.0, 0.0, 0.0, 0.0, 1925.0
000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2992.0000000000005, 0.0, 0.0, 0.0,
0.0, 352.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 704.0, 3256.0000000000005, 0.0, 0.0,
2728.0, 891.0000000000001, 891.0000000000001, 0.0, 0.0, 0.0, 0.0, 220.00000000
000003, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2420.0, 0.0, 0.0, 0.0, 0.0, 1496.0000000
000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1617.0000000000002, 2233.0, 0.0, 0.0, 55
0.0, 2101.0, 2299.0, 0.0, 0.0, 0.0, 0.0, 3245.0000000000005, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 814.0000000000001, 0.0, 0.0, 0.0, 0.0, 165.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 3058.0000000000005, 2453.0, 0.0, 0.0, 979.0000000000001, 3146.000000
0000005, 3157.0000000000005, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 2574.0, 0.0, 0.0, 0.0, 0.0, 1177.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1738.00
00000000002, 2805.0, 0.0, 0.0, 2563.0, 1133.0, 495.00000000000006, 0.0, 0.0,
0.0, 0.0, 572.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2662.0, 0.0, 0.0, 0.0, 0.0, 235
4.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1419.0000000000002, 2233.0, 0.0, 0.0, 2739.
0, 605.0, 660.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2387.
0, 1932.0000000000007, 0.0, 0.0, 0.0, 825.0000000000001, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 1672.0000000000002, 2178.0, 0.0, 0.0, 2112.0, 3190.0000000000005, 92
4.0000000000001, 0.0, 0.0, 0.0, 0.0, 2838.0000000000005, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 1034.0, 0.0, 0.0, 0.0, 0.0, 2904.0000000000005, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 748.0000000000001, 1199.0, 0.0, 0.0, 3289.0000000000005, 319.0, 374.
00000000000006, 0.0, 0.0, 0.0, 0.0, 2123.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 305
8.0000000000005, 0.0, 0.0, 0.0, 0.0, 2354.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 276
1.0, 1991.0000000000002, 0.0, 0.0, 1540.0000000000002, 2673.0, 1089.0, 0.0, 0.
0, 0.0, 0.0, 1903.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2112.0, 0.0, 0.

0, 0.0, 0.0, 2871.000000000005, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1331.0, 858.000
0000000001, 0.0, 0.0, 924.0000000000001, 3014.000000000005, 1496.000000000000
2, 0.0, 0.0, 0.0, 0.0, 308.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 638.0, 0.0, 0.0,
0.0, 0.0, 1518.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1518.000000000000
2, 407.00000000000006, 0.0, 0.0, 2299.0, 3278.000000000005, 1969.000000000000
2, 0.0, 0.0, 0.0, 0.0, 2387.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 385.0000000000000
6, 0.0, 0.0, 0.0, 0.0, 869.0000000000001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 3267.0
000000000005, 3036.000000000005, 0.0, 0.0, 1540.0000000000002, 1694.000000000
0002, 3058.000000000005, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 2101.0, 0.0, 0.0, 0.0, 0.0, 550.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2068.0, 10
56.0, 0.0, 0.0, 308.0, 968.0000000000001, 1694.0000000000002, 1419.00000000000
02, 0.0, 0.0, 0.0, 462.00000000000006, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1617.0000
000000002, 0.0, 0.0, 0.0, 0.0, 649.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 682.0, 217
8.0, 0.0, 0.0, 1727.0000000000002, 1408.0, 2310.0, 0.0, 0.0, 0.0, 0.0, 2431.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1991.0000000000002, 0.0, 0.0, 0.0, 0.0, 143.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1474.0000000000002, 2046.0000000000002, 0.0, 0.
0, 2563.0, 1947.0000000000002, 253.00000000000003, 0.0, 849.9999999999984, 0.
0, 0.0, 2805.0, 0.0, 0.0, 0.0, 2387.0, 3102.000000000005, 957.0000000000001,
0.0, 0.0, 836.0000000000001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2937.0000
000000005, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 3212.0000000
000005, 759.0000000000001, 770.0000000000001, 0.0, 1133.0, 2255.0, 88.0, 0.0,
0.0, 132.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1232.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1562.0000000000002, 2013.0000000000
002, 0.0, 1452.0000000000002, 1661.0000000000002, 3212.000000000005, 0.0, 0.
0, 3179.000000000005, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 3289.0000000000
005, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1540.00000000
00002, 99.00000000000001, 0.0, 3223.000000000005, 209.00000000000003, 1826.00
00000000002, 0.0, 0.0, 1881.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 748.0000000000001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 1815.0000000000002, 1958.0000000000002, 0.0, 990.0000000000001, 2354.0, 1
507.0000000000002, 0.0, 0.0, 198.00000000000003, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 3201.000000000005, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 1771.0000000000002, 2409.0, 0.0, 1573.0000000000002, 1903.0000000
000002, 3289.000000000005, 0.0, 0.0, 1529.0000000000002, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 1067.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 1672.0000000000002, 770.0000000000001, 0.0, 2937.000000000005, 124
3.0, 605.0, 0.0, 0.0, 2794.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2794.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2992.000000000000
5, 1111.0, 0.0, 3223.000000000005, 2508.0, 1309.0, 0.0, 0.0, 2981.00000000000
05, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2992.000000000005, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 145.99999999999955, 902.0000000000001,
451.00000000000006, 0.0, 1606.0000000000002, 3234.000000000005, 2970.00000000
00005, 0.0, 0.0, 1023.0000000000001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2
288.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2343.0, 242
0.0, 0.0, 1441.0000000000002, 737.0000000000001, 176.0, 0.0, 0.0, 44.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2266.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 1837.0000000000002, 3201.000000000005, 0.0, 1584.000
0000000002, 726.0000000000001, 473.00000000000006, 0.0, 0.0, 473.000000000000
6, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1067.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1826.0000000000002, 2002.0000000000002, 0.0,
968.0000000000001, 363.00000000000006, 2068.0, 0.0, 0.0, 473.00000000000006,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1639.0000000000002, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 561.0, 2937.000000000005, 3102.00000000
00005, 0.0, 1881.0000000000002, 1243.0, 2255.0, 0.0, 0.0, 660.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 44.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 2541.0, 253.00000000000003, 0.0, 3212.000000000005, 2959.00000
00000005, 2706.0, 0.0, 0.0, 1705.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 902.0000000000001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 3069.000000000005, 2222.0, 2563.0, 0.0, 1441.0000000000002, 2772.0, 946.
0000000000001, 0.0, 0.0, 2717.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1111.

0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2959.0000000000
005, 979.0000000000001, 0.0, 1254.0, 3168.0000000000005, 1166.0, 0.0, 0.0, 192
5.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1573.0000000000002,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2739.0, 2035.00000
00000002, 0.0, 2541.0, 2607.0, 1364.0, 0.0, 0.0, 2310.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 825.0000000000001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 715.0000000000001, 957.0000000000001, 0.0, 2508.0, 3058.00
00000000005, 1793.0000000000002, 0.0, 0.0, 2035.0000000000002, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 2332.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 99.00000000000001, 1397.0, 0.0, 1661.0000000000002, 572.0, 110.
00000000000001, 0.0, 0.0, 1958.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 2816.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 32
12.0000000000005, 2024.0000000000002, 0.0, 1408.0, 2365.0, 1848.0000000000002,
0.0, 0.0, 3234.0000000000005, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 374.0000
0000000006, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 242.0000000
0000003, 1804.0000000000002, 781.0000000000001, 0.0, 2937.0000000000005, 2992.
0000000000005, 3003.0000000000005, 0.0, 0.0, 2871.0000000000005, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 1683.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 275.0, 1705.0000000000002, 0.0, 1243.0, 2860.000
0000000005, 2365.0, 0.0, 0.0, 363.00000000000006, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 3113.0000000000005, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 1672.0000000000002, 352.0, 0.0, 2860.0000000000005, 1958.000000
0000002, 1584.0000000000002, 0.0, 0.0, 1903.0000000000002, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 891.0000000000001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 1727.0000000000002, 3157.0000000000005, 1298.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 1738.0000000000002, 0.0, 0.0, 0.0, 0.0, 572.0, 726.0000000000001, 0.0, 0.0,
0.0, 0.0, 726.0000000000001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2200.0,
0.0, 0.0, 0.0, 0.0, 814.0000000000001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 3124.0000
000000005, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1837.0000000
000002, 0.0, 0.0, 0.0, 0.0, 2046.0000000000002, 1397.0, 0.0, 0.0, 0.0, 0.0, 9
9.00000000000001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1936.0000000000002, 0.0, 0.0, 0.0, 0.0, 231.0000
0000000003, 2475.0, 0.0, 0.0, 0.0, 0.0, 836.0000000000001, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2046.000
0000000002, 0.0, 0.0, 0.0, 0.0, 165.0, 2167.0, 0.0, 0.0, 0.0, 0.0, 2706.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 77.0, 0.0, 0.0, 0.0, 0.0, 1958.0000000000002, 1485.0000000000002, 0.
0, 0.0, 0.0, 0.0, 2607.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1529.0000000000002, 0.0, 0.0, 0.0, 0.0,
1331.0, 1837.0000000000002, 0.0, 0.0, 0.0, 0.0, 1892.0000000000002, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 2761.0, 0.0, 0.0, 0.0, 0.0, 693.0, 990.0000000000001, 0.0, 0.0, 0.0, 0.0, 1
617.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 77.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 693.0, 0.0, 0.0, 0.0, 0.0, 341.0, 396.0000000
0000006, 0.0, 0.0, 0.0, 0.0, 1507.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
1485.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 55
0.0, 0.0, 0.0, 0.0, 0.0, 2937.0000000000005, 1804.0000000000002, 0.0, 0.0, 0.
0, 0.0, 1430.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 979.0000000000001, 0.0, 0.0, 0.0, 0.0,
1485.0000000000002, 1408.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2750.0, 0.0,
0.0, 0.0, 0.0, 1309.0, 726.0000000000001, 0.0, 0.0, 0.0, 0.0, 1276.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 3201.0000000000005, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 319.0, 0.0, 0.0, 0.0, 0.0, 1419.0000000000002, 1661.000000
0000002, 0.0, 0.0, 0.0, 0.0, 154.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1430.0000000000002, 0.0, 0.0,
0.0, 0.0, 264.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1397.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 594.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1001.0000000

00001, 0.0, 0.0, 0.0, 0.0, 1144.0, 1881.0000000000002, 0.0, 0.0, 0.0, 0.0, 303
6.0000000000005, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 484.00000000000006, 0.0, 0.0, 0.0, 0.0, 2046.000
0000000002, 231.00000000000003, 0.0, 0.0, 0.0, 0.0, 671.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 2420.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 935.0
000000000001, 0.0, 0.0, 0.0, 0.0, 1342.0, 440.00000000000006, 0.0, 0.0, 0.0,
0.0, 2013.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2376.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2497.0, 0.0, 0.0, 0.0, 0.0, 1892.00000
00000002, 1287.0, 0.0, 0.0, 0.0, 0.0, 2651.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2475.0, 0.0, 0.0, 0.
0, 0.0, 1320.0, 3058.0000000000005, 0.0, 0.0, 0.0, 0.0, 1617.0000000000002, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 1408.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 2552.0, 0.0, 0.0, 0.0, 0.0, 3113.0000000000005, 957.000000000000
1, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1001.0000000000001, 0.0, 0.0, 0.0, 0.
0, 187.00000000000003, 3003.0000000000005, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 2200.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 2035.0000000000002, 0.0, 0.0, 0.0, 0.0, 1969.0000000000002, 2090.0, 0.0,
0.0, 0.0, 0.0, 3223.0000000000005, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1980.00000000
00002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2288.0, 0.0, 0.
0, 0.0, 0.0, 2329.000000000002, 858.0000000000001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 11.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2233.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 825.0000000000001, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1133.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2046.0000000000002, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1562.00000000
00002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 71
5.0000000000001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 2013.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 1749.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 1276.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 726.000000000001, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 451.00000000000006, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 99.00000000000001, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 3223.0000000000005,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2398.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1078.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 3278.0000000000
005, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 671.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 3157.000
0000000005, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 2816.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.

```
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
2607.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 1496.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 1727.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 143.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 2299.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 1617.0000000000002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
```

In [37]:
```
print("The associated costs for filling new demand is:")
print(np.matmul(np.transpose(c),x_opt2)-np.matmul(np.transpose(c),x_opt1))
```

```
The associated costs for filling new demand is:
4940989.870000012
```

In [38]:
```
# Right-hand side equalities
beq = data2[:,2]*1000
```

In [39]:
```
# Cost vector - here m instead of 5*23*30
c = np.zeros((m,));

# Preallocation of the solution vector for a change in
x = np.zeros((3450,5));
cost = np.zeros((1,5));
```

In [40]:
```
# Iterate through the cost vectors
for k in range(1,6):
    for i in range(23):
        for j in range(30):
            if k == 1:
                c[(k-1)*n + i*30 + j] = c[(k-1)*n + i*30 + j] + 0.85*data4[(k-1
            else:
                c[(k-1)*n + i*30 + j] = c[(k-1)*n + i*30 + j] + data4[(k-1)*30
```

In [41]:
```
# Create a Gurobi model
model = gp.Model()
```

In [42]:
```
# Define the decision variables
x0 = model.addMVar(shape=m, lb=-1e30, ub=gp.GRB.INFINITY, name="x")
```

In [43]:
```
# Set the objective function
model.setObjective(c @ x0, sense=gp.GRB.MINIMIZE)
```

In [44]:
```
# Add the constraints
model.addConstr(Aeq @ x0 == beq, name="eq")
model.addConstr(Aineq @ x0 <= bineq, name="ineq")
```

Out[44]:
```
<MConstr (3455,) *awaiting model update*>
```

In [45]:
```
# Solve the model
model.optimize()
```

```
Gurobi Optimizer version 10.0.1 build v10.0.1rc0 (win64)

CPU model: 12th Gen Intel(R) Core(TM) i5-1240P, instruction set [SSE2|AVX|AVX
2]
Thread count: 12 physical cores, 16 logical processors, using up to 16 threads

Optimize a model with 4145 rows, 3450 columns and 10350 nonzeros
Model fingerprint: 0x1d262bce
Coefficient statistics:
  Matrix range     [1e+00, 1e+00]
  Objective range  [4e+01, 7e+01]
  Bounds range     [0e+00, 0e+00]
  RHS range        [1e+01, 3e+05]
Presolve removed 3451 rows and 5 columns
Presolve time: 0.01s
Presolved: 694 rows, 3445 columns, 6890 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    4.5692423e+07   3.876500e+04   0.000000e+00      0s
     221    4.6682423e+07   0.000000e+00   0.000000e+00      0s

Solved in 221 iterations and 0.02 seconds (0.00 work units)
Optimal objective  4.668242343e+07
```

In [46]:

```
# Get the optimal solution
x[:, 0] = model.getAttr('x')
cost[:, 0] = np.matmul(np.transpose(c),x[:, 0])
```

In [47]:

```
# Iterate through the cost vectors
for k in range(1,6):
    for i in range(23):
        for j in range(30):
            if k == 2:
                c[(k-1)*n + i*30 + j] = c[(k-1)*n + i*30 + j] + 0.85*data4[(k-1
            else:
                c[(k-1)*n + i*30 + j] = c[(k-1)*n + i*30 + j] + data4[(k-1)*30
```

In [48]:

```
# Create a Gurobi model
model = gp.Model()
```

In [49]:

```
# Define the decision variables
x1 = model.addMVar(shape=m, lb=-1e30, ub=gp.GRB.INFINITY, name="x")
```

In [50]:

```
# Set the objective function
model.setObjective(c @ x1, sense=gp.GRB.MINIMIZE)
```

In [51]:

```
# Add the constraints
model.addConstr(Aeq @ x1 == beq, name="eq")
model.addConstr(Aineq @ x1 <= bineq, name="ineq")
```

Out[51]:

```
<MConstr (3455,) *awaiting model update*>
```

In [52]:

```
# Solve the model
model.optimize()
```

```
Gurobi Optimizer version 10.0.1 build v10.0.1rc0 (win64)

CPU model: 12th Gen Intel(R) Core(TM) i5-1240P, instruction set [SSE2|AVX|AVX
2]
Thread count: 12 physical cores, 16 logical processors, using up to 16 threads

Optimize a model with 4145 rows, 3450 columns and 10350 nonzeros
Model fingerprint: 0x676b3443
Coefficient statistics:
  Matrix range     [1e+00, 1e+00]
  Objective range  [8e+01, 1e+02]
  Bounds range     [0e+00, 0e+00]
  RHS range        [1e+01, 3e+05]
Presolve removed 3451 rows and 5 columns
Presolve time: 0.01s
Presolved: 694 rows, 3445 columns, 6890 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    9.2775624e+07   3.002625e+04   0.000000e+00      0s
     205    9.3699580e+07   0.000000e+00   0.000000e+00      0s

Solved in 205 iterations and 0.01 seconds (0.01 work units)
Optimal objective  9.369958032e+07
```

In [53]:
```
# Get the optimal solution
x[:, 1] = model.getAttr('x')
cost[:, 1] = np.matmul(np.transpose(c),x[:, 1])
```

In [54]:
```
# Iterate through the cost vectors
for k in range(1,6):
    for i in range(23):
        for j in range(30):
            if k == 3:
                c[(k-1)*n + i*30 + j] = c[(k-1)*n + i*30 + j] + 0.85*data4[(k-1
            else:
                c[(k-1)*n + i*30 + j] = c[(k-1)*n + i*30 + j] + data4[(k-1)*30
```

In [55]:
```
# Create a Gurobi model
model = gp.Model()
```

In [56]:
```
# Define the decision variables
x2 = model.addMVar(shape=m, lb=-1e30, ub=gp.GRB.INFINITY, name="x")
```

In [57]:
```
# Set the objective function
model.setObjective(c @ x2, sense=gp.GRB.MINIMIZE)
```

In [58]:
```
# Add the constraints
model.addConstr(Aeq @ x2 == beq, name="eq")
model.addConstr(Aineq @ x2 <= bineq, name="ineq")
```

Out[58]:
```
<MConstr (3455,) *awaiting model update*>
```

In [59]:
```
# Solve the model
model.optimize()
```

```
Gurobi Optimizer version 10.0.1 build v10.0.1rc0 (win64)

CPU model: 12th Gen Intel(R) Core(TM) i5-1240P, instruction set [SSE2|AVX|AVX
2]
Thread count: 12 physical cores, 16 logical processors, using up to 16 threads

Optimize a model with 4145 rows, 3450 columns and 10350 nonzeros
Model fingerprint: 0xdde9ede1
Coefficient statistics:
  Matrix range     [1e+00, 1e+00]
  Objective range  [1e+02, 2e+02]
  Bounds range     [0e+00, 0e+00]
  RHS range        [1e+01, 3e+05]
Presolve removed 3451 rows and 5 columns
Presolve time: 0.01s
Presolved: 694 rows, 3445 columns, 6890 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    1.4056036e+08   1.671500e+04   0.000000e+00      0s
     111    1.4091632e+08   0.000000e+00   0.000000e+00      0s

Solved in 111 iterations and 0.02 seconds (0.00 work units)
Optimal objective  1.409163222e+08
```

In [60]:

```
# Get the optimal solution
x[:, 2] = model.getAttr('x')
cost[:, 2] = np.matmul(np.transpose(c),x[:, 2])
```

In [61]:

```
# Iterate through the cost vectors
for k in range(1,6):
    for i in range(23):
        for j in range(30):
            if k == 4:
                c[(k-1)*n + i*30 + j] = c[(k-1)*n + i*30 + j] + 0.85*data4[(k-1
            else:
                c[(k-1)*n + i*30 + j] = c[(k-1)*n + i*30 + j] + data4[(k-1)*30
```

In [62]:

```
# Create a Gurobi model
model = gp.Model()
```

In [63]:

```
# Define the decision variables
x3 = model.addMVar(shape=m, lb=-1e30, ub=gp.GRB.INFINITY, name="x")
```

In [64]:

```
# Set the objective function
model.setObjective(c @ x3, sense=gp.GRB.MINIMIZE)
```

In [65]:

```
# Add the constraints
model.addConstr(Aeq @ x3 == beq, name="eq")
model.addConstr(Aineq @ x3 <= bineq, name="ineq")
```

Out[65]:

```
<MConstr (3455,) *awaiting model update*>
```

In [66]:

```
# Solve the model
model.optimize()
```

```
Gurobi Optimizer version 10.0.1 build v10.0.1rc0 (win64)

CPU model: 12th Gen Intel(R) Core(TM) i5-1240P, instruction set [SSE2|AVX|AVX
2]
Thread count: 12 physical cores, 16 logical processors, using up to 16 threads

Optimize a model with 4145 rows, 3450 columns and 10350 nonzeros
Model fingerprint: 0x15f5ddf2
Coefficient statistics:
  Matrix range     [1e+00, 1e+00]
  Objective range  [2e+02, 3e+02]
  Bounds range     [0e+00, 0e+00]
  RHS range        [1e+01, 3e+05]
Presolve removed 3451 rows and 5 columns
Presolve time: 0.01s
Presolved: 694 rows, 3445 columns, 6890 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    1.8927948e+08   1.192125e+04   0.000000e+00      0s
      67    1.8946867e+08   0.000000e+00   0.000000e+00      0s

Solved in 67 iterations and 0.02 seconds (0.00 work units)
Optimal objective  1.894686720e+08
```

In [67]:

```
# Get the optimal solution
x[:, 3] = model.getAttr('x')
cost[:, 3] = np.matmul(np.transpose(c),x[:, 4])
```

In [68]:

```
# Iterate through the cost vectors
for k in range(1,6):
    for i in range(23):
        for j in range(30):
            if k == 5:
                c[(k-1)*n + i*30 + j] = c[(k-1)*n + i*30 + j] + 0.85*data4[(k-1
            else:
                c[(k-1)*n + i*30 + j] = c[(k-1)*n + i*30 + j] + data4[(k-1)*30
```

In [69]:

```
# Create a Gurobi model
model = gp.Model()
```

In [70]:

```
# Define the decision variables
x4 = model.addMVar(shape=m, lb=-1e30, ub=gp.GRB.INFINITY, name="x")
```

In [71]:

```
# Set the objective function
model.setObjective(c @ x4, sense=gp.GRB.MINIMIZE)
```

In [72]:

```
# Add the constraints
model.addConstr(Aeq @ x4 == beq, name="eq")
model.addConstr(Aineq @ x4 <= bineq, name="ineq")
```

Out[72]:

```
<MConstr (3455,) *awaiting model update*>
```

In [73]:

```
# Solve the model
model.optimize()
```

```
Gurobi Optimizer version 10.0.1 build v10.0.1rc0 (win64)

CPU model: 12th Gen Intel(R) Core(TM) i5-1240P, instruction set [SSE2|AVX|AVX
2]
Thread count: 12 physical cores, 16 logical processors, using up to 16 threads

Optimize a model with 4145 rows, 3450 columns and 10350 nonzeros
Model fingerprint: 0xb9166a02
Coefficient statistics:
  Matrix range     [1e+00, 1e+00]
  Objective range  [2e+02, 3e+02]
  Bounds range     [0e+00, 0e+00]
  RHS range        [1e+01, 3e+05]
Presolve removed 3451 rows and 5 columns
Presolve time: 0.01s
Presolved: 694 rows, 3445 columns, 6890 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    2.3820427e+08   7.553750e+03   0.000000e+00      0s
      44    2.3832070e+08   0.000000e+00   0.000000e+00      0s

Solved in 44 iterations and 0.02 seconds (0.00 work units)
Optimal objective  2.383206999e+08
```

In [74]:

```
# Get the optimal solution
x[:, 4] = model.getAttr('x')
cost[:, 4] = np.matmul(np.transpose(c),x[:, 4])
```

In [75]:

```
# Find the facility with minimal cost
cost = list(cost)
# added 1 to this line where  calculate minimal to represent facilities from 1
minimal = cost.index(min(cost))+1
print("This policy should be implemented at facility: "+ str(minimal))
```

```
This policy should be implemented at facility: 1
```

In [76]:

```
# facility 1 is Alexandria city
```

In [ ]: