In [1]:

```
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
import pprint
import xgboost as xgb
from sklearn import linear_model
from sklearn.model_selection import train_test_split
from sklearn import ensemble
```

In [2]:

```
demands = pd.read_csv('demands.csv')
```

In [3]:

```
df_x = demands.iloc[:, 0:1]
df_y = demands.iloc[:, 1]
```

Figure 1

In [4]:

```
plt.scatter(df_x, df_y)
```

Out[4]:

<matplotlib.collections.PathCollection at 0x23e14be0e80>

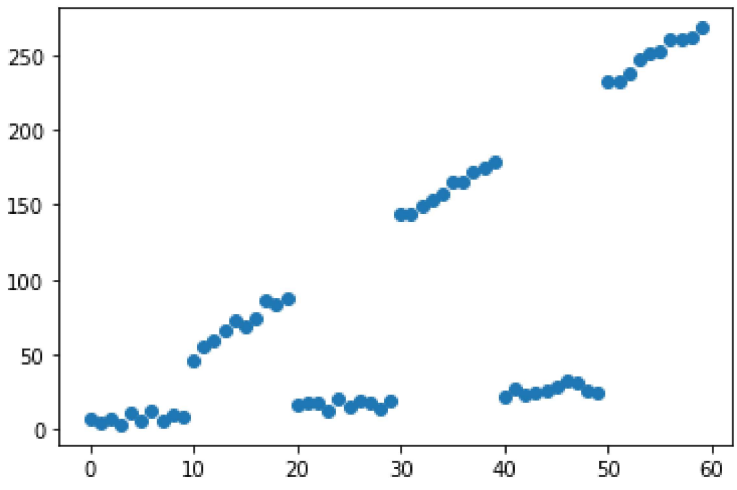


Figure 2

In [5]:

```
model1 = linear_model.LinearRegression()
model1.fit(df_x,df_y)
y_pred = model1.predict(df_x)
plt.scatter(df_x, df_y)
plt.plot(df_x, y_pred, color='red')
plt.show()
```

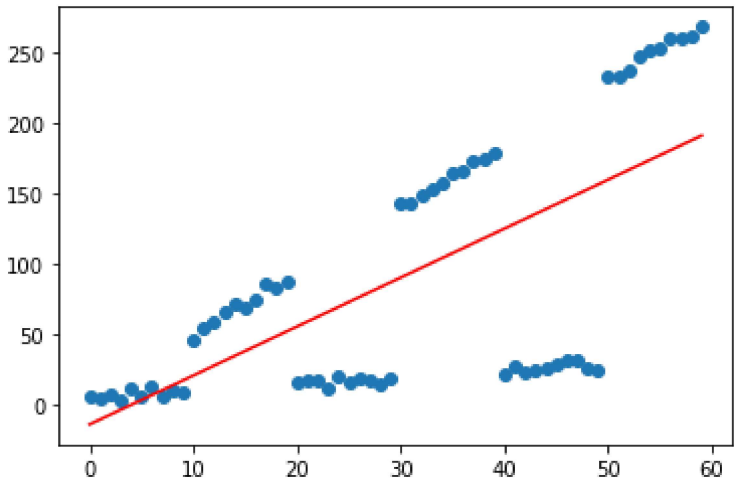


Figure 3

In [6]:

```
params = {'n_estimators': 1, 'max_depth': 1, 'learning_rate': 1, 'loss': 'ls'}
model2 = ensemble.GradientBoostingRegressor(**params)
model2.fit(df_x, df_y)
```

Out[6]:

```
GradientBoostingRegressor(learning_rate=1, max_depth=1, n_estimators=1)
```

In [7]:

```
y_pred = model2.predict(df_x)
plt.scatter(df_x, df_y)
plt.plot(df_x, y_pred, color='red')
plt.show()
```

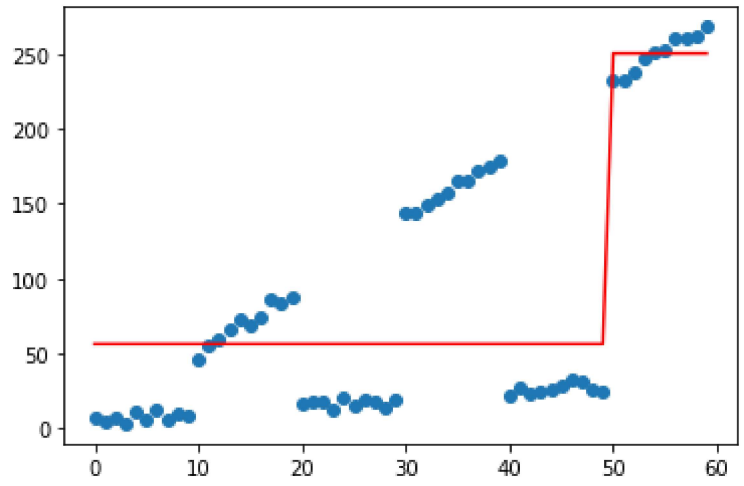


Figure 4

In [8]:

```
params = {'n_estimators': 2, 'max_depth': 1, 'learning_rate': 1, 'loss': 'ls'}
model3 = ensemble.GradientBoostingRegressor(**params)
model3.fit(df_x, df_y)
```

Out[8]:

```
GradientBoostingRegressor(learning_rate=1, max_depth=1, n_estimators=2)
```

In [9]:

```
y_pred = model3.predict(df_x)
plt.scatter(df_x, df_y)
plt.plot(df_x, y_pred, color='red')
plt.show()
```

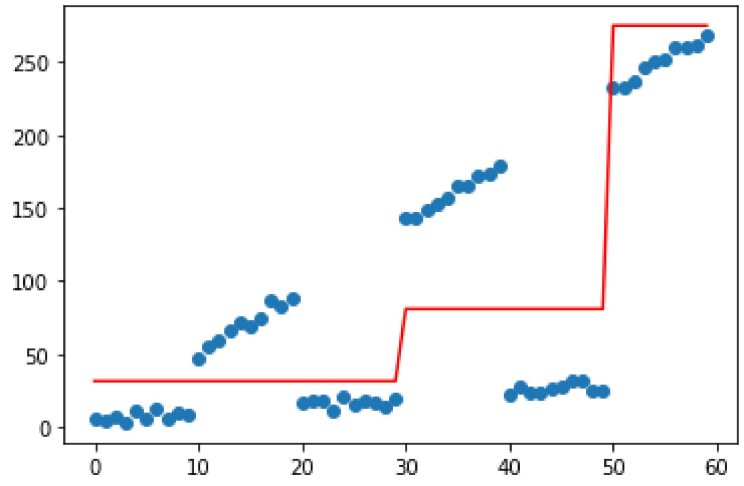


Figure 5

In [10]:

```
params = {'n_estimators': 5, 'max_depth': 1, 'learning_rate': 1, 'loss': 'ls'}
model4 = ensemble.GradientBoostingRegressor(**params)
model4.fit(df_x, df_y)
```

Out[10]:

```
GradientBoostingRegressor(learning_rate=1, max_depth=1, n_estimators=5)
```

In [11]:

```
y_pred = model4.predict(df_x)
plt.scatter(df_x, df_y)
plt.plot(df_x, y_pred, color='red')
plt.show()
```

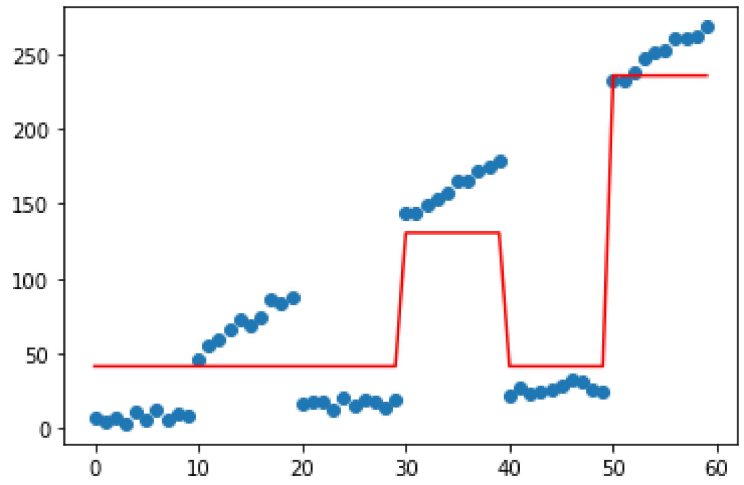


Figure 6

In [12]:

```
params = {'n_estimators': 10, 'max_depth': 1, 'learning_rate': 1, 'loss': 'ls'}
model5 = ensemble.GradientBoostingRegressor(**params)
model5.fit(df_x, df_y)
```

Out[12]:

```
GradientBoostingRegressor(learning_rate=1, max_depth=1, n_estimators=10)
```

In [13]:

```
y_pred = model5.predict(df_x)
plt.scatter(df_x, df_y)
plt.plot(df_x, y_pred, color='red')
plt.show()
```

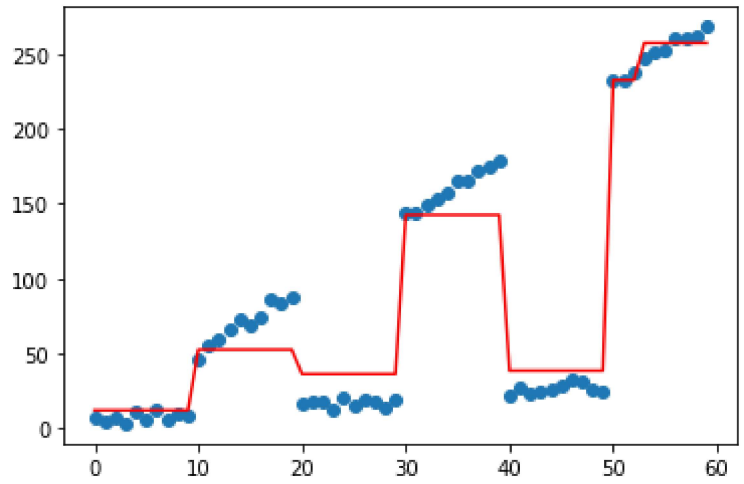


Figure 7

In [14]:

```
params = {'n_estimators': 20, 'max_depth': 1, 'learning_rate': 1, 'loss': 'ls'}
model6 = ensemble.GradientBoostingRegressor(**params)
model6.fit(df_x, df_y)
```

Out[14]:

```
GradientBoostingRegressor(learning_rate=1, max_depth=1, n_estimators=20)
```

In [15]:

```
y_pred = model6.predict(df_x)
plt.scatter(df_x, df_y)
plt.plot(df_x, y_pred, color='red')
plt.show()
```

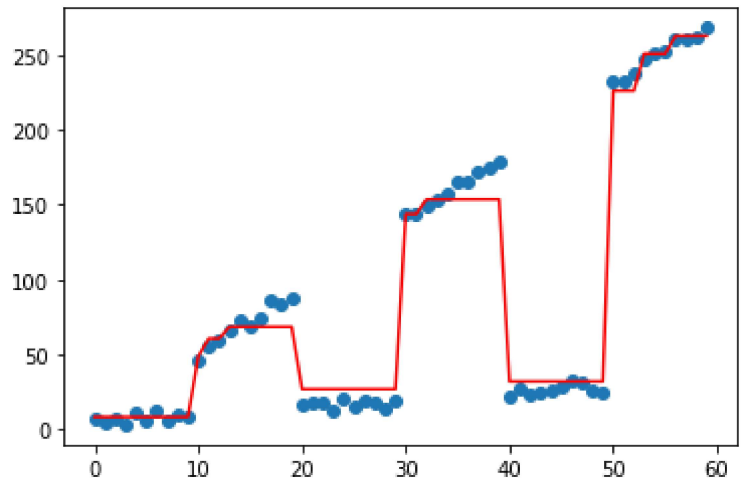


Figure 8

In [16]:

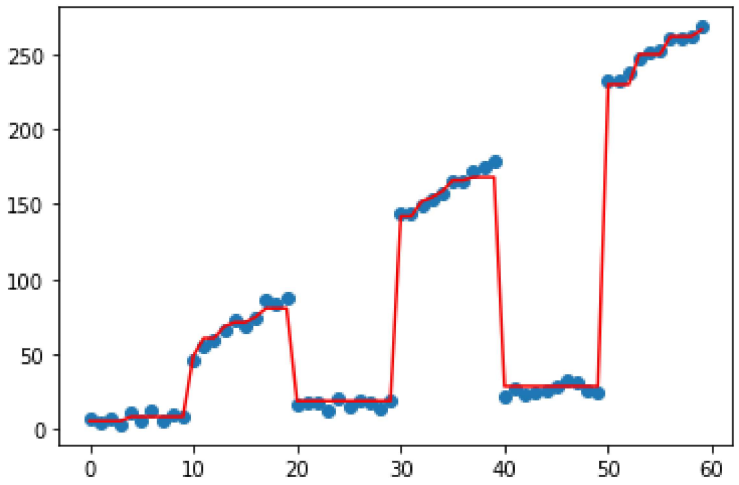
```
params = {'n_estimators': 50, 'max_depth': 1, 'learning_rate': 1, 'loss': 'ls'}
model7 = ensemble.GradientBoostingRegressor(**params)
model7.fit(df_x, df_y)
```

Out[16]:

```
GradientBoostingRegressor(learning_rate=1, max_depth=1, n_estimators=50)
```

In [17]:

```
y_pred = model7.predict(df_x)
plt.scatter(df_x, df_y)
plt.plot(df_x, y_pred, color='red')
plt.show()
```



Midterm Problem 2

(a). $\lambda = 50$, $K = 50$, $iC_j = h = \frac{200}{12}$

$$Q_3^* = \sqrt{\frac{2 \times 50 \times 50}{200/12}} = 17.32$$

$$Q_2^* = \sqrt{\frac{2 \times 50 \times 50}{200/12}} = 17.32$$

$$Q_1^* = \sqrt{\frac{2 \times 50 \times 50}{200/12}} = 17.32 \approx 17$$

$$Q_0^* = \sqrt{\frac{2 \times 50 \times 50}{200/12}} = 17.32$$

Only Q_1^* is realizable, and it has cost

$$g(17) = 510 \times 50 + \frac{50 \times 50}{17} + \frac{\frac{200}{12} \times 17}{2} = 25788.7$$

Next, we calculate the cost of the breakpoints to the right of Q_2^*

$$g_2(65) = 495 \times 50 + \frac{50 \times 50}{65} + \frac{\frac{200}{12} \times 65}{2} = 25330.1$$

$$g_3(129) = 485 \times 50 + \frac{50 \times 50}{129} + \frac{\frac{200}{12} \times 129}{2} = 25344.4$$

Therefore, the optimal order quantity is $Q = 65$, and the cost per month is \$25330.1

(b). $$g(65) = 520 \times 50 + \frac{0 \cdot 50}{65} + \frac{0 \cdot 65}{2} = 2600 > 25330.1$$

Therefore, Zeus should not accept the offer.

Midterm Problem 3

(a). $K = 1000 \quad h = 1.2$

$\theta_8 = 0$

$\theta_7 = K + h(0 \cdot d_4) + \theta_5$

$= 1000 \qquad [S(7) = 8]$

$\theta_6 = \min\{K + h(0 \cdot d_6) + \theta_7,\ K + h(0 \cdot d_6 + 1 \cdot d_7) + \theta_8\}$

$= \min\{2000, 1348\}$

$= 1348 \qquad [S(6) = 8]$

$\theta_5 = \min\{K + h(0 \cdot d_5) + \theta_6,\ K + h(0 \cdot d_5 + 1 \cdot d_6) + \theta_7,\ K + h(0 \cdot d_5 + 1 \cdot d_6 + 2 \cdot d_7) + \theta_8\}$

$= \min\{2348, 2252, 1948\}$

$= 1948 \qquad [S(5) = 8]$

$\theta_4 = \min\{K + h(0 \cdot d_4) + \theta_5,\ K + h(0 \cdot d_4 + 1 \cdot d_5) + \theta_6,\ K + h(0 \cdot d_4 + 1 \cdot d_5 + 2 \cdot d_6) + \theta_7,$
$K + h(0 \cdot d_4 + 1 \cdot d_5 + 2 \cdot d_6 + 3 \cdot d_7) + \theta_8\}$

$= \min\{2948, 2552, 2708, 2752\}$

$= 2552 \qquad [S(4) = 6]$

$\theta_3 = \min\{K + h(0 \cdot d_3) + \theta_4,\ K + h(0 \cdot d_3 + 1 \cdot d_4) + \theta_5,\ K + h(0 \cdot d_3 + 1 \cdot d_4 + 2 \cdot d_5) + \theta_6,$
$K + h(0 \cdot d_3 + 1 \cdot d_4 + 2 \cdot d_5 + 3 \cdot d_6) + \theta_7,\ K + h(0 \cdot d_3 + 1 \cdot d_4 + 2 \cdot d_5 + 3 \cdot d_6 + 4 \cdot d_7) + \theta_8\}$

$= \min\{3552, 3056, 2864, 3272, 3664\}$

$= 2864 \qquad [S(3) = 6]$

$\theta_2 = \min\{K + h(0 \cdot d_2) + \theta_3,\ K + h(0 \cdot d_2 + 1 \cdot d_3) + \theta_4,\ K + h(0 \cdot d_2 + 1 \cdot d_3 + 2 \cdot d_4) + \theta_5,$
$K + h(0 \cdot d_2 + 1 \cdot d_3 + 2 \cdot d_4 + 3 \cdot d_5) + \theta_6,\ K + h(0 \cdot d_2 + 1 \cdot d_3 + 2 \cdot d_4 + 3 \cdot d_5 + 4 \cdot d_6) + \theta_7,$
$K + h(0 \cdot d_2 + 1 \cdot d_3 + 2 \cdot d_4 + 3 \cdot d_5 + 4 \cdot d_6 + 5 \cdot d_7) + \theta_8\}$

$= \min\{3864, 3678, 3290, 3302, 3962, 4702\}$

$= 3290 \qquad [S(2) = 5]$

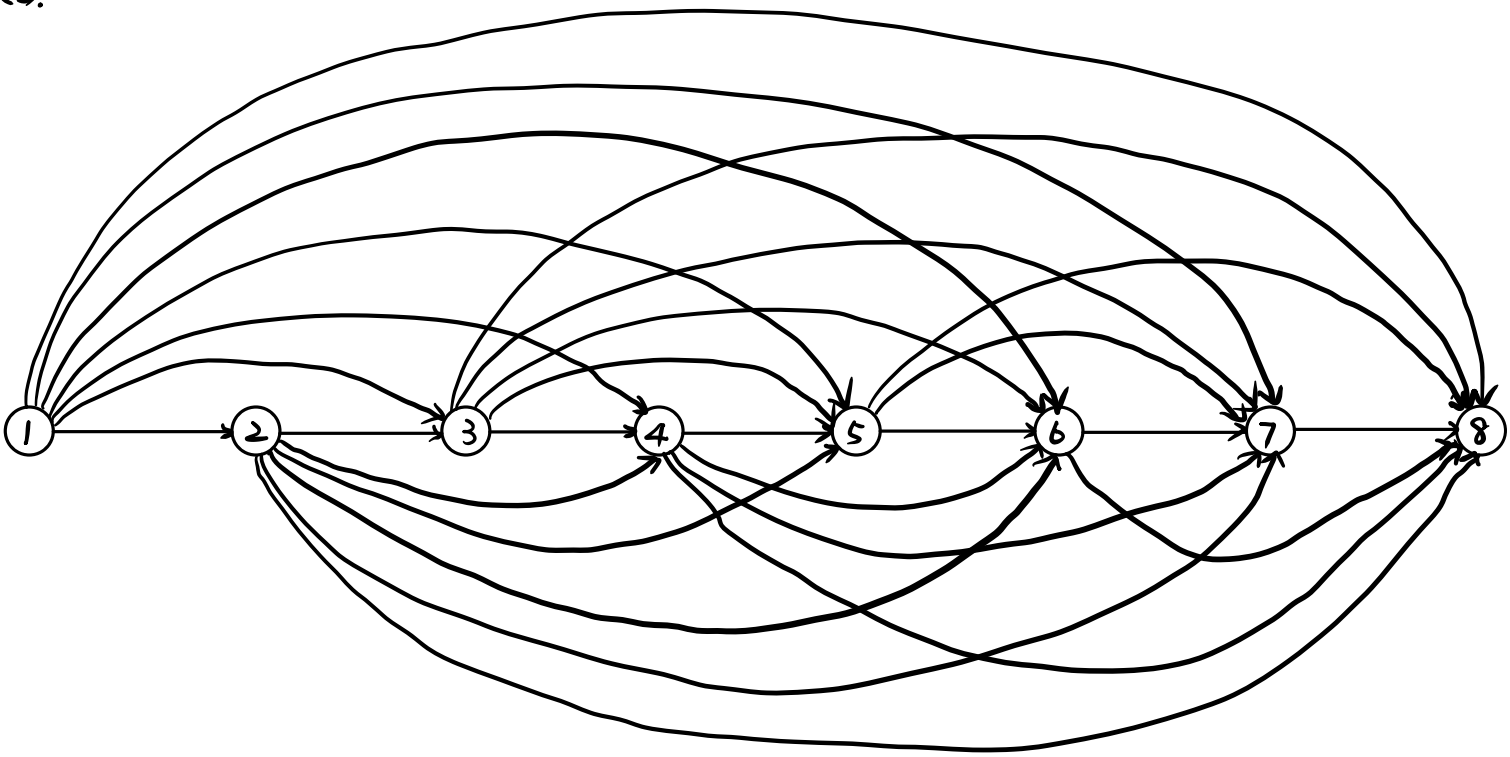
$\theta_1 = \min\{K + h(0 \cdot d_1) + \theta_2,\ K + h(0 \cdot d_1 + 1 \cdot d_2) + \theta_3,\ K + h(0 \cdot d_1 + 1 \cdot d_2 + 2 \cdot d_3) + \theta_4,$
$K + h(0 \cdot d_1 + 1 \cdot d_2 + 2 \cdot d_3 + 3 \cdot d_4) + \theta_5,\ K + h(0 \cdot d_1 + 1 \cdot d_2 + 2 \cdot d_3 + 3 \cdot d_4 + 4 \cdot d_5) + \theta_6,$
$K + h(0 \cdot d_1 + 1 \cdot d_2 + 2 \cdot d_3 + 3 \cdot d_4 + 4 \cdot d_5 + 5 \cdot d_6) + \theta_7,$
$K + h(0 \cdot d_1 + 1 \cdot d_2 + 2 \cdot d_3 + 3 \cdot d_4 + 4 \cdot d_5 + 5 \cdot d_6 + 6 \cdot d_7) + \theta_8\}$

$= \min\{4290, 4050, 3990, 3710, 3926, 4838, 5926\}$

$= 3710 \qquad [S(1) = 5]$

Therefore, on Day #1, we should order 570, and then on Day #5, we should order 670. Total cost is $\$3710$.

(b).



Edge cost:

$1\to2$: $K = 1000$
$1\to3$: $K+hD_2 = 1186$
$1\to4$: $K+hD_2+2hD_3 = 1438$
$1-5$: $K+hD_2+2hD_3+3hD_4 = 1762$
$1\to6$: $K+hD_2+2hD_3+3hD_4+4hD_5 = 2578$
$1\to7$: $K+hD_2+2hD_3+3hD_4+\cdots+5hD_6 = 3838$
$1\to8$: $K+hD_2+2hD_3+3hD_4+\cdots+6hD_7 = 5926$
$2\to3$: $K = 1000$
$2\to4$: $K+hD_3 = 1126$
$2\to5$: $K+hD_3+2hD_4 = 1342$
$2\to6$: $K+hD_3+2hD_4+3hD_5 = 1954$
$2\to7$: $K+hD_3+2hD_4+3hD_5+4hD_6 = 2962$
$2\to8$: $K+hD_3+2hD_4+\cdots+5hD_7 = 4702$
$3\to4$: $K = 1000$
$3\to5$: $K+hD_4 = 1108$
$3\to6$: $K+hD_4+2hD_5 = 1516$
$3\to7$: $K+hD_4+2hD_5+3hD_6 = 2272$
$3\to8$: $K+hD_4+2hD_5+3hD_6+4hD_7 = 3664$
$4\to5$: $K = 1000$
$4\to6$: $K+hD_5 = 1204$
$4\to7$: $K+hD_5+2hD_6 = 1708$
$4\to8$: $K+hD_5+2hD_6+3hD_7 = 2752$
$5\to6$: $K = 1000$
$5\to7$: $K+hD_6 = 1252$
$5\to8$: $K+hD_6+2hD_7 = 1948$
$6\to7$: $K = 1000$
$6\to8$: $K+hD_7 = 1348$
$7\to8$: $K = 1000$

i $A[1]=0$, $B[1]=\emptyset$

ii $A[2]=1000$, $B[2]=\{1\to2\}$

iii $A[3]=1186$, $B[3]=\{1\to3\}$

iv. $A[4]=1438$, $B[4]=\{1\to4\}$

v. $A[5]=1762$, $B[5]=\{1\to5\}$

vi. $A[6]=2578$, $B[6]=\{1\to6\}$

vii. $A[7]=3014$, $B[7]=\{1\to5, 5\to7\}$

viii. $A[8]=3710$, $B[8]=\{1\to5, 5\to8\}$

Therefore, on Day # 1, we should order 570, and then on Day # 5, we should order 670. Total cost is $ 3710.

(c). Let q_t = the number of units ordered in period t

y_t = 1 if we order in period t, 0 otherwise

x_t = the inventory level at the end of period, with $x_0 \equiv 0$.

$$\text{minimize} \sum_{t=1}^{T} (1000 y_t + 1.2 x_t)$$

$$\text{subject to } x_t = x_{t-1} + q_t - d_t \qquad \forall t = 1, \ldots, T$$

$$q_t \le M y_t \qquad \forall t = 1, \ldots, T$$

$$x_t \ge 0 \qquad \forall t = 1, \ldots, T$$

$$q_t \ge 0 \qquad \forall t = 1, \ldots, T$$

$$y_t \in \{0, 1\} \qquad \forall t = 1, \ldots, T$$

where $T = 7$, $M = 100000$

The code is on next page.

In [1]:

```
import numpy as np
from gurobipy import *
```

In [2]:

```
d = [220,155,105,90,170,210,290]
T, K, h = 7, 1000, 1.2
M = 10e5
```

In [3]:

```
WW = Model()

q = WW.addVars(T, lb=np.zeros(7), vtype=GRB.CONTINUOUS, name="order_quantity")
x = WW.addVars(T, lb=np.zeros(7), vtype=GRB.CONTINUOUS, name="inventory_level")
y = WW.addVars(T, vtype=GRB.BINARY, name="if_order")

WW.setObjective(quicksum(K*y[t]+h*x[t] for t in range(T)), GRB.MINIMIZE)

c1 = WW.addConstrs(q[t] <= M*y[t] for t in range(T))
c2 = WW.addConstrs(x[t] == x[t-1]+ q[t] - d[t] for t in range(1,T))
c3 = WW.addConstr(x[0] == q[0] - d[0])
```

```
Restricted license - for non-production use only - expires 2022-01-13
```

In [4]:

```
WW.optimize()
WW.printAttr('X')
```

```
Gurobi Optimizer version 9.1.2 build v9.1.2rc0 (win64)
Thread count: 4 physical cores, 8 logical processors, using up to 8 threads
Optimize a model with 14 rows, 21 columns and 34 nonzeros
Model fingerprint: 0x75f00a53
Variable types: 14 continuous, 7 integer (7 binary)
Coefficient statistics:
  Matrix range     [1e+00, 1e+06]
  Objective range  [1e+00, 1e+03]
  Bounds range     [1e+00, 1e+00]
  RHS range        [9e+01, 3e+02]
Presolve removed 3 rows and 4 columns
Presolve time: 0.01s
Presolved: 11 rows, 17 columns, 27 nonzeros
Variable types: 11 continuous, 6 integer (6 binary)

Root relaxation: objective 1.949020e+03, 7 iterations, 0.00 seconds

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0 1949.02000    0    4          - 1949.02000      -     -    0s
H    0     0                       5948.0000000 1949.02000  67.2%     -    0s
H    0     0                       3926.0000000 1949.02000  50.4%     -    0s
     0     0 3204.90092    0    3 3926.00000 3204.90092  18.4%     -    0s
H    0     0                       3710.0000000 3204.90092  13.6%     -    0s

Cutting planes:
  Implied bound: 7
  MIR: 2
  Flow cover: 4

Explored 1 nodes (15 simplex iterations) in 0.06 seconds
Thread count was 8 (of 8 available processors)

Solution count 3: 3710 3926 5948

Optimal solution found (tolerance 1.00e-04)
Best objective 3.709999999983e+03, best bound 3.709999999983e+03, gap 0.0000%

    Variable            X
-------------------------
order_quantity[0]          570
order_quantity[4]          670
inventory_level[0]          350
inventory_level[1]          195
inventory_level[2]           90
inventory_level[4]          500
inventory_level[5]          290
 if_order[0]            1
 if_order[4]            1
```

Midterm Problem 4.

(a). $C = \begin{pmatrix} 6 \\ 14 \\ 13 \end{pmatrix}$ $A = \begin{pmatrix} 3 & 2 & 1 \\ & & \\ 1 & 2 & 4 \end{pmatrix}$ $x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$ $b = \begin{pmatrix} 24 \\ 60 \end{pmatrix}$

$A^T = \begin{pmatrix} 3 & 1 \\ 2 & 2 \\ 1 & 4 \end{pmatrix}$ $b^T = (24 \quad 60)$

$$\begin{aligned} \min \quad & 24y_1 + 60y_2 \\ \text{s.t.} \quad & 3y_1 + y_2 \geq 6 \\ & 2y_1 + 2y_2 \leq 14 \\ & y_1 + 4y_2 = 13 \\ & y_1 \geq 0 \\ & y_2 \leq 0 \end{aligned}$$

(b). $C = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \Big\} |I|\times|J|$ $b = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \Big\} |I|+|J|$

$A = \begin{pmatrix} 111\cdots100\cdots0000 \\ 000\cdots111\cdots100\cdots0 \\ \vdots \\ 000\cdots00111\cdots1 \\ 1000\cdots1000\cdots1000\cdots \\ 0100\cdots0100\cdots0100\cdots \\ 0010\cdots0010\cdots0010\cdots \\ \vdots \\ 0001\cdots0001\cdots0001 \end{pmatrix}$

$b^T = (\underbrace{1 \cdots 1}_{|I|+|J|})$

$A^T = \begin{pmatrix} \overbrace{1000\cdots}^{|I|}\overbrace{000}^{|J|}100\cdots0 \\ 1000\cdots0000100\cdots0 \\ 1000\cdots00000100\cdots0 \\ \vdots \\ 0100\cdots000100\cdots0 \\ 0100\cdots000010\cdots0 \\ \vdots \\ 000\cdots001000\cdots1 \end{pmatrix} \Big\} |I|\times|J|$

we notice that for A^T, there are two "1" in each row, one in $|I|$ part and one in $|J|$ part.

Therefore: $\min \sum_{i\in I} u_i + \sum_{j\in J} v_j$

$$\begin{aligned} \text{s.t.} \quad & u_i + v_j \geq 1 \quad \text{for all } i\in I,\ j\in J \\ & u_i \geq 0 \quad \text{for } i\in I \\ & v_j \geq 0 \quad \text{for } j\in J \end{aligned}$$