```
# -*- coding: utf-8 -*-
"""
Created on Sun Jul  3 14:51:49 2022

"""

import pandas as pd
import gurobipy as gb
import numpy as np
import copy

N = 42
Q = 4
Months = list(range(1,13))


storage = pd.read_excel(r'C:\Users\bronx\Downloads\data.xlsx',sheet_name = 'Misc. Data',header=None)
F = int((storage.iloc[0][1])[0:3])

items = pd.read_excel(r'C:\Users\bronx\Downloads\data.xlsx',sheet_name = 'Items',header=1)
#print(items)

items_list = list(items.Item)
Potions = [i for i in items_list if "Potion" in i]
Normal = copy.deepcopy(items_list)
Normal.remove("Rations")
#print(Normal)


f = dict(zip(items["Item"],items["Space (cubic feet)"]))
```

```
#print(f)

p = dict(zip(items["Item"],round(items["Cost to Obtain"],4)))
#print(p)

d = {}
for item in range(N):
    d_dict = {1:items.loc[item]["January"],
              2:items.loc[item]["February"],
              3:items.loc[item]["March"],
              4:items.loc[item]["April"],
              5:items.loc[item]["May"],
              6:items.loc[item]["June"],
              7:items.loc[item]["July"],
              8:items.loc[item]["August"],
              9:items.loc[item]["September"],
              10:items.loc[item]["October"],
              11:items.loc[item]["November"],
              12:items.loc[item]["December"]}
    d[items.loc[item]["Item"]] = d_dict
#print(d)



c = {}
for item in range(N):
    c_dict = {1:items.loc[item]["January.1"],
              2:items.loc[item]["February.1"],
              3:items.loc[item]["March.1"],
              4:items.loc[item]["April.1"],
              5:items.loc[item]["May.1"],
```

```
            6:items.loc[item]["June.1"],
            7:items.loc[item]["July.1"],
            8:items.loc[item]["August.1"],
            9:items.loc[item]["September.1"],
            10:items.loc[item]["October.1"],
            11:items.loc[item]["November.1"],
            12:items.loc[item]["December.1"]}
    c[items.loc[item]["Item"]] = c_dict
#print(c)



recipes = pd.read_excel(r'C:\Users\bronx\Downloads\data.xlsx',sheet_name = 'Recipes',header=1)
#print(recipes)

products = list(recipes.Recipe)
#print(products)

b = {}
for j in range(Q):
   b[recipes.loc[j]["Recipe"]] = recipes.loc[j]["# Made in a batch"]
#print(h)



a = {}
for j in products:
    dict = {}
    for i in items_list:
            dict[i] = 0
    a[j] = dict
```

```
for j in range(Q):
   a[recipes.loc[j]["Recipe"]]["Flour"] = recipes.loc[j]["Flour Used"]
   a[recipes.loc[j]["Recipe"]]["Eggs"] = recipes.loc[j]["Egg Used"]
   a[recipes.loc[j]["Recipe"]]["Sugar"] = recipes.loc[j]["Sugar Used"]
   a[recipes.loc[j]["Recipe"]]["Butter"] = recipes.loc[j]["Butter Used"]
   a[recipes.loc[j]["Recipe"]]["Apple"] = recipes.loc[j]["Apples Used"]
#print(a)



e = {}
for product in range(Q):
    e_dict = {1:recipes.loc[product]["January.1"],
              2:recipes.loc[product]["February.1"],
              3:recipes.loc[product]["March.1"],
              4:recipes.loc[product]["April.1"],
              5:recipes.loc[product]["May.1"],
              6:recipes.loc[product]["June.1"],
              7:recipes.loc[product]["July.1"],
              8:recipes.loc[product]["August.1"],
              9:recipes.loc[product]["September.1"],
              10:recipes.loc[product]["October.1"],
              11:recipes.loc[product]["November.1"],
              12:recipes.loc[product]["December.1"]}
    e[recipes.loc[product]["Recipe"]] = e_dict
#print(e)



h = {}
for j in range(Q):
   h[recipes.loc[j]["Recipe"]] = recipes.loc[j]["Hours per Batch"]
```

```
#print(h)
```

```
g = {}
for j in range(Q):
    g_dict = {1:recipes.loc[j]["January"],
              2:recipes.loc[j]["February"],
              3:recipes.loc[j]["March"],
              4:recipes.loc[j]["April"],
              5:recipes.loc[j]["May"],
              6:recipes.loc[j]["June"],
              7:recipes.loc[j]["July"],
              8:recipes.loc[j]["August"],
              9:recipes.loc[j]["September"],
              10:recipes.loc[j]["October"],
              11:recipes.loc[j]["November"],
              12:recipes.loc[j]["December"]}
    g[recipes.loc[j]["Recipe"]] = g_dict
#print(g)
```

```
# model
m = gb.Model('model1')
```

```
# variables
D = {}
x = {}
```

```
y = {}
z = {}
for i in items_list:
    D[i] = {}
    x[i] = {}
    y[i] = {}
    z[i] = {}
    for j in Months:
        D[i][j] = m.addVar(name = 'D_(%s)_(%s)' % (str(i), str(j)),
           vtype = gb.GRB.CONTINUOUS)
        x[i][j] = m.addVar(name = 'x_(%s)_(%s)' % (str(i), str(j)),
            vtype = gb.GRB.CONTINUOUS)
        y[i][j] = m.addVar(name = 'y_(%s)_(%s)' % (str(i), str(j)),
            vtype = gb.GRB.INTEGER)
        z[i][j] = m.addVar(name = 'z_(%s)_(%s)' % (str(i), str(j)),
            vtype = gb.GRB.INTEGER)


A = {}
B = {}
C = {}
E = {}
for i in products:
    A[i] = {}
    B[i] = {}
    C[i] = {}
    E[i] = {}
    for j in Months:
        A[i][j] = m.addVar(name = 'A_(%s)_(%s)' % (str(i), str(j)),
            vtype = gb.GRB.INTEGER)
        B[i][j] = m.addVar(name = 'B_(%s)_(%s)' % (str(i), str(j)),
```

```
            vtype = gb.GRB.INTEGER)
        C[i][j] = m.addVar(name = 'C_(%s)_(%s)' % (str(i), str(j)),
            vtype = gb.GRB.INTEGER)
        E[i][j] = m.addVar(name = 'E_(%s)_(%s)' % (str(i), str(j)),
            vtype = gb.GRB.CONTINUOUS)


G = {}
R = {}
S = {}
T = {}
for month in Months:
    G[month] = m.addVar(name = 'G_(%s)' % (str(month)),
        vtype = gb.GRB.CONTINUOUS)
    R[month] = m.addVar(name = 'R_(%s)' % (str(month)),
        vtype = gb.GRB.CONTINUOUS)
    S[month] = m.addVar(name = 'S_(%s)' % (str(month)),
        vtype = gb.GRB.CONTINUOUS)
    T[month] = m.addVar(name = 'T_(%s)' % (str(month)),
        vtype = gb.GRB.CONTINUOUS)


v = {}
u = {}
n = {}
o = {}
for month in Months:
    v[month] = m.addVar(name = 'v_(%s)' % (str(month)),
        vtype = gb.GRB.BINARY)
    u[month] = m.addVar(name = 'u_(%s)' % (str(month)),
        vtype = gb.GRB.BINARY)
    n[month] = m.addVar(name = 'n_(%s)' % (str(month)),
```

```
        vtype = gb.GRB.BINARY)
    o[month] = m.addVar(name = 'o_(%s)' % (str(month)),
        vtype = gb.GRB.BINARY)


w = {}
q = {}
r = {}
s = {}
t = {}
for i in Potions:
    w[i] = {}
    q[i] = {}
    r[i] = {}
    s[i] = {}
    t[i] = {}
    for month in Months:
        w[i][month] = m.addVar(name = 'w_(%s)_(%s)' % (str(i), str(month)),
            vtype = gb.GRB.BINARY)
        q[i][month] = m.addVar(name = 'q_(%s)_(%s)' % (str(i), str(month)),
            vtype = gb.GRB.INTEGER)
        r[i][month] = m.addVar(name = 'r_(%s)_(%s)' % (str(i), str(month)),
            vtype = gb.GRB.INTEGER,lb=1,ub=3)
        s[i][month] = m.addVar(name = 's_(%s)_(%s)' % (str(i), str(month)),
            vtype = gb.GRB.INTEGER)
        t[i][month] = m.addVar(name = 't_(%s)_(%s)' % (str(i), str(month)),
            vtype = gb.GRB.INTEGER,lb=1,ub=12)


# constraints
for month in Months:
    m.addConstr(gb.quicksum(x[i][month] * f[i] for i in items_list) <= F)
```

```
for i in items_list:
    for month in Months:
        m.addConstr(x[i][month] + z[i][month] >= y[i][month] +
                    gb.quicksum(A[j][month] * a[j][i] for j in products))

for i in items_list:
    m.addConstr(x[i][1] == 0)
    for month in Months[1:]:
        m.addConstr(x[i][month] == x[i][month-1] + z[i][month-1] -
                    gb.quicksum(A[j][month-1] * a[j][i] for j in products) -
                    y[i][month-1])

for month in Months:
    m.addConstr(gb.quicksum(A[j][month] * h[j] for j in products) <= 40)

for j in products:
    for month in Months:
        m.addConstr(B[j][month] == A[j][month] * b[j])

for j in products:
    for month in Months:
        m.addConstr(B[j][month] >= C[j][month])

for i in items_list:
    for month in Months:
        m.addConstr(D[i][month] <= y[i][month] * c[i][month])
        m.addConstr(D[i][month] <= c[i][month] * (.2 * d[i][month] + .8 * y[i][month]))
        m.addConstr(D[i][month] <= c[i][month] * (.65 * d[i][month] + .5 * y[i][month]))
        m.addConstr(y[i][month] <= 2 * d[i][month])
```

```
for j in products:
    for month in Months:
        m.addConstr(E[j][month] <= C[j][month] * g[j][month])
        m.addConstr(E[j][month] <= g[j][month] * (.2 * e[j][month] + .8 * C[j][month]))
        m.addConstr(E[j][month] <= g[j][month] * (.65 * e[j][month] + .5 * C[j][month]))
        m.addConstr(C[j][month] <= 2 * e[j][month])

for month in Months:
    m.addConstr(gb.quicksum(z[i][month] * f[i] for i in items_list) <= F * u[month])
    m.addConstr((gb.quicksum(z[i][month] * f[i] for i in items_list)) - 5 <= F * n[month])
    m.addConstr(S[month] == 2 * u[month] + 18 * n[month])

for month in Months:
    m.addConstr(z["Livestock Feed"][month] >= 50 * o[month])
    m.addConstr(z["Livestock Feed"][month] <= 2500 * o[month])

for i in Potions:
    for month in Months:
        m.addConstr(w[i][month] >= (1/5) * z[i][month])
        m.addConstr(4 * q[i][month] == z[i][month] + r[i][month]*w[i][month])
        m.addConstr(13 * s[i][month] == z[i][month] + t[i][month]*w[i][month])

for month in Months:
    m.addConstr(v[month] >= (1/400) * (z["Rations"][month] - 12))
    m.addConstr(R[month] == (1 - v[month]) * (z["Rations"][month] * p["Rations"]) +
                v[month] * (12 * p["Rations"] + .4 * z["Rations"][month] - 4.8))

for month in Months:
    m.addConstr(T[month] == gb.quicksum(z[i][month] * p[i] for i in Normal))
```

```
for month in Months:
    m.addConstr(G[month] == R[month] + S[month] + T[month])

for i in items_list:
    for month in Months:
        m.addConstr(x[i][month] >= 0)
        m.addConstr(y[i][month] >= 0)
        m.addConstr(z[i][month] >= 0)
        m.addConstr(D[i][month] >= 0)

for j in products:
    for month in Months:
        m.addConstr(A[j][month] >= 0)
        m.addConstr(B[j][month] >= 0)
        m.addConstr(C[j][month] >= 0)
        m.addConstr(E[j][month] >= 0)

for i in Potions:
    for month in Months:
        m.addConstr(q[i][month] >= 0)
        m.addConstr(r[i][month] >= 0)
        m.addConstr(s[i][month] >= 0)
        m.addConstr(t[i][month] >= 0)

# objective function
obj = (gb.quicksum(D[i][month] for i in items_list for month in Months) +
        gb.quicksum(E[j][month] for j in products for month in Months) -
        gb.quicksum(G[month] for month in Months)) / 12
m.setObjective(obj,gb.GRB.MAXIMIZE)
```

```
# update and solve
m.update()
m.optimize()

#print(m.getVars())
#m.display()

print(" ")

print("Optimal Monthly Profit:")
print(round(m.objVal,2))

print(" ")


for month in Months:
    d_temp = round(gb.quicksum(D[i][month] for i in items_list).getValue(),2)
    print("Item Sales (D) in Month {}: {}".format(month,d_temp))
    e_temp = round(gb.quicksum(E[j][month] for j in products).getValue(),2)
    print("Product Sales (E) in Month {}: {}".format(month,e_temp))
    g_temp = round(G[month].X,2)
    print("Costs (G) in Month {}: {}".format(month,g_temp))
    print(" ")
```