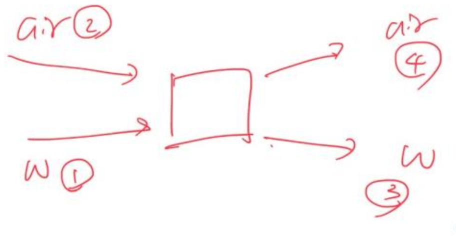
A wastewater treatment plant is designed to remove organic contaminants, including Benzene, Toluene, and Xylene, from wastewater by aerating the water with air. The plant receives a wastewater stream with a flow rate of 1400 moles/hr and concentrations of 13 mole% and 16 mole% for Benzene and Xylene, respectively. The plant also receives a stream of compressed air with a flow rate of 43110 g/hr, containing 1%, 78%, and 21% mole percentages of steam (water), N_2, and O_2, respectively. The treatment process produces two output streams: a treated water stream and an exhaust air stream. The treated water stream has reduced concentrations of Benzene and Toluene at 8 and 1.5 mole%, respectively. The exhaust air stream has a concentration of 72 mole% for N_2. Additionally, the mole % of Toluene in the wastewater stream is 2.9 times that in the exhaust air stream, while the concentration ratio (in mole%) of Xylene in the treated water stream to the exhaust air stream is 7.5.



$\frac{43110}{28.74}$

Stream	F	W	B	T	X	N_2	O_2
1	1400		0.13		0.16	0	0
2	1500	0.01	0	0	0	0.78	0.21
3			0.08	0.015		0	0
4						0.72	

$$\begin{cases} y_{1T} = 2.9 \times y_{4T} \\ y_{3X} = 7.5 \times y_{4X} \end{cases}$$

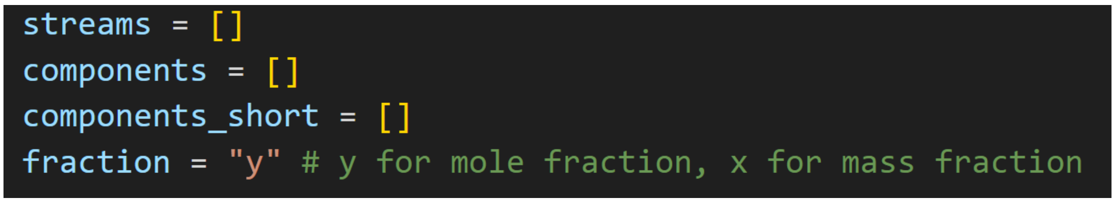
Two process relations
(parameters)

題目簡介:
計算四個管線內(stream1~4)的流率(Flow rate)、各化學物質的莫爾分率。(右上空白處)

解題思路:
利用質量守恆，進料管線的質量(F1~2)=出料管線質量(F3~4)，且各條管線內的化學物質莫爾分率總和(W,B,T,X,N2,O2)=1

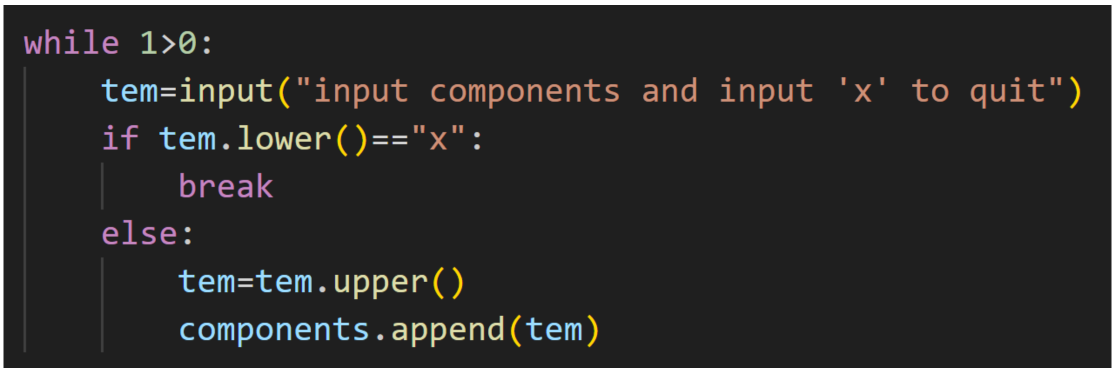
程式碼目的:
用 pandas module 創立一個表格，並將以已知的資料(題目提供)輸入，確認已知的資訊夠不夠求解(變數分析)，再自動列出所有質量守恆等式並用 sympy 求解後，再自動填入表格中，輸出一個完整求解完的 table。

我把預先給的變數像是 streams,components,process，改成全部手動輸入的形式，這樣就可以應付不同題目而不用直接動到 code
清空為預設 0 的 list:



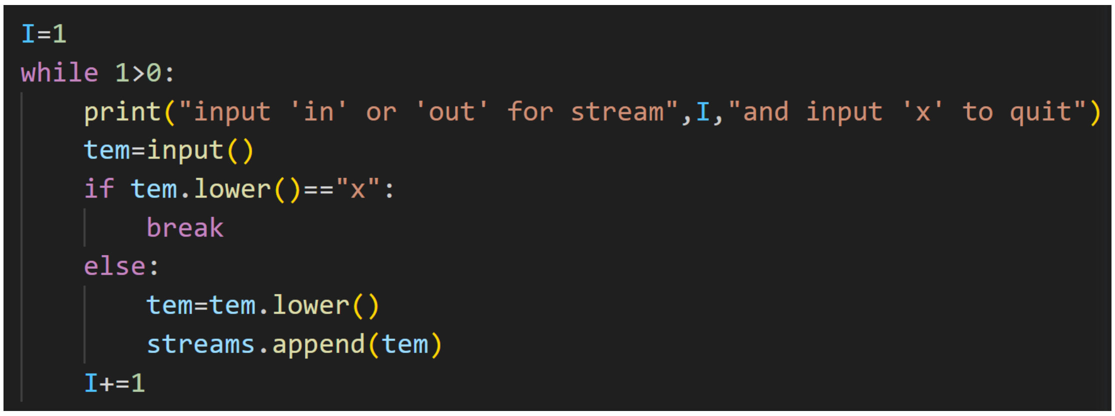
```python
streams = []
components = []
components_short = []
fraction = "y" # y for mole fraction, x for mass fraction
```

自訂 components 並抓取每個 components 字串字首的大寫形式:



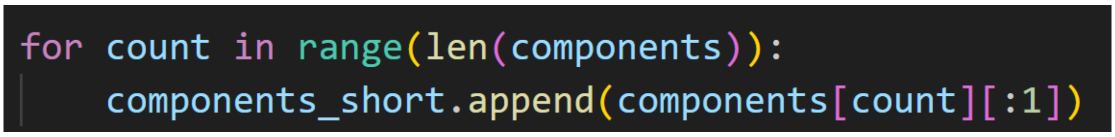
```python
while 1>0:
    tem=input("input components and input 'x' to quit")
    if tem.lower()=="x":
        break
    else:
        tem=tem.upper()
        components.append(tem)
```

自訂 stream 的 input/output:
也可以順便訂下 stream 的數量:



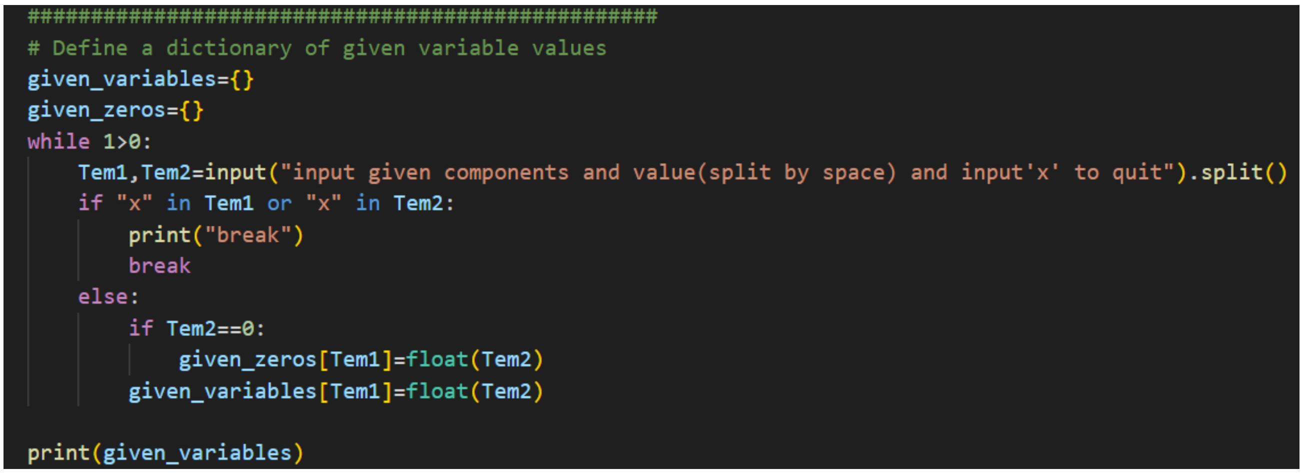
```python
I=1
while 1>0:
    print("input 'in' or 'out' for stream",I,"and input 'x' to quit")
    tem=input()
    if tem.lower()=="x":
        break
    else:
        tem=tem.lower()
        streams.append(tem)
    I+=1
```

component_short 存進抓取的大寫字首:



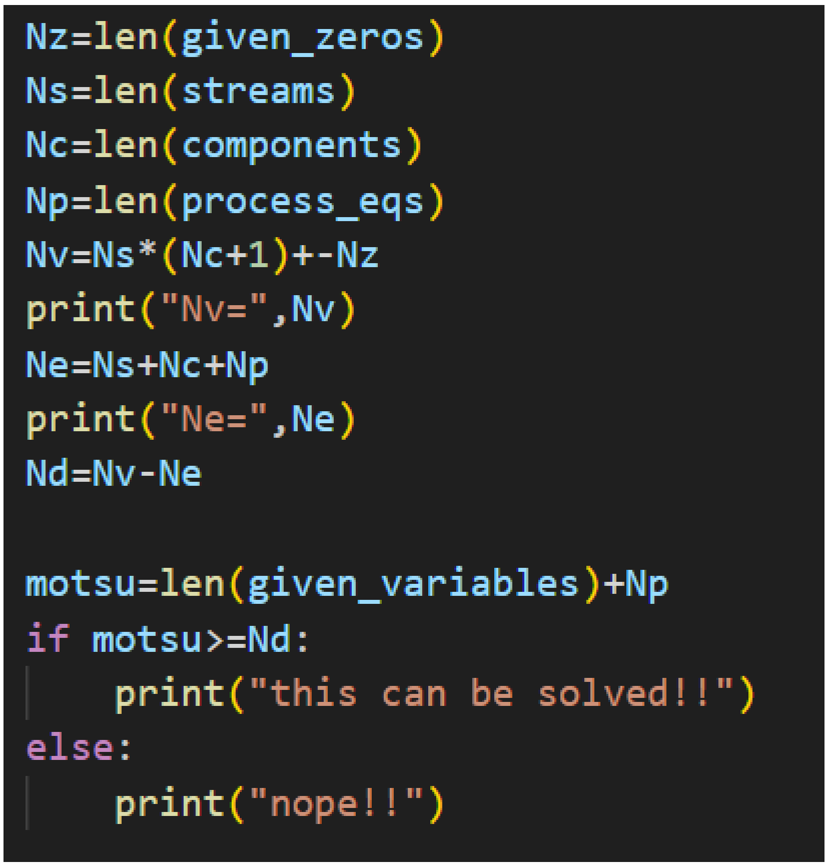
```python
for count in range(len(components)):
    components_short.append(components[count][:1])
```

自訂 given_variables，Tem1 為名稱，Tem2 為 value
如果輸入 x x 就不繼續輸入。
同時也計算 Nz 的數量



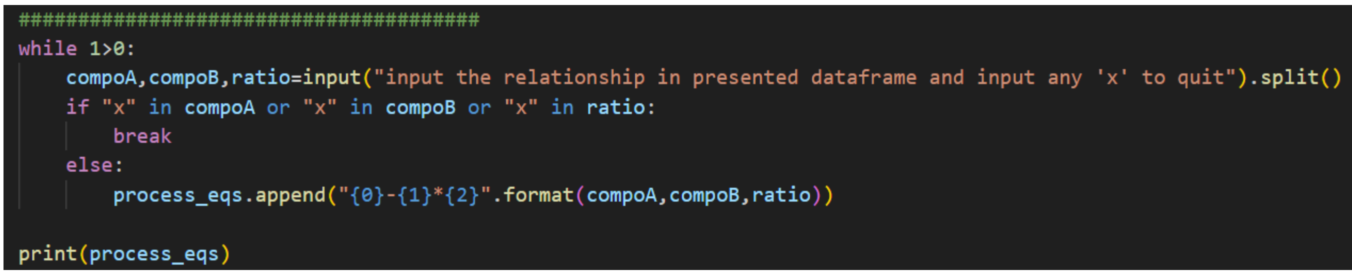
```python
##################################################
# Define a dictionary of given variable values
given_variables={}
given_zeros={}
while 1>0:
    Tem1,Tem2=input("input given components and value(split by space) and input'x' to quit").split()
    if "x" in Tem1 or "x" in Tem2:
        print("break")
        break
    else:
        if Tem2==0:
            given_zeros[Tem1]=float(Tem2)
        given_variables[Tem1]=float(Tem2)

print(given_variables)
```

Variable analysis:



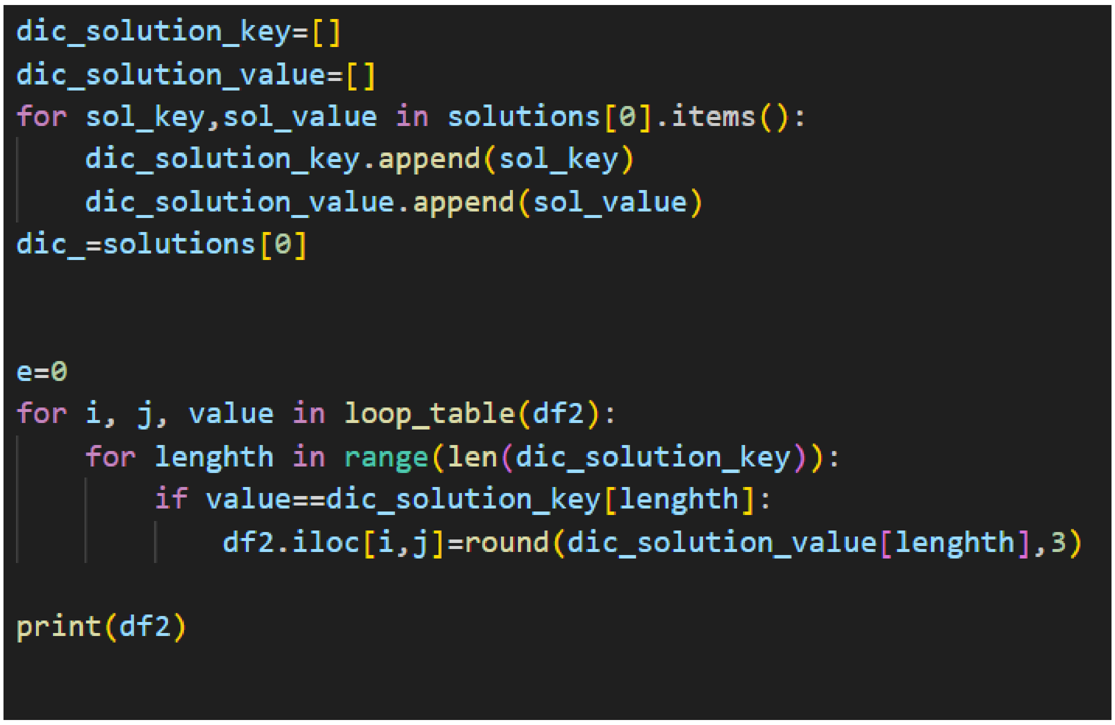
```python
Nz=len(given_zeros)
Ns=len(streams)
Nc=len(components)
Np=len(process_eqs)
Nv=Ns*(Nc+1)+-Nz
print("Nv=",Nv)
Ne=Ns+Nc+Np
print("Ne=",Ne)
Nd=Nv-Ne

motsu=len(given_variables)+Np
if motsu>=Nd:
    print("this can be solved!!")
else:
    print("nope!!")
```

自訂 relationship
預設格式為[A][空格][B][空格][比值
並轉成 equation 的形式



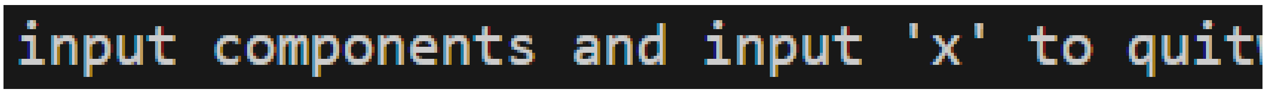
```
########################################
while 1>0:
    compoA,compoB,ratio=input("input the relationship in presented dataframe and input any 'x' to quit").split()
    if "x" in compoA or "x" in compoB or "x" in ratio:
        break
    else:
        process_eqs.append("{0}-{1}*{2}".format(compoA,compoB,ratio))

print(process_eqs)
```

因為發現 solution 在 result 裡是存在於 list 中的 dictionary
所以 solution[0]就是 dictionary 的形式，使用 item()回傳 key 跟 value
再用前面給的寫好的 loop_table()，在 dataframe 裡 search for solutions 裡有的 key，確定在 dataframe 裡的位置後，再將 solution_value 的值丟進去。



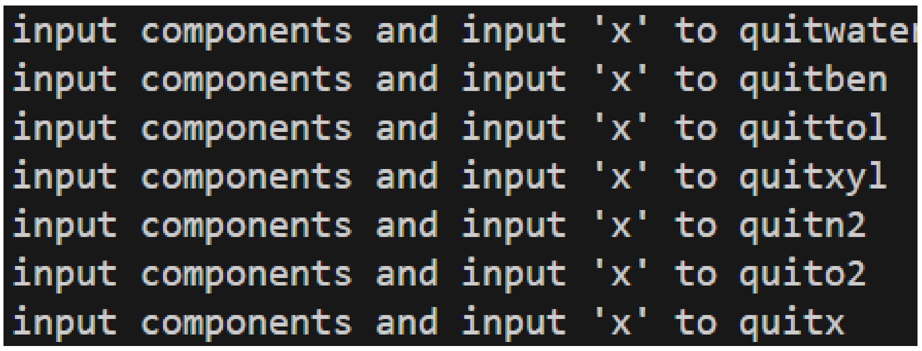
```
dic_solution_key=[]
dic_solution_value=[]
for sol_key,sol_value in solutions[0].items():
    dic_solution_key.append(sol_key)
    dic_solution_value.append(sol_value)
dic_=solutions[0]


e=0
for i, j, value in loop_table(df2):
    for lenghth in range(len(dic_solution_key)):
        if value==dic_solution_key[lenghth]:
            df2.iloc[i,j]=round(dic_solution_value[lenghth],3)

print(df2)
```

Example:



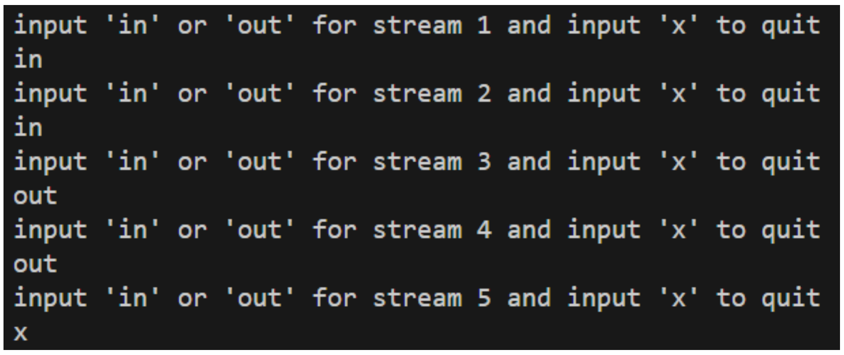
```
input components and input 'x' to quit
```

逐個輸入 components



```
input components and input 'x' to quitwater
input components and input 'x' to quitben
input components and input 'x' to quittol
input components and input 'x' to quitxyl
input components and input 'x' to quitn2
input components and input 'x' to quito2
input components and input 'x' to quitx
```

確認 1 2 3 4..stream 為 in/out

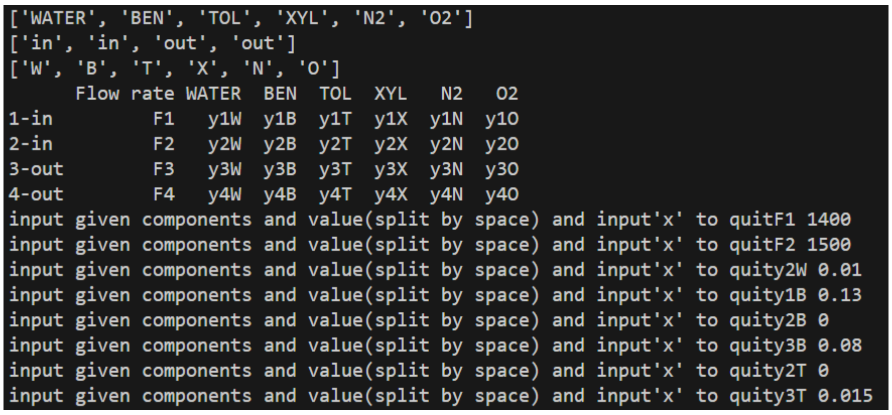


```
input 'in' or 'out' for stream 1 and input 'x' to quit
in
input 'in' or 'out' for stream 2 and input 'x' to quit
in
input 'in' or 'out' for stream 3 and input 'x' to quit
out
input 'in' or 'out' for stream 4 and input 'x' to quit
out
input 'in' or 'out' for stream 5 and input 'x' to quit
x
```

這裡我選擇先 print 一次最初的表格
讓使用者可以知道要輸入的名稱跟值
像
像 0.13 的位置是 y1B、0.16 的位置是 y1X

Stream	F	W	B	T	X	N_2	O_2
1	1400		0.13		0.16	0	0
2	1500	0.01	0	0	0	0.78	0.21
3			0.08	0.015		0	0
4						0.92	

輸入(位置)+(空格)+(value)

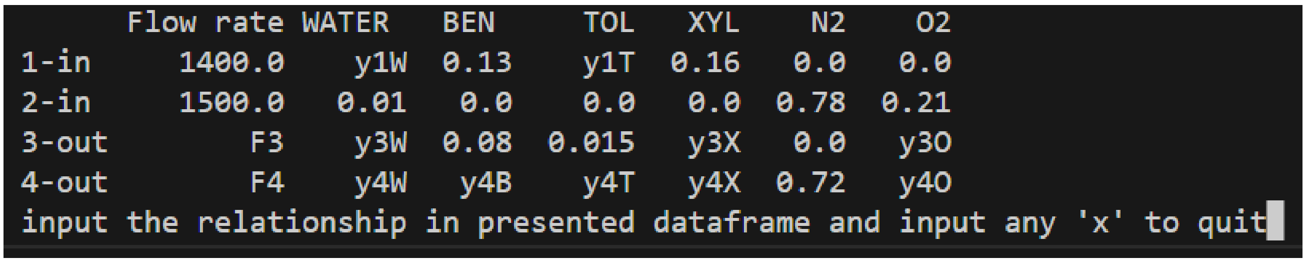


```
['WATER', 'BEN', 'TOL', 'XYL', 'N2', 'O2']
['in', 'in', 'out', 'out']
['W', 'B', 'T', 'X', 'N', 'O']
      Flow rate WATER  BEN  TOL  XYL   N2   O2
1-in         F1   y1W  y1B  y1T  y1X  y1N  y1O
2-in         F2   y2W  y2B  y2T  y2X  y2N  y2O
3-out        F3   y3W  y3B  y3T  y3X  y3N  y3O
4-out        F4   y4W  y4B  y4T  y4X  y4N  y4O
input given components and value(split by space) and input'x' to quitF1 1400
input given components and value(split by space) and input'x' to quitF2 1500
input given components and value(split by space) and input'x' to quity2W 0.01
input given components and value(split by space) and input'x' to quity1B 0.13
input given components and value(split by space) and input'x' to quity2B 0
input given components and value(split by space) and input'x' to quity3B 0.08
input given components and value(split by space) and input'x' to quity2T 0
input given components and value(split by space) and input'x' to quity3T 0.015
```

注意不繼續輸入要打 x x(2 個) 否則將導致錯誤

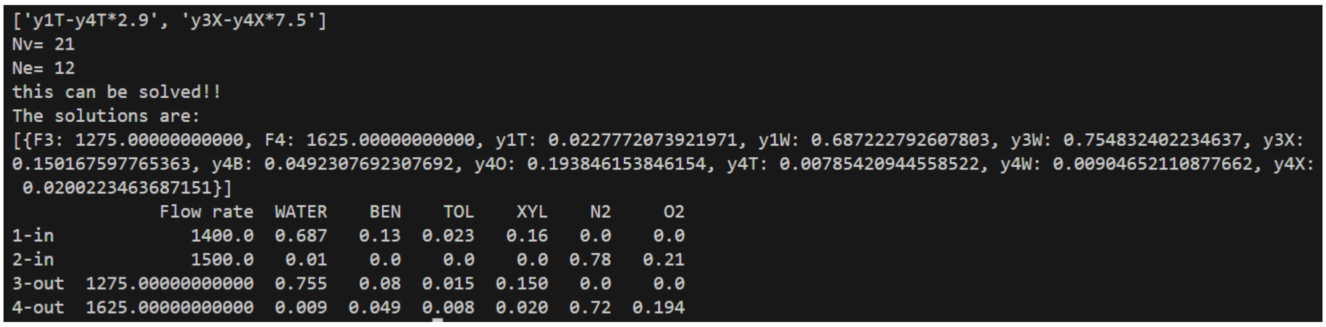
再次印出 dataframe，可以再看到需要的位置並輸入 relationship，
預設格式為:[A][B][比值]
Eg:
Y1T(space)y4T(space)2.9
Y3X(space)y4X(space)7.5 注意不繼續輸入要打 x x x(3 個) 否則將導致錯誤

$$\begin{cases} y_{1T} = 2.9 \times y_{4T} \\ y_{3X} = 7.5 \times y_{4X} \end{cases}$$



```
       Flow rate WATER   BEN     TOL   XYL    N2    O2
1-in      1400.0   y1W  0.13     y1T  0.16   0.0   0.0
2-in      1500.0  0.01   0.0     0.0   0.0  0.78  0.21
3-out         F3   y3W  0.08   0.015   y3X   0.0   y3O
4-out         F4   y4W   y4B     y4T   y4X  0.72   y4O
input the relationship in presented dataframe and input any 'x' to quit
```

求解過程略
結果:



```
['y1T-y4T*2.9', 'y3X-y4X*7.5']
Nv= 21
Ne= 12
this can be solved!!
The solutions are:
[{F3: 1275.00000000000, F4: 1625.00000000000, y1T: 0.0227772073921971, y1W: 0.687222792607803, y3W: 0.754832402234637, y3X:
0.150167597765363, y4B: 0.0492307692307692, y4O: 0.193846153846154, y4T: 0.00785420944558522, y4W: 0.00904652110877662, y4X:
 0.0200223463687151}]
              Flow rate  WATER    BEN    TOL    XYL    N2     O2
1-in             1400.0  0.687   0.13  0.023   0.16   0.0    0.0
2-in             1500.0   0.01    0.0    0.0    0.0  0.78   0.21
3-out  1275.00000000000  0.755   0.08  0.015  0.150   0.0    0.0
4-out  1625.00000000000  0.009  0.049  0.008  0.020  0.72  0.194
```

Dataframe 總變數 Nv 為 21 個，已知變數 Ne 為 12 個，剩下所需變數經過判斷後大於等於 21-12=9 個，因此此題目可求解。求解出來為各個 stream 的 flow rate 與莫爾分率。

完整 code 網址: https://github.com/aiko77777/python.git