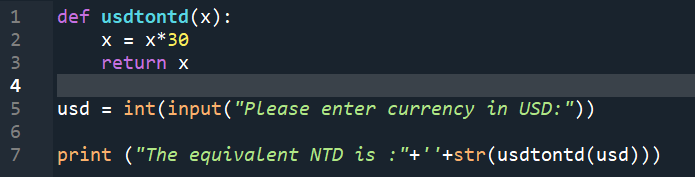
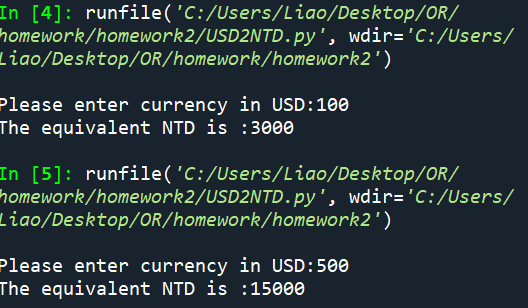
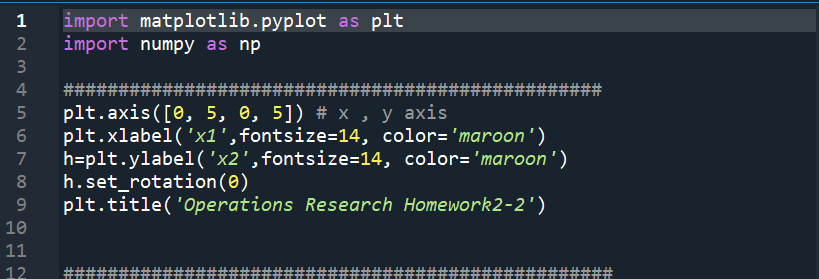
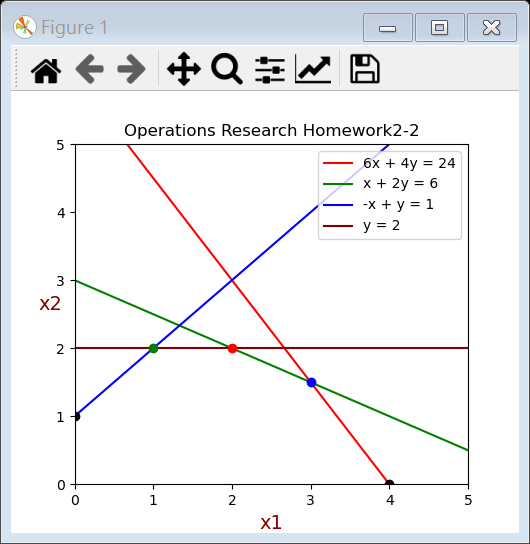
作業研究

1. USD2NTD.py

第一次寫的時候就是寫得很醜的那種，直接得到input\*30後print出來，後來改成函數的寫法，只有看起來好看一點，實際上是一樣的，遇到錯誤的input等等的時候沒有辦法做出回應，也沒有做好擴充的可能，不過題目沒有要求，也就讓它這樣啦。

1.  Graphical.py

1~2行在import基本的module

5~9行在設定圖表的基本參數

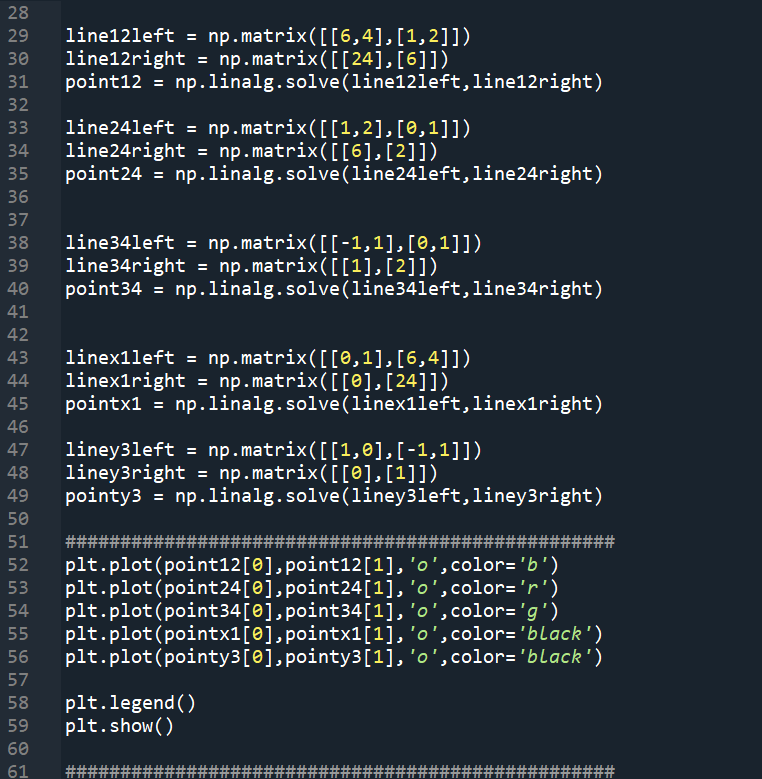
13~15行在圖上畫出紅線

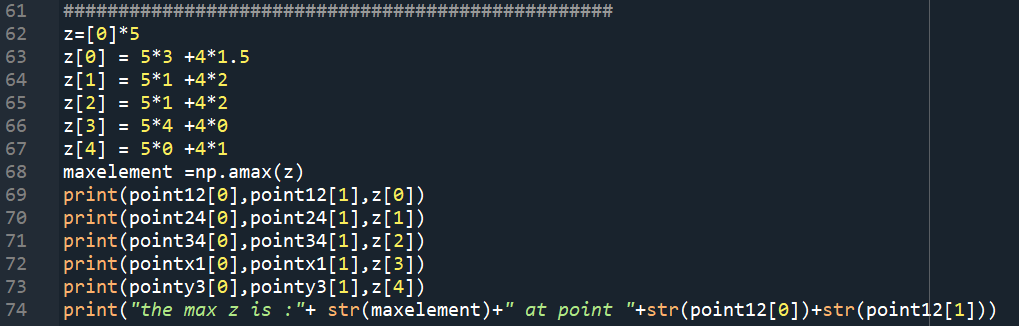
17~19行在圖上畫出綠線

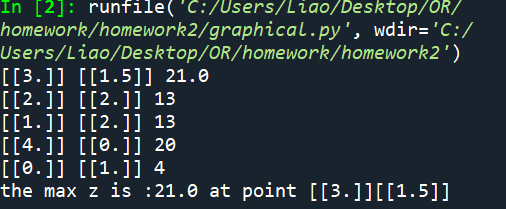
21~23行在圖上畫出藍線

26在圖上畫出棕線



29~49行在圖上看出相交的可行解區域的點，再分別用矩陣運算把點解出後，52~59行把五個點標示在圖上

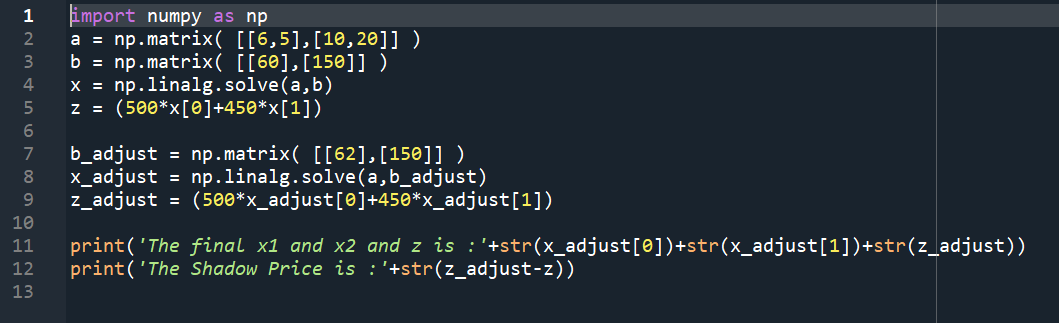


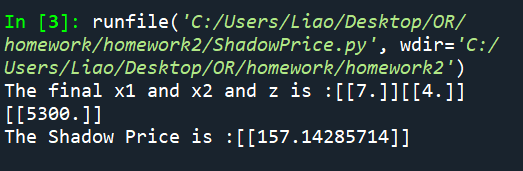


最後把五個點帶入方程式得到解，並取最大值後得到最佳解在(3,1.5)點為

21。

先承認這個程式寫得非常醜，寫得像是把手寫計算過程寫出來而已，完全沒有用到程式的自動化或是幫我們判斷的優點，不過也是因為題目沒有要求要能夠通用在各種情況、不是接收input之後再自動判斷自動計算，所以我也沒有寫函式、判斷式等等較複雜一點的程式，而是像流水帳一樣紀錄運算過程而已。

1. ShadowPrice.py

2~5行在計算未更改的原點的z，7~9行計算更改後的z，之後印出答案

得出的答案也符合理論，即更改為2倍，其shadow price也更改為兩倍，即(78+4/7)的兩倍。

這裡比較像是在驗證理論，比較沒什麼特別的地方。

整個作業算是非常佛心，很簡單，不過我沒有學過python，只有C++的一些基礎，在寫作業前花了很多時間自學一下python，好險大部分程式的邏輯都相同很好上手，只有寫法有些不同，像是array ,matrix的寫法用法就不太一樣，花了一些時間熟悉。