1.

Bob 這學期有選修 python 程式語言入門、線性代數、常微分方程以及邏輯設計實驗。Bob 是位勤奮向上的好學生，所以他花了許多的時間在學習上面，我們假設 Bob「每天」花在 python 的時間為 a 小時，線性代數 b 小時、常微分方程 c 小時以及邏輯設計實驗 d 小時:

1.Bob 希望他每天的讀書時間要超過 5 小時(a+b+c+d>=5)

2.花在常微分方程的時間要比邏設實驗多(b>=c)

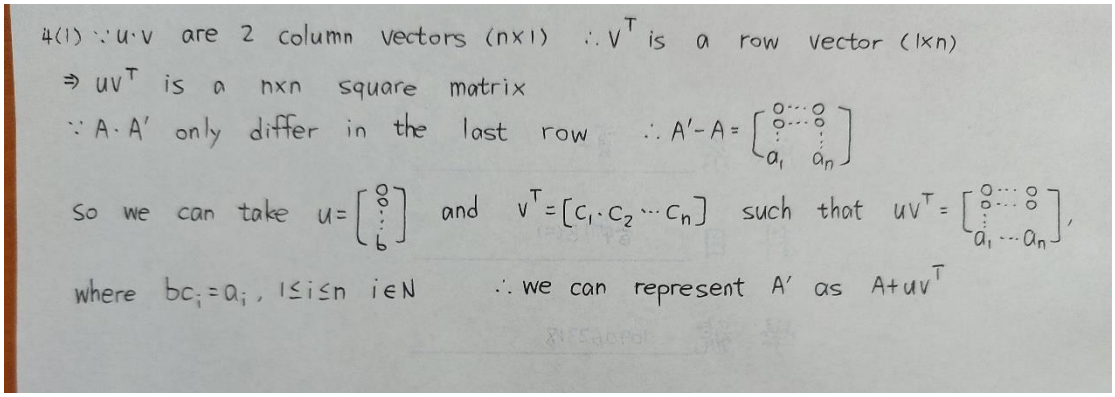
3.因為 Bob 數學不錯，他不需要花太多時間在數學上面，反而想多花時間練習 coding，所以花在 coding 的時間要比算數學的時間還多(a+d>=b+c)

4.又因為邏設實驗對 Bob 來說太困難了，所以他要花的時間比花在 python 上的時間還要多至少 3 倍(d>=3a)

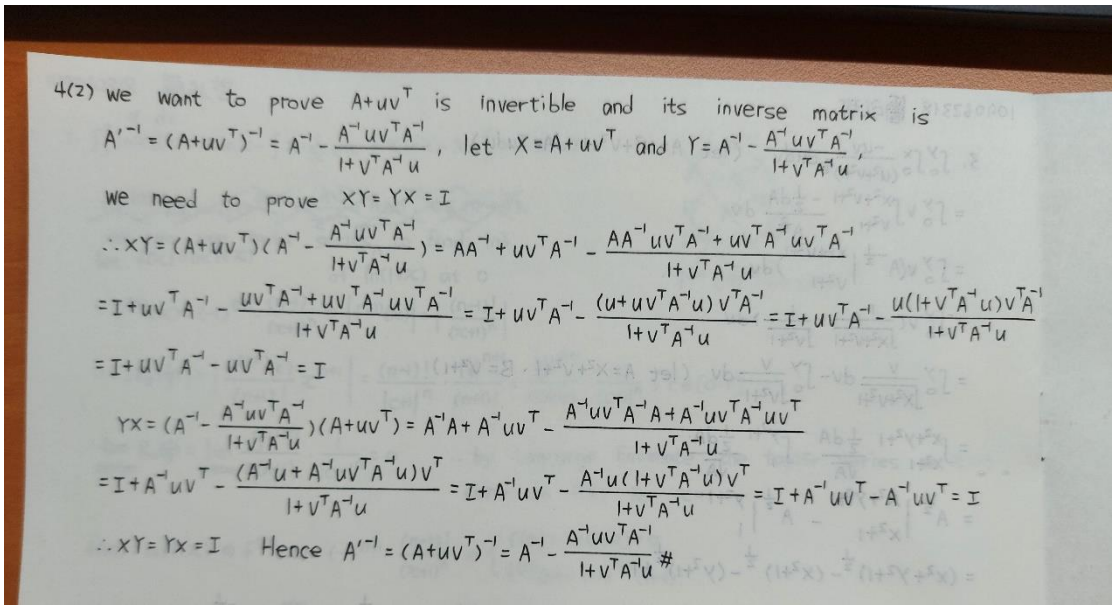
5.最後 Bob 希望他每週的讀書時間至少要 100 小時，已知他一周內有 5 天會讀 python、3 天讀線代、2 天讀常微分方程以及 3 天讀邏設實驗 (5a+3b+2c+3d>=100)

Bob 想知道 a+2b+3c+4d 的最大值為何?

4(1)



4(2)



4(2) we want to prove $A+uv^T$ is invertible and its inverse matrix is $A'^{-1} = (A+uv^T)^{-1} = A^{-1} - \frac{A^{-1}uv^TA^{-1}}{1+v^TA^{-1}u}$, let $X = A+uv^T$ and $Y = A^{-1} - \frac{A^{-1}uv^TA^{-1}}{1+v^TA^{-1}u}$,

we need to prove $XY = YX = I$

$\therefore XY = (A+uv^T)(A^{-1} - \frac{A^{-1}uv^TA^{-1}}{1+v^TA^{-1}u}) = AA^{-1} + uv^TA^{-1} - \frac{AA^{-1}uv^TA^{-1} + uv^TA^{-1}uv^TA^{-1}}{1+v^TA^{-1}u}$

$= I + uv^TA^{-1} - \frac{uv^TA^{-1} + uv^TA^{-1}uv^TA^{-1}}{1+v^TA^{-1}u} = I + uv^TA^{-1} - \frac{(u+uv^TA^{-1}u)v^TA^{-1}}{1+v^TA^{-1}u} = I + uv^TA^{-1} - \frac{u(1+v^TA^{-1}u)v^TA^{-1}}{1+v^TA^{-1}u}$

$= I + uv^TA^{-1} - uv^TA^{-1} = I$

$YX = (A^{-1} - \frac{A^{-1}uv^TA^{-1}}{1+v^TA^{-1}u})(A+uv^T) = A^{-1}A + A^{-1}uv^T - \frac{A^{-1}uv^TA^{-1}A + A^{-1}uv^TA^{-1}uv^T}{1+v^TA^{-1}u}$

$= I + A^{-1}uv^T - \frac{(A^{-1}u + A^{-1}uv^TA^{-1}u)v^T}{1+v^TA^{-1}u} = I + A^{-1}uv^T - \frac{A^{-1}u(1+v^TA^{-1}u)v^T}{1+v^TA^{-1}u} = I + A^{-1}uv^T - A^{-1}uv^T = I$

$\therefore XY = YX = I$ Hence $A'^{-1} = (A+uv^T)^{-1} = A^{-1} - \frac{A^{-1}uv^TA^{-1}}{1+v^TA^{-1}u}$ #

5.

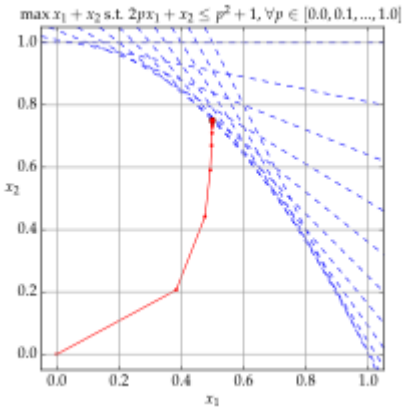
Source: https://en.wikipedia.org/wiki/Karmarkar%27s_algorithm

I found a algorithm that doesn't enumerate all the intersection points.It's Karmarkar's algorithm, introduced by Narendra Karmarkar in 1984 for solving linear programming problems.It can solve problem in polynomial time.Although there exists other polynomial solution,it's inefficient in practical problems.

Denoting n as the number of variables and L as the number of bits of input to the algorithm, Karmarkar's algorithm requires O((n^3.5) * (L^2) * logL * loglogL) by using Fast Fourier Transfer algorithm.

This algorithm doesn't find the optimal answer alone the boundary of feastible set On the other hand,it starts from the interior region and uses a definite fraction to enhance the approximation of the optimal answer on every iteration.
Finally converges to an optimal solution.

(picture source: https://en.wikipedia.org/wiki/Karmarkar%27s_algorithm)



Where blue lines are the constraints,red represent each iteration