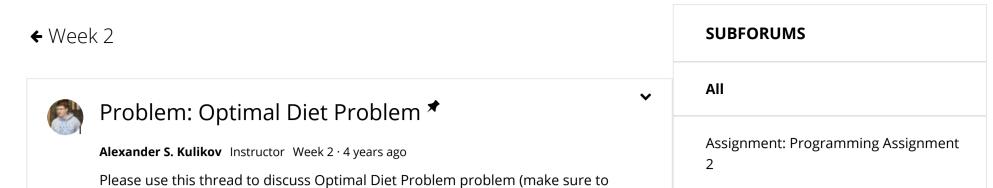
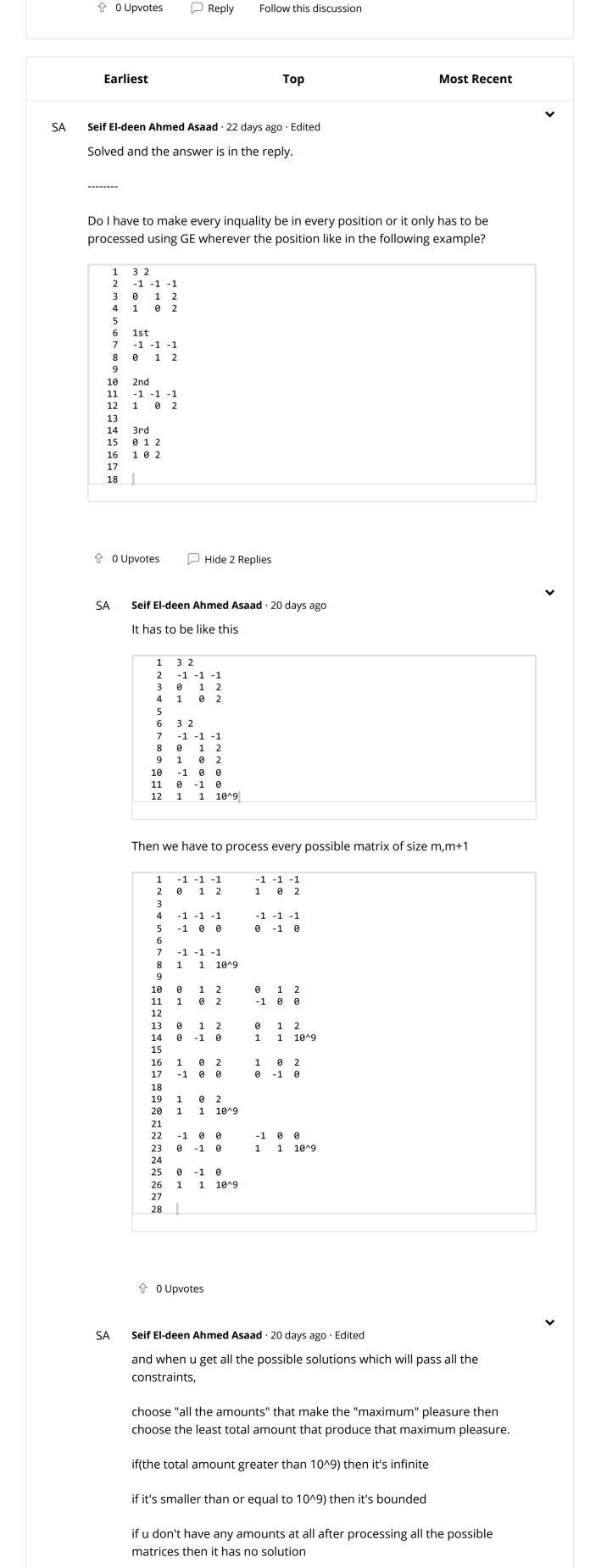
review forum rules before posting).

## Week 2





 ↑ 1 Upvote Reply Reply **Da Shi** · a month ago Finally passed this problem with the "what to do" part. Got stuck on implementing a test for "Infinity" result before realizing that the "what to do" part continues to page 9!!! Reply û Upvotes **Mostafa Nassar** · 3 months ago Hello all, I have found that this links may help to solve those two problems with the same code. The Big M Method: https://www.youtube.com/watch?v=upgpVkAkFkQ Online Calculator: Simplex Method: https://linprog.com/en/main-simplex-method https://cbom.atozmath.com/CBOM/Simplex.aspx?q=sm Optimal Diet Problem c++ (Max time used: 0.00/2.00, max memory used: 9904128/536870912.) Online Advertisement Allocation c++ (Max time used: 0.03/1.00, max memory used: 10842112/536870912.) Reply ↑ 1 Upvote Sanchit Khandelwal · 4 months ago The lectures are not at all helpful and unnecessary complicated. Better go with you tube lectures for linear programming and gaussian eliminations. Reply ~ **Greg G.**  $\cdot$  7 months ago  $\cdot$  Edited Brute force solution in Java: 1 Good job! (Max time used: 2.59/4.00, max memory used: 151121920/536870912.) Unfortunately Dan didn't explain the simplex algorithm properly and it also looked quite complex so I went with the brute force method explained in the What to Do section. Actually, after reusing the Gauss solver from problem 1 (extending it with a "no solution" null return value) it was not complicated to implement the algorithm. The hardest part was actually generating all the subsets, after running out of time limits I needed to look into it - 90% of the time turned out to be subset generation. (Netbeans has an easy to use profiler!) The naive recursive algorithm of powerset generation was too slow. Because if you have, say n+m=32 equations but only 15 variables then the recursive algorithm generates all 2^32 subsets, and adding this many elements to HashSets is too slow. I went with a bitmasking solution <u>like this</u>, since by using a "long" integer, we can represent all subsets with 1s and 0s. Bitwise operations are fast, and I just extended it to check the number of 1s in the bitmask to see if it's equal to our required subset size before going into any set operations. ↑ 1 Upvote Reply **Tova Meystel** · 9 months ago Failing Case #120, the program I am checking against my answer has the same as mine- does anyone have a clue why I would be getting such a different answer?? Input: 6 3 90 61 70 29 -14 41 74 -76 -58 -57 34 2 -71 -6 -97 5 58 -8 8909 1051 -1158 -940 -4919 3610 -72 -91 -16 23.612383375742155000 59.00000000000000000 9.00000000000005300 ↑ 0 Upvotes ☐ Hide 2 Replies TM **Tova Meystel** · 9 months ago I want to add that I also used the python testing script that was added to this group and I passed all of those. û Upvotes

