שאלה 1:

ניתן למצוא את הקוד ב- github שלי בקישור:

https://github.com/Andrey-Ba/Egalitarian-Division

יצרתי מחלקה בשם EgalitarianDivision בפייתון הפותרת את הבעיה.

דוגמה ל- main:

```
from EgalitarianDivision import EgalitarianDivision

mat = [[81, 19, 1],
       [70, 1, 29],
       [98, 1, 1]]

prob = EgalitarianDivision(mat)
prob.run()
```

לקבלת הפלט:

Agent #1 gets 0.318 of resource #1, 1.0 of resource #2, -0.0 of resource #3.

Agent #2 gets 0.225 of resource #1, -0.0 of resource #2, 1.0 of resource #3.

Agent #3 gets 0.457 of resource #1, -0.0 of resource #2, -0.0 of resource #3.

תחילת הקובץ:

```
import cvxpy


class EgalitarianDivision:
    def __init__(self, value_matrix=[]):
        self.preference_table = value_matrix

    def init(self, value_matrix):
        self.preference_table = value_matrix
```

בנייה רגילה של מחלקה.

פונקציית הריצה:

```
def run(self):
    number_of_resources = len(self.preference_table[0])
    number_of_players = len(self.preference_table)
    # Create for each player variables for each resource.
    variables = cvxpy.Variable(number_of_players * number_of_resources)
    min_util = cvxpy.Variable()
    constraint_list = []

    # Add the constraints that a player cannot get less than the minimum util
    # As well as that each variable will not be negative
    for i in range(number_of_players):
        utility_of_player_i = 0
        for j in range(number_of_resources):
            utility_of_player_i = utility_of_player_i +
self.preference_table[i][j] * variables[
                i * number_of_resources + j]
            constraint_list.append(variables[i * number_of_resources + j] >=
0)
        constraint_list.append(utility_of_player_i >= min_util)

    # Add the constraints that that players will take the whole resource.
    for j in range(number_of_resources):
        constraint_for_resource_j = 0
        for i in range(number_of_players):
            constraint_for_resource_j = constraint_for_resource_j +
variables[i * number_of_resources + j]
        constraint_list.append(constraint_for_resource_j == 1)

    problem = cvxpy.Problem(cvxpy.Maximize(min_util),
constraints=constraint_list)
    problem.solve()

    # Print in the given format.
    for i in range(number_of_players):
        print(f"Agent #{i + 1} gets", end=' ')
        for j in range(number_of_resources-1):
            print(f"{round(variables[i * number_of_resources + j].value, 3)}
of resource #{j + 1},", end=" ")
        print(f"{round(variables[i * number_of_resources +
number_of_resources - 1].value, 3)} of resource #{number_of_resources}.")
```

אשר מגדירה לכל שחקן משתנים לכל משאב, משתנה מינימום ורשימת אילוצים. תחילה היא מכניסה את האילוצים לכך שכל שחקן יקבל יותר ממשתנה המינימום ושכל משתנה יהיה חיובי.

לאחר מכן הפונקציה מכניסה לרשימת האילוצים לכל משאב שהסכום של כל השחקנים הלוקחים אותו יהיה 1 (כלומר לקחו הכל).

לבסוף מריצה את הפונקציה עם המשתנים ומוצאת את ערכיהם.

כמו כן, מדפיסה את התוצאות לפי הפורמט הנתון.