AI Python – Software Engineer – Homework

Please solve the problem below using the Python libraries you deem adapted to the resolution. Send your solution in a Jupyter Notebook with descriptions of the different coding steps that can give us a generic view of your understanding of the topic.

Please try to make your solution readable, modular and well coded. Delivering a good implementation following python software engineering best practices is more important than delivering very complex algorithms.

Desk allocation problem (4 to 7 hours)

A company has a fixed amount D of desks spread over an office space and has N employees. D<N as the company relies on absences and work from home and considers that every day the amount n of employees working from the office verifies n<=D, so that everyone shall get a desk.

In order to maximize productivity, the company wants to maximize the collocation of employees from the same team.

The goal is to build a function returning a desk number to each employee as he enters the office. No employee has a fixed desk and the desk assigned can be different every day, anyhow, no one should change of desk during the day, the desk is assigned for the whole day.

For simplicity the office space can be represented by a P x Q matrix on which -1 represents a place that cannot be occupied (kitchen, corridor, hall etc.) and 0 a free desk.

For example

-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	0	0	-1	0	0	0	0	-1	-1
-1	0	0	-1	0	0	0	0	-1	-1
-1	0	0	-1	-1	-1	-1	-1	0	0
-1	-1	-1	-1	0	0	0	-1	0	0

Each employee has a unique employee_ID and belongs to a single team. Pairs employee_ID, team_ID are available in the file “employees”.

The objective is to collocate employees belonging to the same team.

Questions:

1) Write a python function random_office(p,q,d) returning an office of dimension P x Q with d desks placed randomly. (you need to choose the way you want to represent the office and the way you want to place the desks).

2) Imagine metrics that could reflect the level of "collocation" of a set of employees in an office. Implement the pertaining function(s) collocation_score(office, employees)
    a. Where the input argument 'office' is a representation of the office 'filled' with employee_IDs. The input argument 'employees' is the list of pairs employee_ID, team_ID.
    b. and returning a value between 0 and 1 (1 being the best possible collocation score).

   You can create multiple collocation_score functions using multiple metrics if you want.

3) Create a function assign_desk(office, employees, d) returning an 'office' filled with d employees arriving randomly at the office. The decision of desk assignment must be done at the exact moment when the employee arrives at the office and cannot be differed. We don't know who from the employees will really join the office that day.

4) Create multiple random offices of 500 desks using the function random_office(15, 55, 500) created in question 1 and implement an evaluation of your function assign_desk (question 3) using metrics implemented in question 2 and using the employees list provided as attachment and the assumption that 490 random employees will go the office.

We do not expect a perfect work with very elaborated algorithms but an application of standard principles and a robust and understandable implementation. The questions related to the evaluation are as important as the main questions and shall not be neglected.

The 4 questions are there to guide you but you are free to solve the problem the way you want. If you decide to not answer the questions but propose another way to solve this problem this is perfectly fine, just mention it in the comments.

Good luck!