IE400 PRINCIPLES OF ENGINEERING MANAGEMENT
Project Description **Deadline: 11.05.25, 23:59 pm**

There are three dataset files available in Moodle: "**seekers.csv**", "**jobs.csv**" and "**location_distances.csv**".

JobSync aims to improve the job matching process. Data has been collected from job seekers (`seekers.csv`) and for various job openings (`jobs.csv`). The goal is to use optimization to facilitate effective matches. Job openings have associated priority weights (w_j) indicating their importance and a number of available positions (P_j). Both seekers and jobs have associated responses to a 20-question survey.

Data Field Descriptions

Seekers Data (`seekers.csv`): Contains information about each job seeker. Key fields include:

- **Seeker_ID**: Unique identifier.
- **Location**: Current location ('A' through 'F').
- **Desired_Job_Type**: Preferred type (e.g., 'Full-time').
- **Min_Desired_Salary**: Minimum acceptable annual salary.
- **Max_Commute_Distance**: Maximum acceptable commute distance (km).
- **Skills**: List of skills possessed (stored as a string representation, e.g., "['Python', 'SQL']").
- **Experience_Level**: Seeker's experience (e.g., 'Entry-level', 'Mid-level').
- **Questionnaire**: List of 20 responses (0-5) (stored as a string representation).

Jobs Data (`jobs.csv`): Contains information about each job opening. Key fields include:

- **Job_ID**: Unique identifier.
- **Location**: Nominal location ('A' through 'F').
- **Is_Remote**: 1 if the job is remote, 0 otherwise.
- **Job_Type**: Type of employment offered ('Full-time', 'Part-time', 'Contract','Internship').
- **Salary_Range_Min**, **Salary_Range_Max**: Offered annual salary range.
- **Required_Skills**: List of essential skills (stored as a string representation).
- **Required_Experience_Level**: Minimum required experience.
- **Num_Positions** (P_j): Number of available positions (≥ 1).
- **Priority_Weight** (w_j): Importance of filling the job (1-10).
- **Questionnaire**: List of 20 responses (0-5) reflecting job/culture aspects (stored as a string representation).

Note: You will need to parse the string representations of lists (for Skills and Questionnaire) back into actual lists in your code (e.g., using 'ast.literal_eval').

Location Distances(`location_distances.csv`): Stores the location matrix for the cities A, B, C, D, E, F. You can use *pandas.read_csv('location_distances.csv', index_col = 0)* to read it properly.

Project Tasks

Develop and solve two Integer Linear Programming (ILP) models using Gurobi.

Part 1: Maximize Priority-Weighted Matches

Develop an ILP model to assign job seekers to job openings to maximize the total

sum of priority weights (w_j) of the filled positions. The assignment must adhere to these rules:

- Each job seeker can be assigned to at most one job opening.
- The number of seekers assigned to a job opening cannot exceed its available positions (P_j).
- A seeker can only be considered for a job if they meet **all** fundamental requirements:
  a) *Job Type:* Seeker's desired type is compatible with the job's type.
  b) *Salary:* Seeker's minimum desired salary is suitable for the job's salary offering(explain your logic).
  c) *Skills:* Seeker possesses all skills required by the job.
  d) *Experience:* Seeker's experience level meets or exceeds the job's requirement (assume order: Entry $<$ Mid $<$ Senior $<$ Lead $<$ Manager).
  e) *Location:* The job is remote OR the distance between the seeker's location and the job's nominal location is within the seeker's maximum commute distance.

Implement and solve this ILP using Gurobi. Determine the maximum feasible total priority weight (M_w).

Part 2: Minimize Maximum Dissimilarity

First, define a dissimilarity score (d_{ij}) between a potentially compatible seeker i and job j based on their questionnaire responses ($q_{i,k}$ and $q_{j,k}$). Calculate it as the average absolute difference across the 20 questions:

$$d_{ij} = \frac{\sum_{k=1}^{20} |q_{i,k} - q_{j,k}|}{20} \quad \in [0, 5]$$

Develop a second ILP model to find an assignment that minimizes the *worst* (maximum) dissimilarity score (d_{ij}) among all the pairs that are actually matched. This model must ensure:

- All rules from Part 1 (seeker assignment limit, job capacity, fundamental requirements) are still followed.
- The total priority weight achieved by the matches in this model is at least ω% of the maximum value M_w found in Part 1. (ω is a parameter).

Implement and solve this ILP using Gurobi. Experiment with the following ω values $\{70, 75, 80, 85, 90, 95, 100\}$. Graph your findings. Analyze how the minimum achievable maximum dissimilarity changes with ω. Select a suitable value for ω that represents a good trade-off, and justify your choice.

Requirements Deliverables

- You will need an academic license for Gurobi. Ask for help early if you have setup issues.
- Write a report (PDF) detailing your approach:

  - Explain your chosen decision variables for each model.
  - Provide the complete mathematical formulations (objective functions, constraints) you derived and implemented for both ILP models.
  - Describe how you implemented the requirement checks and the location distance rule.
  - It is possible to implement some constraints easier by using Python functionalities. You can even write a very small linear model by doing some pre calculations or using the if else statements in Python. However, try to use these minimally (e.g. dont come up with a very minimal Linear model, otherwise you may lose points)
  - Present your results: M_w (Part 1), analysis of the ω experimentation (Part 2), and your final choice of ω with justification.
  - Include comments on the problem, models, and results.
  - **Do not include source code in the report.**

- Zip your report (PDF) with your well-commented Python codes (.py) and any necessary data files (like your location distance matrix). Name it "**IE400_Project_GroupXX.zip**" (replace XX with your group number). Make your python code understandable and easy to run.

- You will be demonstrating your codes and discussing your report in-class in the last week of lectures. Please be prepared.