Committee Assignment with Excel Data: README

1 Introduction

This document provides a step-by-step guide for using the Python script that

1. Reads an Excel file containing students and their committee ratings,
2. Solves the assignment problem (maximizing total rating, respecting minimum/maximum committee size constraints),
3. Exports the result to a new Excel file that details each committee's assigned students.

The solution is formulated as an Integer Linear Program (ILP) using the `PuLP` library (https://github.com/coin-or/pulp). The default solver is CBC, provided with PuLP. For a typical problem size (e.g., 100 students and 10 committees), the solution often completes in well under a second on a typical laptop.

2 Requirements

- **Python 3.7+** (or later)
- **pip** (Python package manager)

3 Installation

1. Clone or download this repository (containing the Python script).
2. Install required Python libraries:

   ```
   pip install pulp pandas openpyxl
   ```

3. `pulp` is used for ILP solving, `pandas` for reading/writing Excel, and `openpyxl` for `.xlsx` support.

4 Data Format (Excel)

Your **input Excel file** should have:

- **Column A**: Student names (or IDs), one row per student.
- **Columns B..M**: Numerical ratings for each committee. If there are m committees, you should have m columns of ratings after the first name column.

An example with 3 committees:

Name	Comm0	Comm1	Comm2
Alice	10	3	5
Bob	2	9	1
Carol	7	4	10

5 How to Use

5.1 Step 1: Prepare the Excel File

1. Create (or update) a file named, for example, `students_committees.xlsx` with the described format.
2. Make sure there are no extra blank rows or header rows that might break the parsing.

5.2 Step 2: Configure min/max Constraints

Open `committee_assignment_excel_final.py` (or similarly named script) and look for the following lines in the `if __name__ == "__main__":` block:

```
# Example: a single global min=2, max=10 for all committees
global_min_value = 2
global_max_value = 10

# Or define per-committee arrays...
min_sizes = None
max_sizes = None
```

You can set constraints in three ways:

1. **Global constraints for all committees:**
   Set `global_min_value` and `global_max_value` to your desired integers (e.g. 2 and 10). Then keep `min_sizes = None` and `max_sizes = None`.
2. **Per-committee constraints:**
   Let `min_sizes = [m1, m2, ..., m_j]` and `max_sizes = [M1, M2, ..., M_j]`, one entry per committee, and set `global_min_value = None` and `global_max_value = None`.

3. **Combine both:**
   If you pass both global and per-committee values, the code merges them:

$$\text{final\_min}[j] = \max(\text{min\_sizes}[j], \text{global\_min\_value}),$$

$$\text{final\_max}[j] = \min(\text{max\_sizes}[j], \text{global\_max\_value}).$$

5.3 Step 3: Run the Script

After configuring the constraints, run:

```
python committee_assignment_excel_final.py
```

- The script reads the file (e.g. `students_committees.xlsx`) and sets up an ILP to maximize the sum of ratings.
- It solves the assignment using the default CBC solver.
- By default, it writes the results to `results.xlsx`.

You can change file names by editing:

```
excel_input = "students_committees.xlsx"
excel_output = "results.xlsx"
```

5.4 Step 4: Inspect the Output

The script creates `results.xlsx` with a single sheet named *"Assignments"*. Each committee is placed in its own column. Within each column:

- Row 1: `Committee j`
- Row 2: `Total rating: X`
- Row 3: `Number assigned: Y`
- Rows 4+: The names of the assigned students

If committees differ in number of assigned students, extra cells will remain blank. For instance:

Committee_0	**Committee_1**	**Committee_2**
Committee 0	Committee 1	Committee 2
Total rating: 15	Total rating: 22	Total rating: 18
Number assigned: 2	Number assigned: 3	Number assigned: 1
Alice	Bob	Carol
David	Eve	
	Frank	

6 Customization

- **Alternative Solvers**: If you have Gurobi, CPLEX, or GLPK, you can replace `prob.solve()` in the code with e.g. `prob.solve(pulp.GUROBI_CMD())`.
- **Additional Constraints**: You can add new constraints (e.g., a specific student cannot join certain committees) by editing the `solve_committee_assignment` function.
- **Output Format**: If you want each committee on a separate sheet or a different layout, modify the section in `solve_from_excel` that exports the `results.xlsx`.

7 FAQ

Q: Why might I see `"No optimal solution found!"`?

- If the sum of minimum constraints exceeds the number of students, or if the sum of maximum constraints is less than the number of students, the problem is infeasible.
- Alternatively, the solver might fail or be missing. Confirm that `pulp` and `CBC` are installed correctly.

Q: How fast does it run?

- For about 100 students and 10 committees, the default CBC solver typically finds an optimal solution in less than a second.

Q: Can I read from CSV instead of Excel?

- Yes, replace `pd.read_excel(...)` with `pd.read_csv(...)`. Make sure to adjust for any header rows or column differences.