Mini Project — Build and Solve Your Own Optimization Model

What this mini project is about

In this mini project, **you will design, model, and solve your own optimization problem**. You can choose one of the proposed datasets, create your own dataset, or find a small dataset online (for example, from Kaggle or another open data source).

Your goal is to:

- Build a clear mathematical model (variables, objective, constraints).
- Implement it in Python using `pandas` and `scipy.optimize.linprog`.
- Use tools from Lessons 1–8 (variables, loops, functions, lists/dictionaries, if/else, plotting).
- Explain what your optimal solution means in simple words.

1 Your Task and Learning Goals

By the end of this mini project, **you are expected to be able to**:

- Build a linear optimization model from a table of data.
- Clearly define decision variables, an objective function, and constraints.
- Move from a math model to Python code that uses `linprog`.
- Work with data in Excel using `pandas`.
- Use functions, loops, and decisions to explore your problem.
- Create a simple plot that shows your optimal plan.

Important: This is your chance to put all the pieces together from Lessons 1–8. You are not just copying a worked example anymore. You are building and solving your own model.

2 Choosing a Dataset

You have three options for your data:

1. **Use one of the proposed datasets (A–D).**
   These are small Excel files with realistic numbers and several attributes.
2. **Create your own dataset in Excel.**
   You should have at least 4–5 options (rows) and 4–5 numerical attributes (columns).

3. **Use a public dataset (for example, from Kaggle).**
   If you do this, add the link to the dataset in your project, and make sure you only use the columns that allow you to build a linear model.

Your dataset (whichever you pick) must allow you to:

- Define a clear decision for each option ("how many" or "how much").
- Have at least one attribute that becomes your **objective**.
- Have at least three attributes that become **constraints**.
- Include at least one row or set of values that gives you **total availability or minimum requirements** for your constraints.

> **Tip:** If you design your own dataset, keep it simple but meaningful. Think of a small planning or allocation problem: time, money, calories, distance, risk, etc. You may also add a "LIMITS" or "REQUIREMENTS" row in your Excel file with total amounts.

3 Proposed Datasets (Optional)

Below are four example datasets you may use. They are only suggestions. You are free to design or choose something else, as long as you can build a valid model.

In each Excel file, the last row contains summary information such as **available resources** (maximums) or **minimum requirements**. These values exist so that, no matter which objective and constraints you choose, you still have reasonable upper or lower bounds to work with. The modeling ideas given below are **only examples**, not requirements.

Dataset A: School Club Budget Allocation

You are planning activities for a school club. Each activity costs money, needs volunteer hours, and requires some staff. You want to design a plan that makes sense for your goals while staying within the available resources.

Columns (Excel file):

- `activity` – name of the activity
- `cost_per_unit` – dollars needed to run one activity unit
- `volunteer_hours` – volunteer hours per activity unit
- `impact_points` – a score representing the benefit/impact of the activity
- `required_staff` – number of staff members needed per activity unit

Special row in the Excel sheet:

- A final row labeled `LIMITS (available/maximum)` gives: total budget, total volunteer hours, and total staff slots available.

Example modeling ideas (you will choose)

Here are a few example objectives you *could* consider (you are not required to pick any of these):

- Maximize total impact points.
- Minimize total cost while achieving at least a target impact that you choose.
- Maximize impact per volunteer hour.

These are only illustrations to help you think. You will choose your own objective and decide how to use the limits row (budget, hours, staff) in your constraints.

Dataset B: Cafeteria Weekly Menu Planning

You are designing a weekly menu for the school cafeteria. Each type of meal has a cost, calories, protein, and a popularity score. You want to design a menu that balances nutrition, cost, and popularity.

Columns (Excel file):

- `meal` – name of the meal
- `cost_to_prepare` – cost to prepare one serving
- `calories` – calories per serving
- `protein` – grams of protein per serving
- `popularity` – popularity score (higher means students like it more)

Special row in the Excel sheet:

- A final row labeled `REQUIREMENTS (minimums)` gives: minimum total calories, minimum total protein, and minimum total popularity.

Example modeling ideas (you will choose)

Some possible objectives for this dataset are:

- Minimize total cost while meeting the minimum requirements.
- Minimize total calories while keeping protein and popularity high enough.
- Maximize total protein or total popularity subject to a budget limit that you choose.

These are **only examples** to give you ideas. You are free to define your own objective and decide how to turn the "REQUIREMENTS (minimums)" row into constraints.

Dataset C: Makerspace Production Planning

The school makerspace is producing items to sell at a community fair. Each item gives profit but uses machine time, student labor, and materials.

Columns (Excel file):

- `item` – product name
- `profit` – profit per unit sold
- `machine_time` – minutes of machine time per unit
- `labor_time` – minutes of student labor per unit
- `material_units` – material units used per unit

Special row in the Excel sheet:

- A final row labeled `LIMITS (available/maximum)` gives: maximum machine time, maximum labor time, and maximum material units available.

Example modeling ideas (you will choose)

Some example directions you might explore:

- Maximize total profit using machine, labor, and material limits.
- Maximize the total number of items produced while staying under a total time limit.
- Maximize profit per unit of material or per unit of time.

You do not have to choose any of these in particular. They are just ideas to help you design your own objective using the `LIMITS (available/maximum)` row.

Dataset D: School Bus Pickup Planning

The school needs to schedule bus pickup batches to different areas. Each batch serves students, costs fuel, takes time, and has some traffic risk.

Columns (Excel file):

- `area` – area name
- `students_per_batch` – students served in one bus batch
- `fuel_cost` – fuel cost for one batch
- `travel_time` – minutes of driving for one batch
- `traffic_risk` – risk score (higher = more risk)

Special row in the Excel sheet:

- A final row labeled `TARGETS / LIMITS` gives example totals you can use as bounds, such as:
  - a target number of students to serve (minimum),
  - a maximum total travel time,
  - a maximum total traffic risk.

Example modeling ideas (you will choose)

For example, you might:

- Minimize total fuel cost while serving enough students and respecting time and risk limits.
- Minimize total travel time while serving at least the target number of students.
- Maximize students served subject to fuel and risk limits that you set.

These are suggested ideas only. You will choose your own objective and use the `TARGETS / LIMITS` row to build appropriate constraints.

Example modeling ideas (you will choose)

For this dataset, you might:

- Minimize total fuel cost while serving enough students and respecting time and risk limits.
- Minimize total travel time while serving enough students.
- Maximize students served subject to fuel and risk limits.

You will choose your objective and decide how to use the target/limit values in your constraints.

4 Writing Your Mathematical Model

No matter which dataset you choose (A–D, your own, or a public one), you must write a complete model.

(a) Decision variables

Define one decision variable for each option in your dataset. Examples of good names: x_{pasta}, $x_{\text{fieldtrip}}$, x_{north}, etc.

(b) Objective function

You will choose a meaningful objective that matches your story. Some examples:

- Maximize total impact, profit, or students served.
- Minimize total cost, time, or risk.

- Maximize a "benefit per resource" ratio (for example, impact per hour).

Your objective must:

- Be clearly stated as "maximize" or "minimize".
- Be written as a linear expression in your variables (a sum of coefficients times variables).

(c) Constraints

You must include:

- At least three meaningful inequality constraints using your data.
- At least one constraint that uses values from the **LIMITS/REQUIREMENTS** row (if present).
- Non-negativity constraints: $x_i \geq 0$ for every variable.

> **Check yourself:** Do your constraints use the right units? Are you adding up the right columns from the table? Does your objective really match the story you are telling?

5 Python Requirements

Your Python file (or notebook) must show that you can connect data, math, and code.

(a) Read your dataset with pandas

```
import pandas as pd

data = pd.read_excel("your_dataset_file.xlsx")
print(data)
```

If you are using one of the proposed Excel files with a `LIMITS` or `REQUIREMENTS` row at the bottom, you may want to separate the options from the limits:

```
# assume the last row contains the limits / requirements
data_options = data.iloc[:-1] # all rows except the last one
limits_row = data.iloc[-1] # the last row with total limits
```

Make sure the column names in Python match the names you use in your code.

(b) Use tools from Lessons 1–5

Your code must include:

- At least one variable that stores a calculation.
- At least one list *or* dictionary (for example, to store names or results).
- At least one `if`/`else` decision.
- At least one loop (for example, a search for the "best" single option by a simple rule).
- At least one function that you define and call (for example, an efficiency score).

(c) Build and solve the model with `linprog`

You must:

- Create the arrays `c`, `A_ub`, and `b_ub` from your DataFrame.
- Set up `bounds` so that each variable is nonnegative.
- Use the correct sign for the objective (remember the trick for maximization).
- Call `scipy.optimize.linprog` with `method="highs"`.

```
from scipy.optimize import linprog

# c, A_ub, b_ub, bounds should be defined using your data and model
res = linprog(c, A_ub=A_ub, b_ub=b_ub, bounds=bounds, method="highs")

print("Success:", res.success)
print("Optimal value:", res.fun)
print("Decision variables:", res.x)
```

(d) Plot your optimal plan

Make a simple bar chart that shows your decision variables.

```
import matplotlib.pyplot as plt

names = list(data_options.iloc[:, 0]) # assumes first column has option names
plt.bar(names, res.x)
plt.title("Optimal Plan")
plt.xlabel("Options")
plt.ylabel("Quantity")
plt.xticks(rotation=30)
plt.tight_layout()
```

```
plt.savefig("optimal_plan.png", dpi=200)
plt.close()
```

6 Reflection and Deliverables

(a) Reflection questions

Write a short reflection (about one page) where you answer:

- What does your optimal solution tell you to do?
- Which constraint was the most restrictive or important in your problem?
- Did your manual loop or "best guess" match what the solver found? Why or why not?
- What is one thing you learned about optimization?
- What is one thing you learned about Python?

(b) What you will turn in

You will submit:

- Your Python file (or notebook) with:
  - Reading the dataset,
  - Your function, loops, if/else, and calculations,
  - The `linprog` call and printed results,
  - The code that generates your bar chart.
- The Excel file you used (or a link to the public dataset, if applicable).
- Your written reflection (answers to the questions above).
- The image file of your plot (`optimal_plan.png`) or the plot embedded in your notebook.

Final thought: This mini project is your first small "research-style" experience with optimization. You chose a problem, built a model, coded it, and used a solver to make a data-driven decision. That is exactly what people do with optimization in the real world.