Comparison of Solvers:

1. **Gurobi:**
   - A commercial solver with **high efficiency and performance**.
   - Requires a **license** (free for academic use).
   - Offers **robust support for large-scale optimization problems**.
2. **PuLP:**
   - An **open-source solver** that works well for small to medium-sized LP problems.
   - Supports multiple back-end solvers like CBC and GLPK.
   - **Easier to use** but may be slower than Gurobi for large problems.
3. **CVXPY:**
   - A **Python-based convex optimization framework**.
   - More **flexible** for complex optimization problems (e.g., quadratic, cone, and semidefinite programming).
   - **Less optimized** for large-scale LPs compared to Gurobi.

Example Problem

Problem Statement:

A person needs at least **500 calories** and **20 grams of protein** per meal. They can choose between **Food A** and **Food B**, with the following nutritional values and costs:

Food Item	Calories per serving	Protein (g) per serving	Cost per serving (€)
Food A	250	10	3
Food B	200	5	2

We define:

- x_A = number of servings of **Food A**
- x_B = number of servings of **Food B**

Formulation:

Minimize cost:

$$\text{Minimize } Z = 3x_A + 2x_B$$

Subject to:

$$250x_A + 200x_B \geq 500 \quad \text{(Calorie constraint)}$$

$$10x_A + 5x_B \geq 20 \quad \text{(Protein constraint)}$$

$$x_A, x_B \geq 0 \quad (\text{Non} - \text{negativity constraint})$$

Brief Explanation of the Code

The code solves a **Linear Programming (LP) problem** using three different solvers: **Gurobi, PuLP, and CVXPY**.

Libraries & Functions Used:

1. **Gurobi (gurobipy)**
   - `Model()`: Creates an optimization model.
   - `addVar()`: Defines decision variables.
   - `setObjective()`: Sets the objective function (minimizing cost).
   - `addConstr()`: Adds constraints to the model.
   - `optimize()`: Solves the LP problem.
   - **Why?** Gurobi is a high-performance solver, efficient for large-scale problems.

2. **PuLP (pulp)**
   - `LpProblem()`: Defines an LP problem.
   - `LpVariable()`: Creates variables with lower bounds.
   - +=: Defines the objective function and constraints.
   - `solve()`: Runs the solver.
   - `value()`: Extracts solution values.
   - **Why?** PuLP is a simple, open-source solver for LP problems.
3. **CVXPY (cvxpy)**
   - `Variable()`: Creates decision variables.
   - `Minimize()`: Defines the objective function.
   - `Problem()`: Combines objective and constraints into an optimization problem.
   - `solve()`: Finds the optimal solution.
   - **Why?** CVXPY is useful for more complex convex optimization problems.

Results:

```
Gurobi Optimizer version 12.0.1 build v12.0.1rc0 (mac64[arm] - Darwin 24.3.0 24D70)

CPU model: Apple M3
Thread count: 8 physical cores, 8 logical processors, using up to 8 threads

Optimize a model with 2 rows, 2 columns and 4 nonzeros
Model fingerprint: 0xcc35f89a
Coefficient statistics:
  Matrix range     [5e+00, 2e+02]
  Objective range  [2e+00, 3e+00]
  Bounds range     [0e+00, 0e+00]
  RHS range        [2e+01, 5e+02]
Presolve time: 0.00s
Presolved: 2 rows, 2 columns, 4 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    0.0000000e+00   5.125000e+01   0.000000e+00      0s
       2    6.0000000e+00   0.000000e+00   0.000000e+00      0s

Solved in 2 iterations and 0.00 seconds (0.00 work units)
Optimal objective  6.000000000e+00
```

Gurobi Solution:

```
x_A = 2.0
x_B = 0.0
Optimal Cost = 6.0
GLPSOL--GLPK LP/MIP Solver 5.0
Parameter(s) specified in the command line:
 --cpxlp /var/folders/4n/6ydjnj8d16x_m4by160ndjw80000gn/T/7e1970ea859b40458ba18e72a0043249-pulp.lp
 -o /var/folders/4n/6ydjnj8d16x_m4by160ndjw80000gn/T/7e1970ea859b40458ba18e72a0043249-pulp.sol
Reading problem data from '/var/folders/4n/6ydjnj8d16x_m4by160ndjw80000gn/T/7e1970ea859b40458ba18e72a0043249-pulp.lp'...
2 rows, 2 columns, 4 non-zeros
```

```
7 lines were read
GLPK Simplex Optimizer 5.0
2 rows, 2 columns, 4 non-zeros
Preprocessing...
2 rows, 2 columns, 4 non-zeros
Scaling...
 A: min|aij| =  5.000e+00  max|aij| =  2.500e+02  ratio =  5.000e+01
GM: min|aij| =  8.891e-01  max|aij| =  1.125e+00  ratio =  1.265e+00
EQ: min|aij| =  7.906e-01  max|aij| =  1.000e+00  ratio =  1.265e+00
Constructing initial basis...
Size of triangular part is 2
      0: obj =   0.000000000e+00 inf =   5.064e+00 (2)
      1: obj =   6.000000000e+00 inf =   4.189e-17 (0)
OPTIMAL LP SOLUTION FOUND
Time used:   0.0 secs
Memory used: 0.0 Mb (32525 bytes)
Writing basic solution to '/var/folders/4n/6ydjnj8d16x_m4by160ndjw80000gn/T/7e1970ea859b40458ba18e72a0043249-pulp.sol'...
```

PuLP Solution:

```
x_A = 2.0
x_B = 0.0
Optimal Cost = 6.0
```

CVXPY Solution:

```
x_A = 1.9999999986948422
x_B = 2.6830924434355166e-09
Optimal Cost = 6.0000000014507116
```