Name: ________________________________ Section: 12:30pm or 2pm

Final Exam 2023

Learning Objective:

- Create Python code to automate a given task.
- Formulate linear optimization models to inform a business decision.

Instructions:

The final exam tests your mastery of skills taught in Weeks 7-12, which culminates in creating linear optimization models to inform a given business decision. There are three questions, worth a total of 30 points. The exam is 100 minutes, and is open-notes but closed-computer. You can bring paper notes or books of any kind, but no computers, tablets, or cell phones are allowed. **Do not share the exam questions or your solutions with a student who has not yet completed an exam and do not look at other people's solutions. Any violation of academic integrity will result in a zero grade for the exam for everyone involved.**

As long as you fulfill all the specifications described in the problem description, it doesn't matter how you model the problem or how efficient is your code. As with the midterm, partial credits will be given for any fragments of correct solutions, or for English descriptions that would lead to a solution.

Q1. Kitchen Remodeling (Concrete Formulation; 11 Points)

Katherine is remodeling her kitchen and would like to decide on the cabinet design, countertop type, and flooring style. Each option differs in terms of looks, price, and lead time. Her contractor only wants to start the project when all the materials have been delivered, which is based on the maximum lead time of all the options she select. The following table summarizes the options she is considering.

Category	Option	Looks-Score	Price	Lead Time
Cabinet Design	A_1	7	10	3
	A_2	5	8	2
	A_3	4	7	1
Countertop Type	B_1	5	4	2
	B_2	4	2	1
	B_3	1	1	1
Flooring Style	C_1	4	8	1
	C_2	3	7	1

As shown above, there are three cabinet designs to choose from, three countertop types, and two flooring styles. Within each category, she must select exactly one option. For example, if she selects cabinet A_1, countertop B_1 and flooring C_1, then the total looks-score would be $7 + 5 + 4 = 16$. The total price would be $10 + 4 + 8 = 22$. The maximum lead time would be $\max(3, 2, 1) = 3$. In addition to the above, there are certain combinations of options that look better together, as summarized below:

Combo	Additional Looks-Score
$A_1 + B_3$	3
$A_2 + B_2 + C_2$	2

In words, if she selects cabinet A_1 and countertop B_3, then she gets an additional 3 points to the total looks-score, regardless of what flooring she selects. If she selects cabinet A_2, countertop B_2 and flooring C_2, then the total looks-score would be $5 + 4 + 3 + 2 = 14$, which includes the additional looks-score of 2 given in the second row above.

Formulate a linear optimization model to help her optimally trade off the total looks-score, total price, and maximum lead time. The objective function should correspond to

$$(\text{Total Looks-Score}) - \alpha(\text{Maximum Lead Time}),$$

where α is a known constant (not a decision variable). Beside ensuring that she selects exactly one option from each category, your formulation should ensure that (Total Price) $\leq \beta$, where β is her budget and is also a known constant. (You can simply use α and β in your formulation and treat them like numbers, and you don't have to define them.) **You must ensure that the formulation is linear.**

Decision Variables:

Objective and Constraints:

Q2. Email Marketing (Abstract Formulation; 10 Points)

A company would like to use optimization to decide which email ads to send to each potential customer. Suppose that there are n customers and m ads. For each customer i and each ad j, the company used machine learning to estimate two parameters: p_{ij} and s_{ij}. Here, p_{ij} is the expected profit from sending ad j to customer i, and s_{ij} is customer i's interest score for ad j. The company would like to maximize its total expected profit subject to the following constraints:

1. Each customer i receives at most k_i ads. Moreover, a customer cannot get the same ad more than once.
2. The average interest score of ads received by each customer is at least a_i. For example, if a customer receives 3 ads, and her interest scores for them are 5, 1 and 3 respectively, then her average interest score is $(5 + 1 + 3)/3 = 3$.
3. Certain pairs of ads are overlapping and cannot be sent to the same customer. For example, suppose that the set of overlapping pairs of ads is $\{(A, B), (A, C), (B, D)\}$, then ads A and B cannot be sent to the same customer, ads A and C cannot be sent to the same customer, and ads B and D cannot be sent to the same customer.
4. The number of customers who receive each ad is within a factor of 2 of each other. In other words, there does not exist two ads such that the number of customers who receive the first ad is more than double the number of customers who receive the second ad.

Write an abstract formulation of a linear optimization model to implement the above logic. You must ensure that the formulation is linear. For your convenience, some of the data variables are already provided, but you may need to add other data variables.

Data:

- I: the set of customers.
- J: the set of ads.
- p_{ij}: the expected profit of sending ad $j \in J$ to customer $i \in I$.
- s_{ij}: the interest score of ad $j \in J$ for customer $i \in I$.
- k_i: the maximum number of ads that can be sent to customer i.
- a_i: the minimum average interest score of ads sent to customer i.

Decision Variables:

Objective:

Constraints:

Q3. Diet Optimization (Gurobi Coding; 9 points)

This problem asks you to implement the abstract formulation from Q2 of Sample Final Exam C, which is about optimizing one's diet to minimize cost while satisfying minimal nutritional requirements.

Data:

- I: set of foods.
- J: set of nutrients.
- c_i: cost of food i.
- a_{ij}: amount of nutrient j in one serving of food i.
- l_j: lower bound in daily intake of nutrient j.
- u_j: upper bound of nutrient j.
- s_i: minimum number of servings of food i to have if one is to have any such food at all.
- t_i: maximum number of servings of food i to have.

Decision Variables:

- X_i: amount of food i in the diet. (continuous)
- Z_i: whether to include food i at all. (binary)

Objective and Constraints:

$$\text{Min.} \quad \sum_{i \in I} c_i X_i$$

$$\text{s.t.}$$

$$\begin{aligned} l_j \leq \sum_{i \in I} a_{ij} X_i \leq u_j \qquad & \text{for each nutrient } j \in J. \\ s_i Z_i \leq X_i \leq t_i Z_i \qquad & \text{for each food } i \in I. \\ X_i \geq 0 \qquad & \text{for each food } i \in I. \end{aligned}$$

Write a function called `optimize_diet` with one input argument:

- **inputFile**: the name of an input Excel file.

The input Excel file has two sheets. The first sheet is called "Nutrition" and looks like this:

Foods	Calories	Protein	Fat	Sodium	Costs	Minimum	Maximum
1. ice cream	330	8	15	180	1.59	1	2
2. chicken	420	32	10	300	2.89	2	3
3. pizza	320	15	20	820	1.99	2	3
4. fries	380	4	19	270	1.89	1	3
5. macaroni	320	12	10	830	2.09	1	3
6. milk	100	8	2.5	125	0.89	1	3
7. salad	320	31	2	123	2.49	2	4

The first column gives the set of foods I. The last three columns give the cost c_i, the minimum s_i and the maximum t_i for each food $i \in I$. The rest of the table corresponds to a_{ij}. The second sheet is called "Bounds" and looks like this:

	Calories	Protein	Fat	Sodium
Lower bound	1800	91	0	0
Upper bound	2200	9999	65	1779

This gives the lower bound l_j and upper bound u_j for each nutrient $j \in J$.

The function should return a Series which gives the optimal diet. The Series should only include foods that are consumed a positive amount (so foods that not eaten at all should not be included). See the sample run for an illustration. The first few lines of code are already provided.

```
[2]: # Sample run
     optimize_diet('diet.xlsx')
```

```
2. chicken    1.841060
7. salad      3.208609
dtype: float64
```

```
def optimize_diet(inputFile):
    import pandas as pd
    from gurobipy import Model, GRB
```