Exercises - Section 2: Lecture 6 - **Variable Substitution** - Solutions

1. A high-end musical instrument manufacturer projects that it can sell up to 100 trumpets, 40 flugelhorns, 40 baritones, and 20 contrabasses in the next quarter-year. Creating pistons for these instruments requires three precision machining steps: broaching, brazing, and grinding. The table below shows how many minutes are required in each process for each type of musical instrument.

	Trumpet	**Flugelhorn**	**Baritone**	**Contrabass**
Broaching	10	8	12	15
Brazing	20	20	20	20
Grinding	10	12	15	20

Minutes required per instrument on each type of machine

To allow for maintenance and downtime, the company does not want to its machines beyond a certain limit. The total time available on the machines is 2000 hours for broaching, 3000 for brazing, and 2000 hours for grinding.

Once instruments are manufactured, they go to a tester, who is a professional musician; the musician is contracted to test exactly 150 instruments per quarter, no more and no less. Therefore, the company must manufacture exactly 150 instruments this quarter.

The company's profit is approximately $600 per trumpet, $700 per flugelhorn, $1000 per baritone, and $1500 per contrabass.

a. Create a mathematical model and a gurobipy model that the manufacturer can optimize to determine how many instruments of each type to manufacture in order to maximize its quarterly profit. Include an equality constraint to say that the tester can test exactly 150 instruments per quarter (so exactly 150 must be manufactured). Solve the gurobipy model.

There are two ways you might've answered this question, either writing out a specific model or writing the model using subscripts and sums. Below are solutions for both ways.

VERSION 1: WRITING OUT A SPECIFIC MODEL

ENGLISH	**MATH**	**GUROBIPY**	`m=gp.Model("music")`

Data
Number of instruments
Instruments
	N
	t, f, b, c
		`# Hard-code data in file`
		`# Indices will be:`
		`#   0 = trumpets`
		`#   1 = flugelhorns`
		`#   2 = baritones`
		`#   3 = contrabasses`	

Variables Number of each instrument to produce	x_i	`x = m.addVars(4, vtype=GRB.INTEGER,name="x")`	

<u>Objective</u> Maximize profit	Maximize $600x_t + 700x_f + 1000x_b + 1500x_c$	`m.setObjective(600*x[0] + 700*x[1] + 1000*x[2] + 1500*x[3],GRB.MAXIMIZE)`

<u>Constraints</u>
Machine time limits
	$10x_t + 8x_f + 12x_b + 15x_c \leq 2000$
	$20x_t + 20x_f + 20x_b + 20x_c \leq 3000$
	$10x_t + 12x_f + 15x_b + 20x_c \leq 2000$
		`m.addConstr(10*x[0] + 8*x[1] + 12*x[2] + 15*x[3] <= 2000)`
		`m.addConstr(20*x[0] + 20*x[1] + 20*x[2] + 20*x[3] <= 3000)`
		`m.addConstr(10*x[0] + 12*x[1] + 15*x[2] + 20*x[3] <= 2000)`

Manufacture exactly 150 instruments	$x_t + x_f + x_b + x_c = 150$	`m.addConstr(x.sum() == 150)`

Don’t produce more of any instrument than allowed	$x_t \leq 100$
	$x_f \leq 40$
	$x_b \leq 40$
	$x_c \leq 20$
		`m.addConstr(x[0] <= 100)`
		`m.addConstr(x[1] <= 40)`
		`m.addConstr(x[2] <= 40)`
		`m.addConstr(x[3] <= 20)`

Instruments produced must be non-negative and integer	all $x_i \geq 0$ and integer	`# Already part of variable declaration`

VERSION 2: USING SUBSCRIPTS AND SUMS

ENGLISH	**MATH**	**GUROBIPY** `m=gp.Model("music")`
<u>Data</u>		`# Read from data file`
Instruments	I	`# Instruments`
Machines	M	`# Machines`
Maximum production of each instrument type	m_i	`# MaxSales`
Machine availabilities	a_j	`# Available`
Times required for each instrument on each machine	t_{ij}	`# Times`
Profit for each instrument	p_i	`# Profit`
<u>Variables</u> Number of each instrument to produce	x_i	`x = m.addVars(Instruments, ub=MaxSales, vtype=GRB.INTEGER,name="x")`
<u>Objective</u> Maximize profit	Maximize $\sum_{i \in I} p_i x_i$	`m.setObjective(x.prod(Profit),GRB.MAXIMIZE)`

<u>Constraints</u> Machine time limits	$\sum_{i \in I} t_{ij} x_i \leq a_j$ for all machines $j \in M$	`m.addConstrs(sum(Times[j][i] *x[i] for i in Instruments) <= Available[j] for j in Machines)`
Manufacture exactly 150 instruments	$\sum_{i \in I} x_i = 150$	`m.addConstr(x.sum() == 150)`
Don't produce more of any instrument than allowed	$x_i \leq m_i$ for all instruments $i \in I$	`# Already part of variable declaration (ub=MaxSales)`
Instruments produced must be non-negative and integer	all $x_i \geq 0$ and integer	`# Already part of variable declaration (vtype=GRB.INTEGER)`

See `Pt. 2 - 2.6 question 1a.py` and `Pt. 2 - 2.6 question 1a-hardcoded.py` for the gurobipy code.

DISCUSSION: Whichever way you modeled it, the optimal profit is $128,000, achieved by making 50 trumpets, 40 flugelhorns, 40 baritones, and 20 contrabasses.

b. Instead of the equality constraint to say that exactly 150 instruments should be manufactured, use that constraint to substitute out the variable for flugelhorn production. Write the new mathematical and gurobipy models, and solve the gurobipy model.

VERSION 1: WRITING OUT A SPECIFIC MODEL

ENGLISH	**MATH**	**GUROBIPY**	`m=gp.Model("music")`

<u>Data</u>
Number of instruments
Instruments
	N
	t, b, c
		`# Hard-code data in file`
		`# Indices will be:`
		`#   0 = trumpets`
		`#   2 = baritones`
		`#   3 = contrabasses`	

<u>Variables</u> Number of each instrument to produce	x_i	`x = m.addVars(4, vtype=GRB.INTEGER,name="x")`	
<u>Objective</u> Maximize profit	Maximize $600x_t + 700(150 - x_t - x_b - x_c) + 1000x_b + 1500x_c$	`m.setObjective(600*x[0] + 700*(150-x[0]-x[2]-x[3]) + 1000*x[2] + 1500*x[3], GRB.MAXIMIZE)`	

<u>Constraints</u>
Machine time limits
	$10x_t + 8(150 - x_t - x_b - x_c) + 12x_b + 15x_c \leq 2000$
	$20x_t + 20(150 - x_t - x_b - x_c) + 20x_b + 20x_c \leq 3000$
	$10x_t + 12(150 - x_t - x_b - x_c)$
		`m.addConstr(10*x[0] + 8*(150-x[0]-x[2]-x[3]) + 12*x[2] + 15*x[3] <= 2000)`
		`m.addConstr(20*x[0] + 20*(150-x[0]-x[2]-x[3]) + 20*x[2] + 20*x[3] <= 3000)`	

	$+15x_b + 20x_c \leq 2000$	`m.addConstr(10*x[0] + 12*(150-x[0]-x[2]-x[3]) + 15*x[2] + 20*x[3] <= 2000)`
Manufacture exactly 150 instruments	(Implied by substituting $(150 - x_t - x_b - x_c)$ for x_f)	`# Implied by substitution`

Don't produce more of any instrument than allowed	$x_t \leq 100$
	$x_b \leq 40$
	$x_c \leq 20$
		`m.addConstr(x[0] <= 100)`
		`m.addConstr(x[2] <= 40)`
		`m.addConstr(x[3] <= 20)`

Instruments produced must be non-negative and integer	all $x_i \geq 0$ and integer	`# Already part of variable declaration`

VERSION 2: USING SUBSCRIPTS AND SUMS

ENGLISH	MATH	GUROBIPY `m=gp.Model("music")`
Data		`# Read from data file`
Instruments	I	`# Instruments`
Machines	M	`# Machines`
Maximum production of each instrument type	m_i	`# MaxSales`
Machine availabilities	a_j	`# Available`
Times required for each instrument on each machine	t_{ij}	`# Times`
Profit for each instrument	p_i	`# Profit`
Instruments except Flugelhorns	I'	`# InstNoFlugel`
Variables Number of each instrument to produce	x_i for all $i \in I'$	`x = m.addVars(InstNoFlugel, ub=MaxSales, vtype=GRB.INTEGER,name="x")`
Objective Maximize profit	Maximize $p_{\text{Flugel}}(150 - \sum_{i\in I'} x_i) + \sum_{i\in I'} p_i x_i$	`m.setObjective( Profit["Flugelhorn"]*(150-x.sum()) + sum(Profit[i]*x[i] for i in InstNoFlugel),GRB.MAXIMIZE)`

Constraints
Machine time limits
	$t_{\text{Flugel},j}(150 - \sum_{i\in I'} x_i) + \sum_{i\in I'} t_{ij} x_i \leq a_j$ for all machines $j \in M$	`m.addConstrs(Times[j]["Flugelhorn"]*(150-x.sum()) + sum(Times[j][i]*x[i] for i in InstNoFlugel) <= Available[j] for j in Machines)`
		`# Implied by substitution`

Manufacture exactly 150 instruments	(Implied by substituting $(150 - \sum_{i \in I'} x_i)$ for x_{Flugel})	
Don't produce more of any instrument than allowed	$x_i \leq m_i$ for all instruments $i \in I'$	`# Already part of variable declaration (ub=MaxSales)`
Instruments produced must be non-negative and integer	all $x_i \geq 0$ and integer	`# Already part of variable declaration (vtype=GRB.INTEGER)`

See `Pt. 2 - 2.6 question 1b.py` and `Pt. 2 - 2.6 question 1b-hardcoded.py` for the gurobipy code.

DISCUSSION: Either way, this model gives a different solution! The profit is $1000 higher, and the production is 40 trumpets, 40 baritones, and 20 contrabasses (which means producing 150-40-40-20 = 50 flugelhorns). Why is this solution different? Read on...

c. If your solution to model b. produces more than 150 instruments, what do you think went wrong? (Hint: If you plug the solution's recommended number of trumpets, baritones, and contrabasses into the equality constraint for the tester in model a., how many flugelhorns would be manufactured?)

The solution recommends manufacturing 50 flugelhorns, but the limit given above is only 40! Just like it's important to remember to substitute into the non-negativity constraints, it's also important to substitute into the upper-bound constraints for a variable. In this case, that means adding the following.

VERSION 1: WRITING OUT A SPECIFIC MODEL

ENGLISH	**MATH**	**GUROBIPY** `m=gp.Model("music")`
<u>Constraints</u> Can't produce a negative amount of flugelhorns	$150 - x_t - x_b - z_c \geq 0$	`m.addConstr(150 - x[0] - x[2] - x[3] >= 0)`
Can't produce more than the maximum allowable number of flugelhorns	$150 - x_t - x_b - z_c \leq 40$	`m.addConstr(150 - x[0] - x[2] - x[3] <= 40)`

VERSION 2: USING SUBSCRIPTS AND SUMS

ENGLISH	**MATH**	**GUROBIPY** `m=gp.Model("music")`
<u>Constraints</u> Can't produce a negative amount of flugelhorns	$150 - \sum_{i \in I'} x_i \geq 0$	`m.addConstr(150 - x.sum() >= 0)`
Can't produce more than the maximum allowable number of flugelhorns	$150 - \sum_{i \in I'} x_i \leq 40$	`m.addConstr(150 - x.sum() <= 40)`

See `Pt. 2 - 2.6 question 1cd.py` and `Pt. 2 - 2.6 question 1cd-hardcoded.py` for the gurobipy code.

DISCUSSION: After adding these constraints, the gurobipy model gives the correct solution as in part a.

d. On the other hand, if your solution to model b. did produce exactly 150 instruments, you probably remembered to substitute the equality constraint into the nonnegativity and upper-bound constraints for the variable for manufacturing flugelhorns (the variable should be greater than or equal to zero, and less than or equal to 40). If so, remove those constraints to see what wrong solution you might've received if you had forgotten to substitute into the bound.

DISCUSSION: See part c. solution.

2. A marketing firm is purchasing data sets to help train forecasting models. The firm would like to purchase exactly eight out of twelve available data sets. The table below lists the data sets, their cost, and the type of data they contain.

Data Set	Cost	Type of Data
1	$1.1M	Retail, USA
2	$0.5M	Retail, Europe
3	$0.2M	Housing, Canada
4	$0.7M	Credit, Europe
5	$1.1M	Credit, USA
6	$0.9M	Credit, Japan
7	$1.0M	Retail, Japan
8	$1.6M	Credit and Retail, USA
9	$0.8M	Online retail, Canada
10	$0.5M	Housing, USA
11	$0.7M	Online retail, Europe
12	$1.0M	Online retail, USA

To train its models, the company needs at least three data sets that contain retail data (including at least one online retail data set and at least one general retail data set), at least one housing data set, and at least two credit data sets. The company needs at least two data sets from the USA, at least two from Europe, and at least one each from Japan and Canada.

a. Create a mathematical model and a gurobipy model that the company can optimize to determine which data sets to purchase in order to minimize its spend. Include an equality constraint to say that the company must purchase exactly eight of the data sets. Solve the gurobipy model.

There are two ways you might've answered this question, either writing out a specific model or writing the model using subscripts and sums. Below are solutions for both ways.

VERSION 1: WRITING OUT A SPECIFIC MODEL

ENGLISH	MATH	GUROBIPY	`m=gp.Model("market")`
Variables Which datasets to purchase	x_i	`x = m.addVars(range(1,13), vtype=GRB.BINARY, name="x")`	
Objective Minimize cost	Minimize $1.1x_1 + 0.5x_2 + 0.2x_3 + 0.7x_4 + 1.1x_5 + 0.9x_6 + 1.0x_7 + 1.6x_8 + 0.8x_9 + 0.5x_{10} + 0.7x_{11} + 1.0x_{12}$	`m.setObjective(1.1*x[1] + 0.5*x[2] + 0.2*x[3] + 0.7*x[4] + 1.1*x[5] + 0.9*x[6] + 1.0*x[7] + 1.6*x[8] + 0.8*x[9] + 0.5*x[10] + 0.7*x[11] + 1.0*x[12])`	

Constraints Minimum data sets of each category	$x_4 + x_5 + x_6 + x_8 \geq 2$	`m.addConstr(x[4] + x[5] + x[6] + x[8] >= 2) # Credit`
	$x_3 + x_{10} \geq 1$	`m.addConstr(x[3] + x[10] >= 1) # Housing`
	$x_9 + x_{11} + x_{12} \geq 1$	`m.addConstr(x[9] + x[11] + x[12] >= 1) # Online retail`
	$x_1 + x_2 + x_7 + x_8 \geq 1$	`m.addConstr(x[1] + x[2] + x[7] + x[8] >= 1) # Retail`
	$x_1 + x_2 + x_7 + x_8 + x_9 + x_{11} + x_{12} \geq 3$	`m.addConstr(x[1] + x[2] + x[7] + x[8] + x[9] + x[11] + x[12] >= 3) # Total retail`
	$x_5 + x_8 + x_{10} + x_{12} \geq 2$	`m.addConstr(x[1] + x[5] + x[8] + x[10] + x[12] >= 2) # USA`
	$x_3 + x_9 \geq 1$	`m.addConstr(x[3] + x[9] >= 1) # Canada`
	$x_2 + x_4 + x_{11} \geq 2$	`m.addConstr(x[2] + x[4] + x[11] >= 2) # Europe`
	$x_6 + x_7 \geq 1$	`m.addConstr(x[6] + x[7] >= 1) # Japan`
Purchase exactly 8 datasets	$\sum_{i=1}^{12} x_i = 8$	`m.addConstr(x.sum() == 8)`
Each dataset is either purchased or is not purchased	all $x_i \in \{0,1\}$	`# Already part of variable declaration (vtype=GRB.BINARY)`

VERSION 2: USING SUBSCRIPTS AND SUMS

ENGLISH	MATH	GUROBIPY	`m=gp.Model("market")`
Data		`# Read from data file`	
Number of datasets	N	`# DataSets`	
Categories (types, regions) with minimum requirements	T	`# Categories`	
Minimum datasets of each category	m_j	`# Minimums`	
Which categories is each dataset in?	a_{ij} = 1 if dataset i is in category j, 0 if not	`# Groups`	
Cost of each dataset	c_i	`# Costs`	
Total datasets required	M	`# NumToBuy`	
Number of Categories	C	`# NumCategories`	
Variables Which datasets to purchase	x_i	`x = m.addVars(DataSets, vtype=GRB.BINARY, name="x")`	

Objective Minimize cost	Minimize $\sum_{i=1}^{N} c_i x_i$	`m.setObjective(x.prod(Costs))`
Constraints Minimum data sets of each category	$\sum_{i=1}^{N} a_{ij} x_i \geq m_j$ for all categories j	`m.addConstrs(sum(x[i]*Groups[i][j] for i in range(DataSets)) >= Minimums[j] for j in range(NumCategories))`
Purchase exactly 8 datasets	$\sum_{i=1}^{N} x_i = M$	`m.addConstr(x.sum() == NumToBuy)`
Each dataset is either purchased or is not purchased	all $x_i \in \{0,1\}$	`# Already part of variable declaration (vtype=GRB.BINARY)`

See `Pt. 2 - 2.6 question 2a.py` and `Pt. 2 - 2.6 question 2a-hardcoded.py` for the gurobipy code.

DISCUSSION: Whichever way you modeled it, the minimum cost is $5.3 million, achieved by purchasing datasets 2,3,4,6,9,10,11,12.

b. Instead of the equality constraint to say that exactly eight data sets should be purchased, use that constraint to substitute out the variable for purchasing data set 2. Write the new mathematical and gurobipy models, and solve the gurobipy model.

VERSION 1: WRITING OUT A SPECIFIC MODEL

ENGLISH	**MATH**	**GUROBIPY**	`m=gp.Model("market")`
Variables Which datasets to purchase	x_i	`x = m.addVars(range(1,13), vtype=GRB.BINARY, name="x")`	
Objective Minimize cost	Minimize $1.1x_1 + 0.5(8 - x_1 - x_3 - x_4 - x_5 - x_6 - x_7 - x_8 - x_9 - x_{10} - x_{11} - x_{12}) + 0.2x_3 + 0.7x_4 + 1.1x_5 + 0.9x_6 + 1.0x_7 + 1.6x_8 + 0.8x_9 + 0.5x_{10} + 0.7x_{11} + 1.0x_{12}$	`m.setObjective(1.1*x[1] + 0.5*x[2] + 0.2*x[3] + 0.7*x[4] + 1.1*x[5] + 0.9*x[6] + 1.0*x[7] + 1.6*x[8] + 0.8*x[9] + 0.5*x[10] + 0.7*x[11] + 1.0*x[12])`	

Constraints Minimum data sets of each category	$x_4 + x_5 + x_6 + x_8 \geq 2$	`m.addConstr(x[4] + x[5] + x[6] + x[8] >= 2) # Credit`
	$x_3 + x_{10} \geq 1$	`m.addConstr(x[3] + x[10] >= 1) # Housing`
	$x_9 + x_{11} + x_{12} \geq 1$	`m.addConstr(x[9] + x[11] + x[12] >= 1) # Online retail`
	$x_1 + (8 - x_1 - x_3 - x_4 - x_5 - x_6 - x_7 - x_8 - x_9 - x_{10} - x_{11} - x_{12}) + x_7 + x_8 \geq 1$	`m.addConstr(x[1] + (8-(x[1] + x[3] + x[4] + x[5] + x[6] + x[7] + x[8] + x[9] + x[10] + x[11] + x[12])) + x[7] + x[8] >= 1) # Retail`
	$x_1 + (8 - x_1 - x_3 - x_4 - x_5 - x_6 - x_7 - x_8 - x_9 - x_{10} - x_{11} - x_{12}) + x_7 + x_8 + x_9 + x_{11} + x_{12} \geq 3$	`m.addConstr(x[1] + (8-(x[1] + x[3] + x[4] + x[5] + x[6] + x[7] + x[8] + x[9] + x[10] + x[11] + x[12])) + x[7] + x[8] + x[9] + x[11] + x[12] >= 3) # Total retail`
	$x_5 + x_8 + x_{10} + x_{12} \geq 2$	`m.addConstr(x[1] + x[5] + x[8] + x[10] + x[12] >= 2) # USA`
	$x_3 + x_9 \geq 1$	`m.addConstr(x[3] + x[9] >= 1) # Canada`
	$(8 - x_1 - x_3 - x_4 - x_5 - x_6 - x_7 - x_8 - x_9 - x_{10} - x_{11} - x_{12}) + x_4 + x_{11} \geq 2$	`m.addConstr((8-(x[1] + x[3] + x[4] + x[5] + x[6] + x[7] + x[8] + x[9] + x[10] + x[11] + x[12])) + x[4] + x[11] >= 2) # Europe`
	$x_6 + x_7 \geq 1$	`m.addConstr(x[6] + x[7] >= 1) # Japan`
Purchase exactly 8 datasets	(Implied by substitution)	`Implied by substitution`
Each dataset is either purchased or is not purchased	all $x_i \in \{0,1\}$	`# Already part of variable declaration (vtype=GRB.BINARY)`

VERSION 2: USING SUBSCRIPTS AND SUMS

ENGLISH	MATH	GUROBIPY	`m=gp.Model("market")`
Data		`# Read from data file`	
Number of datasets	N	`# DataSets`	
Categories (types, regions) with minimum requirements	T	`# Categories`	
	m_j	`# Minimums`	

Minimum datasets of each category		
Which categories is each dataset in?	a_{ij} = 1 if dataset i is in category j, 0 if not	`# Groups`
Cost of each dataset	c_i	`# Costs`
Total datasets required	M	`# NumToBuy`
Number of Categories	C	`# NumCategories`
<u>Variables</u> Which datasets to purchase	x_i	`x = m.addVars(DataSets, vtype=GRB.BINARY, name="x")`
<u>Objective</u> Minimize cost	Minimize $c_2\left(M - \sum_{i\in\{1,\dots,N\},i\neq 2} x_i\right) + \sum_{i\in\{1,\dots,N\},i\neq 2} c_i x_i$	`m.setObjective(sum(Costs[i]*x[i] for i in range(DataSets) if i != 1) + Costs[1]*(NumToBuy - sum(x[i] for i in range(DataSets) if i != 1)))`
<u>Constraints</u> Minimum data sets of each category	$a_{2j}\left(M - \sum_{i\in\{1,\dots,N\},i\neq 2} x_i\right) + \sum_{i\in\{1,\dots,N\},i\neq 2} a_{ij} x_i \geq m_j$ for all categories j	`m.addConstrs(sum(x[i]*Groups[i][j] for i in range(DataSets) if i != 1) + Groups[1][j]*(NumToBuy - sum(x[i] for i in range(DataSets) if i != 1)) >= Minimums[j] for j in range(NumCategories))`
Purchase exactly 8 datasets	Implied by substituting $M - \sum_{i\in\{1,\dots,N\},i\neq 2} x_i$ for x_2	`# Implied by substitution`
Each dataset is either purchased or is not purchased	all $x_i \in \{0,1\}$	`# Already part of variable declaration (vtype=GRB.BINARY)`

See `Pt. 2 - 2.6 question 2b.py` and `Pt. 2 - 2.6 question 2b-hardcoded.py` for the gurobipy code.

DISCUSSION: Either way, this model gives a different solution! The cost is \$0.5 million lower, and the datasets purchased are 3,4,6,10,12. Why is this solution different? Read on...

c. If your solution to model b. purchases fewer than seven data sets, what do you think went wrong? (Hint: check your solution against all of the original constraints and objective. How many duplicates of data set 2 does the model seem to be purchasing?)

The substitution implies that dataset 2 is not purchased if eight of the other datsets are purchased, and datset 2 is purchased if seven of the others are. Mathematically, that assumption is $x_2 = M - \sum_{i\in\{1,\dots,N\},i\neq 2} x_i$. But since $\sum_{i\in\{1,\dots,N\},i\neq 2} x_i$ equals 5 in the solution to part b., the solution seems to be implying that $x_2 = 3$, that dataset 2 must be purchased three times to get a total of eight dataset purchases! The reason is that the binary constraints $x_i \in \{0,1\}$ implicitly contain two bound constraints plus an integer restriction.

Mathematically, $x_i \in \{0,1\}$ really means three things:

(1) x_i is integer
(2) $x_i \geq 0$
(3) $x_i \leq 1$.

In gurobipy, saying `vtype=GRB.BINARY` for a variable also really means three things:

(1) `vtype=GRB.BINARY`
(2) `lb=0`
(3) `ub=1`

So, just like the model in part b. substituted $M - \sum_{i \in \{1,\ldots,N\}, i \neq 2} x_i$ for x_2 everywhere else, the same substitution needs to be done in the constraints $x_i \geq 0$ and $x_i \leq 1$.

VERSION 1: WRITING OUT A SPECIFIC MODEL

ENGLISH	**MATH**	**GUROBIPY**	`m=gp.Model("market")`
Constraints Can't purchase dataset 2 fewer than 0 times	$8 - x_1 - x_3 - x_4 - x_5 - x_6 - x_7 - x_8 - x_9 - x_{10} - x_{11} - x_{12} \geq 0$	`m.addConstr(8-(x[1] + x[3] + x[4] + x[5] + x[6] + x[7] + x[8] + x[9] + x[10] + x[11] + x[12]) >= 0)`	
Can't purchase dataset 2 more than one time	$8 - x_1 - x_3 - x_4 - x_5 - x_6 - x_7 - x_8 - x_9 - x_{10} - x_{11} - x_{12} \leq 40$	`m.addConstr(8-(x[1] + x[3] + x[4] + x[5] + x[6] + x[7] + x[8] + x[9] + x[10] + x[11] + x[12]) <= 1)`	

VERSION 2: USING SUBSCRIPTS AND SUMS

ENGLISH	**MATH**	**GUROBIPY**	`m=gp.Model("market")`
Constraints Can't purchase dataset 2 fewer than 0 times	$M - \sum_{i \in \{1,\ldots,N\}, i \neq 2} x_i \geq 0$	`NumToBuy - sum(x[i] for i in range(DataSets) if i != 1) >= 0`	
Can't purchase dataset 2 more than one time	$M - \sum_{i \in \{1,\ldots,N\}, i \neq 2} x_i \leq 1$	`NumToBuy - sum(x[i] for i in range(DataSets) if i != 1) <= 1`	

See `Pt. 2 - 2.6 question 2cd.py` and `Pt. 2 - 2.6 question 2cd-hardcoded.py` for the gurobipy code.

d. On the other hand, if your solution to model b. did purchase exactly eight data sets, you probably remembered to substitute the equality constraint into the implied bound constraints for the variable for purchasing data set 2 (the variable should be greater than or equal to zero, and the variable should be less than or equal to one). If so, remove those constraints to see what wrong solution you might've received if you had forgotten to substitute into the bounds.

DISCUSSION: See part c. solution.

NOTES: