Question 15.2

In the videos, we saw the "diet problem". (The diet problem is one of the first large-scale optimization problems to be studied in practice. Back in the 1930's and 40's, the Army wanted to meet the nutritional requirements of its soldiers while minimizing the cost.) In this homework you get to solve a diet problem with real data. The data is given in the file `diet.xls`.

1. Formulate an optimization model (a linear program) to find the cheapest diet that satisfies the maximum and minimum daily nutrition constraints, and solve it using PuLP. Turn in your code and the solution. (The optimal solution should be a diet of air-popped popcorn, poached eggs, oranges, raw iceberg lettuce, raw celery, and frozen broccoli. UGH!)
2. Please add to your model the following constraints (which might require adding more variables) and solve the new model:
    1. If a food is selected, then a minimum of 1/10 serving must be chosen. (Hint: now you will need two variables for each food *i*: whether it is chosen, and how much is part of the diet. You'll also need to write a constraint to link them.)
    2. Many people dislike celery and frozen broccoli. So at most one, but not both, can be selected.
    3. To get day-to-day variety in protein, at least 3 kinds of meat/poultry/fish/eggs must be selected. [If something is ambiguous (e.g., should bean-and-bacon soup be considered meat?), just call it whatever you think is appropriate – I want you to learn how to write this type of constraint, but I don't really care whether we agree on how to classify foods!]

Answer 15.2

```
# Load and preview dataset. Import relevant packages.
from pulp import *
import pandas as pd
raw = pd.read_excel(open('/Users/…/data 15.2/diet.xls','rb'),sheet_name='Sheet1')
raw.head()
```

	Foods	Price/ Serving	Serving Size	Calories	Cholesterol mg	Total_Fat g	Sodium mg	Carbohydrates g	Dietary_Fiber g	Protein g	Vit_A IU	Vit_C IU	Calcium mg	Iron mg
0	Frozen Broccoli	0.16	10 Oz Pkg	73.8	0.0	0.8	68.2	13.6	8.5	8.0	5867.4	160.2	159.0	2.3
1	Carrots,Raw	0.07	1/2 Cup Shredded	23.7	0.0	0.1	19.2	5.6	1.6	0.6	15471.0	5.1	14.9	0.3
2	Celery, Raw	0.04	1 Stalk	6.4	0.0	0.1	34.8	1.5	0.7	0.3	53.6	2.8	16.0	0.2
3	Frozen Corn	0.18	1/2 Cup	72.2	0.0	0.6	2.5	17.1	2.0	2.5	106.6	5.2	3.3	0.3
4	Lettuce,Iceberg,Raw	0.02	1 Leaf	2.6	0.0	0.0	1.8	0.4	0.3	0.2	66.0	0.8	3.8	0.1

```
# Extract relevant data to be processed. Convert df to list
data = raw[0:64]
data = data.values.tolist()

# Read relevant vectors and define optimization function as func parameter. The goal is to build constraints on the minimization function to find the cheapest diet within the max_th and min_th threshold extracted from min/max daily intake values.
foods = [x[0] for x in data]
cost = dict([(x[0], float(x[1])) for x in data])
length = [x[0] for x in data]
```

```
func = LpProblem('Cost_Minimize', LpMinimize)
k = []
for j in range(0,11):
   k.append(dict([(x[0], float(x[j+3])) for x in data]))
min_th = [1500, 30, 20, 800, 130, 125, 60, 1000, 400, 700, 10]
max_th = [2500, 240, 70, 2000, 450, 250, 100, 10000, 5000, 1500, 40]
```

	Foods	Price/ Serving	Serving Size	Calories	Cholesterol mg	Total_Fat g	Sodium mg	Carbohydrates g	Dietary_Fiber g	Protein g	Vit_A IU	Vit_C IU	Calcium mg	Iron mg
65	NaN	NaN	Minimum daily intake	1500.0	30.0	20.0	800.0	130.0	125.0	60.0	1000.0	400.0	700.0	10.0
	Foods	**Price/ Serving**	**Serving Size**	**Calories**	**Cholesterol mg**	**Total_Fat g**	**Sodium mg**	**Carbohydrates g**	**Dietary_Fiber g**	**Protein g**	**Vit_A IU**	**Vit_C IU**	**Calcium mg**	**Iron mg**
66	NaN	NaN	Maximum daily intake	2500.0	240.0	70.0	2000.0	450.0	250.0	100.0	10000.0	5000.0	1500.0	40.0

```
# Define the continuous and binary variables for foods and foods eaten. Func is defined to be the total cost of foods.
var_co = LpVariable.dicts('foods', foods,0)
var_bi = LpVariable.dicts('eaten',foods,0,1,LpBinary)
x = LpVariable.dicts('x', foods, 0)
func += lpSum([cost[c] * x[c] for c in length])

# Add constraint to the cost function by restricting cost function to be within the max and min daily nutrition constraints.
for i in range(0,10):
   cost = pulp.lpSum([k[i][j] * x[j] for j in foods])
   constraint = min_th[i] <= cost
   func += constraint
   cost = pulp.lpSum([k[i][j] * x[j] for j in foods])
   constraint = max_th[i] >= cost
   func += constraint

# Solve the function with the given constraint. Yields the optimal combination of foods and total cost.
func.solve()
for opti in func.variables():
   if opti.varValue > 0:
         print(opti.varValue, opti)
print('Total Cost = $' + str(round(func.objective.value(),2)))
52.64371 x_Celery,_Raw
0.25960653 x_Frozen_Broccoli
63.988506 x_Lettuce,Iceberg,Raw
2.2929389 x_Oranges
0.14184397 x_Poached_Eggs
13.869322 x_Popcorn,Air_Popped
Total Cost = $4.34

# For Part 2, the three new constraints are added to the cost function. For selected food in foods, the food must be greater than or equal than .1 serving. Combination of celery and frozen broccoli cannot be greater than 1, and the selection of foods that satisfy meat/poultry/fish/egg
```

must be greater than or equal to 3. The total cost with the new constraint is the same as the previous total cost.

```
for s in foods:
   func += var_co[s] >= .1 * var_bi[s]
func += var_bi['Celery, Raw'] + var_bi['Frozen Broccoli'] <= 1
func += var_co['Roasted Chicken'] + var_co['Poached Eggs'] \
      + var_co['Scrambled Eggs'] + var_co['Bologna,Turkey'] \
      + var_co['Frankfurter, Beef'] + var_co['Ham,Sliced,Extralean'] \
      + var_co['Kielbasa,Prk'] + var_co['Pizza W/Pepperoni'] \
      + var_co['Hamburger W/Toppings'] + var_co['Hotdog, Plain'] \
      + var_co['Pork'] + var_co['Sardines in Oil'] \
      + var_co['White Tuna in Water'] + var_co['Chicknoodl Soup'] \
      + var_co['Splt Pea&Hamsoup'] + var_co['Vegetbeef Soup'] \
      + var_co['Neweng Clamchwd'] + var_co['New E Clamchwd,W/Mlk'] \
      + var_co['Beanbacn Soup,W/Watr'] >= 3
func.solve()
print('Total Cost = $' + str(round(func.objective.value(),2)))
```

```
Total Cost = $4.34
```