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. UG!)

In [1]:
```
# import libraries
import pandas as pd
import pulp
```

In [2]:
```
# import data
data = pd.read_excel("week 7 data-summer/data 15.2/diet.xlsx")
data
```

Out[2]:

	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
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
62	Crm Mshrm Soup,W/Milk	0.65	1 C (8 Fl Oz)	203.4	19.8	13.6	1076.3	15.0	0.5	6.1	153.8	2.2	178.6	0.6
63	Beanbacn Soup,W/Watr	0.67	1 C (8 Fl Oz)	172.0	2.5	5.9	951.3	22.8	8.6	7.9	888.0	1.5	81.0	2.0
64	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
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
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

67 rows × 14 columns

In [3]:
```
# separate data into data1(food) data2(minimum intake)
data1 = data[:-3]
data2 = data[-3:]
```

In [4]:
```
# get list of nutrients
nutrients = list(data.columns[3:])

# get minimum intake
min_data = data2[1:2][nutrients]
min_dict = {i : j for (i,j) in zip(list(min_data.columns), min_data.values[0])}

# get maximum intake
max_data = data2[2:3][nutrients]
max_dict = {i : j for (i,j) in zip(list(max_data.columns), max_data.values[0])}
```

In [5]:
```
# define variables for optimization later
ingred = list(data1['Foods'])
costs = {i : j for (i,j) in zip(ingred, list(data1["Price/ Serving"]))}
prob = pulp.LpProblem(name="Diet Problem", sense=pulp.LpMinimize)
ingred_var = pulp.LpVariable.dicts("Ingr", ingred, 0, 100)
chosen = pulp.LpVariable.dicts("Chosen", ingred, 0, 1, cat="Binary")
```

```
C:\Users\User\anaconda3\Lib\site-packages\pulp\pulp.py:1316: UserWarning: Spaces are not permitted in the name. Converted to '_'
  warnings.warn("Spaces are not permitted in the name. Converted to '_'")
```

In [6]:
```
# optimization
prob += pulp.lpSum(costs[i] * ingred_var[i] for i in ingred), 'Total Cost'
```

In [7]:
```
# for each nutrient
for nutrient in nutrients:
    # constraints
    assert len(ingred) == len(data1)
    ingredient_content = {i : j for (i,j) in zip(ingred, list(data1[nutrient]))}
    assert len(ingredient_content) == len(ingred_var)
    prob += pulp.lpSum([ingredient_content[i] * ingred_var[i] for i in ingred]) >= min_dict[nutrient], 'min{}'.format(nutrient)
    prob += pulp.lpSum([ingredient_content[i] * ingred_var[i] for i in ingred]) <= max_dict[nutrient], 'max{}'.format(nutrient)
```

In [8]:
```
# solve
prob.solve()
result_dict = {}
ingr_name = []
ingr_amount = []

for i in prob.variables():
    result_dict[i.name] = i.varValue
    if i.varValue > 0:
        ingr_name.append(i.name[5:])
        ingr_amount.append(i.varValue)

ingr_name.append('Total Cost')
ingr_amount.append(pulp.value(prob.objective))

results = pd.DataFrame({'Amount' : ingr_amount})
results.index = ingr_name
print(results)
```

```
                    Amount
Celery,_Raw          52.643710
Frozen_Broccoli       0.259607
Lettuce,Iceberg,Raw  63.988506
Oranges               2.292939
Poached_Eggs          0.141844
Popcorn,Air_Popped   13.869322
Total Cost            4.337117
```

---

2. Please add to your model the following constraints (which might require adding more variables) and solve the new model:

- 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 is is chosen, and how much is part of the diet. You'll also need to write a constraint to link them.)
- Many people dislike celery and frozen broccoli. So at most one, but not both, can be selected.
- 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 dont really care whether we agree on how to classify foods!

In [9]:
```
# import data
data = pd.read_excel("week 7 data-summer/data 15.2/diet.xlsx")
data
```

Out[9]:

	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
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
62	Crm Mshrm Soup,W/Milk	0.65	1 C (8 Fl Oz)	203.4	19.8	13.6	1076.3	15.0	0.5	6.1	153.8	2.2	178.6	0.6
63	Beanbacn Soup,W/Watr	0.67	1 C (8 Fl Oz)	172.0	2.5	5.9	951.3	22.8	8.6	7.9	888.0	1.5	81.0	2.0
64	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
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
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

67 rows × 14 columns

In [10]:
```
# separate data into data1(food) data2(minimum intake)
data1 = data[:-3]
data2 = data[-3:]
```

In [11]:
```
# get list of nutrients
nutrients = list(data.columns[3:])

# get minimum intake
min_data = data2[1:2][nutrients]
min_dict = {i : j for (i,j) in zip(list(min_data.columns), min_data.values[0])}

# get maximum intake
max_data = data2[2:3][nutrients]
max_dict = {i : j for (i,j) in zip(list(max_data.columns), max_data.values[0])}
```

In [12]:
```
# define variables for optimization later
ingred = list(data1['Foods'])
costs = {i : j for (i,j) in zip(ingred, list(data1["Price/ Serving"]))}
prob = pulp.LpProblem(name="Diet Problem", sense=pulp.LpMinimize)
ingred_var = pulp.LpVariable.dicts("Ingr", ingred, 0, 100)
chosen = pulp.LpVariable.dicts("Chosen", ingred, 0, 1, cat="Binary")
```

```
C:\Users\User\anaconda3\Lib\site-packages\pulp\pulp.py:1316: UserWarning: Spaces are not permitted in the name. Converted to '_'
  warnings.warn("Spaces are not permitted in the name. Converted to '_'")
```

In [13]:
```
# optimization
prob += pulp.lpSum(costs[i] * ingred_var[i] for i in ingred), 'Total Cost'
```

In [14]:
```
# for each nutrient
for nutrient in nutrients:
    # constraints
    assert len(ingred) == len(data1)
    ingredient_content = {i : j for (i,j) in zip(ingred, list(data1[nutrient]))}
    assert len(ingredient_content) == len(ingred_var)
    prob += pulp.lpSum([ingredient_content[i] * ingred_var[i] for i in ingred]) >= min_dict[nutrient], 'min{}'.format(nutrient)
    prob += pulp.lpSum([ingredient_content[i] * ingred_var[i] for i in ingred]) <= max_dict[nutrient], 'max{}'.format(nutrient)
```

In [15]:
```
# add part 2 constraints
for ingr in ingred:
    prob += chosen[ingr] <= ingred_var[ingr] * 99999999999999
    prob += ingred_var[ingr] >= chosen[ingr] * 0.1

prob += chosen["Celery, Raw"] + chosen["Frozen Broccoli"] <= 1
prob += chosen["Tofu"] + chosen["Roasted Chicken"] + chosen["Poached Eggs"] + chosen["Scrambled Eggs"] + chosen["Bologna,Turkey"] +\
        chosen["Frankfurter, Beef"] + chosen["Ham,Sliced,Extralean"] + chosen["Kielbasa,Prk"] + chosen["Hamburger W/Toppings"] +\
        chosen["Hotdog, Plain"] + chosen["Pork"] + chosen["Sardines in Oil"] + chosen["White Tuna in Water"] >= 3
```

In [16]:
```
# solve
prob.solve()
result_dict = {}
ingr_name = []
ingr_amount = []

for i in prob.variables():
    result_dict[i.name] = i.varValue
    if i.varValue > 0:
        ingr_name.append(i.name[5:])
        ingr_amount.append(i.varValue)

ingr_name.append('Total Cost')
ingr_amount.append(pulp.value(prob.objective))

results = pd.DataFrame({'Amount' : ingr_amount})
results.index = ingr_name
print(results)
```

```
                    Amount
n_Poached_Eggs       1.000000
n_Scrambled_Eggs     1.000000
n_Tofu               1.000000
Celery,_Raw         52.298911
Frozen_Broccoli      0.248768
Lettuce,Iceberg,Raw  64.546946
Oranges              2.325006
Poached_Eggs         0.100000
Popcorn,Air_Popped   13.852241
Scrambled_Eggs       0.100000
```

Tofu	0.100000
Total Cost	4.375539