1. Initially, I was able to solve this problem by setting up **many** dictionaries (refer the 15.2.1diet.pdf attached to this assignment). The dictionaries I created were: price_per_serving (which used the foods and the price per foods, this dictionary would be used in the main problem as it's the part we are trying to minimize), calories, cholesterol, total_fat, sodium, carbs, fiber, protein, vit_A, vit_C, calcium, iron, and a food_nutr_dict (which was a nested dictionary that took every food and paired it with the the nutrient listing and values it corresponded with). Since I created a dictionary for every single nutrient value I was able to set up 22 constraint equations (a minimum and maximum constraint equation for the 11 nutrients).

Solving these problem gave me the following solution:

- AmountFood_Celery,_Raw = 52.64371
- AmountFood_Frozen_Broccoli = 0 AmountFood_Celery,_Raw = 52.64371
- AmountFood_Frozen_Broccoli = 0.25960653
- AmountFood_Lettuce,Iceberg,Raw = 63.988506
- AmountFood_Oranges = 2.2929389
- AmountFood_Poached_Eggs = 0.14184397
- AmountFood_Popcorn,Air_Popped = 13.869322
- Total cost of diet = 4.337116797399999

However, I was not satisfied with my code for this problem because it did not seem like an effective approach (manually creating dictionaries per nutrient and manually creating constraint equations for min/max nutrient values). I knew that a loop could be implemented and would allow me to work with larger data sets more effectively.

I was able to create a for loop (refer to the 15.2.1.dietupdate.pdf attached to this assignment) that looped through all the different min and max values provided for the different nutrients. This for loop also did not need me to create individual dictionaries for the nutrients values and was able to work solely with the nested food_nutr_dict. It provides the same solution and sets me up for working with the large diet data set.

1. I added these three constraints to the original code (without the for loop), but would be implemented the same way in my updated code. Please refer to the 15.2.2.diet.pdf file.

If a food is chosen the diet must include at least 1/10 of the serving. To include this constraint we created a new variable (an integer variable that can take the value of 0 or 1, so it acts as a binary decision). We then update our diet minimization problem to compare the binary chosen food variables to our original chosen food variable that we used in the first problem. This is the code:

```
# Setting up our decision variables
#1. we do not have to select all of the foods, so we will set a decision variable for either using the food or not using the food
chosen_food_vars = LpVariable.dicts("Food Chosen", foods, 0, 1, cat = "Integer")

#2. choosing foods variable decision
food_vars = LpVariable.dicts("Food", foods, 0)

# if a food is chosen at least 1/10 of the serving size must be selected
for f in foods:
    prob += food_vars[f] >= chosen_food_vars[f]*0.1
    prob += food_vars[f] <= chosen_food_vars[f]*1000000
```

Looking at the last three lines, the for loop, we are comparing the original chosen foods to the binary variable. If the food is chosen (Integer variable is 1), then both constraint equations are true; however, if the food is not chosen (Integer variable is 0), then the second equation will always be false and that food will not make it to the solution.

To ensure only celery or broccoli was chosen I added this constraint to the code:

```
# choose only celery or broccoli
prob += chosen_food_vars['Frozen Broccoli'] + chosen_food_vars['Celery, Raw'] <= 1
```

Using our binary variable, we are telling the problem that Broccoli and Celery cannot both be equal to 1. In other words, both Broccoli and Celery cannot be chosen.

To make sure our diet includes at least 3 kinds of meat/poultry/fish/eggs:

```
# at least 3 kinds of meat/poultry/fish/eggs
prob += chosen_food_vars['Roasted Chicken'] + chosen_food_vars['Poached Eggs'] +
chosen_food_vars['Scrambled Eggs'] + chosen_food_vars['Bologna,Turkey'] + chosen_food_vars['Frankfurter, Beef']
+ chosen_food_vars['Ham,Sliced,Extralean'] + chosen_food_vars['Kielbasa,Prk'] + chosen_food_vars['Pizza W/Pepperoni']
+ chosen_food_vars['Taco'] + chosen_food_vars['Hamburger W/Toppings'] + chosen_food_vars['Hotdog, Plain']
+ chosen_food_vars['Pork'] + chosen_food_vars['Sardines in Oil'] + chosen_food_vars['White Tuna in Water']
+ chosen_food_vars['Chicknoodl Soup'] + chosen_food_vars['Splt Pea&Hamsoup'] + chosen_food_vars['Vegetbeef Soup']
+ chosen_food_vars['Beanbacn Soup,W/Watr'] >= 3››
```

All the foods that I considered to be part of those food groups I added to this constraint and, similar to the Broccoli/Celery constraint, I indicated that at least 3 from this list must be chosen.

The solution with our added constraints is:

- Food_Celery,_Raw = 42.399358
- Food_Chosen_Celery,_Raw = 1.0
- Food_Chosen_Kielbasa,Prk = 1.0
- Food_Chosen_Lettuce,Iceberg,Raw = 1.0
- Food_Chosen_Oranges = 1.0
- Food_Chosen_Peanut_Butter = 1.0
- Food_Chosen_Poached_Eggs = 1.0
- Food_Chosen_Popcorn,Air_Popped = 1.0
- Food_Chosen_Scrambled_Eggs = 1.0
- Food_Kielbasa,Prk = 0.1
- Food_Lettuce,Iceberg,Raw = 82.802586
- Food_Oranges = 3.0771841
- Food_Peanut_Butter = 1.9429716
- Food_Poached_Eggs = 0.1
- Food_Popcorn,Air_Popped = 13.223294
- Food_Scrambled_Eggs = 0.1
- Total cost of diet = 4.512543427000001

Large Diet

From my updated code for the first problem I was able to use the same constraint for loop that uses the food_nutr_dict (this helped me avoid having to create ~60 constraint equations, actually 54 because 3 nutrients did not have indicated min/max constraints).

I also had to run a quick excel macro to insert 0 into all missing values. I made the assumption that if the food did not have the corresponding nutrient then that nutrient's value is 0.

Please refer to the dietlarge.pdf file attached to this assignment.

The solution from running the optimization code is:

- AmountFood_Beans,_adzuki,_mature_seeds,_raw = 0.26287569
- AmountFood_Broccoli_raab,_raw = 0.050577004
- AmountFood_Cocoa_mix,_no_sugar_added,_powder = 0.71520838
- AmountFood_Egg,_white,_dried,_flakes,_glucose_reduced = 0.087630722
- AmountFood_Infant_formula,_MEAD_JOHNSON,_ENFAMIL,_NUTRAMIGEN,_with_iron,_p = 0.24888274
- AmountFood_Infant_formula,_MEAD_JOHNSON,_LOFENALAC,_with_iron,_powder,_not = 0.14987872
- AmountFood_Infant_formula,_NESTLE,_GOOD_START_ESSENTIALS__SOY,__with_iron, = 0.54857268
- AmountFood_Infant_formula,_ROSS,_ISOMIL,_with_iron,_powder,_not_reconstitu = 0.30522487
- AmountFood_Lupins,_mature_seeds,_raw = 0.25659834
- AmountFood_Margarine,_vegetable_oil_spread,_60%_fat,_stick,_no_salt = 0.25233075
- AmountFood_Margarine_like_spread,_approximately_60%_fat,_tub,_soybean_(hyd = 0.13465246
- AmountFood_Oil,_vegetable,_teaseed = 0.78334552
- AmountFood_Snacks,_potato_chips,_reduced_fat = 0.75686107
- AmountFood_Tomatoes,_sun_dried,_packed_in_oil,_drained = 0.0017327317
- AmountFood_Water,_bottled,_non_carbonated,_CALISTOGA = 9999.6508
- AmountFood_Wheat,_durum = 0.10897091
- Total cholesterol of diet = 0.0