# ~\Documents\gt2\ISYE6501\HW7\testing.ipynb

```python
from pulp import LpProblem, LpVariable, LpMinimize, lpSum, value
import pandas as pd

# Load the diet data
file_path = r'C:\Users\robed\Documents\gt2\ISYE6501\HW7\diet.xls'
df = pd.read_excel(file_path, sheet_name='Sheet1')

data = df.iloc[:64]
data_list = data.values.tolist()

# Create dictionaries for each nutrient and cost
foods = [item[0] for item in data_list]
cost = {item[0]: float(item[1]) for item in data_list}
calories = {item[0]: float(item[3]) for item in data_list}
cholesterol = {item[0]: float(item[4]) for item in data_list}
total_fat = {item[0]: float(item[5]) for item in data_list}
sodium = {item[0]: float(item[6]) for item in data_list}
carbs = {item[0]: float(item[7]) for item in data_list}
fiber = {item[0]: float(item[8]) for item in data_list}
protein = {item[0]: float(item[9]) for item in data_list}
vitaminA = {item[0]: float(item[10]) for item in data_list}
vitaminC = {item[0]: float(item[11]) for item in data_list}
calcium = {item[0]: float(item[12]) for item in data_list}
iron = {item[0]: float(item[13]) for item in data_list}

# Define minimum and maximum intake requirements
min_intake = [1500, 30, 20, 800, 130, 125, 60, 1000, 400, 700, 10]
max_intake = [2500, 240, 70, 2000, 450, 250, 100, 10000, 5000, 1500, 40]

# Create a list of nutrient constraints
nutrients = [calories, cholesterol, total_fat, sodium, carbs, fiber, protein, vitaminA, vitaminC, calcium, iron]

# Initialize the linear programming problem
diet_problem = LpProblem('DietOptimization', LpMinimize)

# Define the continuous and binary variables for the foods
food_amounts = LpVariable.dicts("Amount", foods, lowBound=0)
chosen = LpVariable.dicts("Chosen", foods, lowBound=0, upBound=1, cat='Binary')

# Define the objective function to minimize the total cost
diet_problem += lpSum([cost[food] * food_amounts[food] for food in foods]), "Total Cost"

# Add nutrient constraints
for i in range(len(min_intake)):
    nutrient = nutrients[i]
    diet_problem += lpSum([nutrient[food] * food_amounts[food] for food in foods]) >= min_intake[i], f"Min_{i}"
    diet_problem += lpSum([nutrient[food] * food_amounts[food] for food in foods]) <= max_intake[i], f"Max_{i}"

# Add constraint to enforce that if a food is chosen, at least 0.1 servings must be included
for food in foods:
    diet_problem += food_amounts[food] >= 0.1 * chosen[food]
    diet_problem += food_amounts[food] <= 1000 * chosen[food]

# Add constraint to allow at most one of celery and frozen broccoli
diet_problem += chosen['Frozen Broccoli'] + chosen['Celery, Raw'] <= 1, "At most one of Broccoli and Celery"

# Add constraint to ensure at least 3 types of protein sources are chosen
protein_sources = ['Roasted Chicken', 'Poached Eggs', 'Scrambled Eggs', 'Frankfurter, Beef', 'Kielbasa,Prk',
                   'Hamburger W/Toppings', 'Hotdog, Plain', 'Pork', 'Bologna,Turkey', 'Ham,Sliced,Extralean',
                   'White Tuna in Water']
diet_problem += lpSum([chosen[food] for food in protein_sources]) >= 3, "At least three proteins"

# Solve the problem
diet_problem.solve()

# Print the optimal diet solution
print('Optimization Solution:')
for var in diet_problem.variables():
    if var.varValue > 0 and 'Amount' in var.name:
        print(f"{var.varValue:.4f} units of {var.name.replace('Amount_', '')}")

# Print the total cost of the optimal diet
print(f"Total cost of food = ${value(diet_problem.objective):.4f}")
```