Computer Science
Computing for Medicine

April 2018
University of Toronto

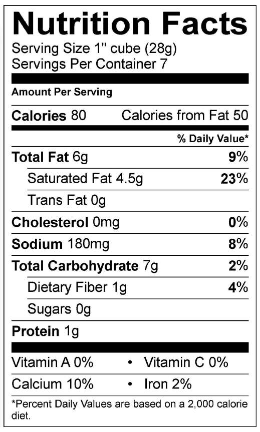
Computing for Medicine, Phase 3, Project #6

Visualizing Information on Nutrition Labels

This assignment will give you experience with visualization and Python programming. Your task is to allow the user to query a collection of nutrition facts about various foods from a JSON file, and plot the result in a visual representation.

———— ♦ ————

Nutrition labels on food products are a major resource for people to make informed decisions about the foods they consume. In their current tabular format (see Figure 1), nutrition facts labels are difficult to navigate and understand without training [1]. Many experts believe it's because of information overload, and a lack of a clear, simple message. For the average consumer, "is 6g of fat too much for 1 serving of cheese?" and "is 1g of protein a good amount?" are difficult questions to answer, due to an undeveloped sense of scale regarding nutrients[1] and one's personal needs.



Nutrition Facts
Serving Size 1" cube (28g)
Servings Per Container 7

Amount Per Serving	
Calories 80	Calories from Fat 50
	% Daily Value*
Total Fat 6g	**9%**
Saturated Fat 4.5g	**23%**
Trans Fat 0g	
Cholesterol 0mg	**0%**
Sodium 180mg	**8%**
Total Carbohydrate 7g	**2%**
Dietary Fiber 1g	**4%**
Sugars 0g	
Protein 1g	
Vitamin A 0%	• Vitamin C 0%
Calcium 10%	• Iron 2%

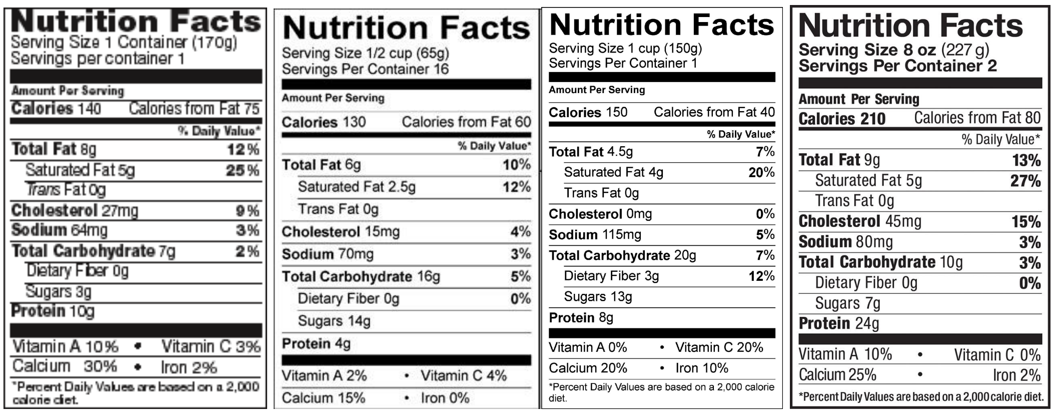
*Percent Daily Values are based on a 2,000 calorie diet.

Figure 1 - Nutrition labels are currently displayed as a large textual table on food packages. Additional quick facts are presented as front-of-package labels, mostly for marketing purposes.

The difficulty in interpreting the information conveyed on food packages is one of the reasons why most consumers ignore or misunderstand these labels [2]. Front-of-package labels have been introduced to provide easy-to-understand nutritional facts, such as "good source of calcium", or "rich in fiber" (see Figure 1). While more accessible to the layman consumer, these labels are loosely regulated and typically designed for marketing purposes, rather than aimed at

[1] See for instance, how much 39g of sugar (contained in a can of coke) represents in terms of sugar cubes: http://www.sugarstacks.com/beverages.htm

better informing consumers. Not a single food package ever advertises that it contains a lot of sugar or a lot of fat!
Another problem is the somewhat deceptive practice of arbitrarily choosing the number of servings per package—often with the goal of lowering the number of calories that appear. Even for a consumer with a fair understanding of nutrition, it remains difficult to assess how much nutrients they truly consume, because serving sizes on labels rarely correspond to their actual intake. A typical example is bottles of a sweetened beverage, listing calories and sugar for 'about half' of the bottle. Many consumers look at the number and mistakenly assume it refers to the entire contents, adding to the confusion. The lack of consistency across packages also makes it tedious to compare the nutritional values of similar foods.



Nutrition Facts
Serving Size 1 Container (170g)
Servings per container 1

Amount Per Serving	
Calories 140	Calories from Fat 75
	% Daily Value*
Total Fat 8g	**12**%
Saturated Fat 5g	**25**%
Trans Fat 0g	
Cholesterol 27mg	**9**%
Sodium 64mg	**3**%
Total Carbohydrate 7g	**2**%
Dietary Fiber 0g	
Sugars 3g	
Protein 10g	
Vitamin A 10% •	Vitamin C 3%
Calcium 30% •	Iron 2%

*Percent Daily Values are based on a 2,000 calorie diet.

Nutrition Facts
Serving Size 1/2 cup (65g)
Servings Per Container 16

Amount Per Serving	
Calories 130	Calories from Fat 60
	% Daily Value*
Total Fat 6g	**10%**
Saturated Fat 2.5g	**12%**
Trans Fat 0g	
Cholesterol 15mg	**4%**
Sodium 70mg	**3%**
Total Carbohydrate 16g	**5%**
Dietary Fiber 0g	**0%**
Sugars 14g	
Protein 4g	
Vitamin A 2% •	Vitamin C 4%
Calcium 15% •	Iron 0%

Nutrition Facts
Serving Size 1 cup (150g)
Servings Per Container 1

Amount Per Serving	
Calories 150	Calories from Fat 40
	% Daily Value*
Total Fat 4.5g	**7%**
Saturated Fat 4g	**20%**
Trans Fat 0g	
Cholesterol 0mg	**0%**
Sodium 115mg	**5%**
Total Carbohydrate 20g	**7%**
Dietary Fiber 3g	**12%**
Sugars 13g	
Protein 8g	
Vitamin A 0% •	Vitamin C 20%
Calcium 20% •	Iron 10%

*Percent Daily Values are based on a 2,000 calorie diet.

Nutrition Facts
Serving Size 8 oz (227 g)
Servings Per Container 2

Amount Per Serving	
Calories 210	Calories from Fat 80
	% Daily Value*
Total Fat 9g	**13%**
Saturated Fat 5g	**27%**
Trans Fat 0g	
Cholesterol 45mg	**15%**
Sodium 80mg	**3%**
Total Carbohydrate 10g	**3%**
Dietary Fiber 0g	**0%**
Sugars 7g	
Protein 24g	
Vitamin A 10% •	Vitamin C 0%
Calcium 25% •	Iron 2%

*Percent Daily Values are based on a 2,000 calorie diet.

Figure 2 - A series of nutrition facts labels found on various yogourts. Note the inconsistencies in serving size across products.

Together, the ineffective representation of nutritional contents and the lack of standards regarding serving size are major barriers to nutrition literacy. While there is a drive to improve food labels, in practice change takes time due to the various government agencies involved in ensuring the quality of any new labeling.

———— ♦ ————

The goal of this project is to develop an interactive visualization that conveys nutritional contents of food in a more accessible and engaging way, with an ultimate goal of promoting healthier lifestyles.

REFERENCES

[1] Peter Helfer, Thomas R. Shultz. The effects of nutrition labeling on consumer food choice: a psychological experiment and computational model. Annals of the New York Academy of Sciences, 2014; 1331 (1): 174.
[2] Graham, D.J., Hodgin, K., & Heidrick, C. (2015). Nutrition label viewing during a food selection
task: Front-of-package labels vs. Nutrition Facts Panels. Journal of the Academy of Nutrition and Dietetics, 115(10), 1636-1646.
[3] Graham, D.J., Mohr, G.S. (2014). When zero is greater than one: Consumer misinterpretations of nutrition labels. Health Psychology, 33(12) 1579-1587.

PART I : Load and visualize the nutritional information of a food item.

This first part consists of 1) loading data in a structured data format, 2) selecting relevant information from this file (i.e. food item and specific nutritional facts), and 3) plotting this information in a bar chart showing daily percentage value.

1. Read and display the data

In this project, we will work with food labels stored in a JSON format since most APIs (e.g. Open Food Facts, MyNetDiary, Spoonacular's food API) provide detailed information in this format.

1.1. Open the `foodfacts.json` file in a text editor, and observe its structure. How are each of the food items characterized?
Draw the corresponding hierarchical data representation of the JSON file.

1.2. In a file named `seminar6_part1.py`, begin by importing the `json` module. Next, write a `load_file` function that loads the JSON file, given a file name. Include the following code snippet in your script:

```
with open(filename, 'r') as open_file:
    json_data = json.load(open_file)
```

In the main block of your script, call your newly defined `load_file` function and store the result in a variable named `data`.

1.3. We will now display the object that `data` refers to.

- Display the `data` object using `print`. Note how difficult it is to parse the hierarchical structure of the data with the basic print function.

To make this more readable, we will use `prettyprint`[2], the "data pretty printer" module that prints the *formatted* representation of an object on the stream.

- Import the `pprint` function from the `pprint` module as `pp`

```
from pprint import pprint as pp
```

- Use pretty print to display the hierarchical data contained in the JSON file that you loaded.

```
pp(data)
```

[2] https://docs.python.org/2/library/pprint.html

Note the difference with `print`.

2. Select and display the nutritional information of one ingredient

The data you loaded contains a lot of information, for numerous food items. We are interested in a subset of this information (e.g., major nutrients), for a single food item. We will now filter the data of interest and plot it in a chart.

2.1. We will first select the food item the user is interested in.

- Define a new `select_food` function with two parameters representing the JSON data and the name of a food, respectively. Return the JSON data for that food as a dictionary.

- In the main block, prompt the user to enter a food, call the `select_food` function to get the data for that food, and then pretty print the JSON data for that food.

2.2. We will now select the nutrition facts that the user is interested in.

- Define a `select_facts` function which takes the JSON data for a food and a list of nutrients, and returns a list of dictionaries with the data for those nutrients.

- Update the main block to repeatedly prompt the user to enter nutrients. Once the user types `quit`, stop prompting. Call the `select_facts` function to get the JSON data for those nutrients and pretty print it.

Here is an example interaction:

```
Enter the food item you want: Carrot juice
Enter a nutrient or type 'quit' to exit: Protein
Enter a nutrient or type 'quit' to exit: Fiber
Enter a nutrient or type 'quit' to exit: Calcium
Enter a nutrient or type 'quit' to exit: quit
[{'amount': 1.7, 'fact': 'Protein', 'percent': 3.4},
 {'amount': 2.5, 'fact': 'Fiber', 'percent': 10},
 {'amount': 44, 'fact': 'Calcium', 'percent': 4.4}]
```

3. Plot the data in a bar chart.

Now that we have the nutritional information for the food item of our interest, the next step will consist of visualizing this information in the form of a bar plot, where each bar corresponds to a nutrient, and its size corresponds to the percent of the recommended daily value. This way, a consumer can better evaluate, out of 100%, whether a food contains a lot, or too little of a specific nutrient.

To do so, we will use the `pyplot`[3] interface of the `matplotlib`[4] module, a popular Python 2D plotting library similar to MATLAB.
We want to plot the data in a horizontal bar chart as follows (the order in which nutrients are displayed is arbitrary here):

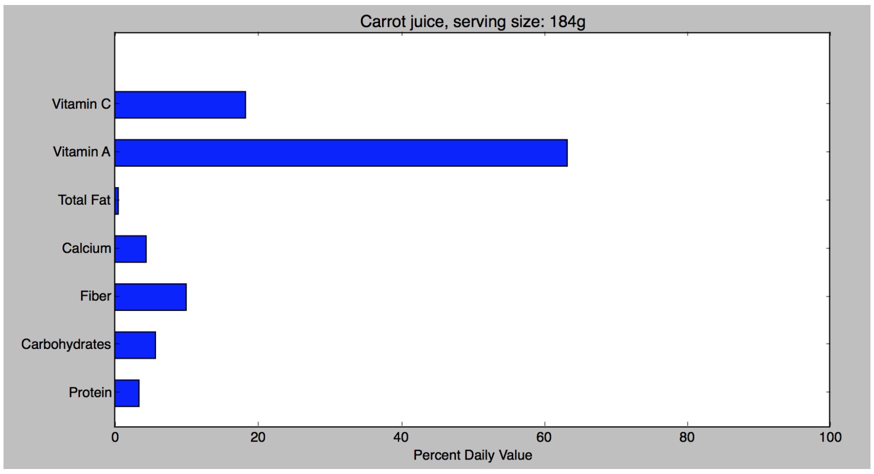


Figure 3 - A horizontal bar chart showing the percent daily value of a series of nutrients, for the 'Carrot juice' food item.

3.1 Search the `matplotlib` documentation for which function of `pyplot` displays a *horizontal bar chart*, and identify the data structure this function needs to build the bars.

A typical task in Python is to *prepare* and *format* the data to pass it to other functions. Here, we will build two tuples: the first tuple containing the nutrients names, and the second containing the daily percentage for these nutrients.

- Using an iterator, build the tuple of nutrient names in a variable named `names` and the tuple of percent daily values in a variable named `values.`

3.2 Now that you have the data at hand in a practical format, plot the horizontal bar chart for the food item and list of nutrients of your choice, as shown in Figure 3. Complete the following steps in order:

- Build a simple horizontal bar chart where each bar corresponds to a nutrient

- Add the `yticks` in your chart, for the viewer to be able to associate the bars with the corresponding nutrient

- Change the `xlimit` of your chart so that one can better read the values out of 100%

3 http://matplotlib.org/api/pyplot_api.html
4 http://matplotlib.org/

- Add a label to the x axis, to specify that the data corresponds to the Percentage Daily Value

- Add a title to your chart so that the viewer knows which food item he/she visualizes the data of, and the corresponding serving size

Part II. Show the data of two different foods for comparison.

In this part, we aim to facilitate comparison of two food items by plotting the nutrition facts concurrently, in the same plot.

In a file named `seminar6_part2.py`, begin by importing `seminar6_part1`. This will allow you to use the functions you defined in that module without duplicating code.

For foods `'Yogurt lowfat fruit added'` and `'Yogurt whole milk'` and nutrients *`Total Fat, Carbohydrates, Cholesterol, Sodium, Protein, Magnesium`* and *`Fiber`*, the expected result is shown in Figure 4.

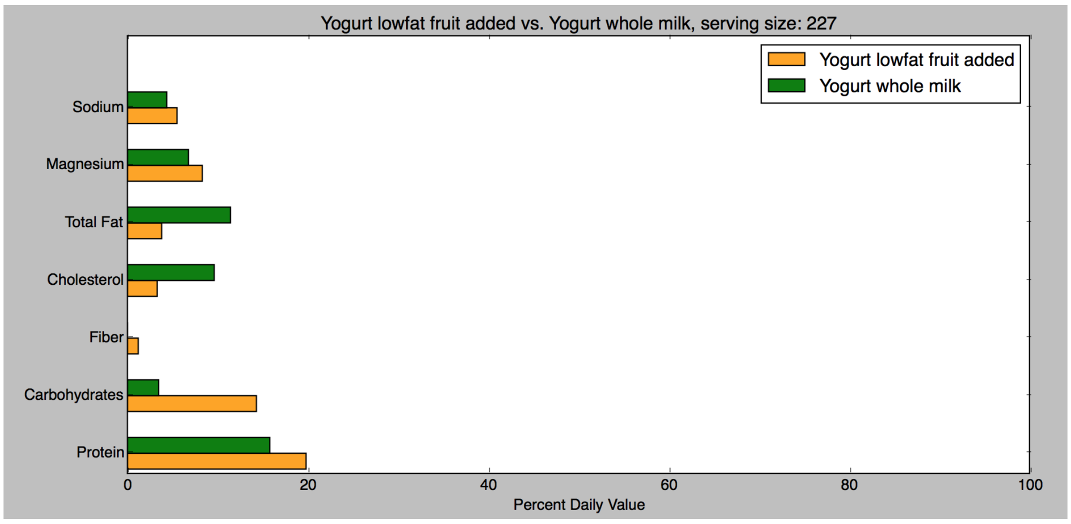


Figure 4 - Comparison of the percent daily value of a series of nutrients for the `'Yogurt lowfat fruit added'` and `'Yogurt whole milk'` food items

1. Normalize the data

Prompt the user for two food items to compare and for nutrients to compare, and load the raw data for those two food items.

Since we are comparing two food items of possibly different serving size, it is necessary to normalize the values to enable comparison. Create a function `normalize_nutrients` which takes the nutrients data (list of dictionaries) and a normalization factor, and returns the same list of dictionaries where the values for 'amount' and 'percent' are multiplied by the normalization factor.

Here is an example of the data before and after normalization:

```
Enter the first food item you want: Yogurt lowfat fruit added
Enter the second food item you want: Yogurt whole milk
Enter a nutrient or type 'quit' to exit: Carbohydrates
Enter a nutrient or type 'quit' to exit: Sodium
Enter a nutrient or type 'quit' to exit: Protein
Enter a nutrient or type 'quit' to exit: quit

------- Values for Yogurt lowfat fruit added, serving size: 227
[{'amount': 9.9, 'fact': 'Protein', 'percent': 19.8},
 {'amount': 43, 'fact': 'Carbohydrates', 'percent': 14.3},
 {'amount': 133, 'fact': 'Sodium', 'percent': 5.5}]

------- Values for Yogurt whole milk, serving size: 151
[{'amount': 5.3, 'fact': 'Protein', 'percent': 10.5},
 {'amount': 7.0, 'fact': 'Carbohydrates', 'percent': 2.3},
 {'amount': 69.2, 'fact': 'Sodium', 'percent': 2.9}]




------- After normalization: Values for Yogurt lowfat fruit added, serv
size: 227
[{'amount': 9.9, 'fact': 'Protein', 'percent': 19.8},
 {'amount': 43, 'fact': 'Carbohydrates', 'percent': 14.3},
 {'amount': 133, 'fact': 'Sodium', 'percent': 5.5}]

------- After normalization: Values for Yogurt whole milk, serving size: 22
[{'amount': 7.967549668874171,
  'fact': 'Protein',
  'percent': 15.784768211920529},
 {'amount': 10.52317880794702,
  'fact': 'Carbohydrates',
  'percent': 3.457615894039735},
 {'amount': 104.02913907284768,
  'fact': 'Sodium',
  'percent': 4.359602649006622}]
```

2. Plot the data

Plot the data for the two food items in the same chart as shown in Figure 3, including the title, the different colors for the bars, and the legend.

Part III. Personalized visualization.

In this part, our goal is to plot the data for one or two food items, this time with values corresponding to one's personal needs.

———— ♦ ————

BACKGROUND

The daily values shown on food packages are based on a 2,000-kilocalorie diet for healthy adults. However, depending on one's sex, age, weight and activity level, the 2,000-kilocalorie reference might be quite off. Energy requirements are based on multiple factors including body composition, training and goals (i.e. weight loss, muscle building). While the formulas to compute the recommended intake are not set in the stone, they constitute a more accurate model to convey percent daily values than the traditional 2,000-kilocalorie per day reference.

In this work, we will use the Harris-Benedict principle [1] to compute an estimate of an individual's basal metabolic rate (BMR) and daily kilocalorie requirements, considering Mifflin and St Jeor equation for BMR [2], which provides a better prediction for modern lifestyles than the original equation.

The BMR Harris–Benedict equations revised by Mifflin and St Jeor in 1990 are:

Men: BMR = (10 × weight in kg) + (6.25 × height in cm) - (5 × age in years) + 5
Women: BMR = (10 × weight in kg) + (6.25 × height in cm) - (5 × age in years) - 161
An individual's recommended daily kilocalorie intake to maintain current weight depending on activity is computed as follows:

Activity level	**Daily kilocalories needed**
Little to no exercise :	BMR x 1.2
Light exercise (1-3 days per week):	BMR x 1.375
Moderate exercise (3-5 days per week):	BMR x 1.55
Heavy exercise (6-7 days per week):	BMR x 1.725
Very heavy exercise (twice per day, extra heavy workouts)	BMR x 1.9

REFERENCES

[1] Harris JA, Benedict FG (1918). "A Biometric Study of Human Basal Metabolism". Proceedings of the National Academy of Sciences of the United States of America. 4 (12): 370–3..
[2] Mifflin MD, St Jeor ST, Hill LA, Scott BJ, Daugherty SA, Koh YO (1990). "A new predictive equation for resting energy expenditure in healthy individuals". The American Journal of Clinical Nutrition. 51 (2): 241–7.

1. Compute the user's recommended daily kilocalorie intake

In a file named `seminar6_part3.py`, begin by prompting the user for their personal information (sex, height, weight, age and activity level), and use the above formulas to compute their recommended daily kilocalorie intake.

An example of interaction is as follows:

```
Enter your sex (F-M): F
Enter your age (in years): 32
Enter your weight (in kg): 62
Enter your height (in cm): 163
Activity levels:
 1. Little to no exercise
 2. Light exercise (1-3 days per week)
 3. Moderate exercise (3-5 days per week)
 4. Heavy exercise (6-7 days per week)
 5. Very heavy exercise (twice per day, extra heavy workouts)
Enter your level of activity (1-5): 2
'BMR: 1317.75'
'Recommended intake (kcal): 2503.725'
```

2. Compute personalized daily values

We will now use the recommended intake information to improve the visualization of Part I.

- Begin by importing `seminar6_part1` to be able to reuse functions from your previous code.

- Prompt the user for the food item and nutrients they want to visualize, and load the corresponding data.

The next step consists of computing the personalized daily value for each nutrient, using the recommended intake computed before.

- The `'percentage'` value in your data must be updated according to the recommended intake. Is it the case for the `'amount'` value as well? Why?

- Let us call `reference_intake` the traditional 2000 kcal reference, `recommended_intake` one's personal recommended intake as computed above, and `reference_value` the daily percent value of a nutrient, based on the 2,000kcal diet. What is the formula to compute the personalized daily percent value of a nutrient?

- In your code, create a `personalized_nutrients` function that takes the list of nutrients and the personalization factor of the precedent question and returns the same list of dictionaries with updated daily percent values.

An example of the expected result is as follows:

```
[...]
'Recommended intake (kcal): 2755.0'

Enter the food item you want: Chocolate Chip cookies commercial

Enter a nutrient or type 'quit' to exit: Iron
Enter a nutrient or type 'quit' to exit: Protein
Enter a nutrient or type 'quit' to exit: Total Fat
Enter a nutrient or type 'quit' to exit: quit
selected nutrients:  ['Iron', 'Protein', 'Total Fat']

[{'amount': 2.3, 'fact': 'Protein', 'percent': 4.6},
 {'amount': 8.8, 'fact': 'Total Fat', 'percent': 13.5},
 {'amount': 0.8, 'fact': 'Iron', 'percent': 4.4}]

--- Personalized values ---
[{'amount': 2.3, 'fact': 'Protein', 'percent': 3.3393829401088926},
 {'amount': 8.8, 'fact': 'Total Fat', 'percent': 9.800362976406534},
  {'amount': 0.8, 'fact': 'Iron', 'percent': 3.194192377495463}]
```

Finally, call the `plot_nutrients` function from Part I to plot the data.

3. Extend your code to support food comparison, using personalized data
In this last step we will extend the code so as to consolidate Part I, Part II and Part III in a single program.

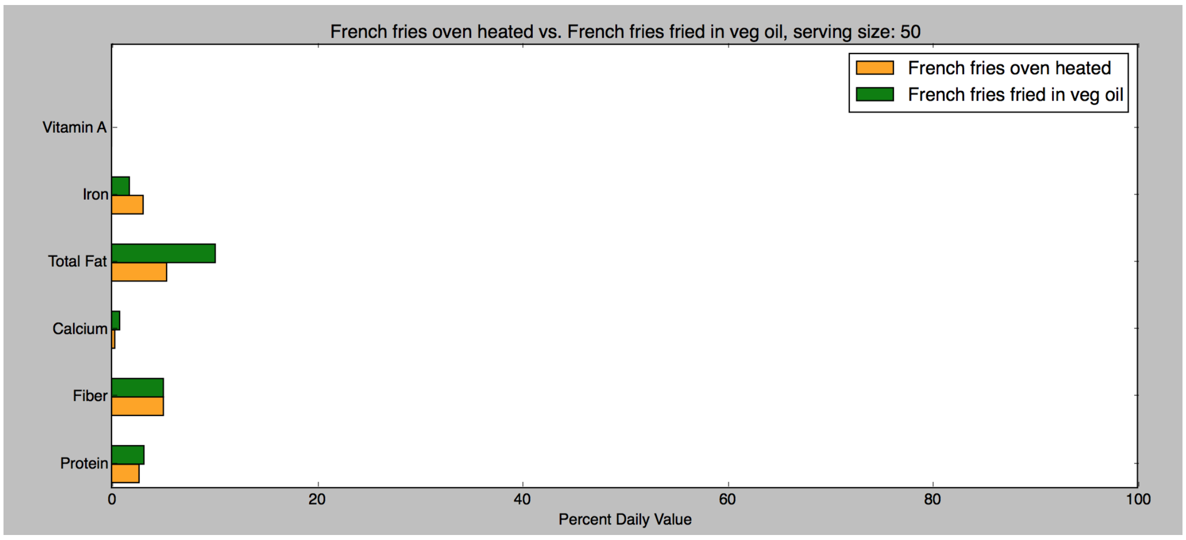
- Modify your code to prompt the user for the choice of visualizing a single food item, or compare two food items, after computing the BMR and recommended intake.

- Add the corresponding `if` statement to either display the information for one food item (i.e. the code of part III.2), or compare two food items (code to add, based on part II, and the `personalized_nutrients` function of part III.2)

An example of the expected result is as follows:

```
Enter your sex (F-M): F
Enter your age (in years): 32
Enter your weight (in kg): 61
Enter your height (in cm): 167
Activity levels:
 1. Little to no exercise
 2. Light exercise (1-3 days per week)
 3. Moderate exercise (3-5 days per week)
 4. Heavy exercise (6-7 days per week)
 5. Very heavy exercise (twice per day, extra heavy workouts)
Enter your level of activity (1-5): 3
'BMR: 1332.75'
'Recommended intake (kcal): 2532.225'
Do you want to visualize
 1. one food item
 2. two food items
Answer:2
Enter the first food item you want: French fries oven heated
Enter the second food item you want: French fries fried in veg oil
Enter a nutrient or type 'quit' to exit: Total Fat
Enter a nutrient or type 'quit' to exit: Protein
Enter a nutrient or type 'quit' to exit: Calcium
Enter a nutrient or type 'quit' to exit: Fiber
Enter a nutrient or type 'quit' to exit: Iron
Enter a nutrient or type 'quit' to exit: Vitamin A
Enter a nutrient or type 'quit' to exit: quit
------- After normalization: Values for French fries oven heated, serving size: 50
[{'amount': 1.7, 'fact': 'Protein', 'percent': 2.6853853824205984},
 {'amount': 1.6, 'fact': 'Fiber', 'percent': 5.054843072791716},
 {'amount': 4, 'fact': 'Calcium', 'percent': 0.31592769204948223},
 {'amount': 4.4, 'fact': 'Total Fat', 'percent': 5.370770764841197},
 {'amount': 0.7, 'fact': 'Iron', 'percent': 3.080294997482451},
 {'amount': 0, 'fact': 'Vitamin A', 'percent': 0.0}]


------- After normalization: Values for French fries fried in veg oil, serving size: 50
[{'amount': 2.0, 'fact': 'Protein', 'percent': 3.159276920494822},
 {'amount': 1.6, 'fact': 'Fiber', 'percent': 5.054843072791716},
 {'amount': 10.0, 'fact': 'Calcium', 'percent': 0.7898192301237055},
 {'amount': 8.3, 'fact': 'Total Fat', 'percent': 10.109686145583431},
 {'amount': 0.4, 'fact': 'Iron', 'percent': 1.7376023062721522},
 {'amount': 0.0, 'fact': 'Vitamin A', 'percent': 0.0}]
```


French fries oven heated vs. French fries fried in veg oil, serving size: 50
French fries oven heated
French fries fried in veg oil
Vitamin A
Iron
Total Fat
Calcium
Fiber
Protein
0
20
40
60
80
100
Percent Daily Value

APPENDIX

List of foods contained in the `foodfacts.json` file, by category

. Baked goods
Apple pie
Bagel, plain
Bread crumbs, dry grated
Carrot cake, cream cheese
Cheese crackers
Chocolate Chip cookies commercial
Corn Chips
Croissant 4 1/2 x 4 x 1 3/4 in
English muffin, plain
Gingerbread
Graham crackers
Melba toast, plain
Mixed grain toasted
Muffins Blueberry, commercial
Peach pie, piece
Pretzels, thin sticks
Raisin bread slice
Rye wafers, whole grain
Rye, sliced
Saltine crackers
Snack cakes, chocolate
Wheat bread, sliced
Wheat cracker, thin
White bread, sliced
Yellow cake

. Fruits
Apple juice
Apples raw with peel 2 3/4 diam
Apricot nectar canned
Avocado Calif 1/2 lb with refuse
Banana raw without peel
Blackberries raw
Blueberries raw
Cantaloupe
Cherries sweet raw without pits
Dates whole without pits
Honeydew
Mango raw edible part
Nectarines raw without pits
Orange juice
Peaches juice pack
Peaches raw whole
Pears raw Bartlett
Pineapple raw chunks, dices
Pineapple, juice canned
Plantains, cooked, boiled,sliced
Plums
Prune juice bottled or canned
Raisins, seedless
Rhubarb cooked added sugar
Strawberries thawed measure

. Beverages
Beer Light (12 fl oz)
Beer Regular (12 fl oz)
Carbonated Cola (12 fl oz)
Carbonated Diet Cola (12 fl oz)
Coffee Brewed
Coffee Instant
Gin, Rum, Vodka, Whiskey 80 proof
Grape Drink Canned
Lemonade concentrate
Orange (12 fl oz)
Root Beer (12 fl oz)
Tea Brewed

. Eggs
Raw white
Raw whole without shell
Raw Yolk
Scrambled with milk and butter

. Grain products
Egg noodles, cooked
Grape Nuts cereal
Lucky Charms cereal
Macaroni, cooked firm stage hot
Popcorn, air popped plain
Rice, brown cooked
Spaghetti, cooked firm stage hot
Trix cereal
White rice, raw dry

. Meat
Beef - Liver
Beef - Steak lean and fat
Beef dried chipped
Canadian style Bacon
Ground beef lean
Ground beef regular
Ham luncheon meat canned
Lamb - Chops lean and fat
Lamb leg roasted lean and fat
Pork - Chops, Broiled lean & fat
Pork, Bacon medium slices
Pork, rib roasted lean and fat
Veal cutlet braised or broiled

. Misc
Chili powder
Pepper black
Pickles dill medium

. Dairy
Buttermilk
Cheese - Blue
Cheese - Cheddar cut pieces
Cheese - Cream
Cheese - Monterey Jack
Cheese - Swiss
Cheese Gouda
Cheese Mozzarella whole milk
Cottage Creamed small curd
Cream sour cultured
Custard baked
Egg Nog commercial
Ice Cream Vanilla
Ice Milk Vanilla
Kefir
Milk - 2% Low Fat
Milk - Whole
Pudding canned Chocolate
Sherbert (2% fat)
Skim Milk
Yogurt lowfat fruit added
Yogurt whole milk

. Mixed dishes and fast food
Burrito, beef and bean
Cole slaw
Corn dog
Enchilada
Spaghetti in sauce w/cheese

. Nuts, seeds and products
English walnuts, chopped
Peanut Butter
Pine nuts/pinyon dried

. Poultry
Chicken - Roasted whole
Chicken breast fried w/batter
Duck meat only roasted
Turkey - Roasted whole, slices

. Sausages and lunchmeat
Frankfurter, beef and pork
Frankfurter, turkey

. Soups, sauces and gravies
Beef gravy canned
Bouillon

. Sweets
Marshmallows

. Fats and oils
Butter - Stick
Lard
Margarine - Regular, hard
Margarine Spread (60% fat) hard
Oil - Corn
Oil - Olive
Oil - Peanut
Safflower
Salad Dressing - Blue Cheese
Salad Dressing - French Regular
Salad Dressing - Italian Regular
Salad Dressing 1000 Island
Salad Dressing Mayonnaise
Soybean

. Fish and shellfish
Fishsticks
Oysters Eastern raw
Salmon, canned pink solids
Trout broiled w/butter & lemon
Tuna, oil packed
Clams - raw
Oysters - raw Pacific
Salmon - Broiled or baked

. Vegetables and legumes
Dandelion greens (raw)
Asparagus, raw cuts and tips
Bamboo shoots, canned and sliced
Beets cooked sliced or diced
Blackeyed peas frozen drained
Broccoli - Cooked, raw, spears
Brussel sprouts cooked raw
Carrot juice
Carrots whole raw
Cauliflower - Cooked, raw
Celery, pascal large stalk
Corn cooked raw on cob
Cucumber with peel
Eggplant cooked
French fries fried in veg oil
French fries oven heated
Lettuce Butterhead/Boston
Lima Beans, thick seeded
Mushrooms raw sliced
Onions raw chopped
Peas, green canned drained
Pinto Beans cooked from dry
Potato chips
Potatoes mashed
Spinach raw chopped
Tomatoes raw whole
Vegetable juice cocktail canned
Water chestnuts canned sliced