CS50's Introduction to Programming with Python

OpenCourseWare

Donate (https://cs50.harvard.edu/donate)

David J. Malan (https://cs.harvard.edu/malan/)
malan@harvard.edu
(https://www.facebook.com/dmalan) (https://github.com/dmalan)
(https://www.instagram.com/davidjmalan/) (https://www.linkedin.com/in/malan/)
(https://www.reddit.com/user/davidjmalan)
(https://www.threads.net/@davidjmalan) (https://twitter.com/davidjmalan)

Nutrition Facts

The U.S. Food & Drug Adminstration (FDA) offers downloadable/printable posters (https://www.fda.gov/food/food-labeling-nutrition/nutrition-information-raw-fruits-vegetables-and-fish) that "show nutrition information for the 20 most frequently consumed raw fruits … in the United States. Retail stores are welcome to download the posters, print, display and/or distribute them to consumers in close proximity to the relevant foods in the stores."

In a file called `nutrition.py`, implement a program that prompts ~~consumers~~ users to input a fruit (case-insensitively) and then outputs the number of calories in one portion of that fruit, per the FDA's poster for fruits, which is also available as text (https://www.fda.gov/food/food-labeling-nutrition/raw-fruits-poster-text-version-accessible-version). Capitalization aside, assume that users will input fruits exactly as written in the poster (e.g., `strawberries`, not `strawberry`). Ignore any input that isn't a fruit.

▼ **Hints**

- Rather than use a conditional with 20 Boolean expressions, one for each fruit, better to use a `dict` to associate a fruit with its calories!
- If `k` is a `str` and `d` is a `dict`, you can check whether `k` is a key in `d` with code like:

```
if k in d:
    ...
```

- Take care to output the fruit's calories, not calories from fat!

Demo

Before You Begin

Log into cs50.dev (https://cs50.dev/), click on your terminal window, and execute `cd` by itself. You should find that your terminal window's prompt resembles the below:

```
$
```

Next execute

```
mkdir nutrition
```

to make a folder called `nutrition` in your codespace.

Then execute

```
cd nutrition
```

to change directories into that folder. You should now see your terminal prompt as `nutrition/ $`. You can now execute

```
code nutrition.py
```

to make a file called `nutrition.py` where you'll write your program.

How to Test

Here's how to test your code manually:

- Run your program with `python nutrition.py`. Type `Apple` and press Enter. Your program should output:

  ```
  Calories: 130
  ```

- Run your program with `python nutrition.py`. Type `Avocado` and press Enter. Your program should output:

  ```
  Calories: 50
  ```

- Run your program with `python nutrition.py`. Type `Sweet Cherries` and press Enter. Your program should output

  ```
  Calories: 100
  ```

- Run your program with `python nutrition.py`. Type `Tomato` and press Enter. Your program should output nothing.

Be sure to try other fruits and vary the casing of your input. Your program should behave as expected, case-insensitively.

You can execute the below to check your code using `check50`, a program that CS50 will use to test your code when you submit. But be sure to test it yourself as well!

```
check50 cs50/problems/2022/python/nutrition
```

Green smilies mean your program has passed a test! Red frownies will indicate your program output something unexpected. Visit the URL that `check50` outputs to see the input `check50` handed to your program, what output it expected, and what output your program actually gave.

How to Submit

In your terminal, execute the below to submit your work.

```
submit50 cs50/problems/2022/python/nutrition
```