Practicum 2: Input, Variables, and Food Rescue

Due Date: September 17th, 2021 at 11:59pm EST

In the United States, food waste is estimated to be between 30 and 40% of the food supply. This waste comes from all points along the supply chain from the time it is grown to the time that consumers acquire it. A significant portion of this waste comes from grocery stores throwing away or composting produce that is not aesthetically pleasing or that has gotten a bit old. Boston has a significant number of organizations who work to do “just in time food rescue,” where volunteers and employees bring still edible food that would have been disposed of to communities experiencing food insecurity.

Today, we’ll use the case of food rescue to practice the following:

1. Handling user input
2. Printing and formatting strings
3. Converting types of variables

1 How Many Potatoes?

Write Python code that does the following:

1. Prompts the user for the number of volunteers that they have available (an integer)
2. Calculates the total number of pounds that can be rescued, assuming two different conditions:
    a. Volunteers have cars and can transport **500 pounds of potatoes**
    b. Volunteers have bicycles with trailers and can transport **260 pounds of potatoes**
3. Prints the total number of pounds that can be rescued under both conditions

The program output should look like this (user input in **bold underlined text**):

```
How many volunteers are available? 8
If they have cars, your 8 volunteers can rescue 4000 lbs of potatoes!
If they have bicycles, your 8 volunteers can rescue 2080 lbs of
potatoes!
```

Now, instead of telling the user how many pounds of potatoes they can rescue, we want to tell them the number of potatoes they can rescue, assuming that the average potato weighs **0.38 pounds.**

Write Python code that does the following:

1. Calculates the number of potatoes that can be rescued by volunteers with cars, using the same number of volunteers from the input to the previous set of code

2. Calculates the number of potatoes that can be rescued by volunteers with bikes
3. Prints the total number of potatoes that can be rescued under both conditions.

The program output should look like this:

```
If they have cars, your 8 volunteers can rescue 10562.3 potatoes!
If they have bicycles, your 8 volunteers can rescue 5473.7 potatoes!
```

2 Recommending Volunteers

Next, we'll calculate the inverse of this problem. Suppose that you have a grocery store that calls up your food rescue organization and says, "We have 1,307 pounds of potatoes for you today!" How many volunteers do you need to rescue it?

Write Python code that does the following:

1. Prompts the user for the name of the grocery store
2. Prompts the user for the number of pounds of potatoes that the store has available (an integer or a float)
3. Calculates the minimum number of volunteers that are needed to rescue the food, assuming two different conditions:
    a. Volunteers have cars and can transport 500 pounds of food
    b. Volunteers have bicycles with trailers and can transport 260 pounds of food
4. Prints the number of volunteers needed to rescue the food

The program output should look like this (user input in **bold underlined text**):

```
What is the store's name? Happy Market
How many pounds of potatoes does Happy Market have to pick up? 75435
If they have cars, you need 150.87 volunteers!
If they have bicycles, you need 290.13461538461536 volunteers!
```

3 No Partial Volunteers

We can't have a fractional number of volunteers, so we want to clean up our code so that we report the smallest whole number of volunteers that are needed to pick up all of the food.

First, try out Python's built-in `round()` function. Does it behave how you expect it to?

Next, try the `math.ceil()` and `math.floor()` functions. To use these, you'll need to import the `math` module, which is a collection of functions that come with Python by default, but that we need to explicitly include if we want to use them. I recommend including, or "importing," it as the very first line of code.

For example:

```
```
    DS2001
    Ryan Gallagher
```
import math

# Part 1
# part 1 code
# ...

# Part 2
# part 2 code
# ...


# Part 3
example_math_ceil = math.ceil(4.13)
print(example_math_ceil)
example_math_floor = math.floor(4.13)
print(example_math_floor)
```

Edit your Python code from the previous section to print out a whole number of volunteers.

4 Using Constants

Constants are variables that hold unchanging values, like conversion factors or a car's carrying capacity. We typically name constants with all capital letters, `LIKE_THIS`. We're going to add some constants at the beginning of our program.

Define three constants:

1. A volunteer's carrying capacity if they have a car, in pounds
2. A volunteer's carrying capacity if they have a bicycle with a trailer, in pounds
3. The weight of an average potato, in pounds

Wherever you refer to any of these values in your code, replace them with the appropriate constant.

For example:

```
import math
```

```
# A volunteer with a bike can carry 100 lbs
BIKE_CAPACITY = 100

# Part 1
# part 1 code
# ...
max_lbs_bikes = volunteers * BIKE_CAPACITY
```

5 Submit Your Work

When you are done, submit the following to Canvas in a zip folder:

1. The Python file (.py) containing your code answers for this practicum.
2. A text file (.txt, .doc, .pdf, etc.) with any comments about anything you were not able to get working and what you tried to solve it.

This handout was originally created by Felix Muzny and updated by Ronald Robertson.
It has been updated here by Ryan Gallagher.