Homework 02: Function decomposition

Introduction

This assignment is designed to give you practice with the following new topics:

- Use built-in python functions (`input()`, `round()`)
- Implement basic computations in Python: math operations, type conversion, string concatenation
- Translate verbal descriptions of more complex computations and implement them as functions
- Compose different functions to solve some given task.

Important Tips

- Remember, computer science is a science. Always write code with a prediction in mind. While you can sparingly code to try and see output, you should focus on thinking through what you want your code to do, what you expect as a result, and compare what your code actually does to your result.
- Before you type, trace the execution of the starter code already provided
- Sometimes sub-tasks go together, like computation followed by printing. You should consider them simultaneously if that helps you follow the correctness of your work.
- Use extra steps if needed for your thought process, printing results to test your code. Just remember to remove these extra prints once you know your code is correct.
- Match your output to the output given. Precision is important in computer science.
- Don't change file names. You're generally given `.py` files to work in. Don't change the filenames of those files.

Your assignment

Your task is to complete following steps:

1. Download the `hw2.zip` file and open it. You will see three python files, `hw2_age.py`, `hw2_bill.py`, and `hw2_house.py` in the unzipped folder. You are expected to write your programs in these files.
2. Complete `hw2_age.py`. This file is used in Part 1.
3. Complete `hw2_gratuity.py`. This file is used in Part 2.
4. Complete `hw2_house.py`. This file is used in Part 3.
5. Review the grading criteria at the end of this assignment.
6. Submit your completed programs.

Notice that each starter `.py` file has a header with some information for you to fill in. Please do so. Your feedback helps the instructors better understand your experiences doing the homeworks and where we can provide better assistance.

Part 1: Age madlibs

In `hw2_age.py` write a program that asks a user for a `name` and `age` and prints out two things: - A greeting with the person's name - The person's age if they lived on different planets

Example output 1

```
What is your name? Aang
How old are you (in years)? 12

Hi Aang! Here is a fun fact:

You would be 6.38 years if you lived on Mars
You would be 19.32 years if you lived on Venus
You would be 0.4 years if you lived on Saturn
```

Example output 2

```
What is your name? Avatar
How old are you (in years)? 112
```

```
Hi Avatar! Here is a fun fact:

You would be 59.57 years if you lived on Mars
You would be 180.32 years if you lived on Venus
You would be 3.77 years if you lived on Saturn
```

Task A: Implement functions

We've given you two function headers with docstrings. Implement those functions.

Task B: Compose the functions in `main()`

Use the functions you've written, along with other built-in python functions to perform the task.

Background information for the task

Age conversion formulae source

- 1 Earth year = 1.61 Venus years
- 1 Mars year = 1.88 Earth years
- 1 Venus year = 47.82 Saturn years

`round()` function documentation This is a built-in function in Python that takes two inputs —`number` and `ndigits` — and returns the number rounded down to `ndigits`.

Here are some examples:

- `round(3.61271, 1)` gives `3.6`
- `round(2.675, 2)` gives `2.67` (note it doesn't round up to `2.68`)

Part 2: Bill Splitting

When you go out to dinner with friends, one person pays the entire bill and everyone else sends that person money (e.g., using Venmo) for their share of the bill. However, computing how much each person owes requires accounting for the cost of individual dishes, tax, and tip. Specifically, you need to:

- Determine the percentage of gratuity you want added.
- Determine the *tax rate* that was applied to the total bill. This is the amount of tax divided by the total amount before tax.
- The gratuity and tax rate can be combined with the `cost` of your dish to determine the total amount you owe using this formula: `cost + cost * (tax_rate + gratuity)`

In this part you will write a program that will perform these computations for you.

Example output 1

```
This program assumes gratuity is not included in the bill
What percentage gratuity should be added? 15

How much is the total bill before tax? 73.56
How much is the total bill after tax? 79.45
Tax rate: 8.0%

How much is your dish? 14.99
You owe $18.44
```

Example output 2

```
This program assumes gratuity is not included in the bill
What percentage gratuity should be added? 18
```

```
How much is the total bill before tax? 53.46
How much is the total bill after tax? 56.93
Tax rate: 6.5%

How much is your dish? 10
You owe $12.45
```

Task A: Implement functions

We've given you two function headers with docstrings that are not yet implemented `get_gratuity` and `you`. Implement these two functions.

Task B: Compose the functions in `main()`

Use the functions you've written, along with other functions we've provided to perform the task.

Part 3: House drawing

Your task is to design and implement (in `hw2_house.py`) a program that creates a drawing of a house as text art.

Task A: Basic house

Write code to draw a basic house:

```
|__________________|
|   _____  _____   |
|   [___]  [___]   |
|   [___]  [___]   |
|   [___]  [___]   |
|                  |
|   _____  _____   |
|   [___]  [___]   |
|   [___]  [___]   |
|   [___]  [___]   |
|                  |
|__________________|
|   _____  _____   |
|   [___]  [___]   |
|   [___]  [___]   |
|   [___]  [___]   |
|__________[___]__|
```

The drawing is made using only five distinct characters: spaces (), underscores (_), vertical bars (`|`), left square brackets (`[`), and right square brackets (`]`).

You **must** use separate functions for distinct features and to increase reusability while reducing repetition in your code. Note that there are distinct segments to the figure as well as *repeated elements* which lend themselves naturally to implementing as separate functions.

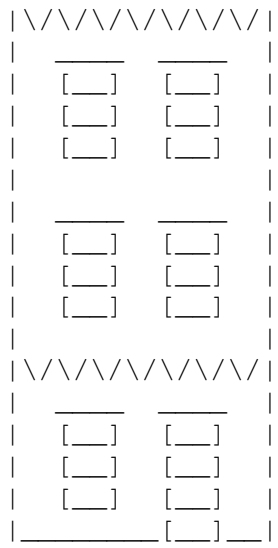
Task B: Custom house

Modify your code to allow the user to customize the house. In particular, the user must be able to input:

- The character to use for glass
- The pair of characters to use as the building roof and top floor divider

Example output 1

```
Enter a character to use for glass: _
Enter two characters to use as the building roof and top floor divider: \/
```



Example output 2

```
Enter a character to use for the glass: |
Enter two characters to use as the building roof and top floor divider: xo
```