Homework 5 - Advanced Lists, Tuples & Modules

CS 1301 - Intro to Computing - Spring 2020

Important

- Due Date: **Tuesday, February 18th, 11:59 PM**.
- This is an individual assignment. High-level collaboration is encouraged, **but your submission must be uniquely yours.**
- Resources:
    - TA Helpdesk
    - Email TA's or use class Piazza
    - How to Think Like a Computer Scientist
    - CS 1301 YouTube Channel
- Comment out or delete all function calls. Only import statements, global variables, and comments are okay to be outside of your functions.
- **Read the entire document before starting this assignment.**

Responsible Coding: Some functions in this assignment will have a 'Responsible Coding' badge next to them. These functions are written to help you think about how you can use your skills as a programmer to analyze problems related to *sustainability*, such as ethics, health, and the environment.

Hidden Test Cases: In an effort to encourage debugging and writing robust code, we will be including hidden test cases on Gradescope for some functions. You will not be able to see the input or output to these cases. Below is an example output from a failed hidden test case:

```
Test failed: False is not true
```

Modules: It is often important to use outside modules or call pre-written code within your function. Python has many modules including `calendar`, which we will ask you to use in this homework. In short, a module is a file containing a set of functions you want to use in your application. Modules must first be imported into the code. Different aspects of the module can then be accessed by calling a method of this function, such as `math.sqrt()`.

Written by Jasmine Yap (jyap6@gatech.edu) & Brae Davies (bdavies34@gatech.edu)

Summer Job

Function Name: summer_job()
Parameters: a list of job tuples (`list`)
Returns: highest paying job (`str`)
Description: You are looking to find a job on campus for this summer to save money for a study abroad. Write a function that takes in a list of tuples in the format of (job title (`str`), hourly pay (`float`), hours per week (`int`)), and determines in which job you would make the most money weekly. If two jobs have the same weekly pay, return the job that appears first in the list. Assume that tax is already accounted for.

Note: These numbers are not representative of the true salaries.

```
>>> jobs = [("Teaching Assistant", 8.50, 12), ("Office of International Affairs
", 9, 8), ("CRC", 7.50, 15)]
>>> print (summer_jobs(jobs))
'CRC'
```

```
>>> jobs = [("Tour Guide", 15, 4), ("CULC Help Desk", 9.25, 10), ("Tutor", 8.00
, 7)]
>>> print (summer_jobs(jobs))
'CULC Help Desk'
```

Wasted Food Responsible Coding

Function Name: wasted_foods()
Parameters: a list of a list and tuples
Returns: least wasteful food and percent waste (`tuple`)
Description: Georgia Tech dining has noticed that some meals have been producing more leftovers than others. They want to figure out what meals students prefer, so that more food is not wasted. Write a function that takes in a list of one list with meals, followed by tuples co-ordinated to each meal in the list, formatted as:

`[ [meal name 1, meal name 2, etc], (meal 1 lbs of food purchased, meal 1 lbs of food left over), (meal 2 lbs of food purchased, meal 2 lbs of food left over), etc]`

This function should find the percent of food wasted and return a tuple with the meal that is least wasteful and its percent waste rounded to 2 decimal places. If two meals have the same percent waste, return the meal that has the least amount of leftovers. No two meals will have the same amount of leftovers.

```
>>> foods = [["Chicken Parmesan", "Spaghetti", "Cheeseburger"], (545, 86), (462
, 200), (620, 107)]
```

```
>>> print (wasted_foods(foods))
('Chicken Parmesan', 15.78)
```

```
>>> foods = [["Pizza", "Sandwiches"], (702, 156), (490, 100)]
>>> print (wasted_foods(foods))
('Sandwiches', 20.41)
```

Ancestors

Function Name: ancestors()
Parameters: list of names (`list`), surname (`str`)
Returns: matching names (`list`)
Description: Seeing people participate in mail-in DNA ancestry tests lately, you have decided you want to participate, but you would rather save money and code your own test. Write a function that takes in a list of names and a specific surname to search for. Return a list of the first names of all the names in the original list that have the correct surname **AND** do not have middle names. There should be no repeats in the returned list. It should be case-sensitive ("Yap" is not the same surname as "yap").

```
>>> names = ["Damian Henry", "Henry Ford", "Warren Elliot Henry", "Heather Fren
ch Henry"]
>>> surname = "Henry"
>>> print (ancestors(names, surname))
['Damian']
```

```
>>> names = ["Kim Kardashian West", "Kylie Jenner", "Kourtney Mary Kardashian",
 "Jasmine Kardashian", "Brae Kardashian", "Jasmine Kardashain"]
>>> surname = "Kardashian"
>>> print (ancestors(names, surname))
['Jasmine', 'Brae']
```

Trigonometry

Function Name: trigonometry()
Parameters: degrees (`str`), operation (`str`)
Returns: answer (`float`)
Description: You will be given a string which indicates what degree angle you are working with. Using **Python's math module**, write a function that converts the degrees to radians, and return either the cosine, sine, or tangent of 'x' radians, depending on what the user passes in. If the user does not pass in an acceptable trigonometric function (`cosine` , `sine` , or

`tangent`), just return the radian value. All values returned should be rounded to 2 decimal places.

```
>>> print (trigonometry("38.7°", "cosine"))
0.78
```

```
>>> print (trigonometry("180°", "arc sine"))
3.14
```

Days of the Week

Function Name: days_of_the_week()
Parameters: list of birthdays (`list`), year (`int`)
Returns: list of names (`list`)
Description: You and your friends all want to celebrate your birthdays together, but you don't want to stay up late on a school night. Write a function that takes in a list of tuples formatted as (day (`int`), month (`int`), name (`string`)), and a year (`int`). Use **Python's calendar module** to find the day of the week that the date falls on in the given year (0 being Monday, 6 being Sunday). Return of a list of the names of your friends whose birthday falls on a Friday (4) or a Saturday (5).

```
>>> birthdays = [(28, 2, "Molly"), (19, 3, "Abby"), (4, 9, "Kevin"), (15, 1, "C
laudia")]
>>> year = 2020
>>> print (days_of_the_week(birthdays, year))
['Molly', 'Kevin']
```

```
>>> birthday = [(10, 11, "Sayler"), (18, 9, "Chad"), (29, 10, "Tammie")]
>>> year = 2027
>>> print (days_of_the_week(birthdays, year))
['Chad', 'Tammie']
```

Donation Amount Responsible Coding

Function Name: donation_amount()
Parameters: list of donations (`list`)
Returns: total (`float`)
Description: With the recent wildfires in Australia, more than 17.9 million acres have been burned across the country's six states. Thousands of homes have been destroyed and an estimated one billion animals have suffered the wrath of the fires. Opportunities of donation and international aid have helped Australia's forces combat this climate disaster.

Write a function that takes in a list of donation amounts and calculates the total of all donations. You will need to call the function from the provided Python file, and pass in each of the donation amounts as arguments. The donation amounts in the list will be strings, but the provided python file will contain a function that will return an addable number. If you do not use the function in `convert.py` to solve this function then you will not receive credit for this function.

```
>>> donations = ["$13.45", "$32.00", "$100.00", "$10.50"]
>>> print (donation_amount(donations))
155.95
```

```
>>> donations = ["$10", "$5", "$1", "$15"]
>>> print (donation_amount(donations))
31.0
```

Grading Rubric

Function	Points
summer_job()	15
wasted_food()	20
ancestors()	20
trigonometry()	20
days_of_the_week()	15
donation_amount()	10
Total	**100**

Provided

The `HW05.py` skeleton file has been provided to you. This is the file you will edit and implement. All instructions for what the functions should do are in this skeleton and this document.

You have also been provided `convert.py`. This is a file you will use to implement the `donation_amount()` function. Please do not turn this back in.

Submission Process

For this homework, we will be using Gradescope for submissions and automatic grading. When you submit your `HW05.py` file to the appropriate assignment on Gradescope, the autograder will run automatically. The grade you see on Gradescope will be the grade you get, unless your grading TA sees signs of you trying to defeat the system in your code. You can resubmit this assignment an unlimited number of times until the deadline; just click the "Resubmit" button at the lower right-hand corner of Gradescope. You do not need to submit your `HW05.py` on Canvas.