Important

1. Due Date: **Thursday, August 31st.**
2. This homework is graded out of 100 points.
3. This is an Individual Assignment. You may collaborate with other students in this class. Collaboration means talking through problems, assisting with debugging, explaining a concept, etc. Students may only collaborate with fellow students currently taking CS 1301, the TA's and the lecturer. You should not exchange code or write code for others. For individual assignments, each student must turn in a unique program. Your submission must not be substantially similar to another student's submission. Collaboration at a reasonable level will not result in substantially similar code.
4. For Help:
    - TA Helpdesk (Schedule posted on class website.)
    - Email TA's or use Piazza Forums Notes:
    - How to Think Like a Computer Scientists [http://openbookproject.net/thinkcs/python/english3e/]
    - CS 1301 Python Debugging Guide [http://www.cc.gatech.edu/classes/AY2016/cs1301_spring/CS-1301-Debugging-Guide/index.html]
5. Don't forget to include the required collaboration statement (outlined on the syllabus). Failing to include the Collaboration Statement will result in no credit.
6. Do not wait until the last minute to do this assignment in case you run into problems.
7. **Read the entire specifications document before starting this assignment.**

Introduction

The goal of this homework is for you to allow your TAs to get to know you as well as allow you to practice and understand how to write functions and evaluate expressions. Part 1 is simply uploading a photo of your face given the specifications listed. Part 2 will consist of 7 functions for you to implement. **You have been given HW1.py** to fill out with instructions in the docstrings. However, below you will find **more detailed information** to complete your assignment. Read it thoroughly before you begin. You have until **Thursday, August 31st** to complete this assignment.

String Formatting

A concept that will be very helpful for this homework is **string formatting**. The purpose of this is to allow you to manipulate statements using variables so that values can change within depending on the different outputs of the function. In order to do this, let's use an example where I ask the user for their name, and then print it out. Try this examples out:

```
name = input("What is your name?")
print("Your name is {}".format(name))

num = 3.141592
print("pi is {:.2f}".format(num))
```

What is happening is that anywhere in a string, I can put `{}` to indicate a placeholder for a variable. After the end quotation marks, I would write `.format()` to the end, and inside the parenthesis will be the variables that you want to include, in the order that `{}` are placed in the sentence. Read this for more info: [https://pyformat.info/]

print vs. return

Distinguishing the difference between print and return is extremely important now and in the future. Print takes a python object and outputs a printed representation of it in the output window. You can think of a print statement as something for the person. Printing has no ongoing execution of a program. It doesn't assign a value to the variable. For example, if you have this function:

```
def printFunc():
      print(2)

In your shell, you should get this:

>>> a = printFunc()
2
>>> print(a)
None
```

However, return is used for a function when you need to save the result of the function to be used for later. By returning a value, the function is producing a value for use *by the program*, in particular for use in the part of the code where the function was invoked. For example, if you have this function:

```
def returnFunc():
      return 2

In your shell, you should get this:

>>> b = returnFunc()
>>> print(b)
2
```

Read this for more info: [http://interactivepython.org/runestone/static/pip2/Functions/Printvs.return.html]

Part 1: Mugshot

Find a picture of yourself. Crop the picture to include just your face. (Your face must be at least 1/3 of the image.) Resize the picture to be exactly 300 pixels by 300 pixels in size. Save the picture as a JPG, using your first and last name as the file name, (e.g. Jay_Summet.jpg). Submit this picture to t-square. You may use an image editing tool you are familiar with to manipulate your mugshot picture. If you do not have a preference, we recommend using the GNU Image Manipulation Program (GIMP): http://www.gimp.org/downloads/

Part 2: Functions

Function name (1): **`weight_on_mars`**
Parameters: **N/A**
Return value: **N/A**
Description: Write a Python program which accepts your weight on Earth and prints out your weight on Mars. Assume that the weight inputted is not negative. The weight can be

1. Ask the user to input their weight on Earth in pounds.
2. Your weight on Mars is 0.38 of your weight on Earth. For example, if you weight 100 pounds on Earth, you would weigh 38 pounds on Mars.
3. Print out the answer in the format: "At a weight of 100 pounds on Earth, you would weigh 38 pounds on Mars".

Test Cases (not an exhaustive list):

1.
```
>>> What is your weight on Earth (in pounds)?
      150
At a weight of 150 pounds on Earth, you would weigh 57 pounds on Mars.
```

2.
```
>>> What is your weight on Earth (in pounds)?
      100.8
At a weight of 100.8 pounds on Earth, you would weigh 38.304 pounds on Mars.
```

Function name (2): **`volume_of_cone`**
Parameters: **N/A**
Return value: **volume of the cone**
Description:
Write a function that takes the radius and height from the user and calculates the volume of a cone.

1. Get the length of the radius of the cone from the user.
2. Get the height of the cone from the user.
3. Calculate the volume of a cone with the radius length and height entered by the user using the following formula:

$$volume = \frac{\pi * radius^2 * height}{3}$$

4. Return the volume of the cone rounded to three decimal places.

Hints:

- You can round a number by using the `round(number, number of decimal places)` function or by using string formatting. For example, you can round 56.14523 to 56.145 by doing `round(56.14523,3)`. [https://docs.python.org/3/library/functions.html#round]

- Importing the `math` module to get the value of π might be helpful. In other words, you must use the math module π. You will lose points if you hardcode 3.14.

Test Cases (not an exhaustive list):

1.
```
>>> a = volume_of_cone()
>>> What is the length of the radius of the cone?
      3
>>> What is the height of the cone?
      4
>>> print(a)
37.699
```

```
2. >>> b = volume_of_cone()
   >>> What is the length of the radius of the cone?
         10
   >>> What is the height of the cone?
         2
   >>> print(b)
   209.440
```

Function name (3): **`calculate_velocity`**
Parameters: **initial_position, final_position, time**
Return value: **N/A**
Description: Write a function that finds the velocity using the initial position, final position, and time passed in the function.

1. Calculate the velocity using the following formula:

$$velocity = \frac{final\_position - initial\_position}{time}$$

2. Print the velocity in the following format: "With an final position of 100 meters, initial position of 0 meters, and a time of 80 seconds, the velocity is 12.5 m/s."

Test Cases (not an exhaustive list):

```
1. >>> calculate_velocity(0,100,80)
   With a final position of 100 meters, initial position of 0 meters, and a
   time of 80 seconds, the velocity is 12.5 m/s.
2. >>> calculate_velocity(2,10,0.5)
   With a final position of 10 meters, initial position of 2 meters, and a time
   of 0.5 seconds, the velocity is 16 m/s.
```

Function name (4): **`liquid_converter`**
Parameters: **N/A**
Return value: **N/A**
Description: Write a user-interactive function to convert any number of fluid ounces to the equivalent number of gallons, quarts, pints, and gills.

1. Get the number of fluid ounces as an integer from the user. You can assume that the user input an int.
2. Calculate the total number of gallons, quarts, pints, and gills represented by the original number of fluid ounces, using the following information:
   a. There are 128 fluid ounces in a gallon.
   b. There is 32 fluid ounces in a quart.
   c. There are 16 fluid ounces in a pint.
   d. There are 4 fluids ounces in a gill.
3. Print the calculated number of gallons, quarts, pints, and gills in the following format (assuming the user entered 6523): "6523 fluid ounces is 50 gallon(s), 3 quart(s), 1 pint(s), and 2 gill(s)"

Hints:

- The modulus (%) and integer division (//) operators will be very useful for this function.

Test Cases (not an exhaustive list):

```
1. >>> How many fluid ounces would you like to convert?
         6523
   6523 fluid ounces is 50 gallon(s), 3 quart(s), 1 pint(s), and 2 gill(s)
```

```
2. >>> How many fluid ounces would you like to convert?
         1250
   1250 fluid ounces is 9 gallon(s), 2 quart(s), 2 pint(s), and 0.5 gill(s)
3. >>> How many fluid ounces would you like to convert?
         180
   >>> 180 fluid ounces is 1 gallon(s), 1 quart(s), 1 pint(s), and 1 gill(s)
```

Function name (5): **`calorie_counter`**
Parameters: **N/A**
Return value: **N/A**
Description:
Write a function that takes the number of meals and number of miles to calculate a person's caloric intake in a day.

1. Get the number of meals a person ate from the user.
2. Get the number of miles a person ran.
3. Calculate a person's caloric intake using the following:
    a. The average calories gained per meal is 500 calories.
    b. A person that has done no exercise has burned about 1600 calories.
    c. The average calories burned per mile of running is 95 calories.
4. If a person gains the same or more calories than they burn then you should print a positive number. If a person burns more than they gain you should print a negative number.
5. Print the result in the following format: "After eating 5 meals and running 2 miles, a person gained 2500 calories and burned 1790 calories, leading to an intake of 710 calories."

Test Cases (not an exhaustive list):

```
1. >>> How many meals did you eat today?
         4
   >>> How many miles did you run today?
         0
   After eating 4 meal(s) and running 0 mile(s), a person gained 2000 calories
   and burned 1600 calories, leading to an intake of 400 calories.
2. >>> How many meals did you eat today?
         1
   >>> How many miles did you run today?
         1
   After eating 1 meal(s) and running 1 mile(s), a person gained 500 calories
   and burned 1695 calories, leading to an intake of -1195 calories.
```

Function name (6): **paycheck_computation**
Parameters: **pay_rate, hours_worked, tax_rate**
Return value: **person's take home pay**
Description: A person's pay is determined by the hours they worked times the hourly rate. Unfortunately, a percentage of their pay is taken out as taxes. Using the parameters listed above, calculate and return the amount that a person's paycheck will be for after taxes have been taken out.

1. The first parameter, pay_rate, contains a person's hourly pay.
2. The second parameter, hours_worked, contains a person's number of hours worked.
3. The third parameter, tax_rate, gives the amount taken from taxes as a fraction between 0 and 1.

4. Calculate the person's take home pay, which is their paycheck total after taxes have been deducted.
5. Return the person's take home pay.

Test Cases (not an exhaustive list):

```
1. >>> a = paycheck_computation(10,4,0.8)
   >>> print(a)
   8
2. >>> b = paycheck_computation(18,40,0.2)
   >>> print(b)
   576
```

Function name (7): **main**
Parameters: **N/A**
Return value: **N/A**
Description:
This function has been started for you. It serves as an introduction for conditional statements (learned later in the course). All you have to do is ask the user what function they will like to call and make the function call yourself.

1. Ask the user to input the number of the function (any number from 1 through 6 should be valid)
2. Replace the `pass` statements with your function calls.

Grading Rubric

-**Mugshot**: 20 points
    File Named Correctly: 5 Points
    Correct File Size: 10 Points
    Face Recognizable: 5 Points
- **weight_on_mars**: 10 points
    Correct function header: 2 Points
    Prints in the correct format: 3 Points
    Correct Value is calculated: 5 Points
-**volume_of_cone**: 10 points
    Correct function header: 2 Points
    Prints in the correct format: 3 Points
    Returns the correct value: 5 Points
-**calculate_velocity**: 10 points
    Correct function header: 2 Points
    Prints in the correct format: 3 Points
    Correct response is calculated: 5 Points
-**liquid_converter**: 10 points
    Correct function header: 2 Points
    Prints in the correct format: 3 Points
    Correct response is calculated: 5 Points
-**calorie_counter**: 15 points
    Correct function header: 2 Points
    Prints in the correct format: 3 Points
    Correct response is calculated: 7 Points
-**paycheck_computation:** 20 points
    Correct function header: 2 Points
    Prints in the correct format: 3 Points
    Returns correct value: 8 Points
-**main**: 5 points
    Calls the correct function: 5 Points

Provided

The following file(s) have been provided to you. There are several, but you will only edit one of them:

1. `HW1.py`
   This is the file you will edit and implement. All instructions for what the methods should do are in the docstrings.

Deliverables

You must submit all of the following file(s). Please make sure the filename matches the filename(s) below. Be sure you receive the confirmation email from T-Square, and then download your uploaded files to a new folder and run them.

1. `HW1.py`
2. `First_Last.jpg`