CMSC 201 Spring 2016
Homework 3 – Branching

Assignment: Homework 3 – Branching
Due Date: Monday, February 22nd, 2016 by 8:59:59 PM
Value: 40 points

Homework 3 is designed to help you practice branching (one-way and two-way selection structures) in a Python environment.

Remember to enable Python 3 before you run your programs:

```
scl enable python33 bash
```

Instructions

In this homework, we will be doing a series of exercises designed to make you practice using variables, expressions, and if/else statements. Each one of these exercises should be in a **separate python file**. For this assignment, you may assume that all the input you get will be of the correct type (*e.g.,* if you ask the user for a whole number, they will give you an integer).

For this assignment, you'll need to follow the class coding standards, a set of rules designed to make your code clear and readable. The class coding standards are on Blackboard under “Assignments” in a file titled “CMSC 201 - Python Coding Standards.”

A very important piece of following the coding standards is the file header comments. Make sure that each file has a comment block at the top (see the coding standards document for an example).

*NOTE: **You must use `main()`** as seen in the `lab1.py` file, and as discussed in class.*

Details

Homework 3 is broken up into five parts. **Make sure to complete all 5 parts.**

NOTE: Your filenames for this homework must match the given ones exactly.

And remember, filenames are case sensitive!

hw3_part1.py **(Worth 4 points)**

Read in a single float from the user. Print out the number and whether it is positive, negative, or equal to zero.

Here is some sample output, with the user input in blue.
(Yours does not have to match this exactly, but it should be similar.)

```
bash-4.1$ python hw3_part1.py
Please enter a floating point number: 0
The number 0.0 is equal to zero

bash-4.1$ python hw3_part1.py
Please enter a floating point number: 5.6
The number 5.6 is positive

bash-4.1$ python hw3_part1.py
Please enter a floating point number: -4.7
The number -4.7 is negative
```

hw3_part2.py **(Worth 8 points)**

For this part of the homework you will need to create a grade calculator.

Your program should prompt the user for these inputs, **exactly in this order:**

1. Homework weight
2. Homework grade
3. Exam Weight
4. Exam Grade
5. Discussion Weight
6. Discussion Grade

(HINT: A weighted total is computed by multiplying each grade by the corresponding weight and adding them together.)

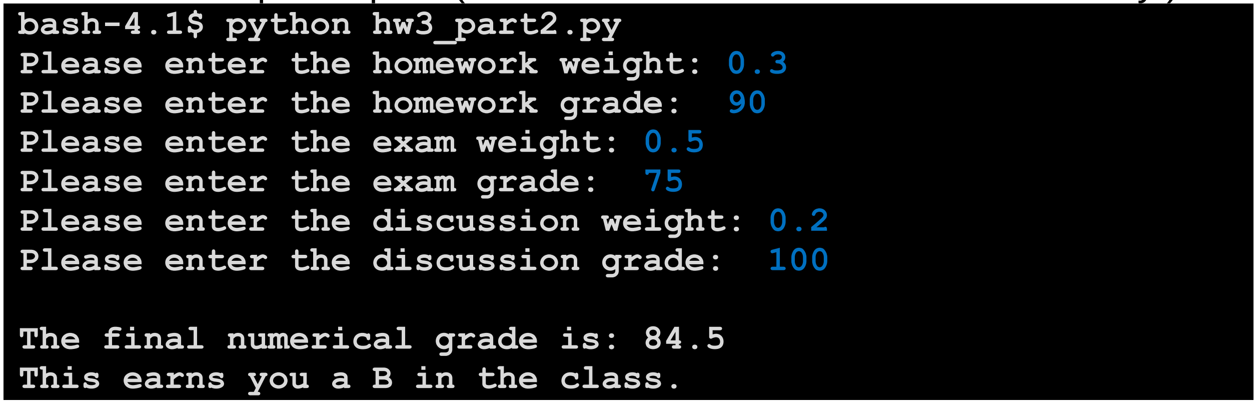
For these inputs, you can assume the following:

- Weights will be between 0 and 1 (inclusive)
    - You don't need to check that the weights add up to 1
- Grades will be between 0 and 100 (inclusive)

Once you have computed the student's final grade, print out the numerical grade and their letter grade.

- A = 90 – 100 (any grade 90 and above)
- B = 80 – 89.9
- C = 70 – 79.9
- D = 60 – 69.9
- F = 0 – 59.9 (any grade below 60)

Here is some sample output. (Yours does not have to match this exactly.)



```
bash-4.1$ python hw3_part2.py
Please enter the homework weight: 0.3
Please enter the homework grade:  90
Please enter the exam weight: 0.5
Please enter the exam grade:  75
Please enter the discussion weight: 0.2
Please enter the discussion grade:  100

The final numerical grade is: 84.5
This earns you a B in the class.
```

hw3_part3.py **(Worth 10 points)**

Next you are going to create a program that plays a simple "Guess Who"-like game. The user has selected a character, and your program will narrow down who that character could be by asking questions about them. For simplicity's sake, there are only five possible characters.
(WARNING: This part of the homework is the most challenging, so budget plenty of time and brain power. And read the instructions carefully!)

Your program can ask the other player about four characteristics. Your program should ask the ***minimum*** number of questions needed to guess the character. *(HINT: Your program should need to ask no less than two questions and no more than three questions to find the right character.)*

- Is the character a woman?
- Does the character have blue eyes?
- Does the character wear glasses?
- Does the character have a beard?

For these inputs, you can assume the following:

- The user will only ever enter either lowercase **`y`** (for "yes") or lowercase **`n`** (for "no")

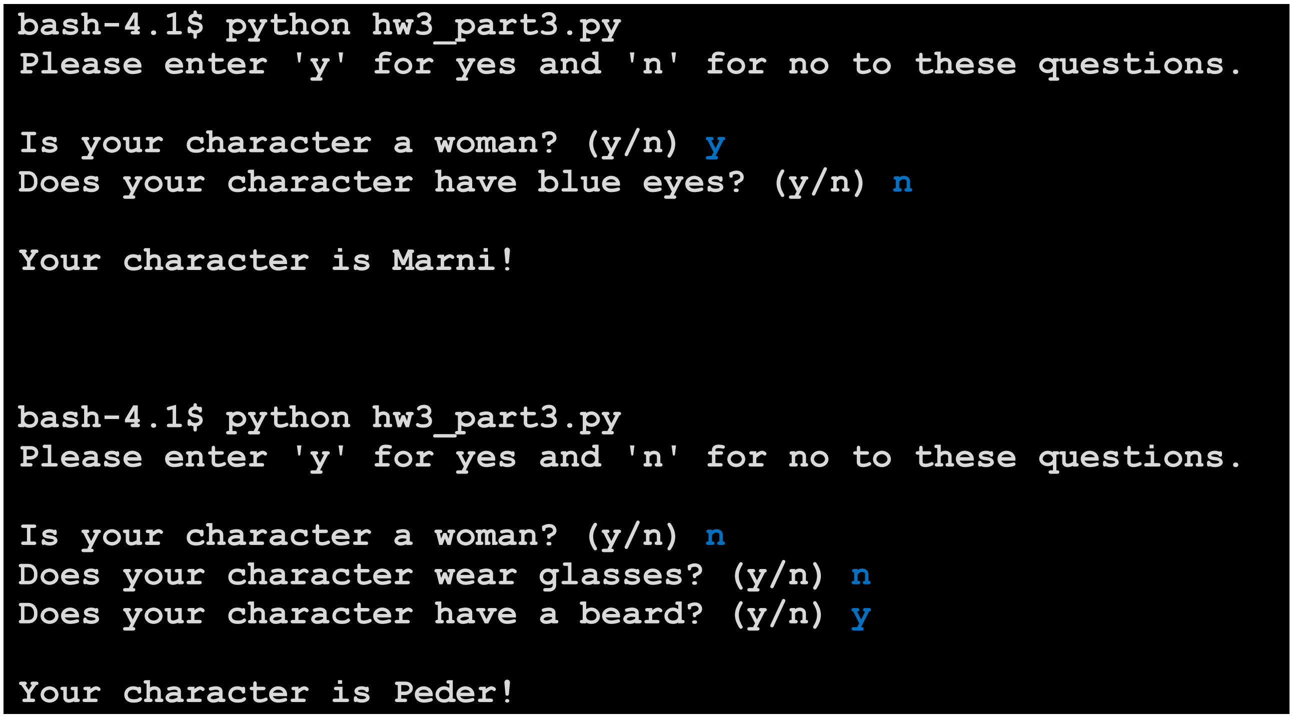
Based on their responses, you must select the correct character and print out your guess to the screen. Here are the possibilities for the characters:

- Character is a woman and has blue eyes: Jane
- Character is a woman and doesn't have blue eyes: Marni
- Character is not a woman and wears glasses: Adrian
- Character is not a woman, does not wear glasses, and has a beard: Peder
- Character is not a woman, does not wear glasses, and does not have a beard: Zhang

(HINT: Review Lecture 02 (Algorithmic Thinking) and either create a flowchart or write some pseudocode for how you want your program to work. Do ***not*** *start coding this part without having a plan!)*

(See the next page for sample output.)

Here is some sample output for hw3_part3.py, with the user input in blue. (Yours does not have to match this exactly, but it should be similar.)



```
bash-4.1$ python hw3_part3.py
Please enter 'y' for yes and 'n' for no to these questions.

Is your character a woman? (y/n) y
Does your character have blue eyes? (y/n) n

Your character is Marni!



bash-4.1$ python hw3_part3.py
Please enter 'y' for yes and 'n' for no to these questions.

Is your character a woman? (y/n) n
Does your character wear glasses? (y/n) n
Does your character have a beard? (y/n) y

Your character is Peder!
```

hw3_part4.py **(Worth 6 points)**

For this part, you are going to ask the user for the temperature and the scale it’s measured in (Fahrenheit or Celsius). Then you will print out whether water is a liquid, gas, or solid at that temperature.

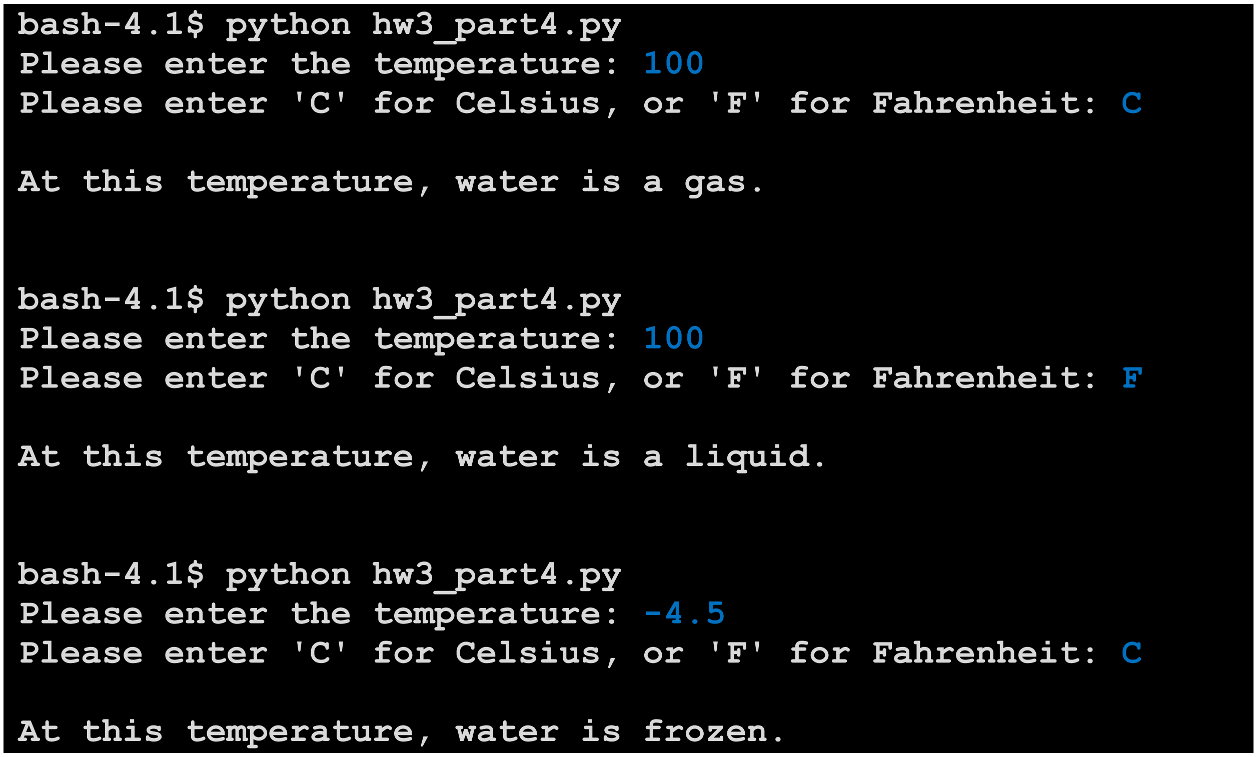
For the input to these two questions, you can assume the following:

- The user will enter a number (it may be decimal)
- The user will only ever enter either uppercase `C` (for “Celsius”) or uppercase `F` (for “Fahrenheit”)

Make sure that you handle both of the temperature scales.

IMPORTANT: You will need to do your own research on water’s states. (You may assume that the water is at sea level.)

Here is some sample output. (Yours does not have to match this exactly.)



```
bash-4.1$ python hw3_part4.py
Please enter the temperature: 100
Please enter 'C' for Celsius, or 'F' for Fahrenheit: C

At this temperature, water is a gas.


bash-4.1$ python hw3_part4.py
Please enter the temperature: 100
Please enter 'C' for Celsius, or 'F' for Fahrenheit: F

At this temperature, water is a liquid.


bash-4.1$ python hw3_part4.py
Please enter the temperature: -4.5
Please enter 'C' for Celsius, or 'F' for Fahrenheit: C

At this temperature, water is frozen.
```

hw3_part5.py **(Worth 8 points)**

Finally, we will create a (very simplified) day of the week calculator.

Ask the user to enter a day of the month.

For the input to this question, you can assume the following:

- The user will enter a number.
    - You can assume that the number will be an integer.
    - You cannot assume that the number will be valid!

We will assume the month starts on Monday and has 31 days.
If the day of the month the user entered is not a valid day of the month (less than 1 or greater than 31), print a short error message to the user.
Otherwise print the day of the week that day falls on. For instance, the 2nd would be a Tuesday, the 10th would be a Wednesday, etc.

IMPORTANT: Do not write a case for each day of the month. If your program uses 31 individual `if-else` statements, you will lose significant points.

Here is some sample output. (Yours does not have to match this exactly.)

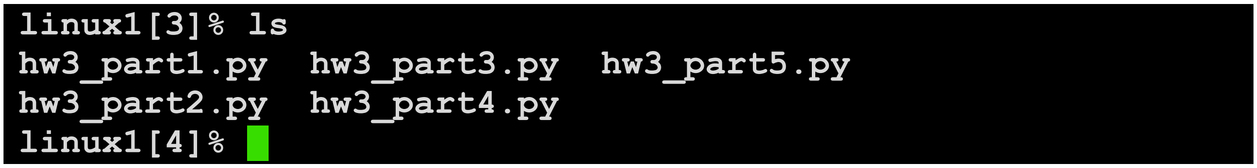
```
bash-4.1$ python hw3_part5.py
Please enter the day of the month: 0
Invalid day

bash-4.1$ python hw3_part5.py
Please enter the day of the month: 9
Today is a Tuesday!

bash-4.1$ python hw3_part5.py
Please enter the day of the month: 24
Today is a Wednesday!
```

Submitting

Once all five parts of your Homework 3 are complete, it is time to turn them in with the **`submit`** command.

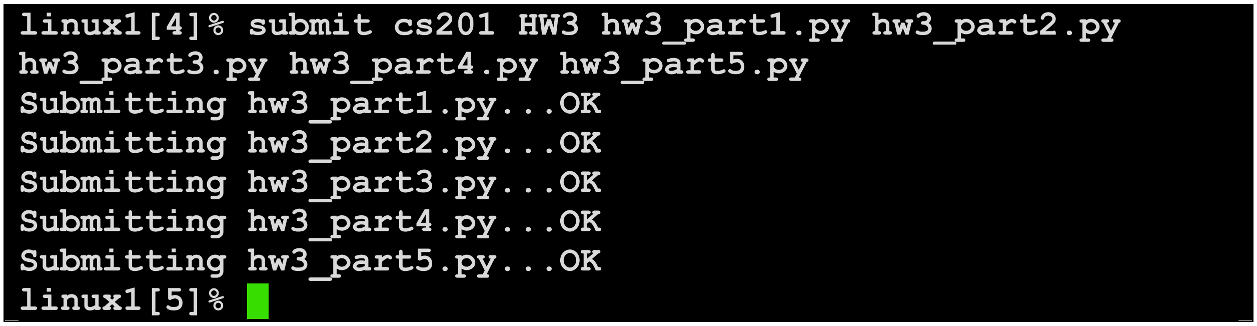
Don't forget to complete the header block comment for each file! Make sure that you updated the header block's file name and description for each file.

You must be logged into your GL account, and you must be in the same directory as the Homework 3 files. To double check this, you can type **`ls`**.



```
linux1[3]% ls
hw3_part1.py  hw3_part3.py  hw3_part5.py
hw3_part2.py  hw3_part4.py
linux1[4]% 
```

To submit your files, we use the **`submit`** command, where the class is **`cs201`**, and the assignment is **`HW3`**. Type in (all on one line)
`submit cs201 HW3 hw3_part1.py hw3_part2.py hw3_part3.py hw3_part4.py hw3_part5.py`
and press enter.



```
linux1[4]% submit cs201 HW3 hw3_part1.py hw3_part2.py
hw3_part3.py hw3_part4.py hw3_part5.py
Submitting hw3_part1.py...OK
Submitting hw3_part2.py...OK
Submitting hw3_part3.py...OK
Submitting hw3_part4.py...OK
Submitting hw3_part5.py...OK
linux1[5]% 
```

If you don't get a confirmation like the one above, check that you have not made any typos or errors in the command.

You can double-check that all five homework files were submitted by using the **`submitls`** command. Type in **`submitls cs201 HW3`** and hit enter.

And you're done!