CMSC 201 Fall 2016
Homework 3 – Branching

Assignment: Homework 3 – Branching
Due Date: Wednesday, September 28th, 2016 by 8:59:59 PM
Value: 40 points

Collaboration: For Homework 3, collaboration is allowed. Make sure to consult the syllabus about the details of what is and is not allowed when collaborating. You may not work with any students who are not taking CMSC 201 this semester. If you work with someone, remember to note their name, email address, and how you collaborated at the top of your file.

If you did not work with anyone else on this assignment, your collaboration statement should state that
`I did not collaborate with anyone on this assignment.`

If you did work with someone else (or multiple other people), your collaboration statement should state something *similar* to the following:
`I collaborated with Fox Mulder (fmulder1@umbc.edu); I helped him understand the loop.`

Make sure that you have a complete file header comment at the top of each file, and that all of the information is correctly filled out.

```
# File:    FILENAME.py
# Author:  YOUR NAME
# Date:    THE DATE
# Section: YOUR DISCUSSION SECTION NUMBER
# E-mail:  YOUR_EMAIL@umbc.edu
# Description:
#    DESCRIPTION OF WHAT THE PROGRAM DOES
# Collaboration:
#    COLLABORATION STATEMENT GOES HERE
```

Homework 3 is designed to help you practice branching in a Python environment. You will need to use one-way and two-way selection structures, as well as nested selection structures. You may also need to use multi-way selection structures for this assignment.

Remember to enable Python 3 before running and testing your code:

```
    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 the website, linked at the top of the "Assignments" page. You can also access them directly via this URL (http://goo.gl/yEoGfC).

NOTE: ***You must use*** **`main()`** *as seen in your* **`lab2.py`** *file, and as discussed in class.*

At the end, your Homework 3 files must run without any errors.

Details

Homework 3 is broken up into four separate parts. **Make sure to complete all 4 parts.**

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

And remember, filenames are case sensitive!

Questions

Each question is worth the indicated number of points. Following the coding standards, having complete file headers, and having correctly named files is worth 6 points.

hw3_part1.py **(Worth 5 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 12 points)**

Next you are going to create a program that plays a simple game, where you ask the player about their dog, and attempt to guess the dog's breed. Your program will narrow down the possibilities by asking questions about the player's dog. For simplicity's sake, there are only five possible dog breeds.

*(**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 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 breed.)*

- Is the dog a big dog?
- Does the dog have a fluffy coat?
- Does the dog have a curly tail?
- Do the dog's ears stick up?

For these inputs, you can assume the following:

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

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

- Dog is a big dog and has a fluffy coat: Samoyed
- Dog is not a big dog and has a curly tail: Shiba Inu
- Dog is not a big dog and does not have a curly tail: Corgi
- Dog is a big dog, does not have a fluffy coat, and does not have ears that stick up: Borzoi
- Dog is a big dog, does not have a fluffy coat, and does have ears that stick up: German Shepherd

*(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_part2.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_part2.py
Please enter 'yes' or 'no' to these questions.

Is your dog a big dog? yes
Does your dog have a fluffy coat? no
Do your dog's ears stick up? no

Your dog is a Borzoi!



bash-4.1$ python hw3_part2.py
Please enter 'yes' or 'no' to these questions.

Is your dog a big dog? no
Is your dog's tail curly? yes

Your dog is a Shiba Inu!
```

hw3_part3.py **(Worth 8 points)**

For this part, you are going to ask the user for the temperature and the scale it's measured in (Celsius or Kelvin). 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 a decimal)
- The user will only ever enter either uppercase **`C`** (for "Celsius") or uppercase **`K`** (for "Kelvin")

Make sure that you program handles 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, 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 the temperature: 60
Please enter 'C' for Celsius, or 'K' for Kelvin: C
At this temperature, water is a liquid.

bash-4.1$ python hw3_part3.py
Please enter the temperature: 60
Please enter 'C' for Celsius, or 'K' for Kelvin: K
At this temperature, water is a (frozen) solid.

bash-4.1$ python hw3_part3.py
Please enter the temperature: 373.2
Please enter 'C' for Celsius, or 'K' for Kelvin: K
At this temperature, water is a gas.
```

hw3_part4.py **(Worth 9 points)**

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

You will ask the user to enter the day of the month, and will respond with which the correct day or the week.

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 Sunday and has 30 days.

If the day of the month the user entered is not a valid day of the month (less than 1 or greater than 30), simply 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 Monday, the 10th would be a Tuesday, etc.

IMPORTANT: Do not write a case for each day of the month. If your program uses 30 individual **`if-else`** statements, you will lose significant points.

(HINT: There is a Python operator that will allow you to write this program without needing to have 30 individual conditionals or statements. Review Lecture 04 (Expressions) to see it in action.)

(See the next page for sample output.)

Here is some sample output for hw3_part4.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_part4.py
Please enter the day of the month: 31
The date 31 is an invalid day.

bash-4.1$ python hw3_part4.py
Please enter the day of the month: 7
Today is a Saturday!

bash-4.1$ python hw3_part4.py
Please enter the day of the month: 18
Today is a Wednesday!
```

Submitting

Once your **`hw3_part1.py`**, **`hw3_part2.py`**, **`hw3_part3.py`**, and **`hw3_part4.py`** files are complete, it is time to turn them in with the **`submit`** command. (You may turn in individual files as you complete them. To do so, only **`submit`** those files that are complete.)

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

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

To submit your Homework 3 python 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`** and press enter.

```
linux1[4]% submit cs201 HW3 hw3_part1.py hw3_part2.py
hw3_part3.py hw3_part4.py
Submitting hw3_part1.py...OK
Submitting hw3_part2.py...OK
Submitting hw3_part3.py...OK
Submitting hw3_part4.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 check that your homework was submitted by following the directions in Homework 0. Double-check that you submitted your homework correctly, since **an empty file will result in a grade of zero for this assignment.**