CS 112 – Fall 2020 – Programming Assignment 3

Branching

Due Date: Sunday, September 20th, 11:59pm

The purpose of this assignment is to get more practice with Boolean logic and branching.

See the **"assignment basics"** file for more detailed information about getting assistance, running the test file, grading, commenting, and many other extremely important things. Each assignment is governed by the rules in that document.

Note: a tester file is included as tester3.py, but the tester is now also integrated with Gradescope!

Background

Selection statements (**if/elif/else** combinations) allow us to write code that can execute different statements based on the current values seen in a particular run of the program. We will use this to write a program that performs calculations and selectively reports on different properties of the calculated values.

Guidelines

- Think carefully about the order you will check different properties. You might want to write some pseudocode or perhaps draw a flowchart if this works better for you. Think first, then implement.
- Be careful what kinds of selection statements you use and be sure to test your code with many examples. For instance, multiple `if` statements will not necessarily behave the same as a chain of `if-elif-elif`.
- When a specific test case is not working, plug your code into the `visualizer` to watch what the code does, which lines run, which branches are taken.
- From built-in functions, you are allowed to call `abs()`, `int()`, `float()`, `str()` only.
- You are **not** allowed to `import` anything.
- You are **not** allowed to use loops, lists, sets, dictionaries and any feature that hasn't been covered in class yet.
- Do not forget to review the "**assignment basics**" file
- Insert comments in the lines you deem necessary

Testing

In this assignment testing will be done the same way as in the previous one. There will be **no user input or print** statements. You are given a number of tasks and for each of them you must implement one Python function. The tester will be calling your functions with certain arguments and will be examining the return values to decide the correctness of your code. Your functions **should not ask for user input and should not print anything**, just return a value based on the specification of the tasks.

Grading Rubric

Submitted correctly:	2	# see assignment basics file for file requirements!
Code is well commented:	8	# see assignment basics file for how to comment!
Tester calculations correct:	90	# see assignment basics file for how to test!
TOTAL:	100	

Note: If your code does not run and crashes due to errors, it will receive **zero** points. Turning in running code is essential.

Scenario

You are working for a game development company and your manager asks you to implement a series of functions that another team will use to build a simple fighter game.

Here are some of the things you need to know about this game:

Players – players of the game can choose to be one of four types of players, with each type having different moves worth different points. Once they choose a type, they can only make the moves associated with that type.

Fighter Type	5 Point Moves	10 Point Moves	15 Point Moves	20 Point Moves	25 Point Moves
earth	rock bullets	dust devil	earth armor	earth levitation	tectonics
fire	fire shield	fly kick	swing kick	fire fly	fireball
water	water wave	waterspout	ice spears	freeze	waterburst
air	air ball	air blast	air bomb	air funnel	air punch

Locations – during gameplay, players will duel at a location. The location might influence player points based on the player's type.

Locations	Influences
full moon bay	Earth = +5 points to each Earth move Air = -2 points from each Air move Water = +3 points to each Water move Fire = -1 points from each Fire move
beach cave	Earth = -2 points from each Earth move Air = +3 points to each Air move Water = -1 points from each Water move Fire = +5 points to each Fire move
foggy swamp	Earth = +3 points to each Earth move Air = -1 points from each Air move Water = +5 points to each Water move Fire = -2 points from each Fire move
air temple	Earth = -1 points from each Earth move Air = +5 points to each Air move Water = -2 points from each Water move Fire = +3 points to each Fire move

Players can only make one move per round, they cannot use the same move consecutively, and they will not know what move the other player is making prior to making their own decision. Players will be moved to a location each round (randomly).

Your task is to implement each of the functions as described below based on the specifications of the game provided. Each function will return the requested output.

Assumptions

You may assume that:

- The types of the values that are sent to the functions are the proper ones, you do not have to validate them.
- All string inputs and outputs are **lower case** and spelled **exactly** as you see in the tables above and in the descriptions of the functions below.
- The functions are going to be called with usable values, you do not have to validate them.

Restrictions

- You **may not import** other modules (like math)
- No **loops** (`for`/`while`) or any other feature we have not yet covered in class.

Functions

The signature of each function is provided below, do ***not*** make any changes to them otherwise the tester will not work properly. Keep in mind that you must ***not*** write a main body for your program. You should only define these functions; it is the tester that will be calling and testing each one of them. The following are the functions you must define:

def whatType(moveName):

Description: Function determines player's type based on the move they made.

Parameters: **moveName** holds the name of the move.

Return value: "earth" or "fire" or "water" or "air"

Examples:
whatType("air ball") -> air
whatType("fireball") -> fire
whatType("tectonics") -> earth

return playerType

def calcPoints(moveName):

Description: Function outputs the point category a particular move is in.

Parameters: **moveName** holds the name of the move.

Return value: 5 or 10 or 15 or 20 or 25

Examples:
calcPoints("air blast") -> 10
calcPoints("freeze") -> 20
calcPoints("earth levitation") -> 20

return points

def whoWins(p1Move, p2Move, location):

Description: After a single move, this function determines the winner out of 2 players.

Parameters: **p1Move** holds the name of the move player 1 made, **p2Move** holds the name of the move player 2 made, **location** holds the name of the location where this duel took place.

Return value: "it's a tie!" or "player 1 wins!" or "player 2 wins!"

Examples:
whoWins("dust devil", "freeze", "beach cave") -> player 2 wins!
whoWins("tectonics", "air punch", "full moon bay") -> player 1 wins!

Hint: You may call function(s) you have already defined in this assignment within this whoWins function to make certain determinations quicker.

return winner

def hasAdvantage(playerType, location):

Description: Determines if a player has an advantage at a certain location based on their type.

Parameters: **playerType** is the player's type (earth, air, fire, water), **location** holds the location name.

Return value: "no advantage" or "has great advantage" or "has some advantage"

Examples:
hasAdvantage("fire", "beach cave") -> has great advantage
hasAdvantage("water", "full moon bay") -> has some advantage
hasAdvantage("fire", "full moon bay") -> no advantage

return result

def countMyCombo(move1, move2, move3, location):

Description: Calculates the total points after a combo move of 3 moves made by one player. Type and location must be considered in the calculation.

Parameters: **move1 move2 move3** holds the names of the three moves. **location** holds the name of the location.

Return value: a numeric calculated point value

Examples:
countMyCombo("waterspout", "ice spears", "freeze", "air temple") -> 39
countMyCombo("air bomb", "air funnel", "air punch", "full moon bay") -> 54

Hint: You may call function(s) you have already defined in this assignment within this countMyCombo function to make certain determinations quicker.

return total

Your professor may or may not have been influenced by Avatar on this assignment, watching every episode during quarantine.