1336 Programming Assignment 2: Combat

General Statement

You must write this assignment starting with **Basic template.py.**

Everything in the template is laid out for you to put the right things in the right place. Leave every comment as it is, from the **# Program <program name>** line to the **# End Program** line (you don't need the template description section above it).

Everything in angle brackets is a placeholder that needs to be changed.

# Program <program name>	becomes	**# Program combat**
# <what the program does>	becomes	**# simple role playing game**
# Date: <today's date>	becomes	**# Date: 24 October 2020**

You will add a line for <revision date> every new date that you work on the program.

Do not leave the angle bracket. It's **combat**, not **< combat >.**

Style

Refer to Style Requirements.docx for conventions you must follow while writing this program.

# Declare Variable types (EVERY variable used)

Because Python does have types, you must remember what kind of value each variable is supposed to hold. Declaring means that you set each variable to int(), float(), str(), or bool() at the beginning. This also makes the variable "real" to Python, which won't recognize a variable until it has been set to a value.

Declarations look like this:

integer_variable = **int()**
decimal_variable = **float()**
string_variable = **str()**
boolean_variable = **bool()**

The Problem

We're building a simple role-playing game.

Understand the Problem

The Task

In this second-level simulation, a warrior is defined as a set of 4 characteristics: strength, energy, speed, and direction. A field is still defined as two locations, a place for each warrior. The locations are numbers that indicate where the warrior is in a line moving left and right only; (field_red == 7) means that the red warrior is in location 7. If a warrior's direction is 'right, when it runs its location will go up; if its direction is 'left' the location goes down when it runs.

1336 Programming Assignment 2: Combat

You will define two warriors and move them around. After any action you will display the game state (all the current values of each warrior's characteristics including the locations). Moving is more complex than in Assignment 1: warriors may change direction. Also, the warriors keep moving until one warrior runs out of energy.

Specifics

Change all the placeholders to be your name, program name, what the program does, and the date.

Declare three integer variables for the strength, energy, and speed, for each of two warriors. The names begin with red_ or blue_ for the two warriors, so they are red_strength, red_energy, and so forth. Declare two integer variables for the field with the warrior name prefixes (field_red, field_blue). Declare string variables red_direction and blue_direction, and another integer variable time. Finally, declare a Boolean variable that will be True as long as both warriors have energy but turns False if either one's energy drops to 0 or below.

You may need to declare other variables as needed.

Setup

The starting red values are strength 45, energy 100, and speed 5. The blue values are 80, 150, and 3, respectively. Time starts at 0 and the Boolean variable is True. Red begins at location 0 with direction "right"; Blue is at location 15 and direction "left."

Play

Display the game state

Print the starting game state in this format:

```
Time is 0.
Red strength is 45, energy is 100, speed is 3, location is 0.
Blue strength is 80, energy is 150, speed is 5, location is 0.
```

Do these things while the Boolean variable is True:

Move the warriors

Move the red warrior in the direction it is pointed (add the warrior's speed to the location if pointed right, subtract the warrior's speed if pointed left).

Move the blue warrior in the same way as described for the red warrior.

Don't worry if the warriors bump into each other: that is Extra Credit (see below).

Update the warriors' conditions

Increase time by 1.
Subtract the square of each warrior's speed from its energy.
Subtract twice the warrior's speed from its strength.

1336 Programming Assignment 2: Combat

Display the new game state

Print the game state as described above

Test the warrior's conditions

If either warrior's energy is 0 or less, set the Boolean variable to False.

Final report

Now that the Boolean variable is False, it means that one or both of the warriors ran out of energy. If both warriors' energy is 0 or less, print "Both died." If only red's or blue's energy is 0 or less, print which one died. If neither warrior's energy is 0 or less, print "ERROR" because you shouldn't get to this point in the program unless one of them died.

Extra Credit (10 points)

If the warriors meet each other, reverse direction. To do this, you will need to change the "move" step: rather than just add or subtract the speed from the warrior's location, move by 1 in a loop (one loop up to the speed of the warrior). Before each step, check to see if the resulting spot would be the other warrior's location. If it is, reverse direction and then take the step.

Special Notes:

1. **Do not use** functions, lists, or files, *even if you know how to do those things*.
2. The purpose of this assignment is to see if you have the skills to use **only** the Python tools described in chapters 1-4 to solve this problem.
3. Using more advanced features demonstrates that you don't know the simpler tools well enough, so using them will **lower** your score.
4. You earn up to 60% by correct operation and up to 40% with good style, readability, and documentation.

Submit (in <u>**Blackboard**</u>) the file **combat.py**.
*Name the file **exactly** as given*. <u>**Do not add anything to the program name. Not** your name, **not** your ID number, **not** words like "assignment 2" or "revised", or **any** additional characters as part of the file name</u>.
The **Python** program is scored with a maximum value of **100 points**.