ENGR 101 - Introduction to Programming

Mini Project 1

December 6, 2017

(Due on Friday, December 22 by 5 pm)

In this mini project, you are going to develop a text-based fighting game (the console version of Mortal Kombat ☺) Called SEHIR KOMBAT. The primary goal of this project is to make you practice functions, loops, conditional execution, and getting user input through keyboard. The detailed explanations about the game that you are going to develop are provided below (Note: example console output from the game is written in BLUE.).

Game Logic:

This game will be played by two players in a turn-based manner. In each turn, one of the player's hero will attack the other player's hero. An attack may be successful by a certain chance rate (details are below). If an attack is successful, the attacked player's health points will decrease by the magnitude of the attack which will be decided by the attacking player. The first player whose health points become zero or negative will lose the game. Initially, each player's hero will have 100 health points. The game will continue in a turn-based manner until one of the players lose.

Text-based User Interface:

The game will be played by two players in a turn-based manner. When the game first starts, the players will be asked to enter the names of their heroes. Names cannot be empty. If a player enters empty name, then the game should warn the user and ask for a valid name again, until the player provides one. Below shows a sample running for this part of the game.

```
----- First Hero -----

Please type your hero's name: Sub-Zero
```

After the first user types the name of his hero, the game will ask for the name of second player's hero.

```
----- Second Hero -----
```

```
Please type your hero's name: Sub-Zero
```

Note that if the second player enters the same name as the first player, as in the above example, the game should warn the user, and then keep asking for a different name until the second user writes a different name. For instance, in the below example, if the second player typed Sub-Zero, the game would provide the following warning.

```
Sub-zero is taken, please choose another name!
----- Second Hero -----
Please write Your hero's name: Scorpion
```

Once the user writes a valid name, the game will start. At the beginning of the game, the game should decide which player is going to attack first by using a random coin toss-like function. (*Hint: use Random Module*). Each player will have equal chance (50%) of being the first attacking player. After the first attacking player is decided randomly, starting hero's name should be printed out as shown below.

```
Coin toss result: Scorpion starts first!
```

Then, the current health points of each hero should be displayed. Initially, each player will have a fixed starting health value of 100, and it should be displayed both with a number and with sticks, where the number of sticks should be half (integer division) of the health points (HP). For instance, 100 health points will be displayed with 50 sticks.

Note: the name of the player who starts the game first as the result of the random coin toss, his health information should be displayed on the left.

```
Scorpion                                                    Sub-Zero
HP[100]:||||||||||||||||||||||||||||||||||||||||||||||||||  HP[100]:||||||||||||||||||||||||||||||||||||||||||||||||||
```

Now, it is the Player1's turn and the player should decide how hard he will hit the enemy (i.e., attack magnitude). See the sample running output below.

```
--------------- Scorpion Attacks !! ---------------
Choose your attack magnitude between 1 and 50:
```

An attack with magnitude M will be successful with a chance of (100-M)%. That is, higher magnitude means higher risk of missing it. For instance, if M is 30, the chance of succeeding would be 70%, whereas, if M is 10, the chance of succeeding would be 90%. If an attack with magnitude

M is successful, the attacked hero’s health points will decrease by M. If the user writes more than 50 or less than 1, the game should warn the player, and re-ask for another attack magnitude as below.

```
--------------- Scorpion Attacks !! ---------------

Choose your attack magnitude between 1 and 50: 60

The attack magnitude must be between 1 and 50.

Choose your attack magnitude between 1 and 50:
```

After providing a valid magnitude, the game will decide if the attack would be successful (based on the above description using the random module). If the player misses the attack, a statement showing that the player missed the attack should be printed, and both of players remaining HP (health points) should be printed as below.

```
Ooopsy! Scorpion missed the attack!

Player1                                          Player2

HP[100]:||||||||||||||||||||||||||||||||||||||||||||||||||    HP[100]:||||||||||||||||||||||||||||||||||||||||||||||||||
```

If the attack is successful, the opponent's health will be decreased by the value of the magnitude, and the result of the attack should be printed followed by the current health point display as shown below.

```
Scorpion hits 30 damage!!

Scorpion                          Sub-Zero

HP[70]:|||||||||||||||||||||||||||||||||||          HP[100]:||||||||||||||||||||||||||||||||||||||||||||||||||
```

The fight should continue in the same way until one of the players’ health point goes below one. When one of the players HP goes below one, the fight should end, and the results should be printed showing the winner’ name as illustrated below.

```
Scorpion                          Sub-Zero

HP[20]:||||||||||                 HP[0]:

##########################################################################

############################## Scorpion Wins !! #################################

##########################################################################
```

After showing the battle result, the game should ask the users whether they want to play another round.

```
Do you want to play another round (Yes or No)? :
```

If the user writes No, the game should print the following and exit.

```
Thanks for playing! See you again!
```

Once the user writes Yes, the game should start again (The new game round should start from the "Coin Tossing Stage" in the above description). That is:

```
Coin toss result: Sub-Zero starts first!

Sub-Zero                                            Scorpion
HP[100]:||||||||||||||||||||||||||||||||||||||||||||||||||   HP[100]:||||||||||||||||||||||||||||||||||||||||||||||||||
```

If the user types something other than "Yes" or "No", the game should ask the same question again.

Implementation Notes:

- You may check this link to get further information about Random module. (http://www.pythonforbeginners.com/random/how-to-use-the-random-module-in-python)

Warnings:

- **Do not** talk to your classmates on project topics when you are implementing your projects (This is serious). **Do not** show or email your code to others (This is even more serious). If you need help, talk to your TAs or the instructor, not to your classmates. If somebody asks you for help, explain them the lecture slides, but do not explain any project related topic or solution. Any similarity in your source codes will have **serious** consequences for both parties.
- Carefully read the project document, and pay special attention to sentences that involve "**should**", "**should not**", "**do not**", and other underlined/bold font statements.
- If you use code from a resource (web site, book, etc.), make sure that you reference those resource at the top of your source code file in the form of comments. You should give details

of which part of your code is from what resource. Failing to do so **may result in** plagiarism investigation.

- Even if you work as a group of two students, each member of the team should know every line of the code well. Hence, it is **important** to understand all the details in your submitted code. You may be interviewed about any part of your code.

How and when do I submit my project? :

- Projects may be done individually or as a small group of two students (doing it individually is recommended for best learning experience). If you are doing it as a group, only **<u>one</u>** of the members should submit the project. File name will tell us group members (Please see the next item for details).
- Submit your own code in a **<u>single</u>** Python file. Name your code file with your and your partner's first and last names (see below for naming). o
    - If your team members are Deniz Barış and Ahmet Çalışkan, then name your code file as deniz_baris_ahmet_caliskan.py (Do **<u>not</u>** use any Turkish characters in file name).
    - If you are doing the project alone, then name it with your name and last name similar to the above naming scheme.
    - Those who **<u>do not</u>** follow the above naming conventions **<u>will get 5 pts off</u>** of their grade.
- Submit it online on LMS by **5 pm on Friday, Dec. 22, 2017**

Late Submission Policy:

- -10%: Submissions between 17:01 – 18:00 on the due date
- -20%: Submissions between 18:01 – midnight (00:00) on the due date
- -30%: Submissions which are 24 hour late.
- -50%: Submissions which are 48 hours late.
- Submission more than 48 hours late will not be accepted.

Grading Criteria? :

Code Organization			**Functionality**				
Meaningful variable names (3 pts)	**Proper use of functions, compact code with no unnecessary repetitions (4 pts)**	**Sufficient commenting (4 pts)**	**Coin Toss (10 pts)**	**Handling an Attack (40 pts)**	**Display Health Stats Properly (20 pts)**	**Making it continue to another round (10 pts)**	**Others (10 pts)**

Interview evaluation: Your grade from interview will be between 0 and 1, and it will be used as a coefficient to compute your final grade. For instance, if your initial grade was 80 before the interview, and your interview grade is 0.5, then your final grade will be 80*0.5 = 40. Not showing up for the interview appointment will **result in** grade 0.

Have further questions?:

Please contact your TAs if you have further questions. If you need help with anything, please use the office hours of your TAs and the instructor to get help. **Do not walk in randomly (especially on the last day) into your TAs' or the instructor's offices. Make an appointment first. This is important. Your TAs have other responsibilities. Please respect their personal schedules!**

Good Luck!