CS 111 – Introduction to Computer Science – Spring 2018

Programming Assignment #5

Jedi and Xwings

Due Date: at 11:59pm on Saturday, May 19.

For this assignment, you will design and implement programs that require the use of classes. Remember assignments are to be done individually, you should not share code, share design ideas, or look at anyone's code. If you have questions about this policy, refer to the syllabus or ask me.

Before getting started with the project, create a folder named `proj5` within your `U:\cs111` folder and use it to save your source file for this project.

Program Description

You will be creating two classes, a `Jedi` class and an `Xwing` class. Every `Jedi` will have a name and may or may not have a lightsaber. Every `Xwing` will have a pilot (a `Jedi`) and a random speed. `Jedi` will be immutable, but `Xwing` instances can have their speed increased, so they are mutable. You will also create a separate function that will allow two `Xwing` instances to do a flight battle simulation.

Remember that every module that uses instances of a class must import that class. Use the file `testJediXwingSimulator.py` to test as you code (comment out the parts you are not ready for.)

Jedi Class: `jedi.py` contains a class called `Jedi` with two instance variables: a name and a Boolean for whether the `Jedi` has a lightsaber. The `Jedi` class will also have the following methods:

- **constructor** – Receives a string parameter for the name and a Boolean for whether the `Jedi` has a lightsaber. Sets the instance variables appropriately. The constructor should allow for a default value of `False` for the lightsaber, so a `Jedi` can be created with a string name and a Boolean or just the string name.
- `getName` – Returns the name of the `Jedi`.
- `getLightsaber` – Returns `True` if the `Jedi` has a lightsaber, and `False` otherwise.
- **equals** – Implements the special method that allows `==` to be used for `Jedi`. Receives a parameter for another `Jedi`. Returns `True` if the `Jedi` have the same name and either both have a lightsaber or both do no have a lightsaber.
- **not equals** – Implements the special method that allows `!=` to be used for `Jedi`. Receives a parameter for another `Jedi`. Returns `True` if the `Jedi` have a different name or if only one of them have a lightsaber.

- **string representation** – Implements the special method that allows `str` and `print` to be used for `Jedi`. Returns a string as follows:

  ```
  Jedi Obi-Wan Kenobi with a lightsaber.
  ```

  if the `Jedi`'s name is Obi-Wan Kenobi and has a lightsaber or

  ```
  Jedi Mace Windu.
  ```

  if the `Jedi`'s name is Mace Windu and does not have a lightsaber.

Xwing Class: `xwing.py` contains a class called `Xwing` with two instance variables, for the pilot (an instance of the `Jedi` class) and the current speed. The `Xwing` class will also have the following methods

- **constructor** – Receives a `Jedi` parameter for the pilot. The current speed is set to a random integer between 200 and 500. Sets the instance variables appropriately.
- `getPilot` – Returns the name of the pilot.
- `isArmed` – Returns `True` if the pilot has a lightsaber, and `False` otherwise.
- `getSpeed` – Returns the current speed.
- `updateSpeed` – Randomly updates the speed by at least 1 and up to a total of 500. For example, if the current speed is 275, then the speed would be updated by a value between 1 and 225. The speed cannot pass 500.
- **string representation** – Implements the special method that allows `str` and `print` to be used for `Xwing`s. Returns a string as follows:

  ```
  Xwing with Jedi Obi-Wan Kenobi with a lightsaber.
  ```

  if the pilot's name is Obi-Wan Kenobi and has a lightsaber or

  ```
  Xwing with Jedi Mace Windu.
  ```

  if the pilot's name is Mace Windu and does not have a lightsaber.

Flight Battle Simulator: `simulator.py` is a module that contains the following function

- `flightBattle(xwing1, xwing2, distance)` – This function receives two `Xwing` instances and a distance and simulates a training battle, and then returns the name of the winner.

  First, it prints the string representation of each `Xwing`. Then it determines the time it will take for the `Xwing` instances to meet, this time is the distance divided by the sum of the speeds of the `Xwing` instances. The time is printed.

  Then, the simulator decides who wins the battle. If only one `Xwing` is armed, the pilot of the armed `Xwing` wins. Otherwise, randomly choose the winner. Print the results, and return the name of the winner. Some example output:

```
********** Flight Battle Simulator **********
X-wing with Jedi Mace Windu.
X-wing with Jedi Obi-Wan Kenobi with a lightsaber.

The X-wings meet after 1.6 minutes.

Obi-Wan Kenobi wins.

********** Flight Battle Simulator **********
X-wing with Jedi Luke Skywalker with a lightsaber.
X-wing with Jedi Yoda.

The X-wings meet after 2.8 minutes.

Luke Skywalker wins.

********** Flight Battle Simulator **********
X-wing with Jedi Yoda.
X-wing with Jedi Mace Windu.

The X-wings meet after 2.5 minutes.

Yoda wins.
```

Program Requirements

Your programs must be well structured and meet the following specifications:

- You programs must be commented appropriately, specifically you must:
  - Include an appropriate file prolog at the top of the source file.
  - Include a comment above each method or function including a description of the parameters, method/function, and return statement.
  - Include appropriate comments throughout the program.
  - Use meaningful variable names.
- Your programs should not include a main function or any floating code.

What to Submit

Please submit the following files to Canvas by the due date and time.

- `jedi.py`
- `xwing.py`
- `simulator.py`

Remember, all of the files must be named exactly as indicated above, with the same case and with no spaces or special characters.