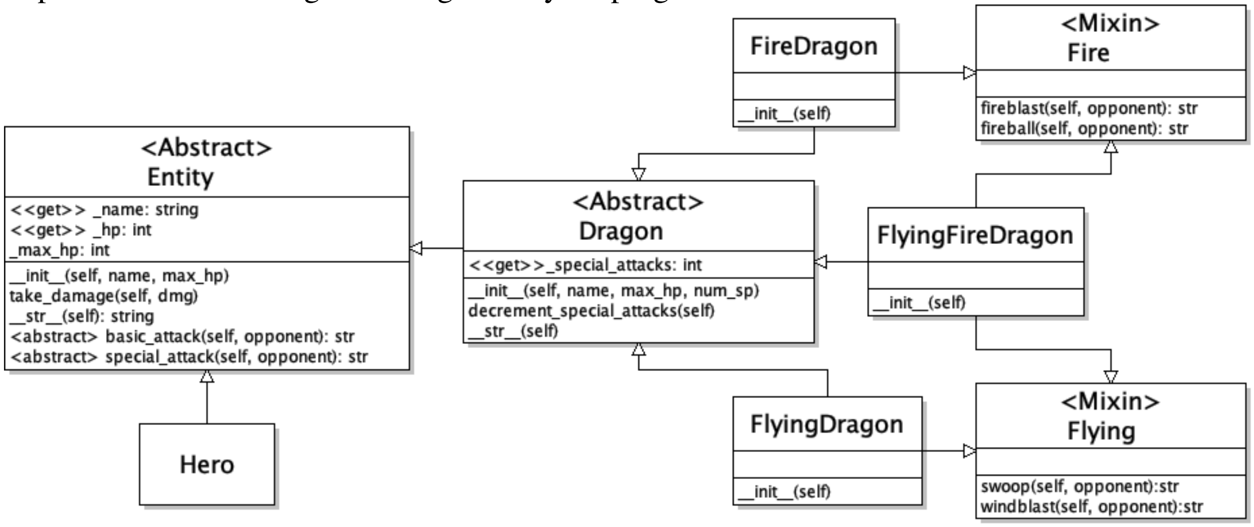
CECS 277 – Lab 9 – Mixins

Dragon Trainer

Create a game where the user must defeat three dragons to pass the trails. Use inheritance to implement the following class diagram in your program.


FireDragon
__init__(self)
<Mixin> Fire
fireblast(self, opponent): str
fireball(self, opponent): str
<Abstract> Entity
<<get>> _name: string
<<get>> _hp: int
_max_hp: int
__init__(self, name, max_hp)
take_damage(self, dmg)
__str__(self): string
<abstract> basic_attack(self, opponent): str
<abstract> special_attack(self, opponent): str
<Abstract> Dragon
<<get>>_special_attacks: int
__init__(self, name, max_hp, num_sp)
decrement_special_attacks(self)
__str__(self)
FlyingFireDragon
__init__(self)
Hero
FlyingDragon
__init__(self)
<Mixin> Flying
swoop(self, opponent):str
windblast(self, opponent):str

Entity Class (entity.py) – abstract class

1. `__init__(self, name, max_hp)` – set the _name, _max_hp, and _hp. Assign the max_hp value to both the _max_hp and _hp attributes.
2. name and hp properties – use decorators to get (not set) the values of _name and _hp.
3. `take_damage(self, dmg)` – the damage the entity takes. Subtract the dmg value from the entity's _hp. Do not let the hp go past 0 (if it's a negative value, reset it to 0).
4. `__str__(self)` – return the entity's name and hp in the format "Name: hp/max_hp".
5. Abstract methods basic_attack and special_attack. No code.

Hero Class (hero.py) – inherits from Entity

1. `basic_attack(self, opponent)` – the drago takes a random amount of damage in the range 2D6 (1-6 + 1-6). Return a string with the description of the attack and the damage dealt to the dragon.
2. `special_attack(self, opponent)` – the dragon takes a random amount of damage in the range 1D12 (1-12). Return a string with the description of the attack and the damage dealt to the dragon.

Dragon Class (dragon.py) – abstract class – inherits from Entity

1. `__init__(self, name, max_hp, num_sp)` – call super's init, then set _special_attacks.
2. `decrement_special_attacks(self)` – subtract 1 from _special_attacks. If the value becomes negative, then reset it to 0.
3. `basic_attack(self, opponent)` – tail attack – the hero takes a random amount of damage in the range 3-7. Return a string with the description of the attack and the damage dealt to the hero.
4. `__str__(self)` – use super to get the __str__ from the entity class, then concatenate on the number of special attacks remaining.

FireDragon Class (fire_dragon.py) – inherits from Dragon and the Fire mixin.

1. `__init__(self)` – call super to set a default name, hp, and number of special attacks.
2. `special_attack(self, opponent)` – randomly choose one of the two fire attack methods from the Fire mixin.

FlyingDragon Class (flying_dragon.py) – inherits from Dragon and the Flying mixin.

1. `__init__(self)` – call super to set a default name, hp, and number of special attacks.
2. `special_attack(self, opponent)` – randomly choose one of the two flying attack methods from the Flying mixin.

FlyingFireDragon Class (flying_fire_dragon.py) – inherits from Dragon, Fire, and Flying.

1. `__init__(self)` – call super to set a default name, hp, and number of special attacks.
2. `special_attack(self, opponent)` – randomly choose one of the four flying and fire attack methods from the Flying and Fire mixins.

Fire Class (fire.py) – Mixin

1. `fireblast(self, opponent)` – if the dragon has any special attacks left, then it blasts the hero with fire and they take a random amount of damage in the range 5-9 and the number of special attacks is decremented. Return a string with a description of the attack and the damage dealt to the hero. Otherwise, no damage is done and a string describing the failure is returned.
2. `fireball(self, opponent)` – if the dragon has any special attacks left, then it spits a fireball at the hero and they take a random amount of damage in the range 4-8 and the number of special attacks is decremented. Return a string with a description of the attack and the damage dealt to the hero. Otherwise, no damage is done and a string describing the failure is returned.

Flying Class (flying.py) – Mixin

1. `swoop(self, opponent)` – if the dragon has any special attacks left, then it does a swoop attack at the hero and they take a random amount of damage in the range 4-8 and the number of special attacks is decremented. Return a string with a description of the attack and the damage dealt to the hero. Otherwise, no damage is done and a string describing the failure is returned.
2. `windblast(self, opponent)` – if the dragon has any special attacks left, then it blasts wind at the hero and they take a random amount of damage in the range 3-7 and the number of special attacks is decremented. Return a string with a description of the attack and the damage dealt to the hero. Otherwise, no damage is done and a string describing the failure is returned.

Main (main.py) – Construct a Hero object and then create a list that contains one of each of the dragons (FireDragon, FlyingDragon, FlyingFireDragon,). Present a menu that allows the user to choose which dragon to attack, and then another menu that gives them the option of attacking with a sword or an arrow. Call the hero's attack method on the dragon they chose and display the attack message returned. If the user defeats the dragon (hp is 0), then remove that dragon from the list. Then choose a random (surviving) dragon that will attack the user, and randomly choose either a basic or special attack and display the attack message returned. Repeat the attacks until the user defeats all three dragons, or until the hero is knocked out. Check all input for validity.

Example Output (user input is in italics)**:**

```
What is your name, challenger?
Astrid

Welcome to dragon training, Astrid
You must defeat 3 dragons.

Astrid: 50/50
1. Gronkle: 15/15
Special attacks remaining: 3
2. Timberjack: 10/10
Special attacks remaining: 3
3. Deadly Nadder: 20/20
Special attacks remaining: 2
Choose a dragon to attack: 3

Attack with:
1. Sword (2 D6)
2. Arrow (1 D12)
Enter weapon: 1

Astrid slashes the Deadly Nadder
with their sword for 7 damage!
Gronkle engulfs Astrid in flames
for 7 damage!

Astrid: 43/50
1. Gronkle: 15/15
Special attacks remaining: 2
2. Timberjack: 10/10
Special attacks remaining: 3
3. Deadly Nadder: 13/20
Special attacks remaining: 2
Choose a dragon to attack: 3

Attack with:
1. Sword (2 D6)
2. Arrow (1 D12)
Enter weapon: 1

Astrid slashes the Deadly Nadder
with their sword for 7 damage!
Deadly Nadder smashes you with its
tail for 5 damage!

Astrid: 38/50
1. Gronkle: 15/15
Special attacks remaining: 2
2. Timberjack: 10/10
Special attacks remaining: 3
3. Deadly Nadder: 6/20
Special attacks remaining: 2
Choose a dragon to attack: 3

Attack with:
1. Sword (2 D6)
2. Arrow (1 D12)
Enter weapon: 2

Astrid hits the Deadly Nadder with
an arrow for 12 damage!
Gronkle engulfs Astrid in flames
for 6 damage!

Astrid: 32/50
1. Gronkle: 15/15
Special attacks remaining: 1
2. Timberjack: 10/10
Special attacks remaining: 3
Choose a dragon to attack: 3
Invalid input - should be within
range 1-2.
Choose a dragon to attack: 2

Attack with:
1. Sword (2 D6)
2. Arrow (1 D12)
Enter weapon: 1

Astrid slashes the Timberjack with
their sword for 8 damage!
Timberjack smashes you with its
tail for 6 damage!

Astrid: 26/50
1. Gronkle: 15/15
Special attacks remaining: 1
2. Timberjack: 2/10
Special attacks remaining: 3
Choose a dragon to attack: 2

Attack with:
1. Sword (2 D6)
2. Arrow (1 D12)
Enter weapon: 1

Astrid slashes the Timberjack with
their sword for 8 damage!
Gronkle smashes you with its tail
for 5 damage!

Astrid: 21/50
1. Gronkle: 15/15
Special attacks remaining: 1
Choose a dragon to attack: 1

Attack with:
1. Sword (2 D6)
2. Arrow (1 D12)
Enter weapon: 2
Astrid hits the Gronkle with an
arrow for 8 damage!
Gronkle spits a fireball at Astrid
for 8 damage!
```

```
Astrid: 13/50
1. Gronkle: 7/15
Special attacks remaining: 0
Choose a dragon to attack: 1

Attack with:
1. Sword (2 D6)
2. Arrow (1 D12)
Enter weapon: 1

Astrid slashes the Gronkle with
their sword for 4 damage!
Gronkle tries to engulf Astrid in
flames, but it is all out of fuel.

Astrid: 13/50
1. Gronkle: 3/15
Special attacks remaining: 0
Choose a dragon to attack: 1

Attack with:
1. Sword (2 D6)
2. Arrow (1 D12)
Enter weapon: 1

Astrid slashes the Gronkle with
their sword for 4 damage!

Congratulations! You have defeated
all three dragons, you have passed
the trials.
```

Notes:

1. You should have 10 different files: entity.py, hero.py, dragon.py, fire_dragon.py, flying_dragon.py, flying_fire_dragon.py, fire.py, flying.py, main.py, and check_input.py.
2. Check all user input using the get_int_range function in the check_input module.
3. Do not create any extra methods or functions or add any extra parameters.
4. Please do not create any global variables or use the attributes globally (ie. do not access any of them using the underscore). Access the name, hp, and special_attacks using the properties, even if they are inherited.
5. Use docstrings to document each of the classes and their methods.
6. Place your names, date, and a brief description of your program in a comment block at the top of your program. Place brief comments throughout your code.
7. Thoroughly test your program before submitting:
    a. Make sure that the Entity and Dragon classes are abstract.
    b. Make sure that your classes are inherited properly. Hero and Dragon should inherit from Entity, and Fire, Flying, and Flying Fire Dragons should inherit from Dragon and the appropriate mixins.
    c. Make sure user input is validated. Including when a dragon is defeated and removed from the list, that dragon should no longer be selectable.
    d. Make sure that the damage dealt is correctly subtracted from the opponent.
    e. Make sure that the randomly selected dragon is alive.
    f. Make sure that the dragons cannot do their special attack once they run out of their respective charges and no damage is dealt to the hero.
    g. Make sure the game ends when the user defeats all 3 dragons, or when the hero runs out of hp.

Dragon Trainer Rubric – Time estimate: 4 hours

Dragon Trainer **10 points**	Correct. 2 points	A minor mistake. 1.5 points	A few mistakes. 1 point	Several mistakes. 0.5 points	No attempt. 0 points
Entity and Hero Classes: 1. Abstract Entity, Hero extends Entity. 2. Has correct attributes, methods, properties, and attack methods. 3. take_damage checks that hp !< 0. 4. Hero's basic_attack -> 2D6 dmg. 5. Hero's special_attack -> 1D12 dmg.					
The Dragon Classes: 1. Dragon is abstract, extends Entity. 2. Has correct attributes, methods, properties, and attack methods. 3. Dragon's init calls super and initializes special_attacks. 4. Dragon's decrement checks !<0. 5. Dragon has basic_attack. 6. Dragons each extend Dragon. 7. Dragons call init with default vals. 8. Dragons special attacks use mixins.					
Fire/Flying Classes: 1. Fire has fireblast and fireball. 2. Flying has windblast and swoop. 3. Methods deal appropriate damage. 4. Methods return attack string.					
Main File: 1. Constructs Hero and a list of each of the three different dragons. 2. Prompts user for attack and attacks specified dragon with specified attack. 3. Random living dragon attacks back. 4. Displays attack strings returned from attack methods. 5. Displays updated hp. 6. Repeats until user dies or all dragons are slain. 7. All user input is validated.					
Code Formatting: 1. All code is in functions/methods. 2. Correct spacing and good naming. 3. No exceptions thrown. 4. No global variables or accessing attributes directly (use methods). 5. Correct documentation.					