OOP Python

March 20, 2021

0.1 What is OOP

- a process in which we can solve problems and abstract away systems

Encapsulation: - all of the internal workings are encased - this means all the data and the methods that modify that data - python does not have true encapsulation

Abstraction: - a particular set of properties and behaviors that are bundeled together in side of an object - alof of the processes are hidden away - you only care about interfaces and how you interact with the object - you dont care about the internals - for abstraction, you only care about the **init** method variables

Inheritance: - you can pass `base` behaviores and properties to another class - inheritance allows you to not repeat your-self

Polymorphism: - the class that is inherting can change - basically objects can override inherited behavior to do different things

0.2 Creating Classses and Objects in Python

- a class is basically a blueprint
- think about different classes of bacteria
- each bacteria has its own DNA and thus belong to a different class
- an object is an initiated class
- `self` refers to the object

[3]:
```
class Bacteria:
    def __init__(self, size):
        self.size = size

    def growth(self, amount):
        size_increase = ('increase', amount)
        self.size += amount
        return f'Bacteria size increased by {amount} to {self.size }'

    def __repr__(self):
        return f"<Bacteria Size: {self.size}>"
```

0.3 Extending Classes

- Extending classes helps us keep clean code bases by rewriting less code

- python can has multiple inheritance that can lead to the diamond of death
- order matter when using multiple inhertance

- favor composition over inheritance
- composition is when you have an instance of another class inside of your class over using inheritance
- problem is your duplicating code
- the better alternative is to use a function

- bacteria use composition with thier energy production
- there delegate the energy production to mitocondria
- but for something like mitosis, they inherit and take the responsibility
- `composition` of different objects

[6]:
```python
class Reader:
    def __init__(self, filepath):
        self.filepath = filepath

    def read(self):
        with open(self.filepath, 'r') as f:
            print(f.read())


class Writer:
    def __init__(self, filepath):
        self.filepath = filepath

    def write(self, data):
        with open(self.filepath, 'w') as f:
            f.write(data)


class Book:
    def __init__(self, title, author, filepath):
        self.title = title
        self.filepath = filepath
        self.author = author

        # notice the composition being done here
        self.reader = Reader(filepath)

        # notice the composition being done here
        self.writer = Writer(filepath)

    # we will proxy our read to the Reader class
    def read(self):
        self.reader.read()
```

```
    def write(self, data):
        self.writer.write(data)
```

0.4 MonsterSlash

[74]:
```
## actions.py

from random import randint

class Actor:
    def __init__(self, name, level):
        self.name = name
        self.level = level
        self.health = 100 * level


    def get_attack_power(self):
        return randint(1, 10) * self.level

    def is_alive(self):
        return self.health > 0

    def attacks(self, other):
        raise NotImplementedError()

    def stats(self):
        print(f'{self.name} has {self.health} hp')

    def __repr__(self):
        return f'<Actor: {self.name} at level {self.level}>'


class Player(Actor):

    def heal(self):
        self.health += 10

    def attacks(self, enemy):
        power = self.get_attack_power()
        print(f'{self.name} attacks {enemy.kind}.')
        print(f'{self.name} has {power} attack power')
        enemy.health -= power

    def __repr__(self):
        return f'<Player: {self.name} at level {self.level}>'

class Enemy(Actor):
```

```
    def __init__(self, name, level, kind):
        super().__init__(name, level)
        self.kind = kind

    def attacks(self, player):
        print(f'{self.name} the {self.kind} attack {player.name}')
        enemies_power = self.get_attack_power()
        print(f'{self.name} has {enemies_power} attack power')
        player.health -= enemies_power


class Ursa(Enemy):
    def __init__(self, name, level, size):
        super().__init__(name, level, 'Ursa')
        self.size = size


    def get_attack_power(self):
        return randint(1, 5) * (self.size * self.level)

class Riki(Enemy):
    def __init__(self, name, level):
        super().__init__(name, level, 'Ursa')

    def get_attack_power(self):
        return  super().get_attack_power() // 4

if __name__ == '__main__':
    player = Player('Naga Siren', 9)
    ursa = Ursa('ursa', 1, 2)
    player.attacks(ursa)
```

```
Naga Siren attacks Ursa.
Naga Siren has 81 attack power
```

[73]:
```
## game.py

# from actors import Player, Enemy
import random

class Game:
    def __init__(self, player, enemies):
        self.player = player
        self.enemies = enemies

    def main(self):
        self.print_intro()
```

```
        self.play()

    def print_intro(self):
        print('''
            Monster Slash!!!
            Ready Player One?
            [Press Enter to Continue]
        ''')
        input()


    def print_linebreak(self):
            print()
            print('*'*30)
            print()

    def play(self):
        while True:
            next_enemy = random.choice(self.enemies)

            cmd = input(f'you see a {next_enemy.kind}. [r]un, [a]ttack, [p]ass?
↪')

            if cmd == 'r':
                print(f'{self.player.name} heals')
                self.player.heal()
            elif cmd == 'a':
                self.player.attacks(next_enemy)
                if not next_enemy.is_alive():
                    self.enemies.remove(next_enemy)
                    next_enemy = None
                if next_enemy:
                    next_enemy.attacks(self.player)
            elif cmd == 'p':
                print('You are still thinking about your next move...')
                if random.randint(1, 11) < 5:
                    next_enemy.attacks(self.player)
            else:
                print('Please choose a valid option!')

            if not self.player.is_alive():
                print('Oh no! You Lose')
                break

            self.print_linebreak()

            self.player.stats()
```

```
            for e in self.enemies:
                e.stats()

            self.print_linebreak()
            if not self.enemies:
                print('You have won! Congratulations')
                break

if __name__ == '__main__':
    #main()
    enemies = [
        Riki('riki', 8),
        Ursa('bear', 4, 6)
    ]
    player = Player('Naga Siren', 6)

    game = Game(player, enemies)
    game.main()
```

```
            Monster Slash!!!
            Ready Player One?
            [Press Enter to Continue]

p
you see a Ursa. [r]un, [a]ttack, [p]ass?p
You are still thinking about your next move…

******************************

Naga Siren has 600 hp
riki has 800 hp
bear has 400 hp

******************************

you see a Ursa. [r]un, [a]ttack, [p]ass?
Please choose a valid option!

******************************

Naga Siren has 600 hp
riki has 800 hp
bear has 400 hp

******************************

you see a Ursa. [r]un, [a]ttack, [p]ass?p
```

```
You are still thinking about your next move…

*****************************

Naga Siren has 600 hp
riki has 800 hp
bear has 400 hp

*****************************

you see a Ursa. [r]un, [a]ttack, [p]ass?r
Naga Siren heals

*****************************

Naga Siren has 610 hp
riki has 800 hp
bear has 400 hp

*****************************

you see a Ursa. [r]un, [a]ttack, [p]ass?r
Naga Siren heals

*****************************

Naga Siren has 620 hp
riki has 800 hp
bear has 400 hp

*****************************

you see a Ursa. [r]un, [a]ttack, [p]ass?r
Naga Siren heals

*****************************

Naga Siren has 630 hp
riki has 800 hp
bear has 400 hp

*****************************

you see a Ursa. [r]un, [a]ttack, [p]ass?a
Naga Siren attacks Ursa.
Naga Siren has 300 attack power
riki the Ursa attack Naga Siren
riki has 188 attack power
```

```
*****************************

Naga Siren has 442 hp
riki has 500 hp
bear has 400 hp

*****************************

you see a Ursa. [r]un, [a]ttack, [p]ass?a
Naga Siren attacks Ursa.
Naga Siren has 558 attack power
riki the Ursa attack Naga Siren
riki has 194 attack power

*****************************

Naga Siren has 248 hp
bear has 400 hp

*****************************

you see a Ursa. [r]un, [a]ttack, [p]ass?a
Naga Siren attacks Ursa.
Naga Siren has 438 attack power
bear the Ursa attack Naga Siren
bear has 480 attack power
Oh no! You Lose
```