ANSWERS

1. A class can be seen as a description of a type. A blueprint for making specific objects of the class. An object therefore is one specific instance or version of a class. A Human can be seen as a class and John and Eric are two versions of Human.

2. Inheritance is a way to let one class have all the attributes and methods from it's parent, in addition to it's own.
This allows us to avoid duplicating code if we want to make many only slightly different types of objects.
We could for instance have an Animal class and then we could have various classes derive the same basic features and methods
from this parent class. They can also add more attributes and methods specific to their class in addition.
For example:

```
class Animal():
    def __init__(self):
    self.heart = True
    self.eyes = 2
    self.legs = random.randint(2,30)

    def heartbeat():
        self.eyes += 1

class Unlaidenedswallow(Animal):
    def __init__(self):
        super().__init__()
        self.average_air_speed_velocity = 56
        self.flying = True
```

3. An instance of an object is always a version of its class.
John is a Human, Eric is also a Human. However, in our Game-class, we make and manage several different
objects, such as two instances of the Player class. It can then be said to have a Player.

4. Encapsulation is a privat method or variable that can only be used or changed within the class, you cant change a private variable or access a private method from another class.
to create a private variable or mathod in python use "_" in front of the variable name/ method name.

5. Polymorhpism is the ability to do different things depending on what type or class an object is.
The perfect example from our implementation is the update method. Every sprite in the all_sprite_list sprite group
has its own update method. They will do different things depending on what they are, Player, Explosion, Bullet etc
But the game need not know how, it simply calls update on all of them, and leave it to the classes to update themselves correctly.
The built in draw method inherited from sprite works in a similar way. It will blit all the self.image on a given surface.