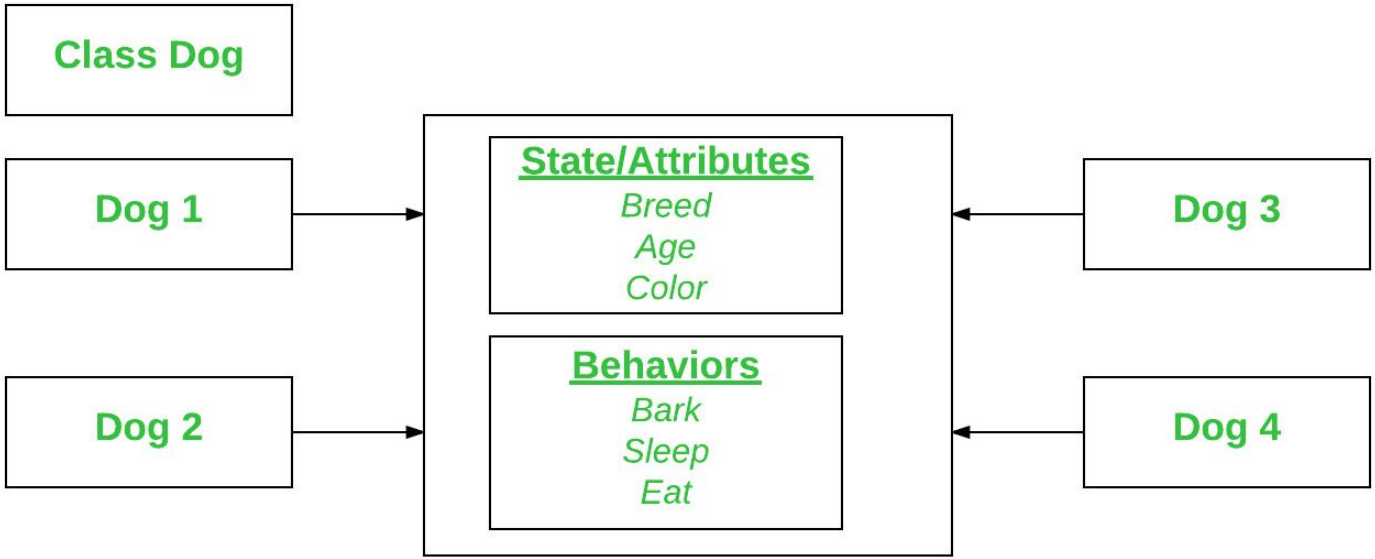
Atributi (stanja) i ponašanja (metode)


Class Dog
Dog 1
Dog 2
State/Attributes
Breed
Age
Color
Behaviors
Bark
Sleep
Eat
Dog 3
Dog 4

Stanja objekata (Atributi)

Kao što smo mogli u proceduralnom programiranju da deklarišemo i inicijalizujemo odgovarajuće promenljive za glavni deo programa ili za pomoćne funkcije, tako možemo definisati i promenljive za same objekte.

Promenljive definisane unutar klase nazivaju se **atributi**.

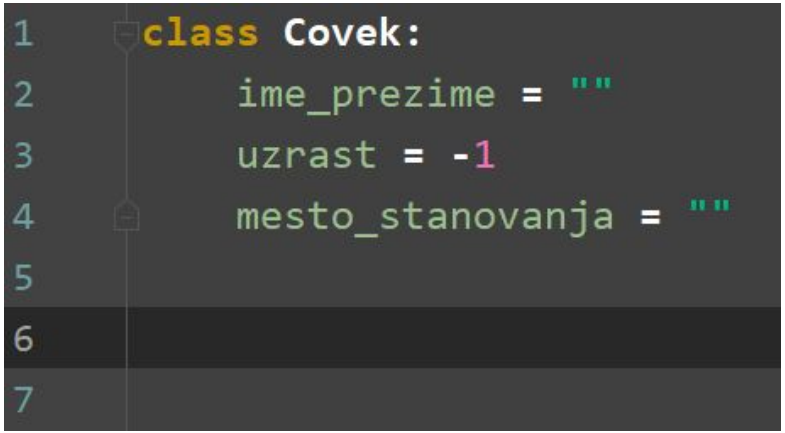
Praksa za Python programere jeste da atribute, vezane za sam objekat, često definišu u konstruktoru pomoću reference **self**.

U slučaju drugih programskih jezika poput Java i C#, odomaćeno je da se atributi deklarišu i/ili inicijalizuju ispod definicije klase, a iznad prve metode.

Primer: **self.naziv**

Primer 4.

Napisati program u kome se definiše klasa **Covek**. Svaki čovek treba posedovati sledeće atribute: **ime_prezime**, **uzrast** i **mesto_stanovanja**.



Ponašanja objekata (Metode)

Za objekte nam je u malo situacija dovoljno da definišemo koje vrednosti poseduje, ako ne postoji način da ih iskoristimo. Zato postoje ponašanja objekata, odnosno metode.

Metode predstavljaju specijalizovane funkcije koje se odnose isključivo na trenutni tip.

Njih definišemo na isti način kao kod regularnih funkcija, samo što ih navodimo u okviru klase i obavezno navodimo prvi argument self, koji je pri pozivu metode podrazumevan i ne mora da se navodi u glavnom delu programa.

Primer: **pas.odazovi_se()**

Primer 5.

Napisati program u kome se definiše klasa **Pas**. Svaki pas treba posedovati sledeće atribute: **rasa**, **uzrast** i **boja**. Takođe, svaki pas treba da ima sledeće metode: **odazovi_se()**, **spavaj()** i **jedi()**.

```
1   class Pas:
2       rasa = ""
3       uzrast = -1
4       boja = ""
5
6       def odazovi_se(self):
7           pass
8
9       def spavaj(self):
10          pass
11
12      def jedi(self):
13          pass
14
15
```

Klasne (statičke) promenljive

Nekada je neophodno da imamo podatak koji je zajednički za sve objekte date klase. Pretpostavimo da hoćemo da čuvamo brojno stanje nekih objekata. Ne bi bilo moguće da takav podatak čuvamo u svakom od objekata, jer bi ažuriranje podatka u jednom objektu povuklo i ažuriranje u svim drugim. U takvim situacijama možemo da definišemo promenljivu strogo vezanu, ne samo za objekat, već i za samu klasu.

Promenljive ovog tipa nazivaju se u Python-u **klasne** ili **statičke** promenljive .

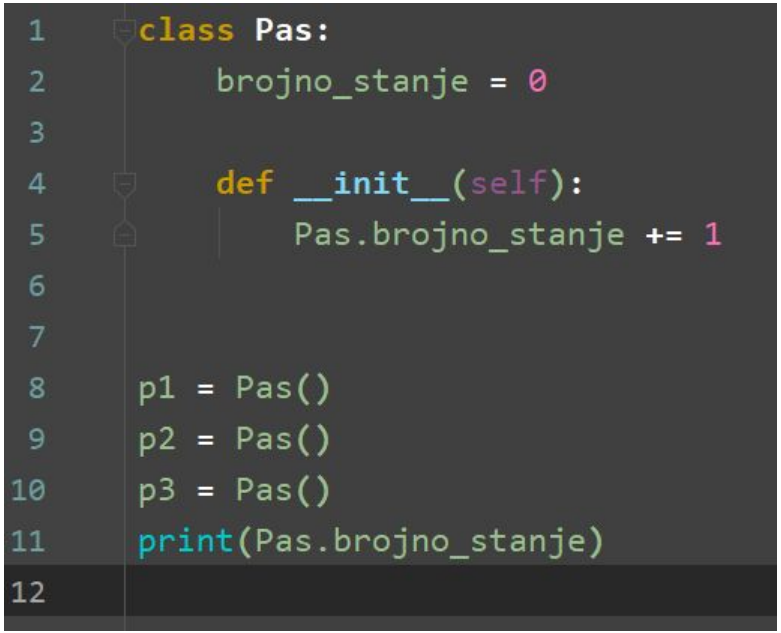
Klasne promenljive obično se definišu u okviru klase iznad definicija metoda. Neophodno je napomenuti da se klasne promenljive ne vezuju za objekat (self).

Pristup vrednosti klasnim promenljivama vrši se preko naziva klase.

Primer: **Pas.broj_pasa**

Primer 6.

Napisati program u kome se definiše klasa **Pas**. U okviru klase voditi brojno stanje o ukupnom broju kreiranih pasa. Napraviti 3 instance klase **Pas** i ispisati brojno stanje.



Statičke metode

Dodatno, nije uvek neophodno da imamo instanciran objekat ako želimo da pozovemo određenu metodu.

Pretpostavimo da imamo klasu Matematika i u okviru nje definisanu metodu koja vraća proizvod 3 broja koji se prosleđuju kao argumenti te metode.

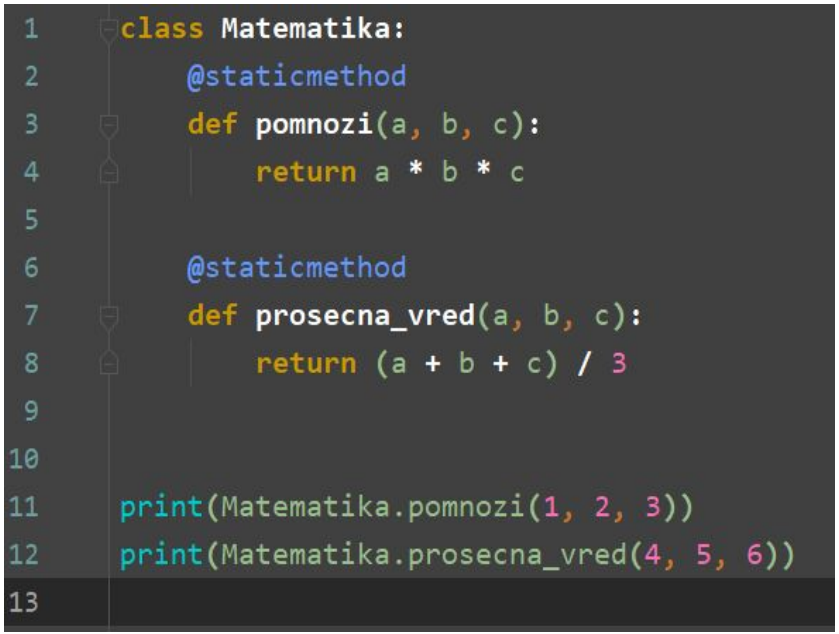
Sama ta metoda može da odredi povratnu vrednost bez uzimanja u obzir druge atribute klase Matematika.

U takvim situacijama, kada nemamo slučaj da funkcija vrši neki rad sa atributima u okviru objekta, takve metode nazivamo **statičkim**.

Statičke metode nemaju referencu na trenutni objekat, te ne prosleđujemo parametar **self**.

Primer 7.

Napisati program u kome se definiše klasa **Matematika**. Definisati dve statičke metode: **pomnozi()** i **prosecna_vred()**. U glavnom delu programa, ispisati vrednosti poziva metoda za redom argumente: 1, 2, 3 i 4, 5, 6.



```python
class Matematika:
    @staticmethod
    def pomnozi(a, b, c):
        return a * b * c

    @staticmethod
    def prosecna_vred(a, b, c):
        return (a + b + c) / 3


print(Matematika.pomnozi(1, 2, 3))
print(Matematika.prosecna_vred(4, 5, 6))

```