**Exercise: 6**

**Name: Pekka Lehtola**

**How many tasks did you do: 8**

**Were the tasks easy, ok, difficult: Easy**

**Do you need help/comments in any task (if yes, to which ones):**

1. Explain the following terms:

**a. Super class**

Super luokalla tarkoitetaan sitä luokkaa mistä ollaan peritty, käytetään myös nimitystä parent

**b. Sub class**

Sub class tarkoittaa luokkaa mikä on peritty Super classista.

**c. Base class**

Oman ymmärrykseni mukaan, Base class = Super class

**d. Derived class**

Derived class = Sub class

**e. “Is a” relationship**

Tällä tarkoitetaan suhdetta luokkien välillä.

Dog **is a** domestic\_animal. Domestic\_animal **is a** Mammal

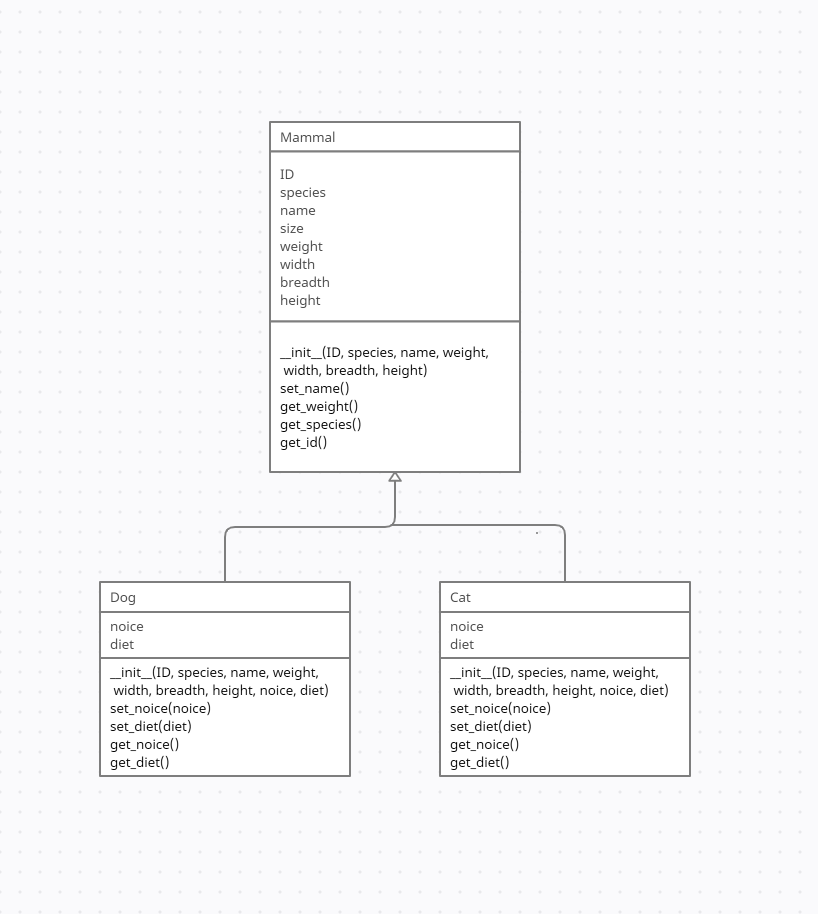
Platypus **is a** Wild\_animal. Wild\_animal **is a** Mammal

Student **is a** participant of OOP course.

Teacher **is a** participant of OOP course.

Jos tämä ei käy järkeen luokissa on jonkinlainen logiikka virhe.

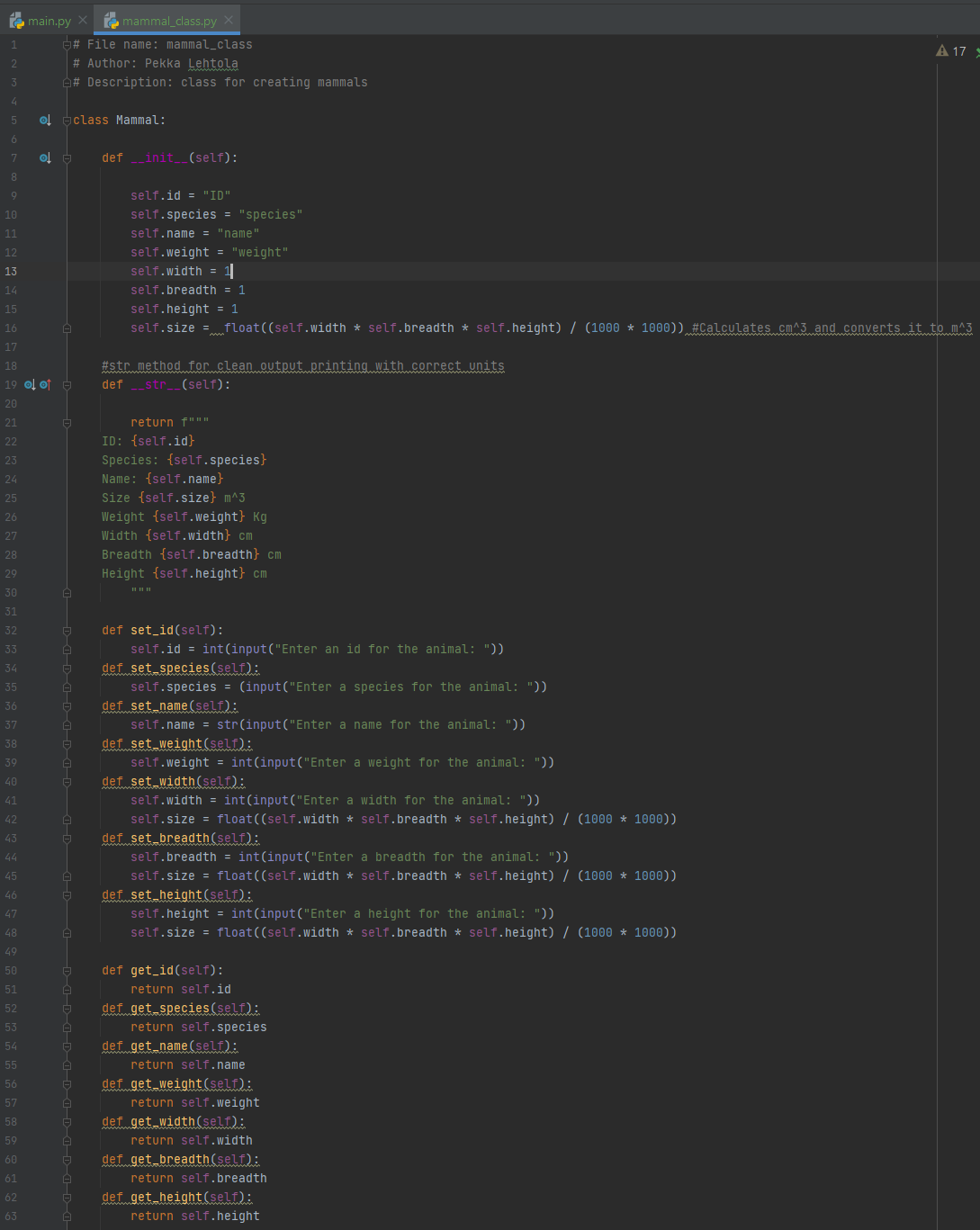
2. Draw a UML diagram of task 3. Use UML syntax and see how data attributes and methods are presented in the example presented on class (about Automobile), can be found on lecture slides. You can use any software or e.g. draw by hand and take a picture. Example in lectures is drawn using MS Visio.

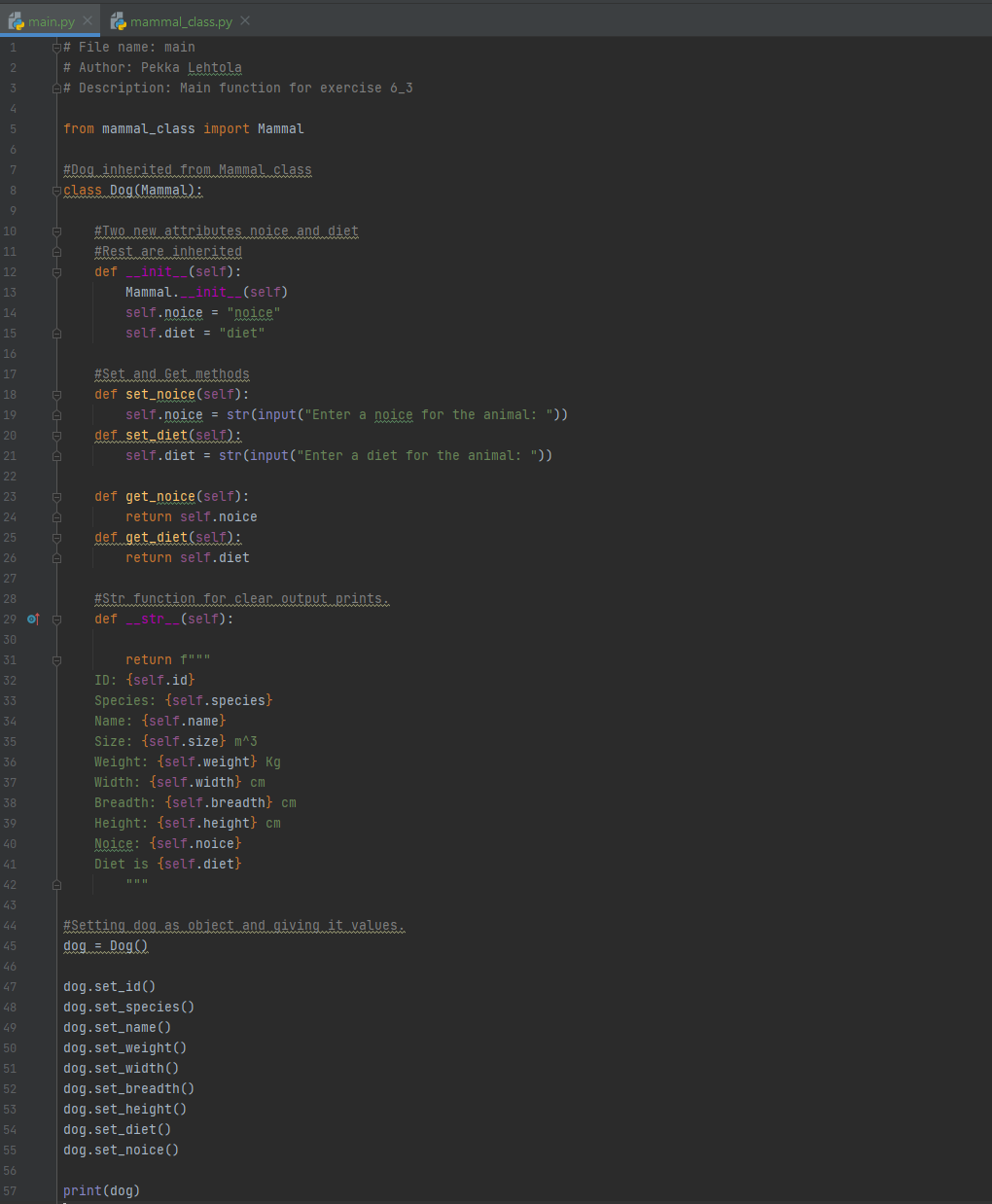


3. Inherit some animals from the Mammal class (that you created in Exercise 4). Add data attribute for the noise the animal makes and the diet they have. Display your objects on screen (= Print out the state of each object (use str-method)).

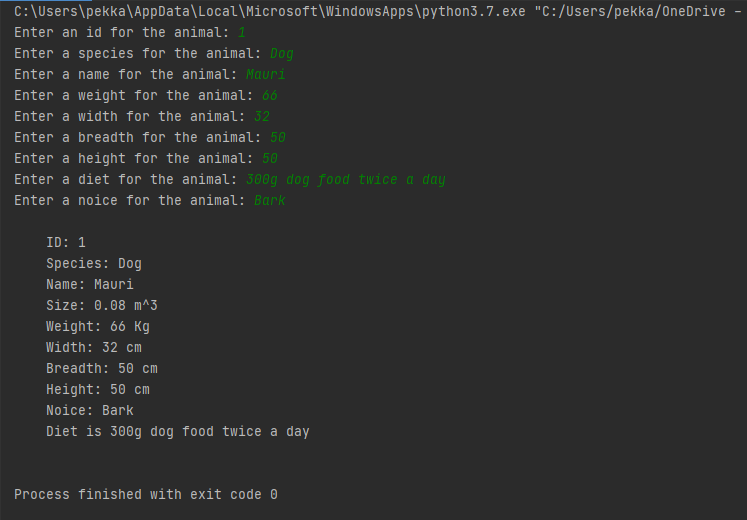
Screen capture of Task 3

Mammal class:

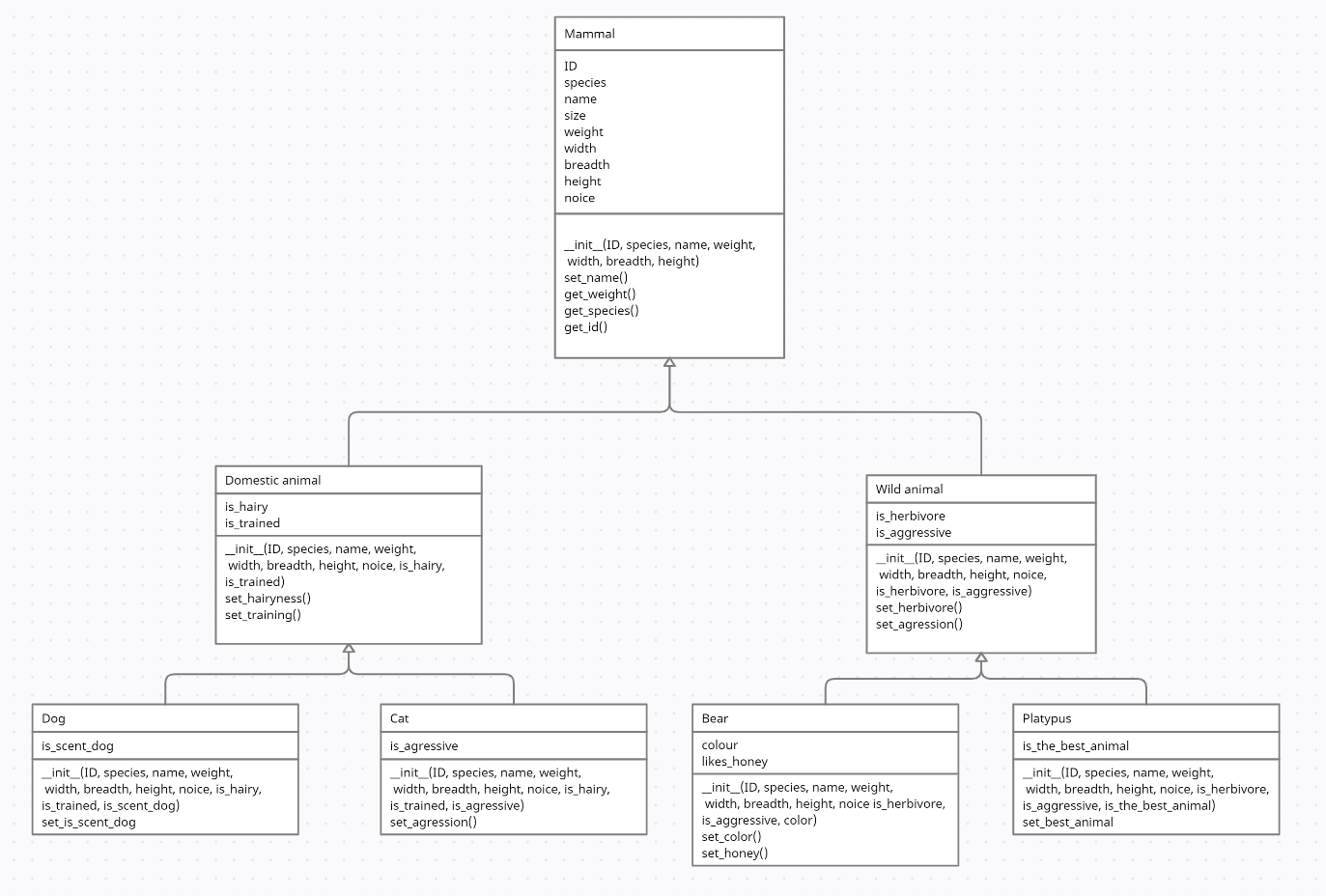


Main:

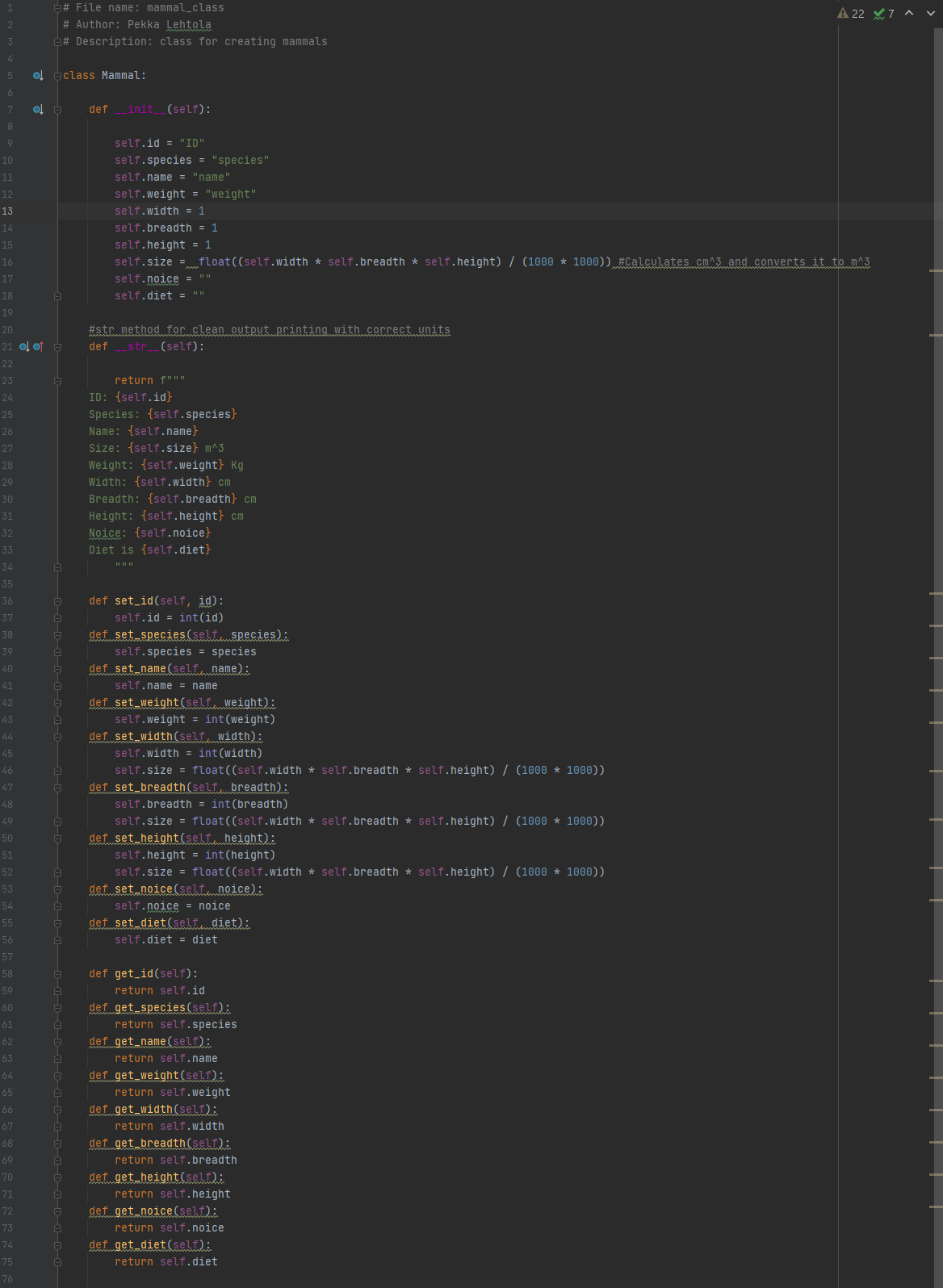
Screen capture of the output of Task 3

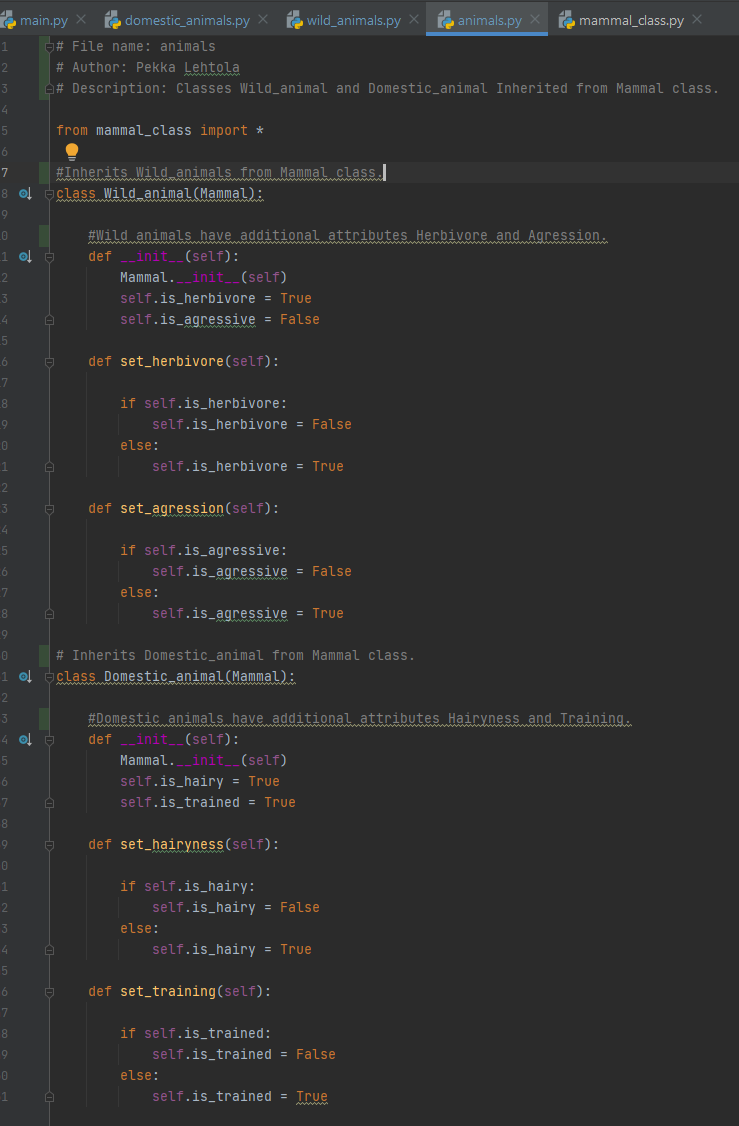


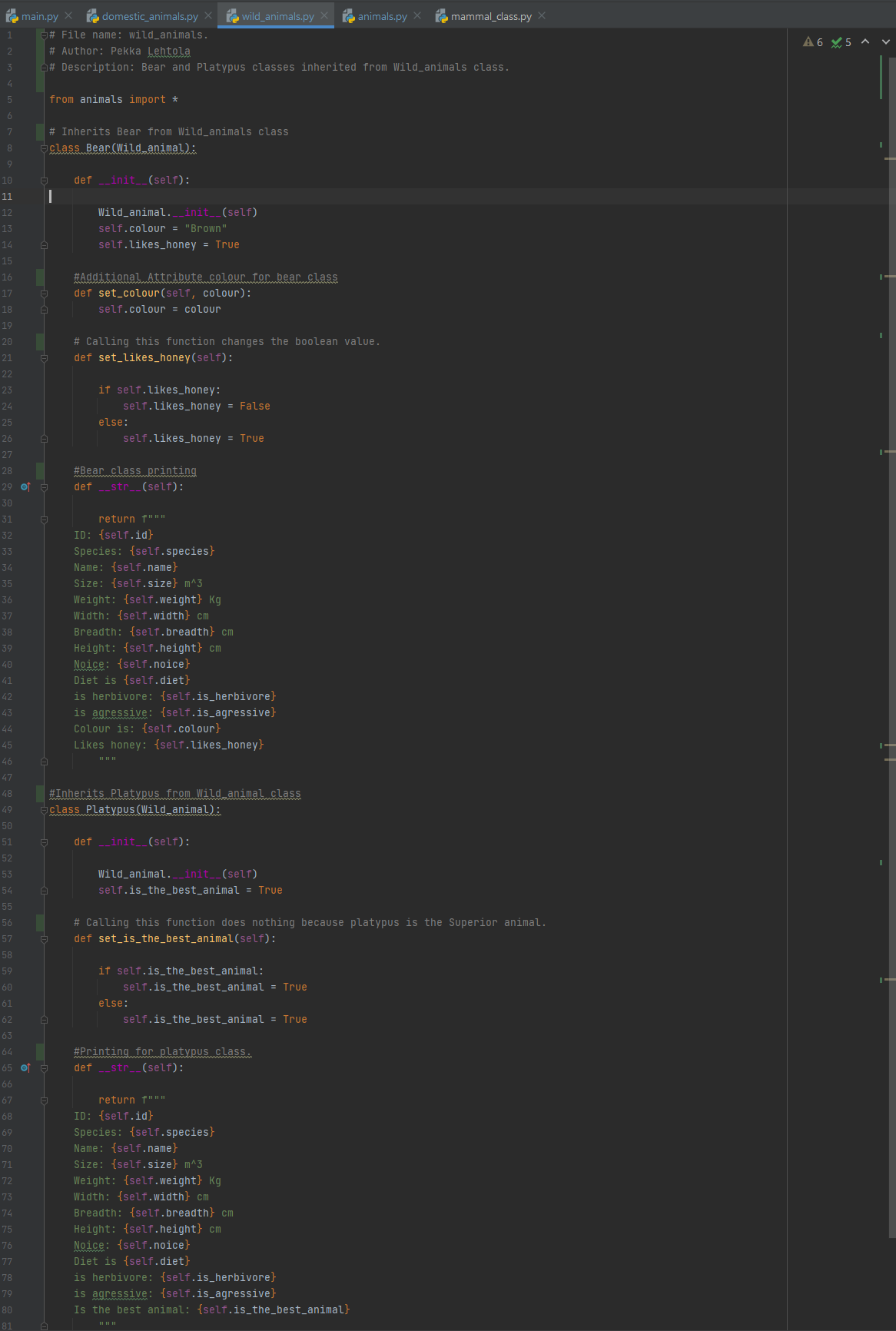
4. Draw a UML diagram of exercise 5. Use UML syntax and see how data attributes and methods are presented in the example presented on class (about Automobile etc.).



5. Change your task 3 code like this: Inherit a domestic animal from Mammal. Also inherit a wild animal from Mammal. Then inherit a few domestic and wild animals from those classes and print them out. Each mammal should make unique noise and have a certain diet as additional data attributes. Add some relevant attributes. Display your objects on screen.

Screen capture of Task 5 Mammal\_class:

Animals:

Wild\_animals:

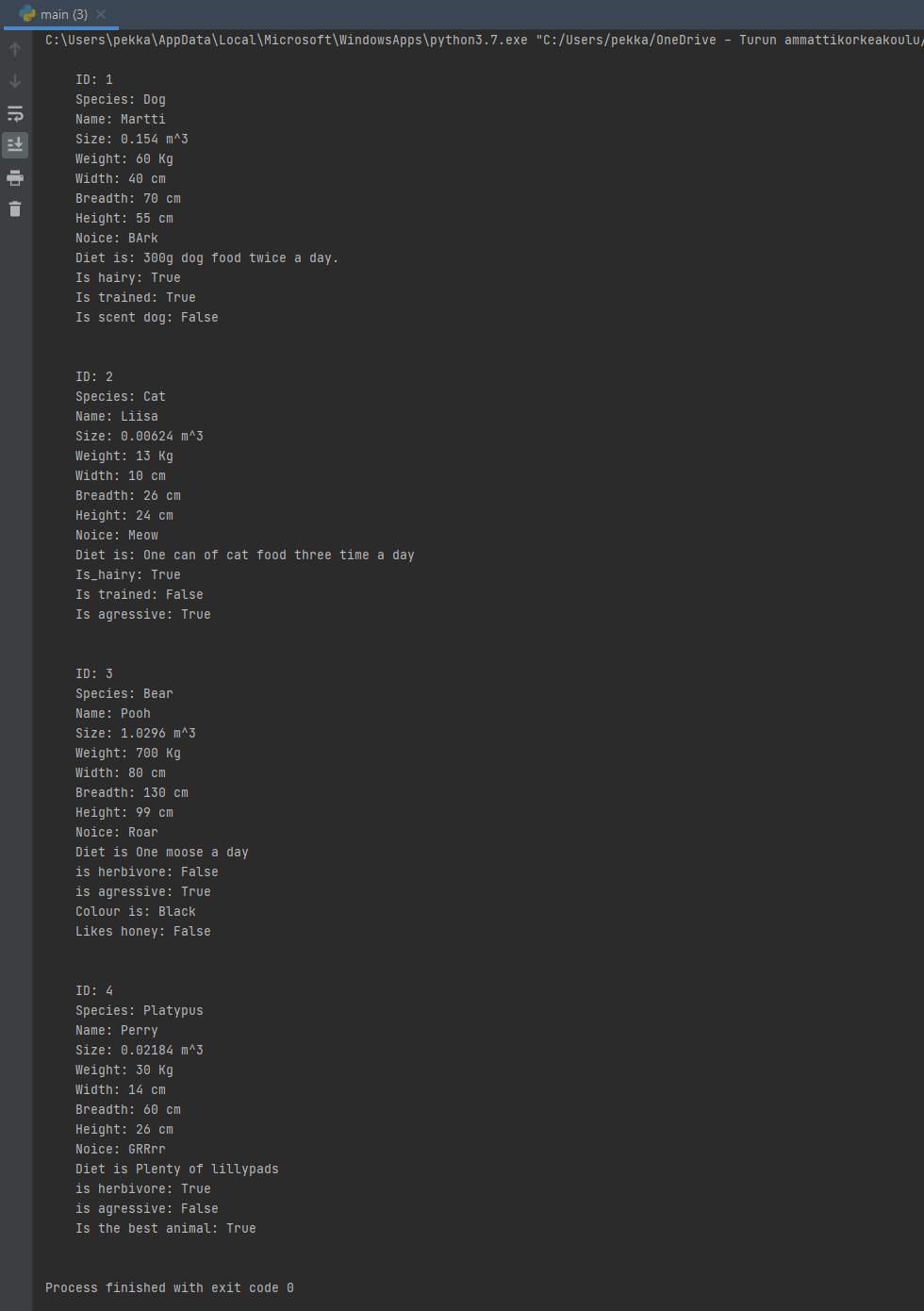
Domestic\_animals:



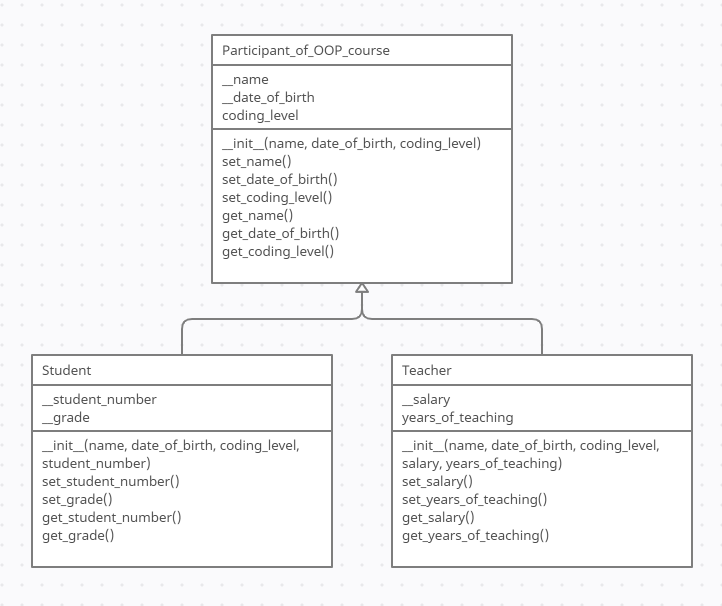
Main part 1: Main part 2:



Screen capture of the output of Task 5



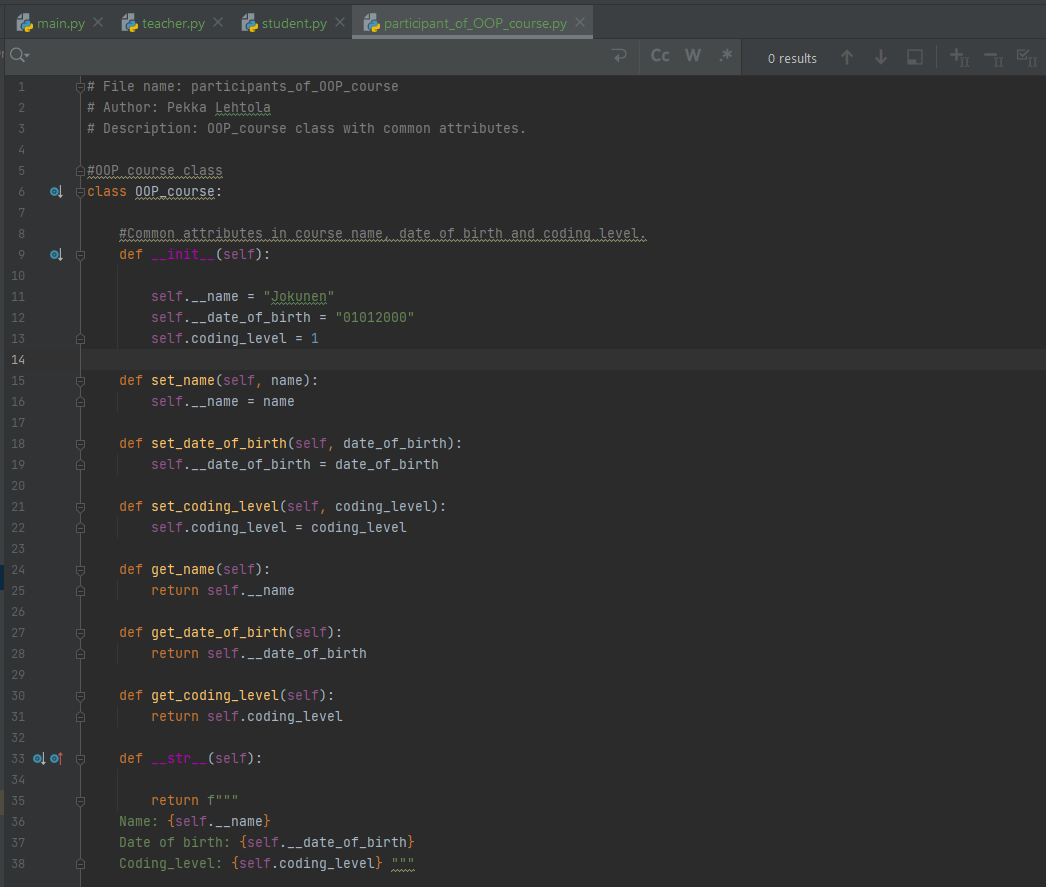
6. Draw a UML diagram of tasks 7. Use UML syntax and see how data attributes and methods are presented in the example presented on class (about Automobile etc.).

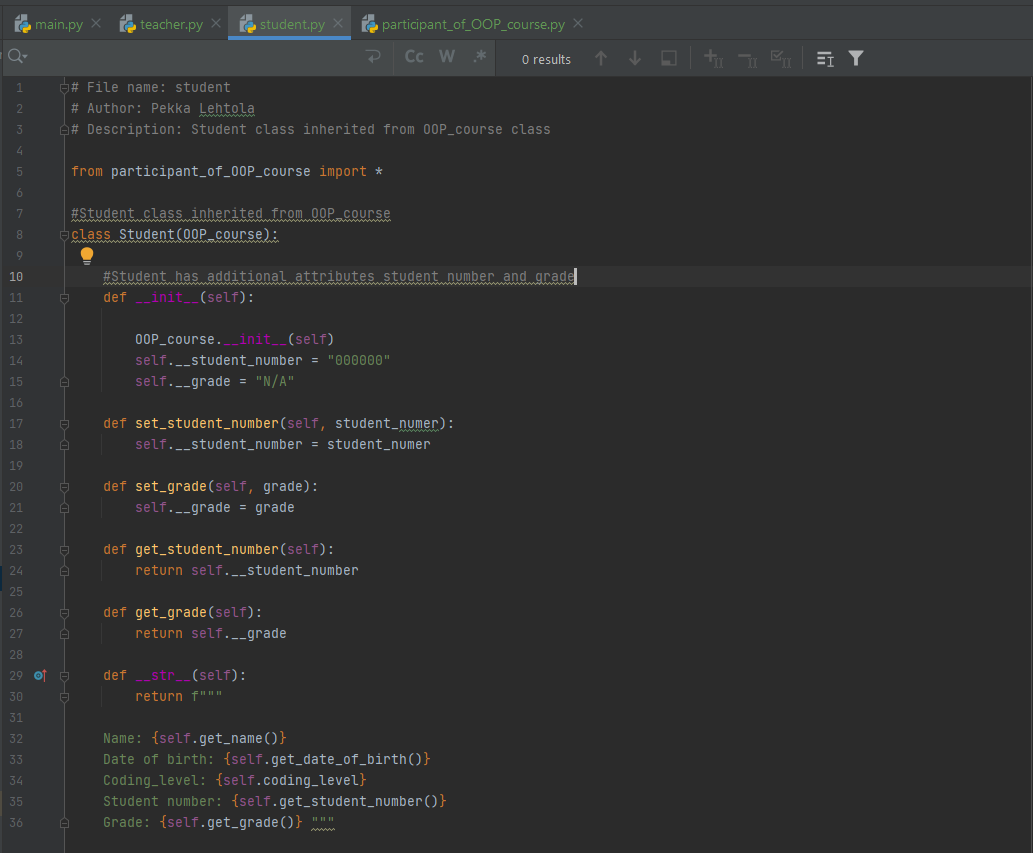


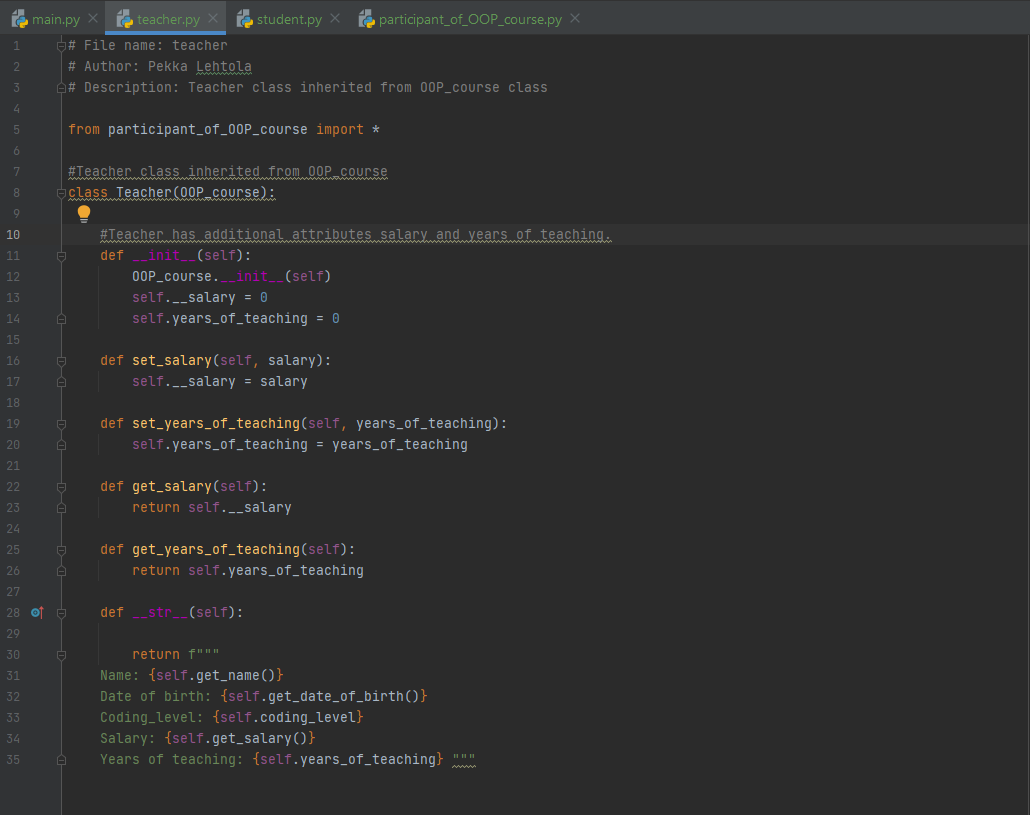
7. Inherit a student and teacher from a participant of OOP course. Think a few proper data attributes that are 1) common for both teachers and students and 2) different between teachers and students.

Screen capture of Task 7

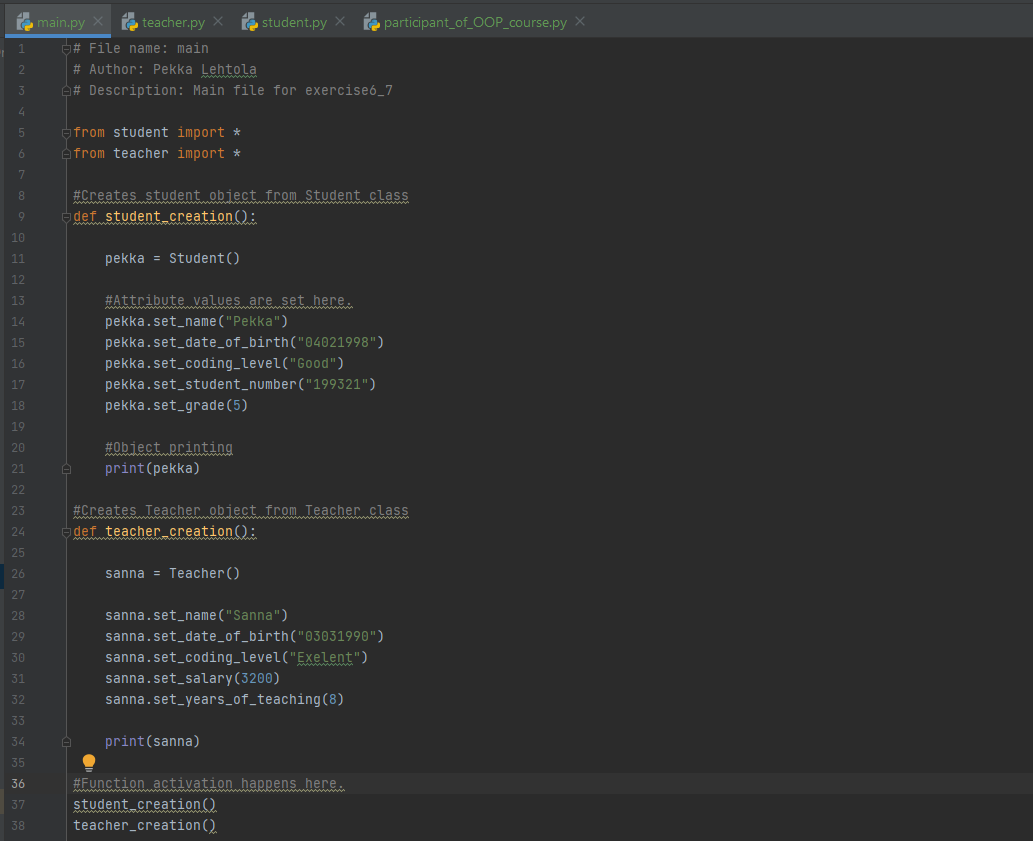
Participant\_of\_OOP\_course:



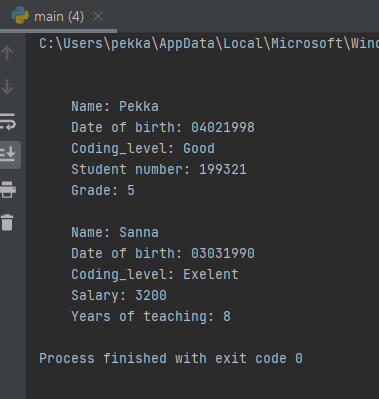
Student:

Teacher:

Main:



Screen capture of the output of Task 7

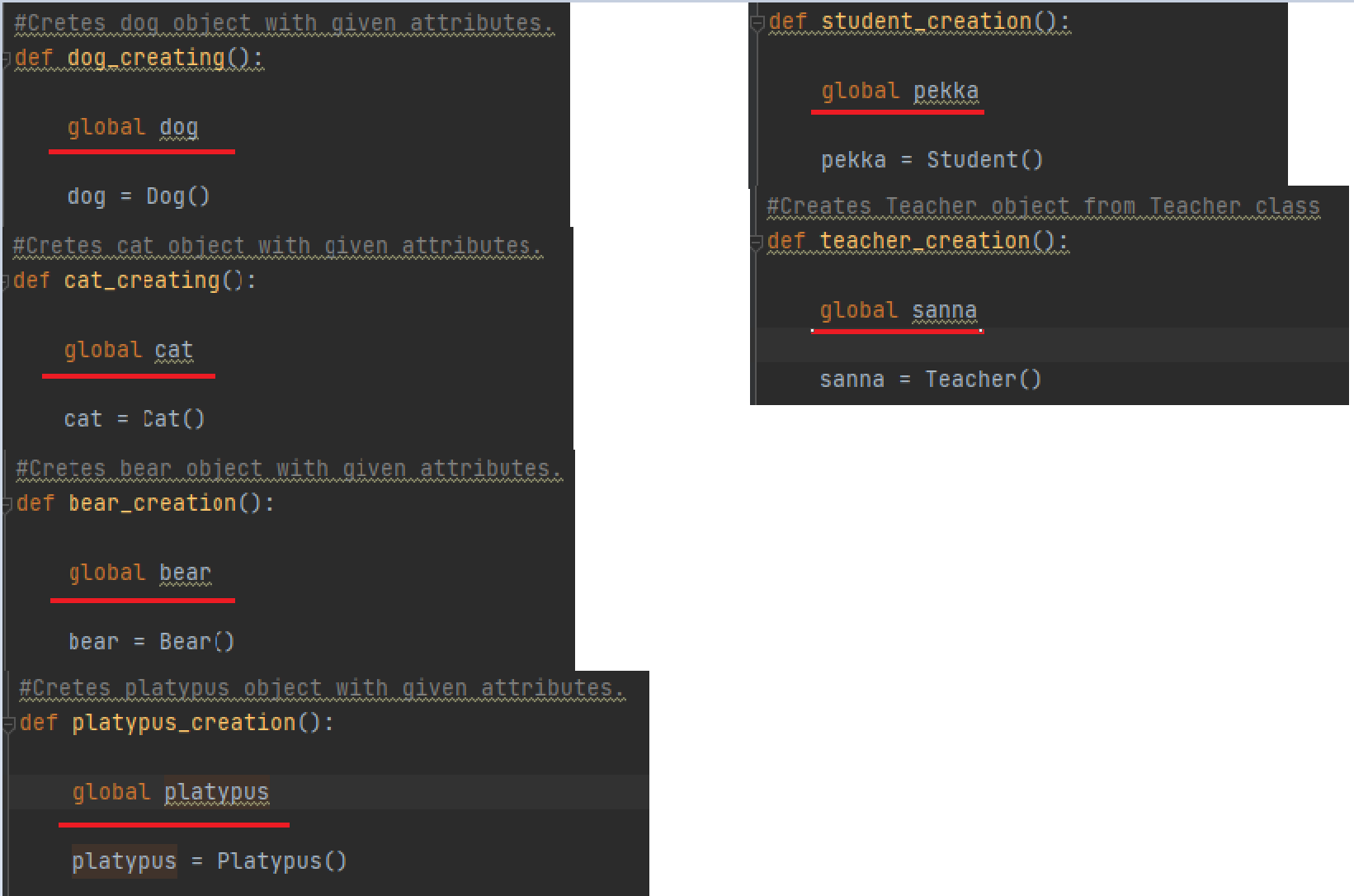


8. Each participant of Task 7 has also 1 domestic animal and 1 wild animal. Display the teachers, students and their information.

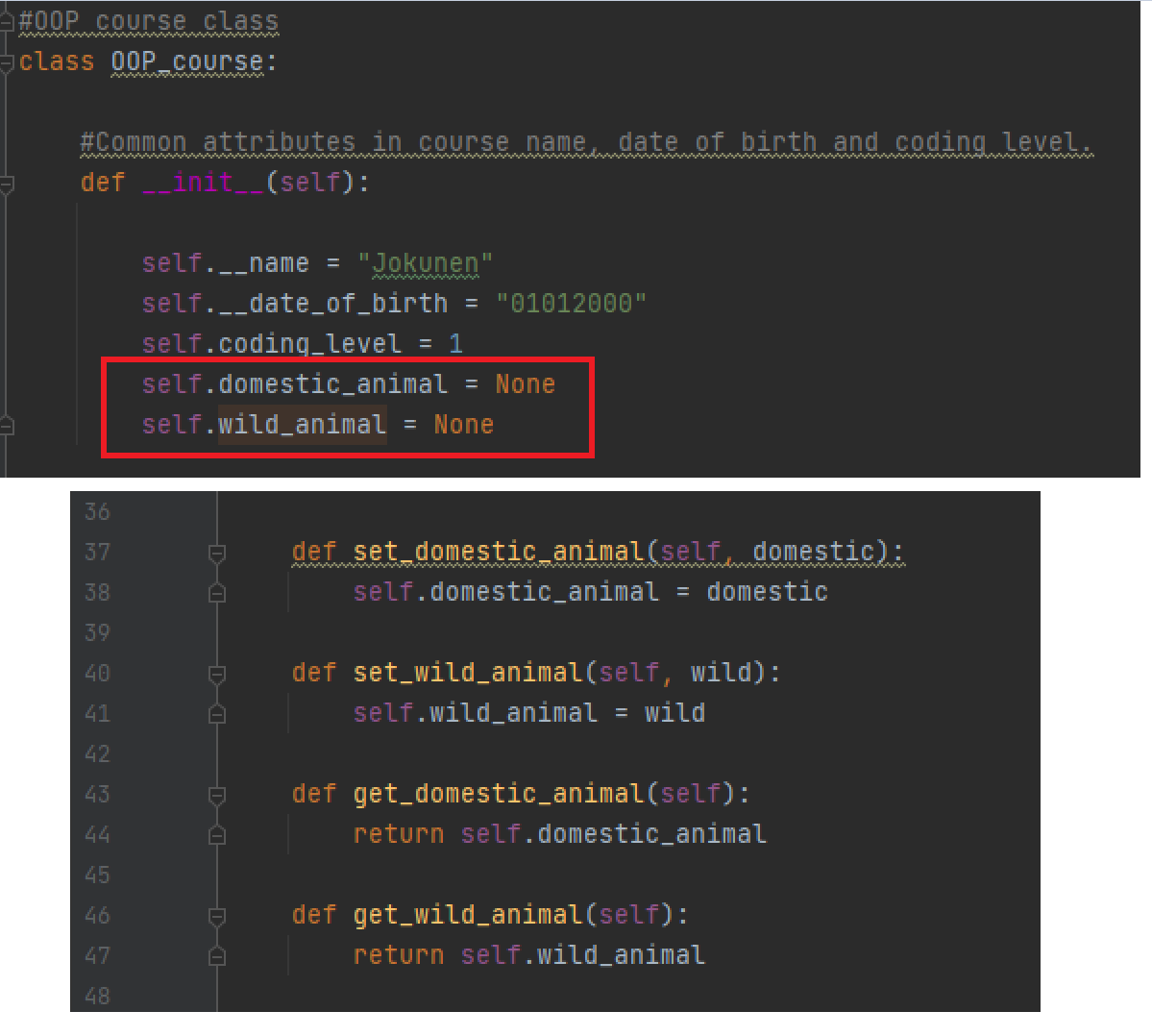
Files that are unchanged : animals, mammal\_class, wild\_animals, teacher, student, wild\_animals, domestic\_animals

Screen capture of Task 8

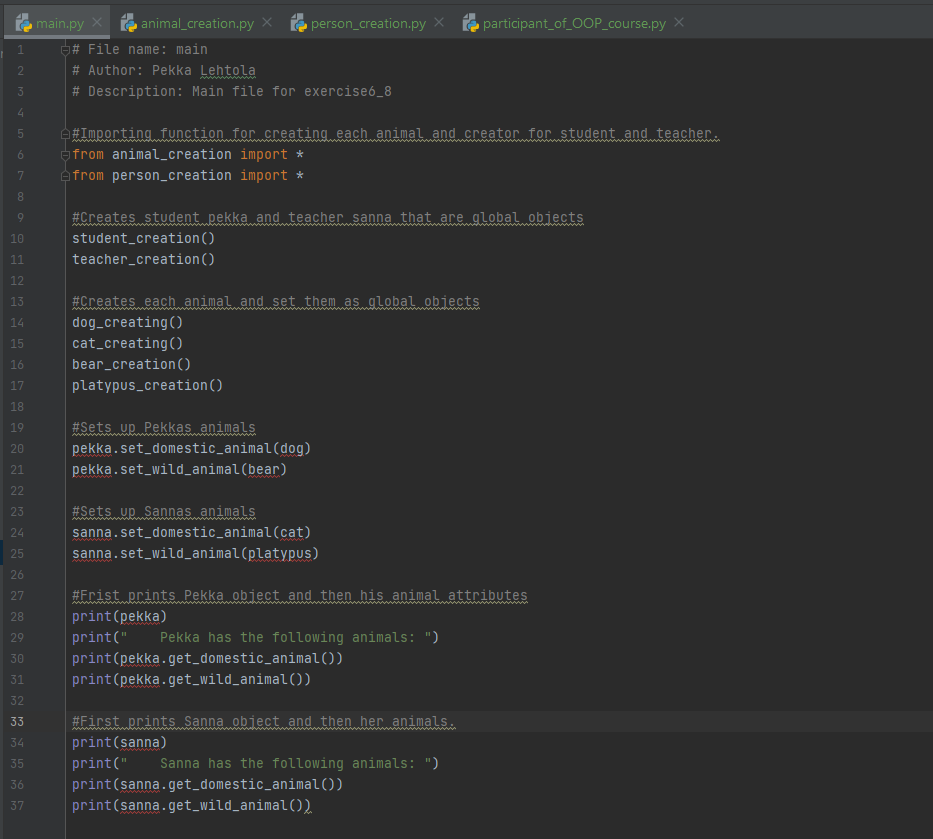
Small edits to animal creation functions **(Main in task 5)** and person creation functions **(Main in task 7):**



Edits to OOP\_course class (From task 7):



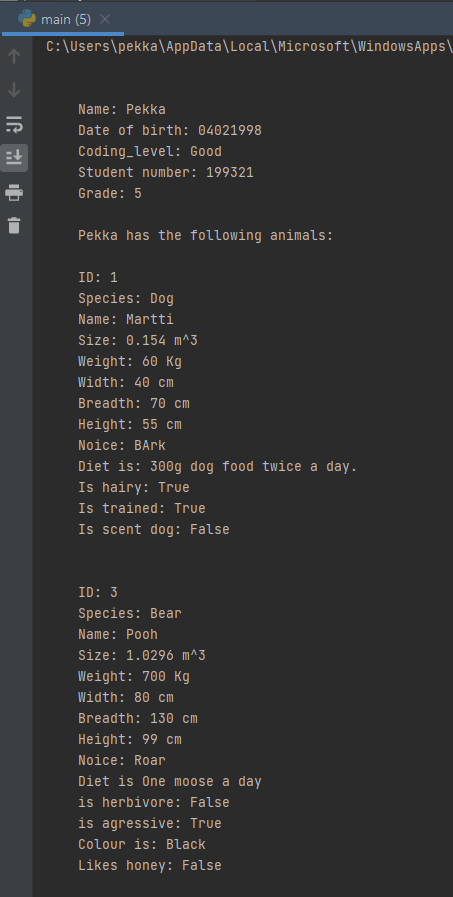
Main:

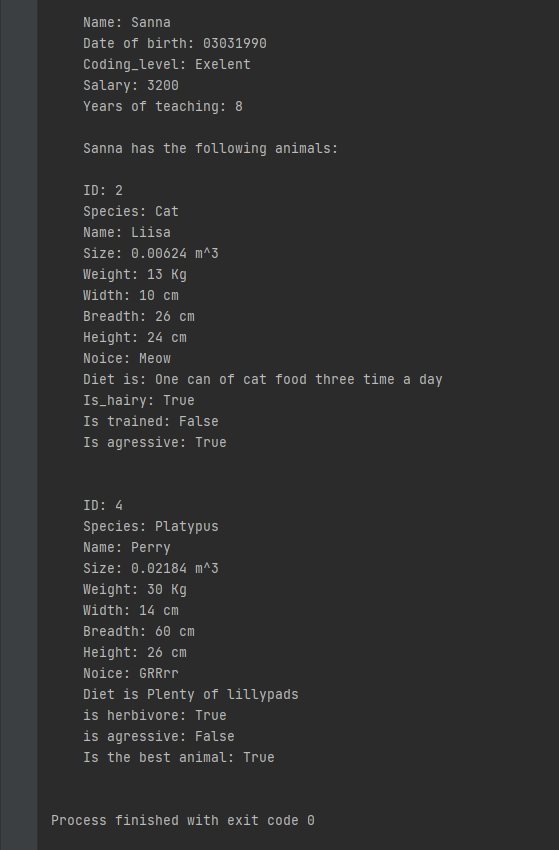


Error highlights are caused from computer not recognizing global objects.

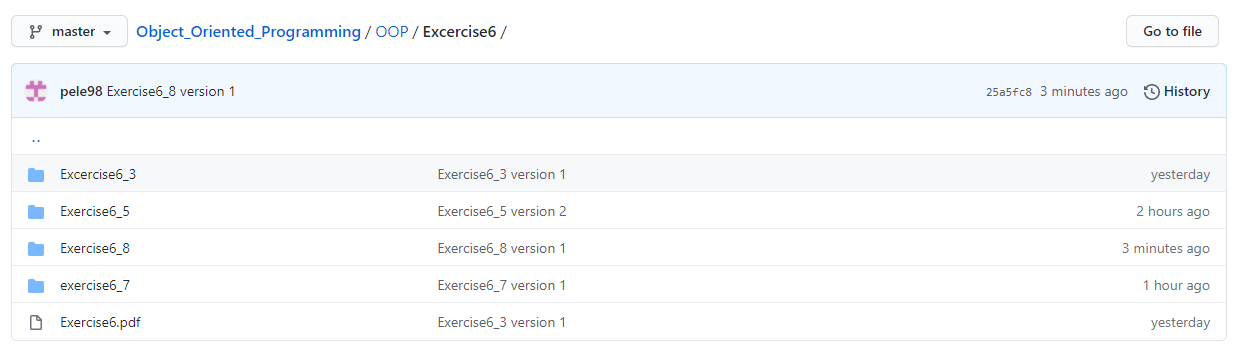
I didn’t add all the code here because document would have been way too long.

Screen capture of the output of Task 8

Output part 1: Output part 2:



Screen capture of git log (showing that you made a commit after every task).



Self-assessment:

**This exercise was easy/difficult/ok/etc. for me because…**

Nämä tehtävä koin melko helpoiksi, haastetta kyllä toi se kun jaoin luokat ja funktiot eri tiedostoihin ja sen kautta tuli muutama ongelma.

**Doing this exercise, I learned…**

Periytyvyyttä ja sen että objectin attribuutti voi olla myös kokonainen objecti.

Oppisin myös käyttämään UML charttia jonka huomasin olevan paljon kätevämpi koodin hahmottelussa, kuin pseudo koodi.

**I am still wondering…**

Voiko \_\_str\_\_ function periä myös ja siihen tehdä muokkauksia perityn luokan sisällä.

**I understood/did not understand that… ; I did/did not know that… ; I did/did not manage to do…**

**-**