Second programming exercise

Task 1)

I start with defining the superclass, which is a more generalized class. The superclass is called “Animal”, the class have a generalized constructor and method. The constructor initializes new instances of “animal\_type”, and the method is taking the “animal\_type” as variable and raises an implementation error to indicate that the method should be inherited and overrided in the subclasses.

A screenshot of a computer program

Description automatically generated

Next I define the subclasses Cat and Dog, which are more specialized classes which inherits variables and methods from the superclass. We can see from the syntax “class Cat(Animal)” and “class Dog(Animal)” that they inherit from the superclass “Animal”.

A screenshot of a computer program

Description automatically generated

As mentioned before the method of the superclass raises an implementation error to indicate that this method should be over-rided in the subclasses. Both subclasses “Cat” and “Dog” inherits the greets method and overrides it, meaning that they provide their own implementation of the method which is specific to the subclass. I was unsure whether to make a new class for the dog size, but it seemd convenient to just make another instance variable “size” in the subclass “Dog” to handle the size issue, which is a valid operation in a subclass.

Polymorphism is the ability of different classes to be treated as instances of the same class trough inheritance. Meaning that both the “Cat” and “Dog” class is treated as instances of the superclass “Animal”. The greets method is the polymorphic “key” in my code, it can be called from any of the subclasses and the correct version of the method (for either “Cat” or “Dog”) will be executed.

A screenshot of a computer program

Description automatically generated

Task 2)

I start with defining the superclass, which is a more generalized class. The superclass is called “Appointment”, the class have a generalized constructor which initialize the object “Appointment” which ensures that every appointment should have a description. It also has two methods which is meant to be over-rided in the subclasses.

A screenshot of a computer program

Description automatically generated

Next I define the subclasses “OneTime”, “Daily” and “Monthly”, they represent the type of frequency for an appointment. All the classes inherit from the superclass. We can see from the syntax “class OneTime(Appointment)”, “class Daily(Appointment)” and “class Monthly(Appointment)” that they inherit from the superclass “Appointment”, also the superclass constructor is initialized in each subclass.

Each of the subclasses also has the methods defined in the superclass, they have their own interpretation in the subclasses and is therefor overridden. I do not have a save method, however the code show polymorphic behavior in that the way that occursOn finds the True of a date and the \_\_repr\_\_ which prints the correct description when occursOn is being applied to a list of appointments. This is because the classes is being treated as instances of the superclass. There is also a test in my code which demonstrates this ability.

A screenshot of a computer program

Description automatically generated