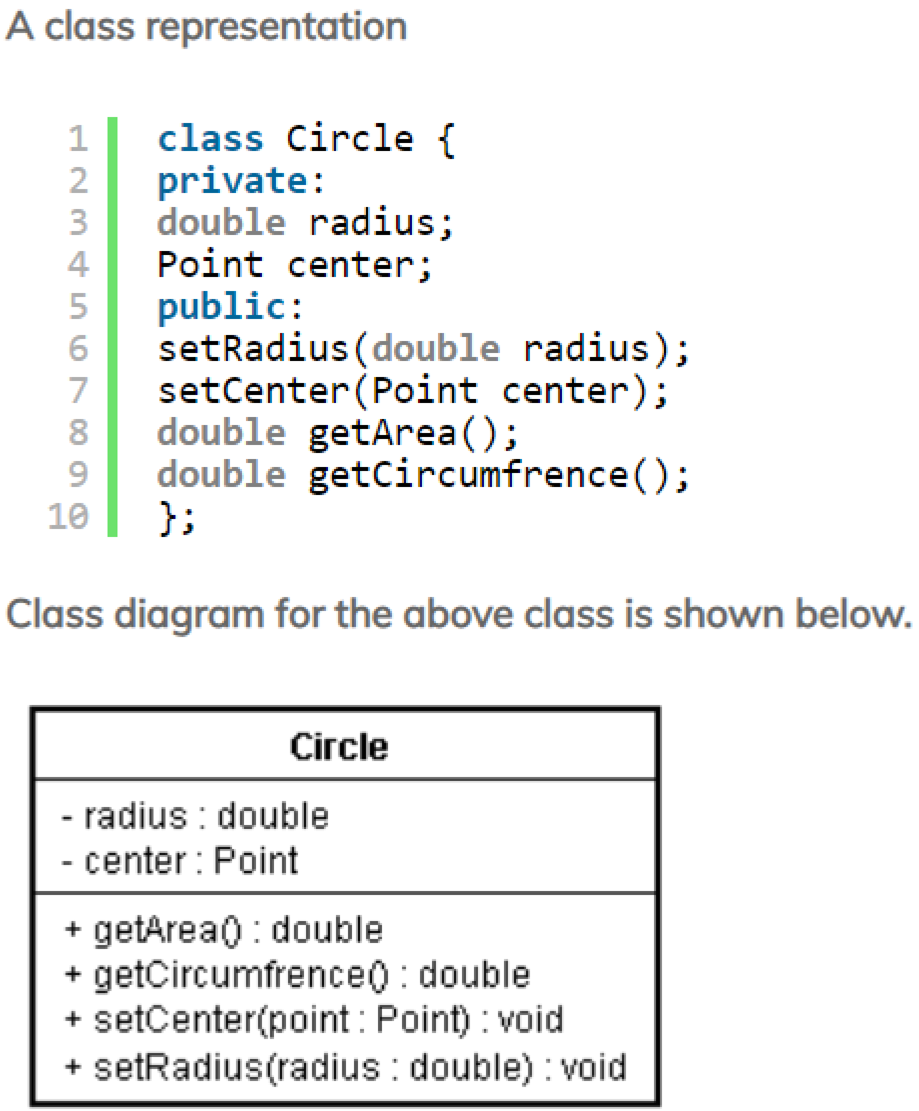
Dijagrami klasa

Zadaci za vjezbu



A class representation

```
class Circle {
private:
double radius;
Point center;
public:
setRadius(double radius);
setCenter(Point center);
double getArea();
double getCircumfrence();
};
```

Class diagram for the above class is shown below.

Circle

- radius : double
- center : Point

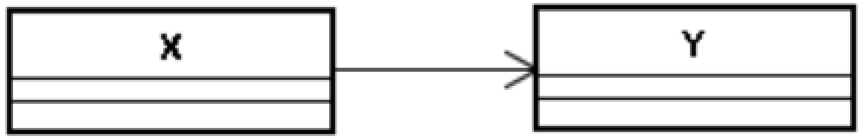
+ getArea() : double
+ getCircumfrence() : double
+ setCenter(point : Point) : void
+ setRadius(radius : double) : void

https://cppcodetips.wordpress.com/2013/12/23/uml-class-diagram-explained-with-c-samples/

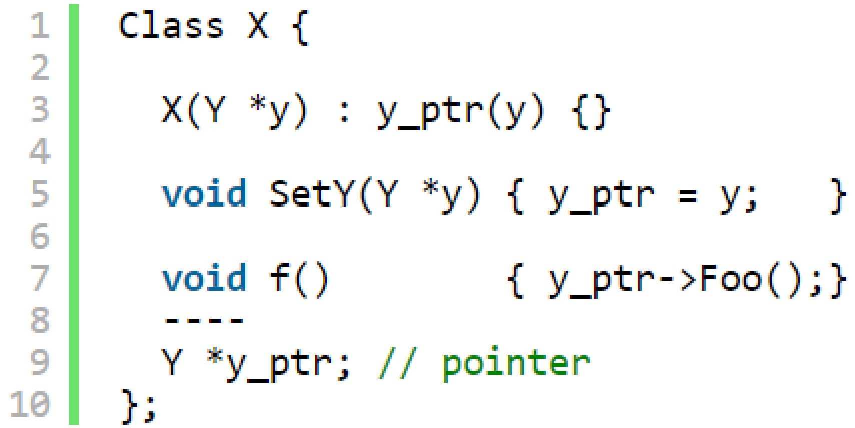
Association

One object is aware of another; it contains a pointer or reference to another object.

Representaion



C++ Example

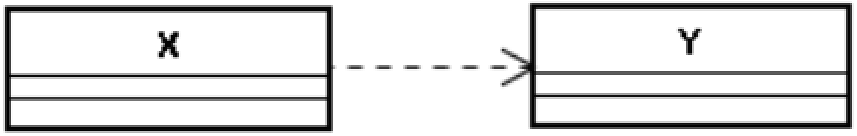


```
Class X {

  X(Y *y) : y_ptr(y) {}

  void SetY(Y *y) { y_ptr = y;   }

  void f()        { y_ptr->Foo();}
  ----
  Y *y_ptr; // pointer
};
```

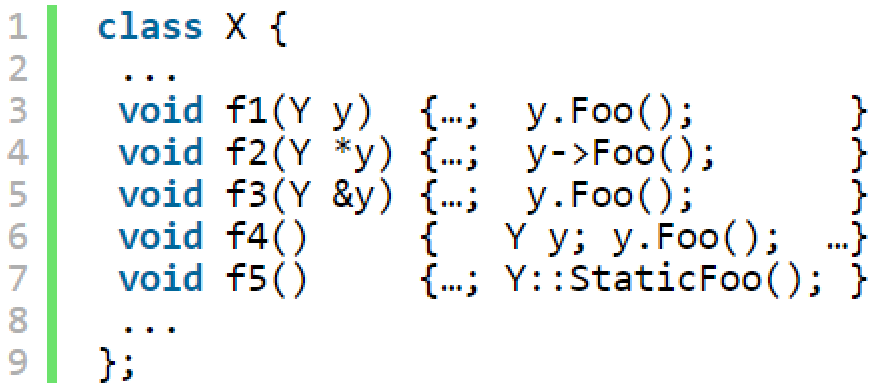
Dependency

One class depends on another if the independent class is a parameter variable or local variable of a method of the dependent class

Representaion



C++ Example

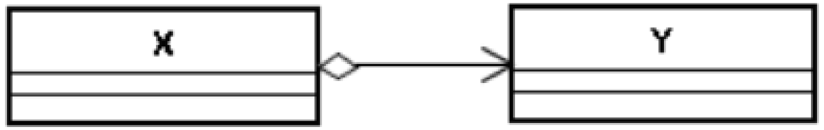


```cpp
class X {
 ...
 void f1(Y y)  {…;  y.Foo();       }
 void f2(Y *y) {…;  y->Foo();      }
 void f3(Y &y) {…;  y.Foo();       }
 void f4()     {   Y y; y.Foo();  …}
 void f5()     {…; Y::StaticFoo(); }
 ...
};
```

Aggregation

Aggregation can occur when a class is a collection or container of other classes, but where the contained classes do not have a strong life cycle dependency on the container—essentially, if the container is destroyed, its contents are not. You may have confusion between aggregation and association .Association differs from aggregation only in that it does not imply any containment.

Representaion

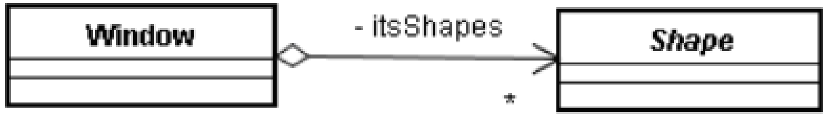


C++ Example

Example 1

```
class Window
{
 public:
  //...
 private:
 vector itsShapes;
};
```

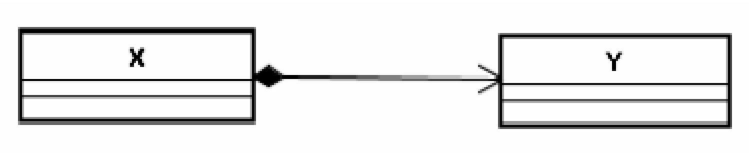
A window class contains a list of its shapes



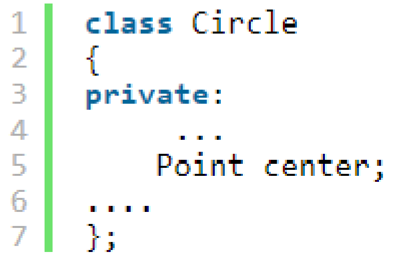
Composition

Composition is the stronger form of aggregation. Composition can occur when a class is a collection or container of other classes, but where the contained classes have a strong life cycle dependency on the container—essentially, if the container is destroyed, its contents are also destroyed

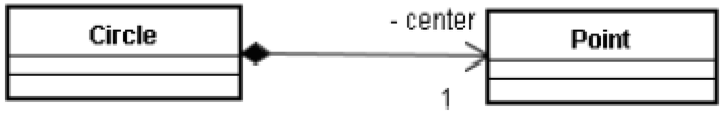
Representation



C++ Example



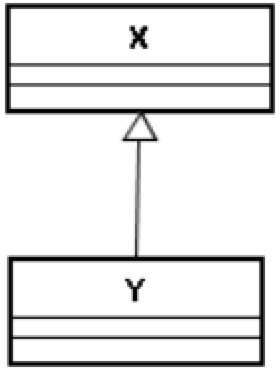
```
class Circle
{
private:
      ...
    Point center;
....
};
```



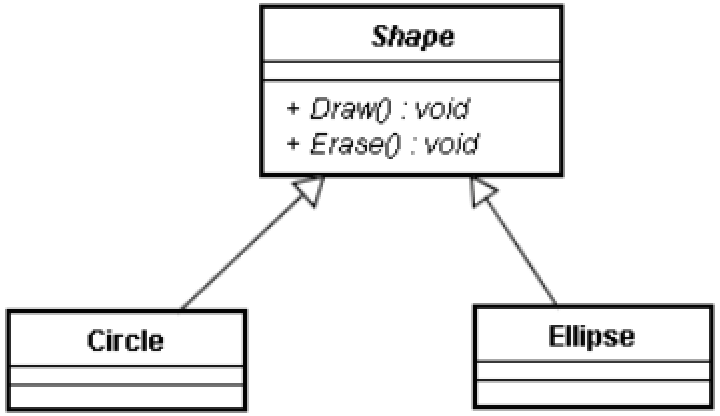
Inheritance (Generalization)

In Inheritance relationship a class is derived from another class. It is a “is a” relationship between two classes.

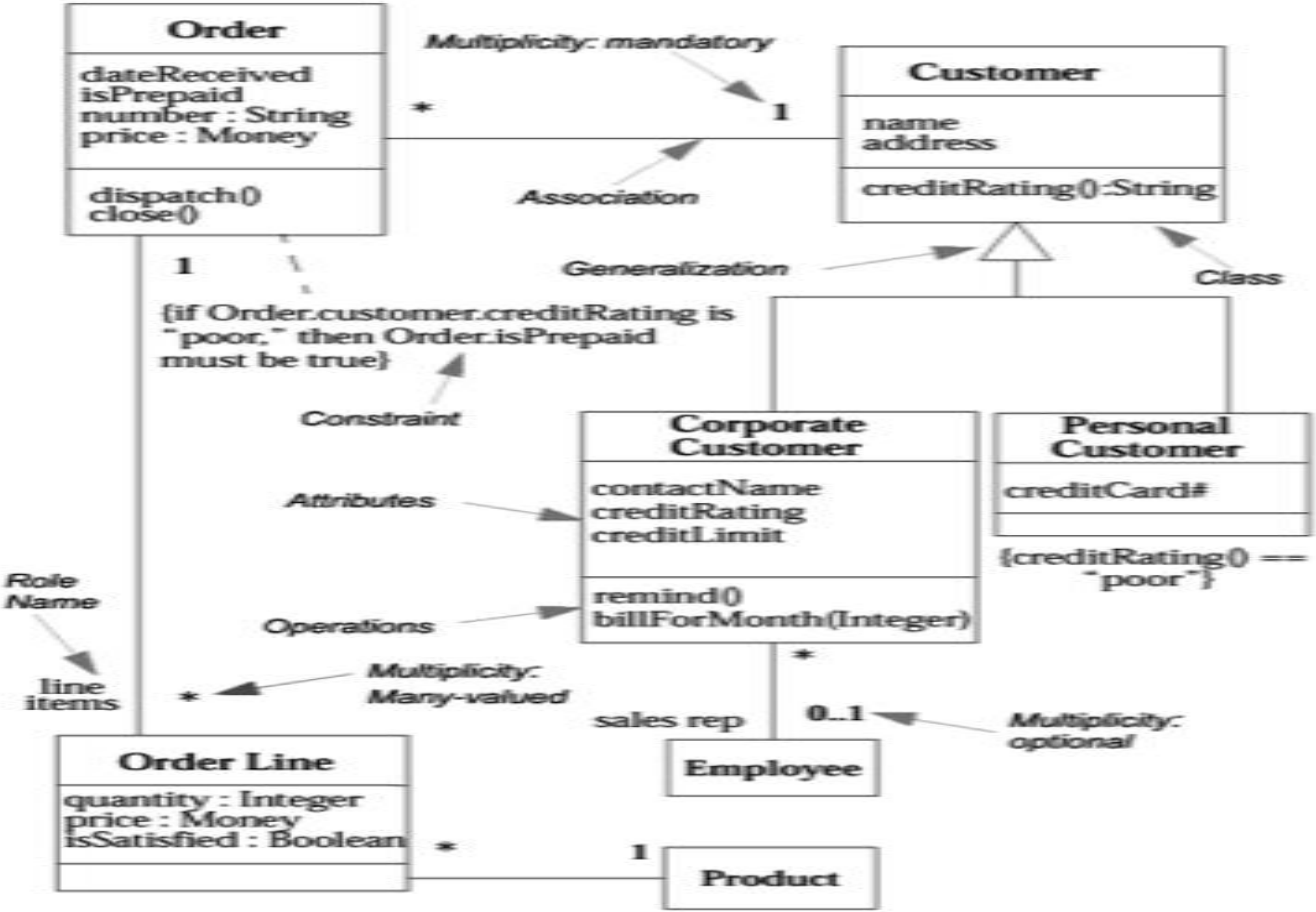
Representation



Here X and Y are normal classes.



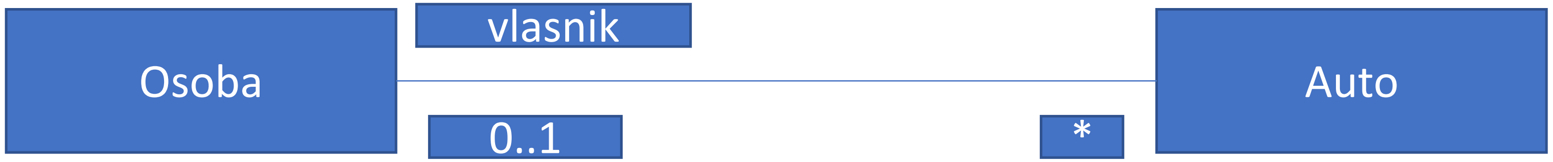
Here Shape is an abstract class that is why it is shown in Italics. Draw () and Erase () methods of Shape class is pure virtual function, so it is also shown as italics.


Order
dateReceived
isPrepaid
number : String
price : Money
dispatch()
close()
Multiplicity: mandatory
*
1
Customer
name
address
creditRating():String
Association
1
Generalization
Class
{if Order.customer.creditRating is "poor," then Order.isPrepaid must be true}
Constraint
Corporate Customer
contactName
creditRating
creditLimit
remind()
billForMonth(Integer)
Personal Customer
creditCard#
{creditRating() == "poor"}
Attributes
Operations
Role Name
line items
*
Multiplicity: Many-valued
sales rep
*
0..1
Multiplicity: optional
Employee
Order Line
quantity : Integer
price : Money
isSatisfied : Boolean
*
1
Product

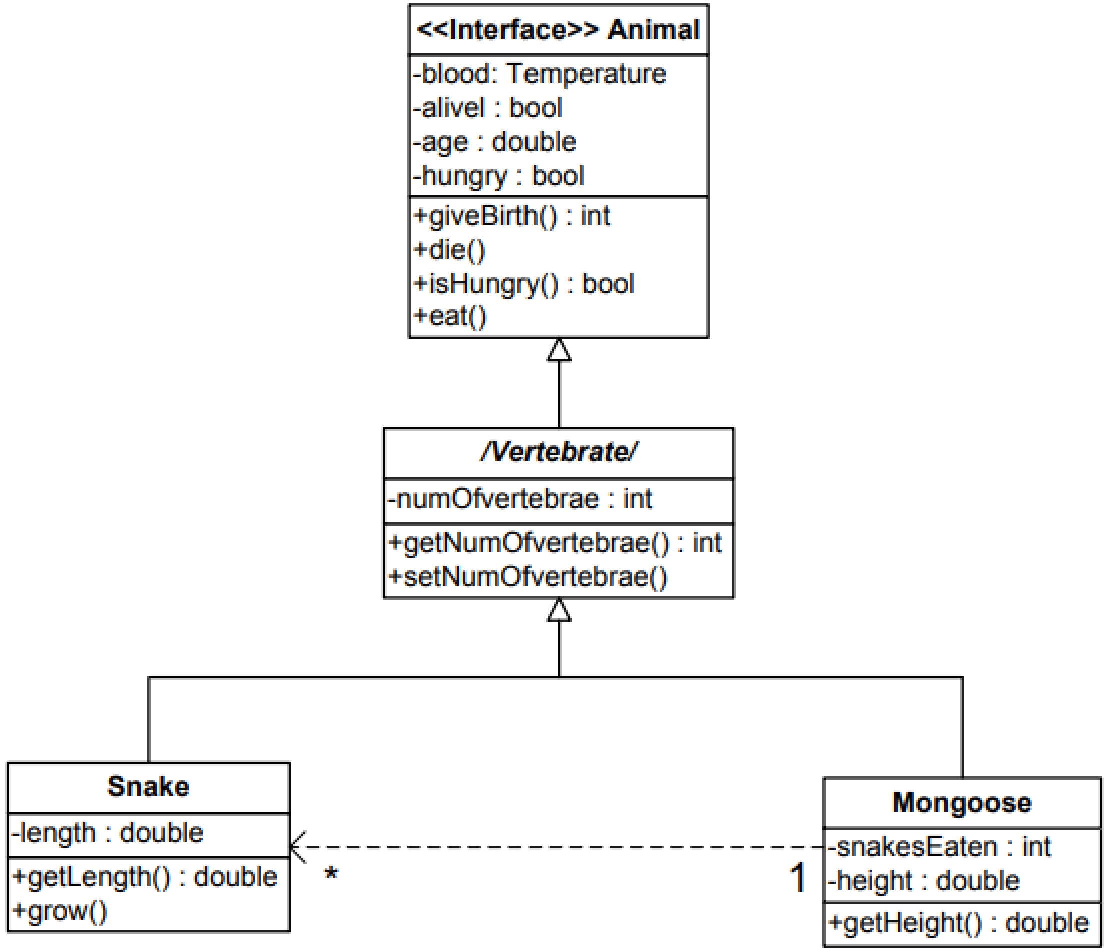
- Prikazati sljedeća svojstva klase Porudzbina kao asocijacije
  - dateReceived
  - isPrepaid
  - lineItems

- Skicirati implementaciju klase Order koja se odnosi na asocijaciju sa klasom OrderLine

- Skicirajte implementaciju sljedećeg dijagrama uz uslov da asocijacijom upravlja klasa Auto



- Skicirati implementaciju


<<Interface>> Animal
-blood: Temperature
-alivel : bool
-age : double
-hungry : bool
+giveBirth() : int
+die()
+isHungry() : bool
+eat()
/Vertebrate/
-numOfvertebrae : int
+getNumOfvertebrae() : int
+setNumOfvertebrae()
Snake
-length : double
+getLength() : double
+grow()
*
1
Mongoose
-snakesEaten : int
-height : double
+getHeight() : double