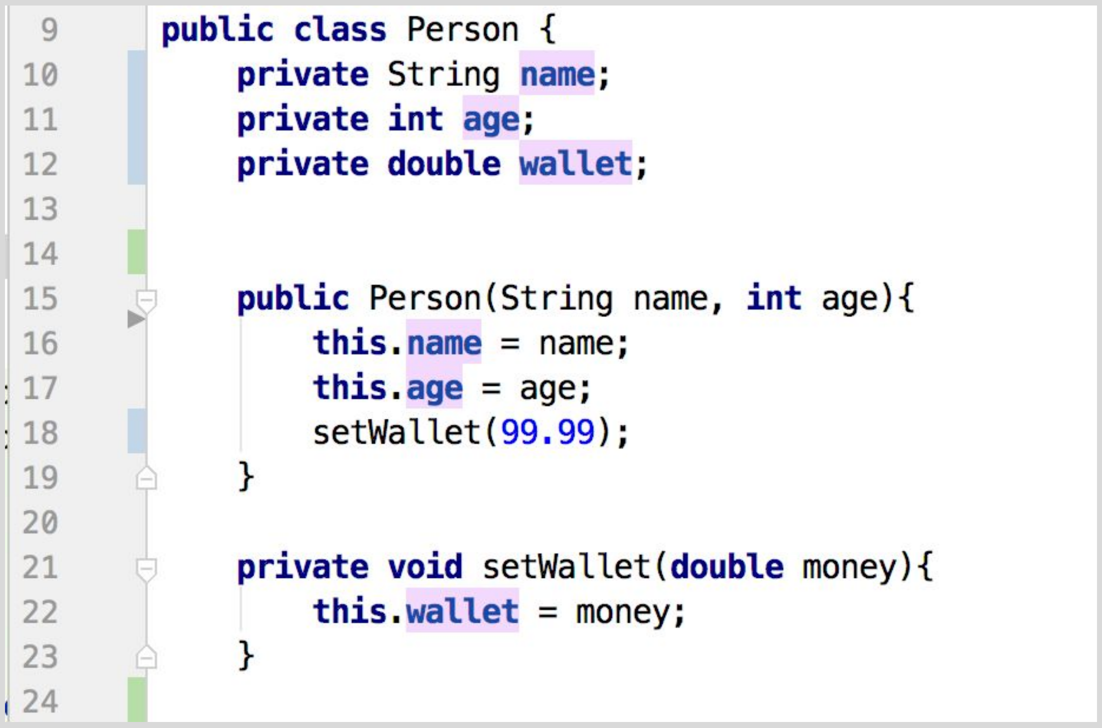
Daniel Garrido-C12

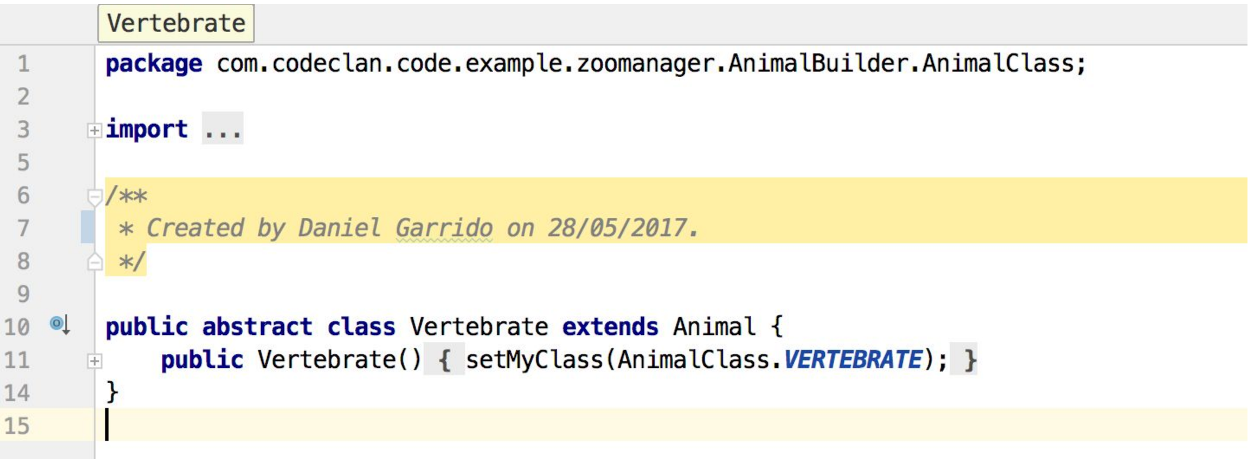
PDA Software Development: **Implementation & Testing unit**
Level 8

I.T. 1 Demonstrate one example of encapsulation that you have written in a program.

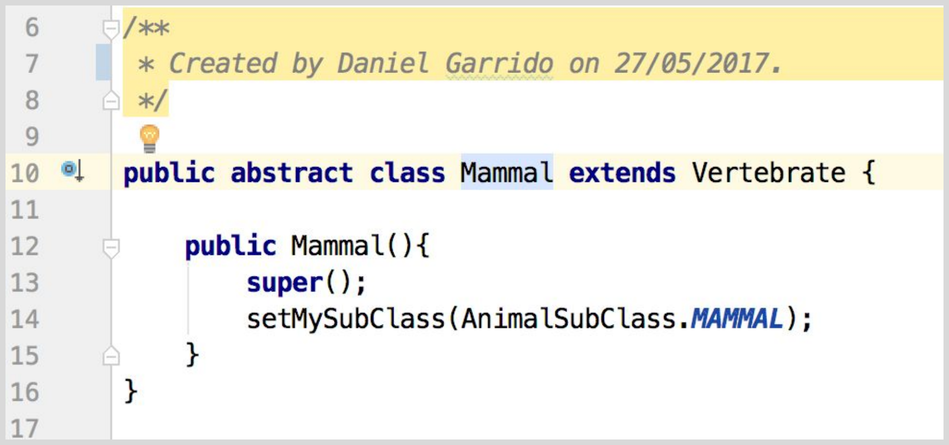


```
public class Person {
    private String name;
    private int age;
    private double wallet;


    public Person(String name, int age){
        this.name = name;
        this.age = age;
        setWallet(99.99);
    }

    private void setWallet(double money){
        this.wallet = money;
    }

```

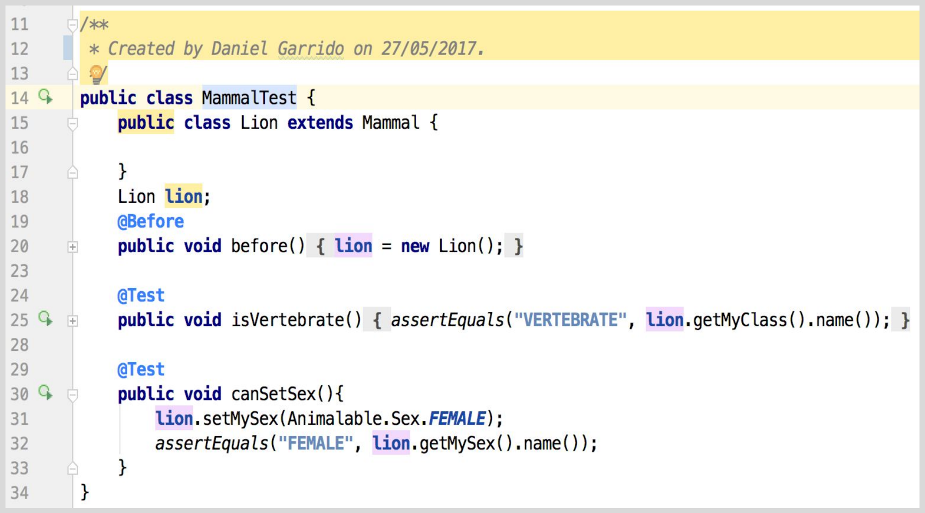
I.T. 2 Demonstrate the user of inheritance in a program you have written by taking a screenshot of a Class, an object and a method, that inherits from another class.



```
package com.codeclan.code.example.zoomanager.AnimalBuilder.AnimalClass;

import ...

/**
 * Created by Daniel Garrido on 28/05/2017.
 */

public abstract class Vertebrate extends Animal {
    public Vertebrate() { setMyClass(AnimalClass.VERTEBRATE); }
}
```



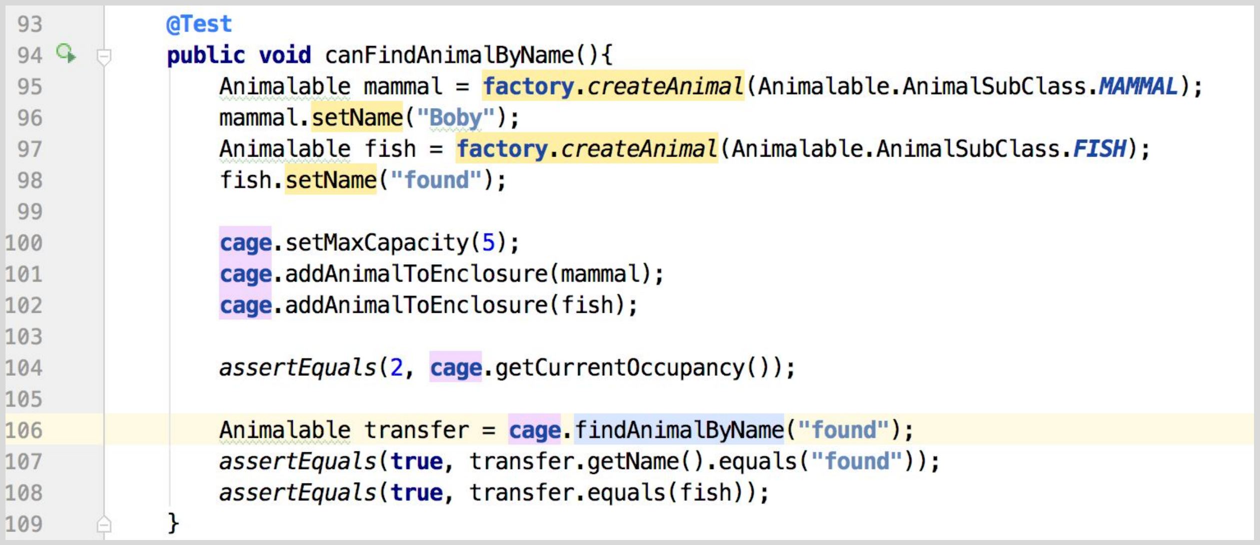
```
/**
 * Created by Daniel Garrido on 27/05/2017.
 */

public abstract class Mammal extends Vertebrate {

    public Mammal(){
        super();
        setMySubClass(AnimalSubClass.MAMMAL);
    }
}
```



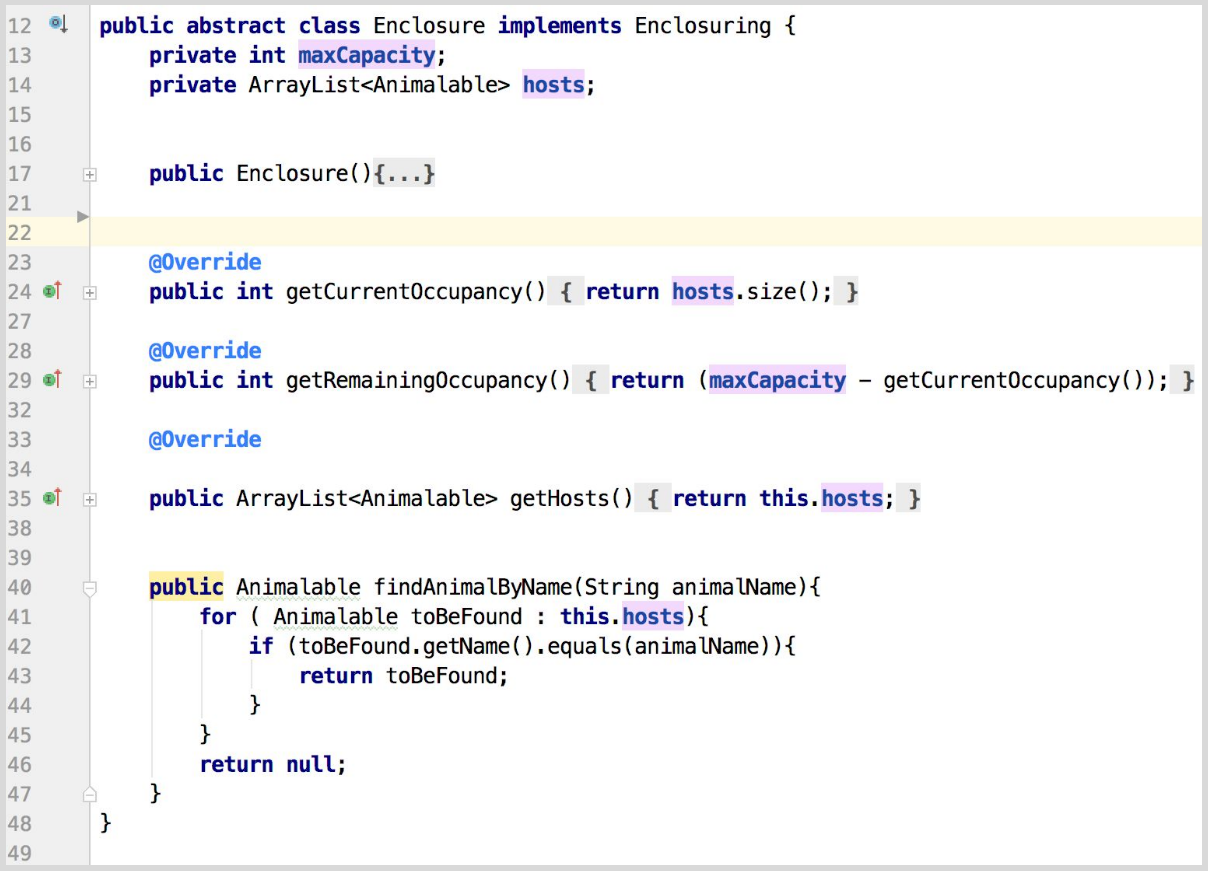
```
/**
 * Created by Daniel Garrido on 27/05/2017.
 */
public class MammalTest {
    public class Lion extends Mammal {

    }
    Lion lion;
    @Before
    public void before() { lion = new Lion(); }

    @Test
    public void isVertebrate() { assertEquals("VERTEBRATE", lion.getMyClass().name()); }

    @Test
    public void canSetSex(){
        lion.setMySex(Animalable.Sex.FEMALE);
        assertEquals("FEMALE", lion.getMySex().name());
    }
}
```

I.T. 3 Demonstrate searching and sorting data in a program you have written.

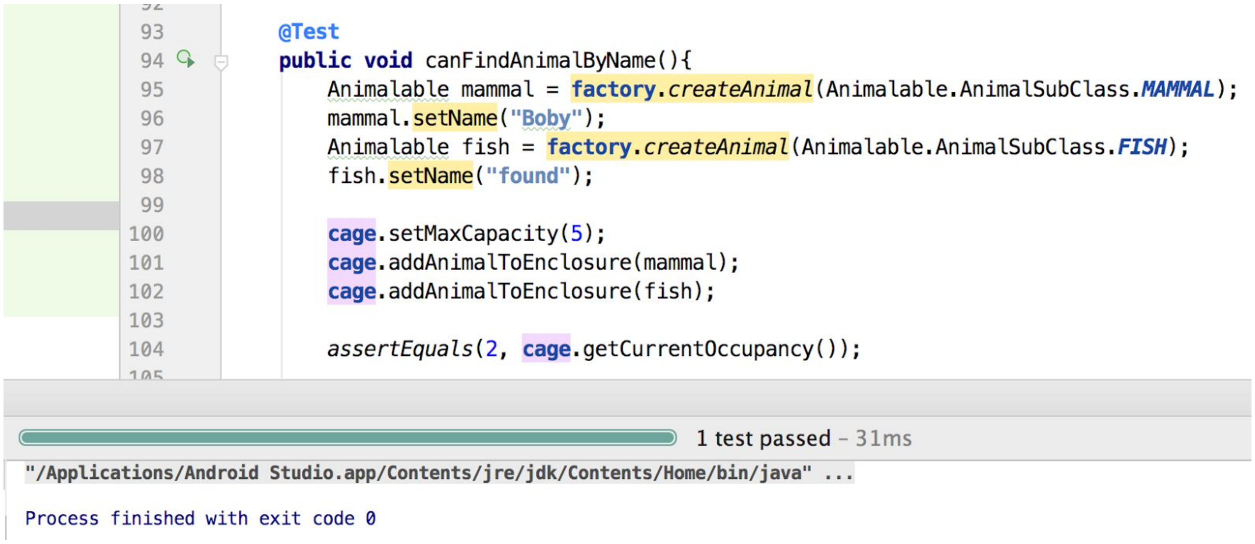
Function that searches data and the result



```java
    @Test
    public void canFindAnimalByName(){
        Animalable mammal = factory.createAnimal(Animalable.AnimalSubClass.MAMMAL);
        mammal.setName("Boby");
        Animalable fish = factory.createAnimal(Animalable.AnimalSubClass.FISH);
        fish.setName("found");

        cage.setMaxCapacity(5);
        cage.addAnimalToEnclosure(mammal);
        cage.addAnimalToEnclosure(fish);

        assertEquals(2, cage.getCurrentOccupancy());

        Animalable transfer = cage.findAnimalByName("found");
        assertEquals(true, transfer.getName().equals("found"));
        assertEquals(true, transfer.equals(fish));
    }
```



```java
public abstract class Enclosure implements Enclosuring {
    private int maxCapacity;
    private ArrayList<Animalable> hosts;


    public Enclosure(){...}


    @Override
    public int getCurrentOccupancy() { return hosts.size(); }

    @Override
    public int getRemainingOccupancy() { return (maxCapacity - getCurrentOccupancy()); }

    @Override

    public ArrayList<Animalable> getHosts() { return this.hosts; }


    public Animalable findAnimalByName(String animalName){
        for ( Animalable toBeFound : this.hosts){
            if (toBeFound.getName().equals(animalName)){
                return toBeFound;
            }
        }
        return null;
    }
}

```

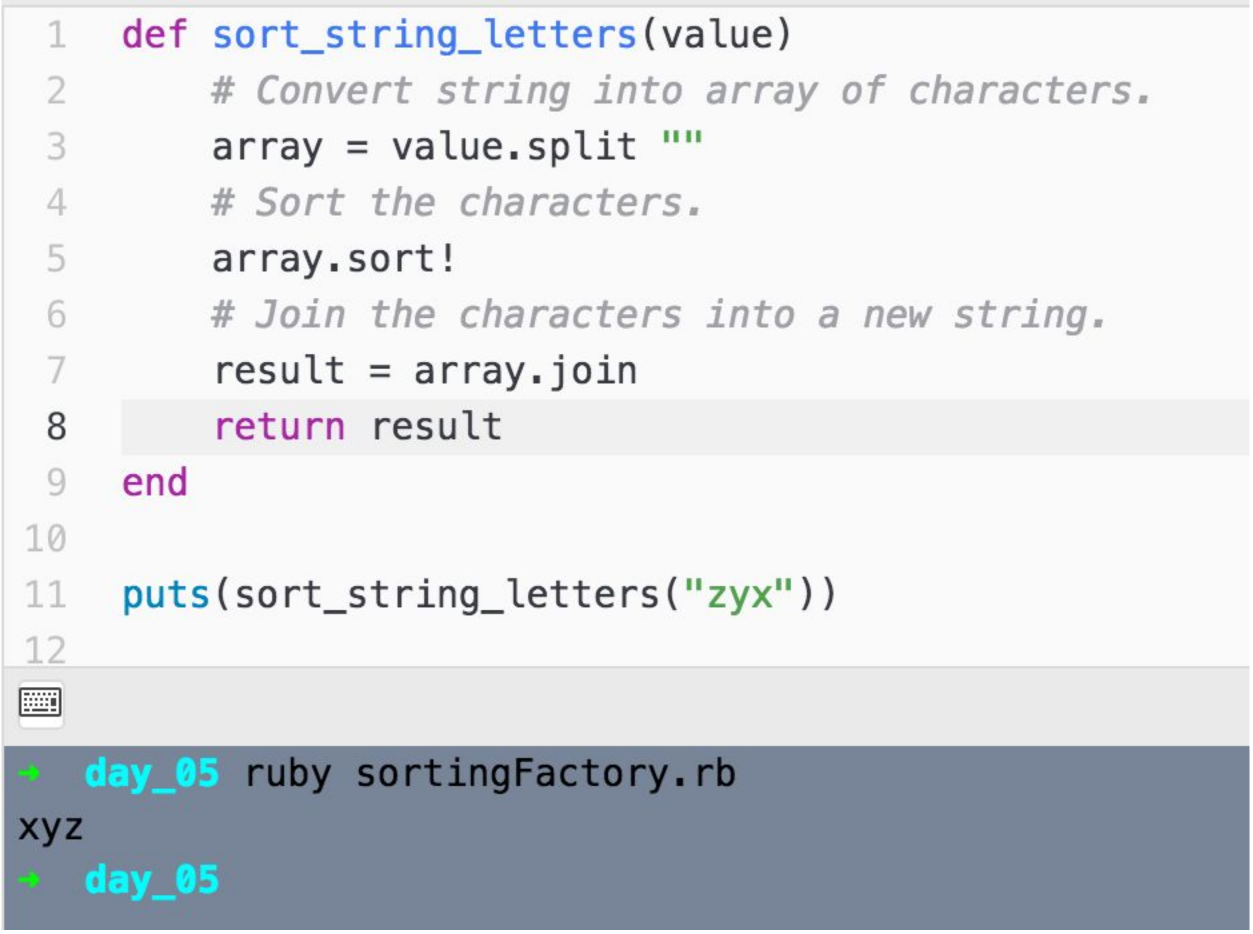


```
    @Test
    public void canFindAnimalByName(){
        Animalable mammal = factory.createAnimal(Animalable.AnimalSubClass.MAMMAL);
        mammal.setName("Boby");
        Animalable fish = factory.createAnimal(Animalable.AnimalSubClass.FISH);
        fish.setName("found");

        cage.setMaxCapacity(5);
        cage.addAnimalToEnclosure(mammal);
        cage.addAnimalToEnclosure(fish);

        assertEquals(2, cage.getCurrentOccupancy());
```

1 test passed – 31ms

```
"/Applications/Android Studio.app/Contents/jre/jdk/Contents/Home/bin/java" ...

Process finished with exit code 0
```

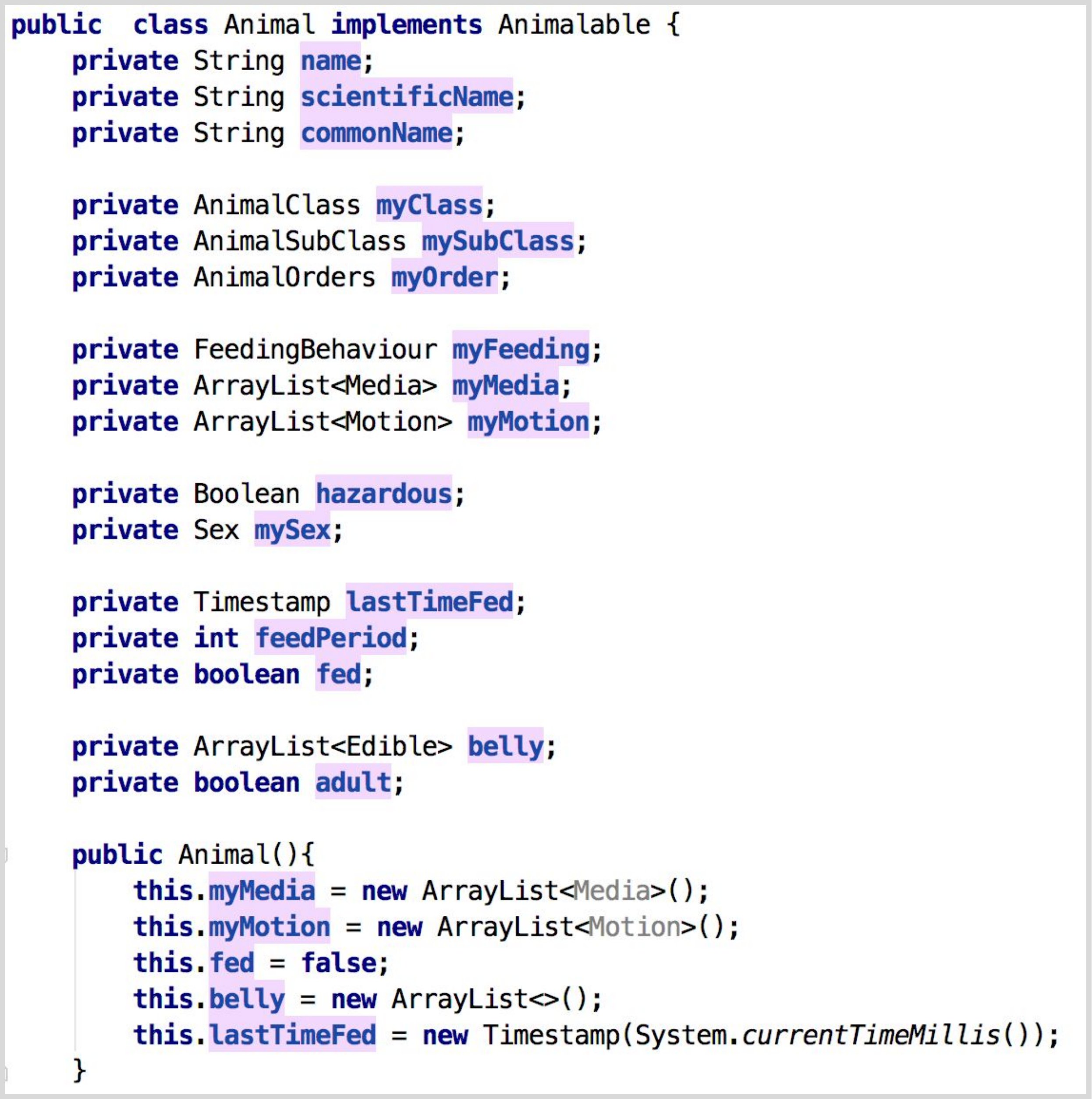
Function that sorts data and the results.



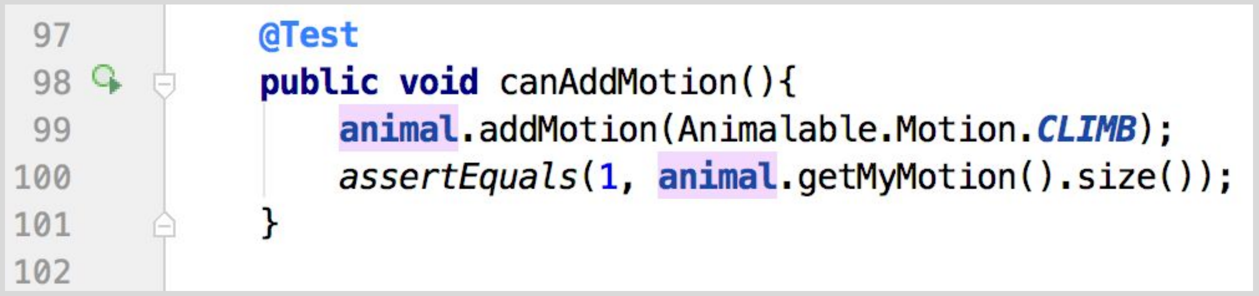
```
def sort_string_letters(value)
    # Convert string into array of characters.
    array = value.split ""
    # Sort the characters.
    array.sort!
    # Join the characters into a new string.
    result = array.join
    return result
end

puts(sort_string_letters("zyx"))
```

```
➜  day_05 ruby sortingFactory.rb
xyz
➜  day_05
```

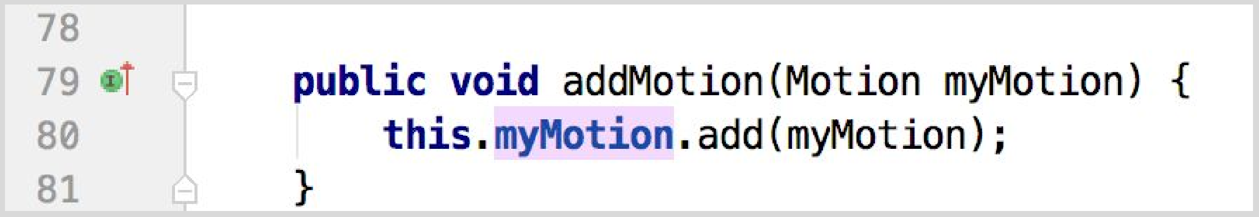
I.T. 4 Take a screenshot of a program where you have created and used; An array, Function that uses it, Result.



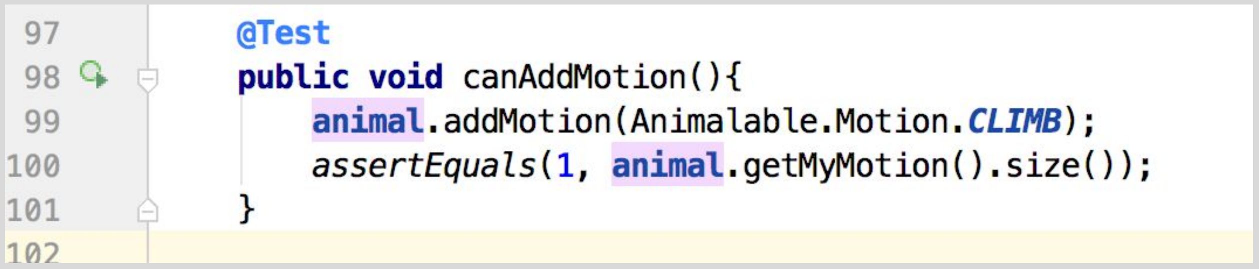
```java
public  class Animal implements Animalable {
    private String name;
    private String scientificName;
    private String commonName;

    private AnimalClass myClass;
    private AnimalSubClass mySubClass;
    private AnimalOrders myOrder;

    private FeedingBehaviour myFeeding;
    private ArrayList<Media> myMedia;
    private ArrayList<Motion> myMotion;

    private Boolean hazardous;
    private Sex mySex;

    private Timestamp lastTimeFed;
    private int feedPeriod;
    private boolean fed;

    private ArrayList<Edible> belly;
    private boolean adult;

    public Animal(){
        this.myMedia = new ArrayList<Media>();
        this.myMotion = new ArrayList<Motion>();
        this.fed = false;
        this.belly = new ArrayList<>();
        this.lastTimeFed = new Timestamp(System.currentTimeMillis());
    }
```



```
    @Test
    public void canAddMotion(){
        animal.addMotion(Animalable.Motion.CLIMB);
        assertEquals(1, animal.getMyMotion().size());
    }

```

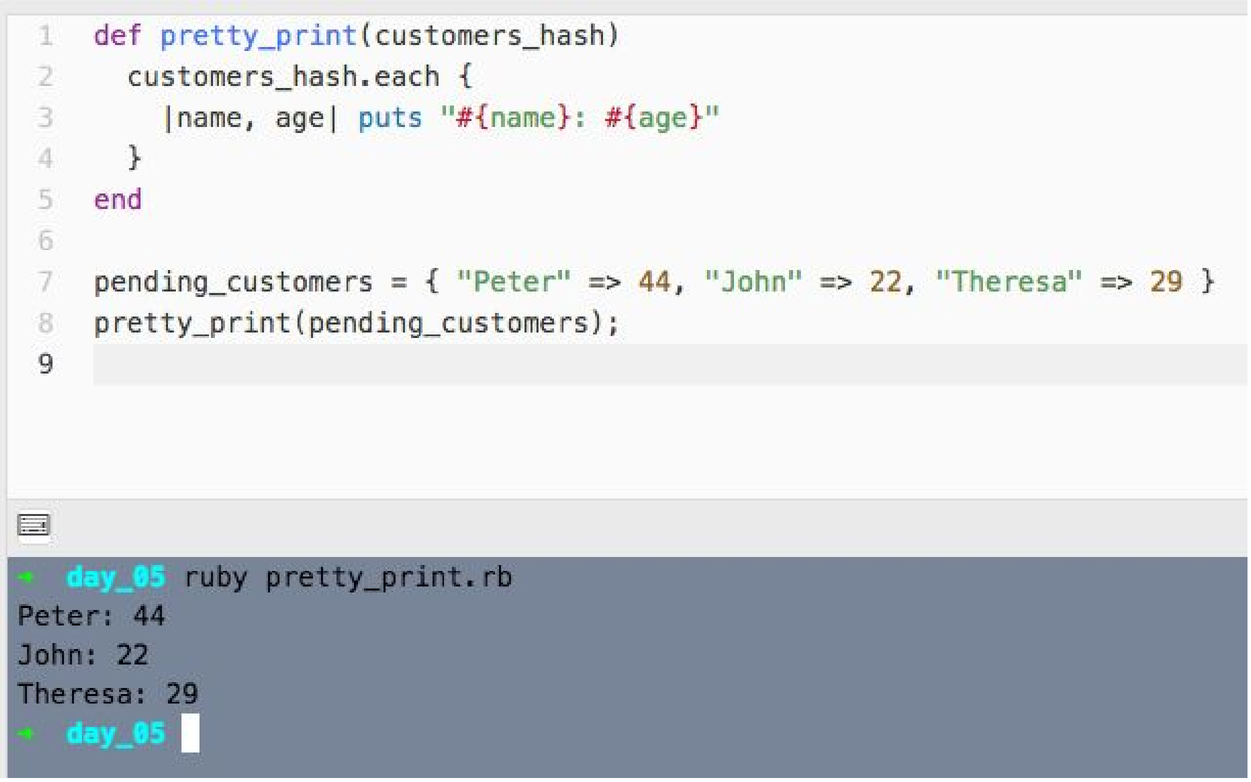


```

    public void addMotion(Motion myMotion) {
        this.myMotion.add(myMotion);
    }
```



```
    @Test
    public void canAddMotion(){
        animal.addMotion(Animalable.Motion.CLIMB);
        assertEquals(1, animal.getMyMotion().size());
    }

```

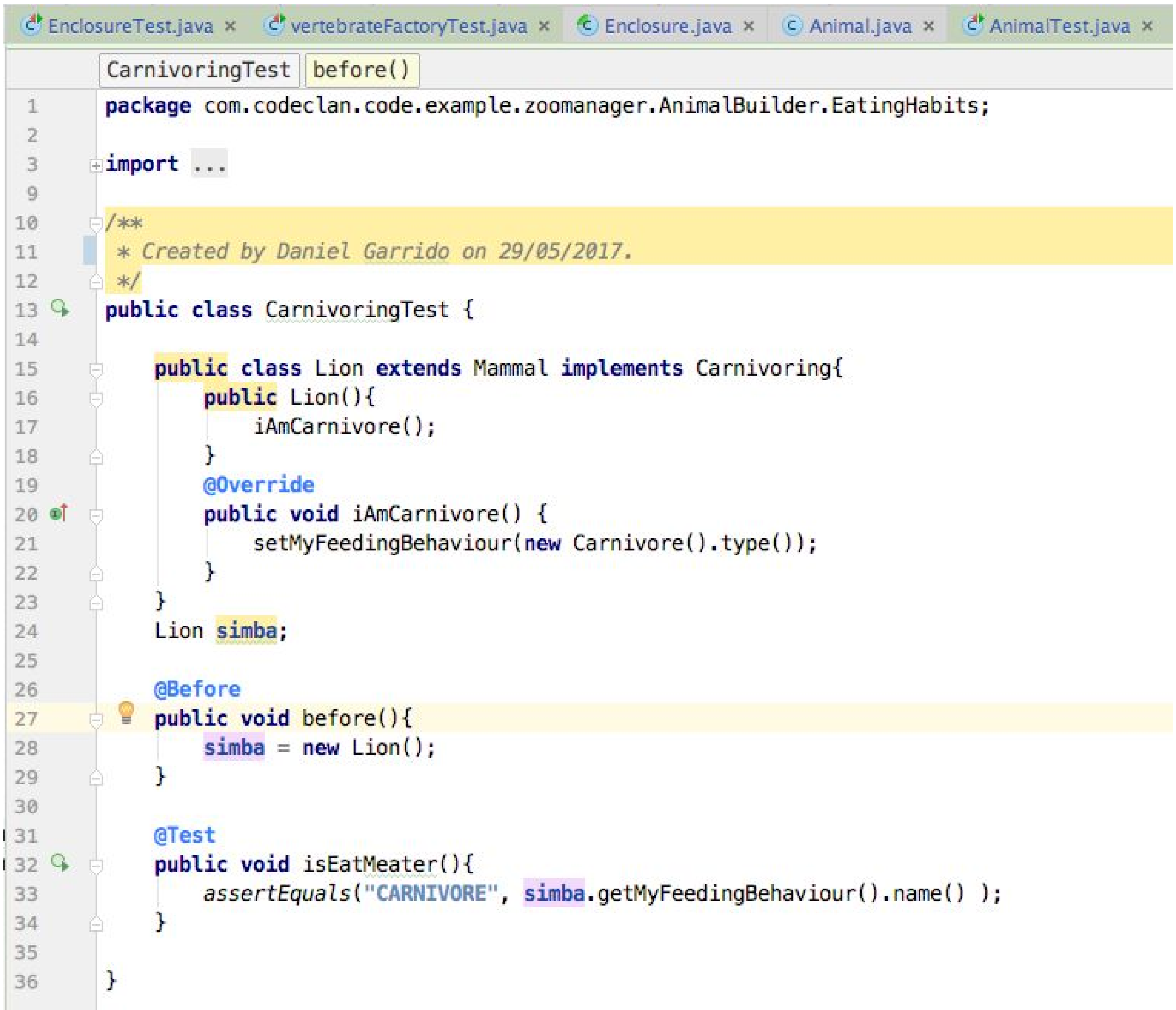
I.T. 4 Take a screenshot of a program where you have created and used; A hash, Function that uses it, Result.



```ruby
def pretty_print(customers_hash)
  customers_hash.each {
    |name, age| puts "#{name}: #{age}"
  }
end

pending_customers = { "Peter" => 44, "John" => 22, "Theresa" => 29 }
pretty_print(pending_customers);

```

```
➜  day_05 ruby pretty_print.rb
Peter: 44
John: 22
Theresa: 29
➜  day_05 
```

I.T. 5 Demonstrate the use of Polymorphism in a program you have written.



```java
package com.codeclan.code.example.zoomanager.AnimalBuilder.EatingHabits;

import ...

/**
 * Created by Daniel Garrido on 29/05/2017.
 */
public class CarnivoringTest {

    public class Lion extends Mammal implements Carnivoring{
        public Lion(){
            iAmCarnivore();
        }
        @Override
        public void iAmCarnivore() {
            setMyFeedingBehaviour(new Carnivore().type());
        }
    }
    Lion simba;

    @Before
    public void before(){
        simba = new Lion();
    }

    @Test
    public void isEatMeater(){
        assertEquals("CARNIVORE", simba.getMyFeedingBehaviour().name() );
    }

}
```