Homework 5 – Java Interfaces

A java interface is a class with all abstract methods. Extending an interface is called Implementing the interface.

An interface cannot have attributes, and none of its methods can have a body.

We cannot Instantiate an interface (meaning, we cannot create an object of this type).

All interface methods are just declarations (signatures) of methods, so it is meant to be implemented by other classes. Interfaces allow the implementing classes to have a uniform API; This is useful for polymorphism, for example when having a collection of multiple types.

Java does not allow a single type (class) to extend multiple classes, however it can implement multiple interfaces – by implementing the methods from all interfaces.

The project "Week_5_Homework" has the package 'pets' with:

A tester,

An interface 'Pet', implemented by the classes 'Dog' and 'Cat'.

Another interface 'Trainable', implemented by the class 'Dog'

Exercises:

1. Read the code and run the method 'petExamples()' in Tester. Read the output.

2. For each of the following lines, what will be printed? If there is a compilation error, explain why.

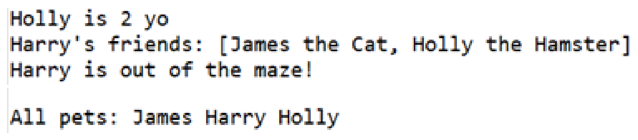
   a. `System.out.println(poppy.getAge());` Prints 3.
   b. `System.out.println(garfield.sayHello());` "Meow" is printed.
   c. `System.out.println(Arrays.toString(poppy.getAllFriends()));` [Garfield the cat, Lily the dog]
   d. `System.out.println(lily.doTrick());` Lily fetches ball

e. `System.out.println(poppy.doTrick());` poppy fetches ball

f.
```
Trainable molly = new Dog("Molly", 5);
System.out.println(molly.doTrick());
System.out.println(molly.getName());
```
doTrick works but not getName, there is no getName in trainable.

3. Implement the class 'Hamster' using the following guidelines:
   a. A hamster is a Pet
   b. A hamster is Trainable
   c. Hamsters can have other pet friends, similarly to dogs and cats
   d. Hamsters say hello by saying "squeak squeak"
   e. toString() should return name + " the hamster"
   f. The hamster's trick is to get out of a maze.

Run the method hamsterTest() in Tester, you should get the following results:



```
Holly is 2 yo
Harry's friends: [James the Cat, Holly the Hamster]
Harry is out of the maze!

All pets: James Harry Holly
```

Comparable interface

”$Week\_5\_Homework$” that contains the package ”sorting” and the package ”comparables”.

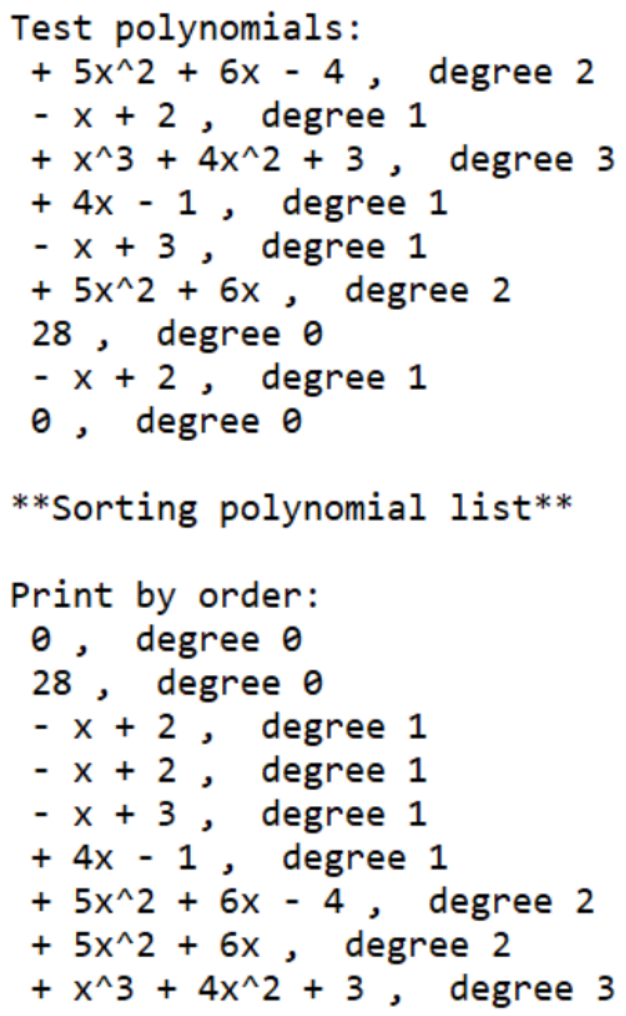
1. The class ’Polynomial’ represent a polynomial with integer coefficients.
   Make the class ’Polynomial’ comparable: a Polynomial P_1 is defined to be greater than polynomial P_2 if

   - P_1 is has a higher degree than P_2.
   - If they have the same degree, compare the coefficients:
     Say $P_1 = a_n \cdot x^n + a_{n-1} \cdot x^{n-1} + ... + a_0$ and $P_2 = b_n \cdot x^n + b_{n-1} \cdot x^{n-1} + ... + b_0$:
     If $a_n > b_n$ (as integers) Then $P_1 > P_2$. If $a_n = b_n$ compare the a_{n-1} and b_{n-1} etc. Two polynomials are the same if they have the same coefficients.

   Example: Let

   - $C_1 = 6x + 7$
   - $C_2 = 6x + 5$
   - $C_3 = -4x^2 - 20x$
   - $C_4 = 2x + 10$
   - $C_5 = -4x^2 - 20x - 5$

   So $C_3 > C_5 > C_1 > C_2 > C_4$
   You can find test cases in the tester class, in the method polynomialTest:

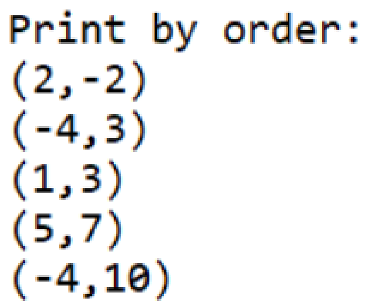


```
Test polynomials:
 + 5x^2 + 6x - 4 ,  degree 2
 - x + 2 ,  degree 1
 + x^3 + 4x^2 + 3 ,  degree 3
 + 4x - 1 ,  degree 1
 - x + 3 ,  degree 1
 + 5x^2 + 6x ,  degree 2
 28 ,  degree 0
 - x + 2 ,  degree 1
 0 ,  degree 0

**Sorting polynomial list**

Print by order:
 0 ,  degree 0
 28 ,  degree 0
 - x + 2 ,  degree 1
 - x + 2 ,  degree 1
 - x + 3 ,  degree 1
 + 4x - 1 ,  degree 1
 + 5x^2 + 6x - 4 ,  degree 2
 + 5x^2 + 6x ,  degree 2
 + x^3 + 4x^2 + 3 ,  degree 3
```

2. The class 'Point' represent a point in 2D space with integer coefficients.
   Make the class 'Point' comparable: a Point P_1 is defined to be larger than point P_2 if P_1 is higher than P_2 on the y axis.
   If they have the same vertical height, the larger point is the point further to the right. Two points are equal if they have the same coordinates.
   Example: Let $C_1 = (6,7)$, $C_2 = (-2,0)$, $C_3 = (-6,5)$, $C_4 = (0,0)$, $C_5 = (20,-10)$.
   So $C_1 > C_3 > C_4 > C_2 > C_5$
   You can find test cases in the tester class, in the method pointTest:



```
Print by order:
(2,-2)
(-4,3)
(1,3)
(5,7)
(-4,10)
```

Your solutions: Upload the class Hamster.java and the package ’comparables’, zipped, to canvas. Make sure to include your implementations to all 3 classes.
Please do not add any other java files.
Please make sure all your code compiles.