Laboratório 13 - Herança e Polimorfismo I (21/10/2014)

1.

```
package lab13;

public class AppAnimal {

	public static void main(String[] args) {
		Animal cat = new Animal("Cat", "Fusca", true);   //criação de um objeto do tipo Animal, associado à variável cat, cujo type é Cat, nome é Fusca e é um Pet

		Animal bear = new Animal("Bear", "Teddy", false);  //criação de um objeto do tipo Animal, associado à variável bear, cujo Type é Bear, nome é Teddy e não é um Pet

		WildlifeAnimal lion = new WildlifeAnimal("Lion", "Simba", 4); //criação de um objeto do tipo WildlifeAnimal, associado à variável lion, cujo Type é Lion, nome é Simba e dangerLevel 4

		Tiger tiger = new Tiger("Raja");    //criação de um objeto do tipo Tiger, associado à variável tiger, cujo nome é Raja

		Chinchilla chin = new Chinchilla("Spooky");    //criação de um objeto do tipo Chinchilla, associado à variável chin, cujo nome é Spooky

		System.out.println(cat.print());   //cat é um objeto da classe Animal, logo é o método print da classe Animal que será utilizado

		System.out.println(bear.print());   //bear é um objeto da classe Animal, logo é o método print da classe Animal que será utilizado

		System.out.println(lion.print());    //lion é um objeto da classe WildlifeAnimal, logo é o método print da classe derivada WildlifeAnimal que será utilizado

		System.out.println(tiger.print());   //tiger é um objeto da classe Tiger que é derivada da classe WildlifeAnimal, logo é o método print da classe derivada WildlifeAnimal que será utilizado uma vez que fez Override do método da super classe Animal

		System.out.println(chin.print());    //chin é um objeto da classe Chinchilla que é derivada da classe HouseAnimal, logo é o método print da classe derivada HouseAnimal que será utilizado  uma vez que fez Override do método da super classe Animal
	}
}
```

Output:

```
CAT: {Name: Fusca; Pet: Yes}     //método print da classe Animal
BEAR: {Name: Teddy; Pet: No}     //método print da classe Animal
LION: {WILD ANIMAL --> Name: Simba; Danger Level: 4}   //método print da classe WildlifeAnimal
TIGER: {WILD ANIMAL --> Name: Raja; Danger Level: 4}   //método print da classe WildlifeAnimal
CHINCHILLA: {HOUSE ANIMAL --> Name: Spooky}   //método print da classe HouseAnimal
```

b)

Para que não seja possível criar instâncias de Animal, basta transformá-la numa classe Abstrata:

Alterações no programa:

```
public abstract class Animal {
(…)
}
```

```
public class AppAnimal {
	public static void main(String[] args) {
	   Animal cat = new HouseAnimal("Cat", "Fusca", true);
	   Animal bear = new WildlifeAnimal("Bear", "Teddy", false);
	(…)
```

```
// Ao alterarmos a classe Animal para abstrata, começa a dar erro nas linhas do Main onde são criados e afetados objetos dessa classe e que temos de alterar. Para não alterar os parâmetros fornecidos na Main, temos de associar o parâmetro isPet a uma das classes derivadas. Apesar do urso não ser na realidade um HouseAnimal, mas sim um WildlifeAnimal, uma vez que não temos o DangerLevel do urso, temos de associar a variável "bear" a um objeto do tipo HouseAnimal e alterar apenas esta classe derivada. Outro método (o que eu escolhi)criamos outro construtor na Classe HouseAnimal e WildlifeAnimal que aceite objetos com estes parâmetros.
```

```
public class HouseAnimal extends Animal{
	public HouseAnimal(String type, String name, boolean pet) {
		super(type, name, true);
		}

	public HouseAnimal(String type, String name) {
		super(type, name, true);
		}
(…)
}
```

```
public class WildlifeAnimal extends Animal{
	private int dangerLevel;

	public WildlifeAnimal(String type, String name, boolean pet){
		super(type, name, false);
		this.isAPet = pet;
	}
(…)
}
```

c) Alterações ao programa:

```
public class AppAnimal {

	public static void main(String[] args) {
(…)
	   EndangeredAnimal panda = new EndangeredAnimal("Panda", "Su Lin", 268);
	   System.out.println(panda.print());
(…)
}
```

```
package lab13;
public class EndangeredAnimal extends Animal{
	int numAnimals;
	public EndangeredAnimal(String type, String name, int numAnimal) {
		super(type, name, false);
		this.numAnimals = numAnimal;
		}

	@Override
	public String print() {
	return this.type.toUpperCase() + ": {ENDANGERED ANIMAL --> Name: " + this.name +
"; Total of animals: " + this.numAnimals + "}";
	}
}
```

d)

```
package lab13;

public class SnowLeopard extends EndangeredAnimal{
     int numAnimal;

       public SnowLeopard(String type, String name, int numAnimal) {
              super(type, name, numAnimal);
              }

}
```

Output final:

```
CAT: {HOUSE ANIMAL --> Name: Fusca}
BEAR: {WILD ANIMAL --> Name: Teddy; Danger Level: 0}
LION: {WILD ANIMAL --> Name: Simba; Danger Level: 4}
TIGER: {WILD ANIMAL --> Name: Raja; Danger Level: 4}
CHINCHILLA: {HOUSE ANIMAL --> Name: Spooky}
PANDA: {ENDANGERED ANIMAL --> Name: Su Lin; Total of animals: 268}
SNOW LEOPARD: {ENDANGERED ANIMAL --> Name: snowball; Total of animals: 6000}
```

```
public static void main(String[] args) {
       //     Animal cat = new Animal("Cat", "Fusca", true);
       //     Animal bear = new Animal("Bear", "Teddy", false);
              Animal cat = new HouseAnimal("Cat", "Fusca", true);
              Animal bear = new WildlifeAnimal("Bear", "Teddy", false);
              WildlifeAnimal lion = new WildlifeAnimal("Lion", "Simba", 4);
       Tiger tiger = new Tiger("Raja");
              Chinchilla chin = new Chinchilla("Spooky");

              System.out.println(cat.print());
              System.out.println(bear.print());
              System.out.println(lion.print());
              System.out.println(tiger.print());
       System.out.println(chin.print());

              EndangeredAnimal panda = new EndangeredAnimal("Panda", "Su Lin", 268);
              System.out.println(panda.print());
              SnowLeopard snowLeopard = new SnowLeopard("Snow Leopard", "snowball", 6000);
              System.out.println(snowLeopard.print());
```