```java
//AnimalAbstract.java

public abstract class AnimalAbstract {
	private String animal_name;

	public abstract void createSound();
//Abstract methods have no body and must be implemented in child classes.
	public String get_animal_name() {
	return animal_name;
	}

	public void setName(String aName)
	{
	animal_name = aName;
	}
}
```

```java
//Camel.java
public class Camel extends AnimalAbstract {
	Camel()
	{
		System.out.println("This is the constructor of the Camel class");
	}
	public void createSound()
	{
		System.out.println("This sound is coming from the Camel class");

	}


}
```

```java
//Mouse.java
public class Mouse extends AnimalAbstract {
	Mouse()
	{
		System.out.println("This is the constructor of the Mouse class");

	}

	public void createSound()
	{
		System.out.println("This sound is coming from the Mouse class");

	}
}
```

```java
//Tiger.java
public class Tiger extends AnimalAbstract{
	Tiger()
	{
		System.out.println("This is the constructor of the Tiger class");

	}
	public void createSound()
	{
		System.out.println("This sound is coming from the Tiger class");
	}


}
```

```java
//TestAnimalAbstract.java
public class TestAnimalAbstract {

	public static void main(String[] args) {


		// AnimalAbstract c1 = new AnimalAbstract();  // You cannot do it
		Camel c1 = new Camel();
		c1.createSound();

		AnimalAbstract c2 = new Camel();
		c2.createSound();

		Mouse m1 = new Mouse();
		m1.setName("Micky");
		System.out.println("The name of this mouse is: " +
m1.get_animal_name());

		AnimalAbstract[] A = new AnimalAbstract[10];
		A[0] = new Camel();
		A[1] = new Mouse();
		A[2] = new Camel();
		A[3] = new Camel();
		A[4] = new Camel();
		A[5] = new Tiger();
		A[6] = new Mouse();
		A[7] = new Camel();
		A[8] = new Tiger();
		A[9] = new Camel();


		for (int i=0; i<10; i++)
			A[i].createSound();
	}
}
```