Bereich: Vererbung (2)

Periodensystem	**Musterlösung**
Package: `de.dhbwka.java.exercise.classes.periodic`	**Klasse:** `PeriodicTable`

```
package de.dhbwka.java.exercise.classes.periodic;

/**
 * @author DHBW lecturer
 * @version 1.0
 *
 * Part of lectures on 'Programming in Java'. Baden-Wuerttemberg
 * Cooperative State University.
 *
 * (C) 2016 by W. Geiger, T. Schlachter, C. Schmitt, W. Suess
 */
public class Element {

	public static final int SOLID = 1;
	public static final int LIQUID = 2;
	public static final int GAS = 3;
	public static final String[] phases =
		{ "Plasma", "fest", "flüssig", "gasförmig" };

	public static final boolean MAIN = true;
	public static final boolean SIDE = false;

	private String name;
	private String symbol;
	private int ordinal;
	private char shell;
	private int phase;
	private boolean group;

	public Element() {
	}

	public Element(String name, String symbol, int ordinal,
			char shell, int phase, boolean group) {
		super();
		this.name = name;
		this.symbol = symbol;
		this.ordinal = ordinal;
		this.shell = shell;
		this.phase = phase;
		this.group = group;
	}

	@Override
	public String toString() {
		return name + " (" + symbol + "," + ordinal + ")" +
				" Schale: " + shell + ", " + phases[phase] +
				", group: "  + (group ? "Hauptgruppe" : "Nebengruppe");
	}
```

```
	@Override
	public boolean equals(Object obj) {
		if (!(obj instanceof Element))
			return false;
		return ((Element) obj).getOrdinal() == ordinal;
	}

	public String getName() {
		return name;
	}

	public void setName(String name) {
		this.name = name;
	}

	public String getSymbol() {
		return symbol;
	}

	public void setSymbol(String symbol) {
		this.symbol = symbol;
	}

	public int getOrdinal() {
		return ordinal;
	}

	public void setOrdinal(int ordinal) {
		this.ordinal = ordinal;
	}

	public char getShell() {
		return shell;
	}

	public void setShell(char shell) {
		this.shell = shell;
	}

	public int getPhase() {
		return phase;
	}

	public void setPhase(int phase) {
		this.phase = phase;
	}

	public boolean isGroup() {
		return group;
	}

	public void setGroup(boolean group) {
		this.group = group;
	}

}
```

```java
package de.dhbwka.java.exercise.classes.periodic;

/**
 * @author DHBW lecturer
 * @version 1.0
 *
 * Part of lectures on 'Programming in Java'. Baden-Wuerttemberg
 * Cooperative State University.
 *
 * (C) 2016 by W. Geiger, T. Schlachter, C. Schmitt, W. Suess
 */
public class Metal extends Element {

	private boolean metalloid;
	private double conductivity;

	public Metal() {
	}

	public Metal(String name, String symbol, int ordinal, char shell, int
phase, boolean group, boolean metalloid, double conductivity) {
		super(name, symbol, ordinal, shell, phase, group);
		this.metalloid = metalloid;
		this.conductivity = conductivity;
	}

    @Override
    public String toString() {
       return super.toString() + (metalloid ? ", Halbleiter" : "") +
                    ", \u03C3: " + conductivity;
    }

	public boolean isMetalloid() {
		return metalloid;
	}

	public void setMetalloid(boolean metalloid) {
		this.metalloid = metalloid;
	}

	public double getConductivity() {
		return conductivity;
	}

	public void setConductivity(double conductivity) {
		this.conductivity = conductivity;
	}

}
```

```java
package de.dhbwka.java.exercise.classes.periodic;

/**
 * @author DHBW lecturer
 * @version 1.0
 *
 * Part of lectures on 'Programming in Java'. Baden-Wuerttemberg
 * Cooperative State University.
 *
 * (C) 2016 by W. Geiger, T. Schlachter, C. Schmitt, W. Suess
 */
public class PeriodicTable {

	private Element[] elements = new Element[119];

	public PeriodicTable() {
	}

	public void addElement(Element e) {
		if (! hasElement(e.getOrdinal()))
			elements[e.getOrdinal()] = e;
	}

	public boolean hasElement(int o) {
		return elements[o] != null;
	}

	public Element getElement(int o) {
		return elements[o];
	}

	public Element[] getMetals() {
		int countMetals = 0;
		for (Element e : elements)
			if (e != null && e instanceof Metal)
				countMetals++;
		Element[] result = new Element[countMetals];
		int pos = 0;
		for (Element e : elements)
			if (e != null && e instanceof Metal)
				result[pos++] = e;
		return result;
	}

	// Fortsetzung s. nächste Seite!
```

```
	public static void main(String[] args) {
		PeriodicTable pt = new PeriodicTable();
		pt.addElement(new Element("Wasserstoff", "H", 1, 'K', Element.GAS,
				Element.MAIN));
		pt.addElement(new Element("Helium", "He", 2, 'K', Element.GAS,
				Element.MAIN));
		pt.addElement(new Metal("Natrium", "Na", 11, 'M', Element.SOLID,
				Element.MAIN,false,21E6));
		pt.addElement(new Metal("Eisen", "Fe", 26, 'N', Element.SOLID,
				Element.SIDE,false,10.02E6));
		pt.addElement(new Metal("Germanium", "Ge", 32, 'N', Element.SOLID,
				Element.SIDE,true,1.45));
		pt.addElement(new Element("Brom", "Br", 35, 'N', Element.LIQUID,
				Element.MAIN));
		pt.addElement(new Metal("Tellur", "Te", 52, 'O', Element.SOLID,
				Element.MAIN,true,0.005));
		pt.addElement(new Metal("Gold", "Au", 79, 'P', Element.SOLID,
				Element.SIDE,false,44E6));

		System.out.println("Elemente:");
		for(Element e : pt.elements)
			if (e != null)
				System.out.println(e);

		System.out.println("\nMetalle:");
		for(Element e : pt.getMetals())
			System.out.println(e);

		System.out.println("\nGold:");
		System.out.println(pt.getElement(79));

	}

}
```