Aufgaben 2

1

```
public class Lumenus {
	private String nummer;
	private double leuchtkraft;
	public double getLeuchtkraft() {
		return leuchtkraft;
	}
	public void setLeuchtkraft(double leuchtkraft) {
		this.leuchtkraft = leuchtkraft;
	}
	public String getNummer() {
		return nummer;
	}
	public Lumenus(String nummer,double leuchtkraft)
	{
		this.nummer=nummer;
		this.leuchtkraft=leuchtkraft;
	}
	public Lumenus()
	{
		this.nummer="nicht registriert";
		this.leuchtkraft=0;
	}
	public void ausgabe()
	{
		System.out.println(nummer+" "+leuchtkraft);
	}
	public String toString() {
		return nummer + ", " + leuchtkraft;
	}
}

public class Calorus extends Lumenus {
	private double wärmemenge;

	public double getWärmemenge() {
		return wärmemenge;
	}

	public void setWärmemenge(double wärmemenge) {
		this.wärmemenge = wärmemenge;
	}

	public Calorus(String nummer, double leuchtkraft, double wärmemenge) {
		super(nummer, leuchtkraft);
		this.wärmemenge = wärmemenge;
	}

	public Calorus() {
		super();
		this.wärmemenge = 0;
	}
	public void ausgabe()
	{
		super.ausgabe();
		System.out.println(wärmemenge);
```

```
	}
	public String toString() {
		return super.toString()+", "+wärmemenge;
	}
}

public class Energiala {

	public static void main(String[] args) {
		Lumenus lum1=new Lumenus("1111",20.5);
		System.out.println(lum1);//lum1.ausgabe();
		Lumenus lum2=new Lumenus();
		System.out.println(lum2);//lum2.ausgabe();
		lum2.setLeuchtkraft(15.33);
		System.out.println(lum2);//lum2.ausgabe();

		Calorus ca=new Calorus("1112",20.5,2.5);
		System.out.println(ca);//ca.ausgabe();
		//als array:
		Lumenus[] arrayL={lum1,lum2,ca,
				new Calorus("1113",20.5,2.6),
				new Calorus("1114",21.5,2.5)
				};
		for (int i=0;i<arrayL.length;i++)
			System.out.println(arrayL[i]);

		//als ArrayList:
		ArrayList<Lumenus> list=new ArrayList<Lumenus>();
		for (int i=0;i<arrayL.length;i++)
			list.add(arrayL[i]);
		for (Lumenus ll:list)System.out.println(ll);
	}
}

2
public class Tiger {
	private String registrierNummer;
	private double schaedelUmfang;
	private double schaedelFestigkeit;

	public Tiger(String registrierNummer, double schaedelUmfang, double
schaedelFestigkeit) {
		super();
		this.registrierNummer = registrierNummer;
		this.schaedelUmfang = schaedelUmfang;
		this.schaedelFestigkeit = schaedelFestigkeit;
	}
	public String getRegistrierNummer() {
		return registrierNummer;
	}
	public double getSchaedelUmfang() {
		return schaedelUmfang;
	}
	public double getSchaedelFestigkeit() {
		return schaedelFestigkeit;
	}
	@Override
	public String toString() {
		return "Tiger [" + registrierNummer + ", " + schaedelUmfang
				+ ", " + schaedelFestigkeit + "]";
	}
```

```
	public double gesamtFestigkeit(){
		return this.getSchaedelFestigkeit();
	}
}

public class Helm {
	private double durchmesser;
	private double festigkeit;
	public Helm(double durchmesser, double festigkeit) {
		super();
		this.durchmesser = durchmesser;
		this.festigkeit = festigkeit;
	}
	public double getDurchmesser() {
		return durchmesser;
	}
	public double getFestigkeit() {
		return festigkeit;
	}
	@Override
	public String toString() {
		return "[durchmesser=" + durchmesser + ", festigkeit=" + festigkeit +
"]";
	}
}

public class HelmTiger extends Tiger {
	private Helm helm;

	public HelmTiger(String registrierNummer, double schaedelUmfang, double
schaedelFestigkeit, Helm helm) {
		super(registrierNummer, schaedelUmfang, schaedelFestigkeit);
		this.helm = helm;
	}

	public Helm getHelm() {
		return helm;
	}

	public void setHelm(Helm helm) {
		if
(helm.getDurchmesser()*2*Math.PI>this.getSchaedelUmfang())this.helm = helm;
		else helm=null;
	}
	@Override
	public String toString() {
		String s=super.toString();
		if (helm!=null)
			return s+" HelmTiger [" + helm + "]";
		else return s;
	}

	public double gesamtFestigkeit(){
		if (helm!=null)
			return this.getSchaedelFestigkeit()+helm.getFestigkeit();
		else
		    return this.getSchaedelFestigkeit();
	}
}

//import java.util.ArrayList;
```

```
public class Register {

	public static void main(String[] args) {
		Tiger ti1=new Tiger("001",25.1,5.0);
		//System.out.println(ti1);

		Tiger[] array=new Tiger[4];
		array[0]=ti1;
		array[1]=new HelmTiger("002",25.1,5.0,new Helm(1,2.0));
		array[2]=new HelmTiger("003",25.4,5.1,new Helm(6,2.0));
		array[3]=new Tiger("004",25.5,6.0);


		for (Tiger t:array) {
			System.out.println(t);
			System.out.println(t.gesamtFestigkeit());
		}

/*		ArrayList<Tiger> list=new ArrayList<Tiger>();
		list.add(ti1);
		list.add(new HelmTiger("002",25.1,5.0,new Helm(1,2.0)));
		list.add(new HelmTiger("003",25.4,5.1,new Helm(6,2.0)));
		list.add(new Tiger("004",25.5,6.0));

		if (list.get(1) instanceof HelmTiger)
System.out.println(list.get(1).getClass());
		//((HelmTiger)list.get(1)).setHelm(new Helm(6,2.0));

		*/
	}
}


3
public interface Mobile {
	public double getGewicht();
	public String toString();
	public void balancieren();
}

public class Stern implements Mobile {
//static IllegalArgumentException error=new IllegalArgumentException("negativer
Argument");
static MyIllegalArgumentException error=new MyIllegalArgumentException("negativer
Argument");
protected double gewicht;

public Stern(double gewicht) throws MyIllegalArgumentException
{//IllegalArgumentException {
		super();
		if (gewicht<0) {
			error.quelle="Konstruktor";
			throw error;
		}
		else this.gewicht = gewicht;
	}

	@Override
	public double getGewicht() {
		return gewicht;
```

```
	}

	@Override
	public void balancieren() {
	}

	@Override
	public String toString() {
		return "Gewicht "+gewicht
	}
}

public class Stab implements Mobile {

	double laenge_l, laenge_r;
	Mobile mobile_l, mobile_r;


	public Stab(double laenge, Mobile mobile_l, Mobile mobile_r) {
		super();
		this.mobile_l = mobile_l;
		this.mobile_r = mobile_r;
		laenge_l=laenge;
		laenge_r=0;
		balancieren();
	}

	@Override
	public double getGewicht() {
		return mobile_l.getGewicht()+mobile_r.getGewicht();
	}

	@Override
	public void balancieren() {

	laenge_r=(mobile_l.getGewicht()*laenge_l)/(mobile_r.getGewicht()+mobile_l.ge
tGewicht());
		laenge_l=laenge_l-laenge_r;
	}

	@Override
	public String toString() {
		return "Links: "+this.laenge_l+", "+this.mobile_l+" Rechts:
"+this.laenge_r+", "+this.mobile_r;
	}
}

public class Test {

	public static void main(String[] args) {
		Stern s1=new Stern(2);
		Glitzerstern s2=new Glitzerstern(4);
		Stern s3=new Stern(9);
		Stab st=new Stab(9,s1,s2);
		Stab st1=new Stab(10,st,s3);
		System.out.println(st1);
		s2.dekorieren();s2.dekorieren();s2.dekorieren();
		st.balancieren();
		st1.balancieren();
		System.out.println(st1);
```

```
		try{
			Stern sf=new Stern(-19);
		}
		catch(MyIllegalArgumentException e){
			System.out.println(e.quelle);
		}


	}

}

public class MyIllegalArgumentException extends IllegalArgumentException {
	public MyIllegalArgumentException(String string) {
		super(string);
	}
	private static final long serialVersionUID = 1L;
	public String quelle;
}
```