Matrikelnummer:	Beispielklausur

Fachrichtung:	Wirtschaftsinformatik
Kurs:	Beispielklausur
Studienhalbjahr:	2

Zur Vorlesung	Fortgeschrittene Programmierung
Dozent:	Matthias Berg-Neels
Datum:	
Bearbeitungszeit:	60 Minuten
Hilfsmittel:	keine

Aufgabenblock:	**1**	**2**	**3**	**4**	**Σ**
Punkte:	11	13	20	16	60
Erreicht:					

Weitere Hinweise:

- Bitte schreiben Sie leserlich
- Keine Hilfsmittel
- Bitte unternehmen Sie keine Täuschungsversuche

Viel Erfolg !!!

Aufgabenblock 1: Interfaces

Aufgabe 1 (4 Punkte)
Worin unterscheiden sich Interfaces von abstrakten Klassen in Java?

Aufgabe 2 (7 Punkte)
Erweitern Sie das nachstehende Interface „Berechnen“ und die Klasse „Quadrat“ so, dass die main()-Methode der Klasse „GeometrieTest“ fehlerfrei ausgeführt werden kann.

```
package KlausurII;

public interface Berechnen {




}
```

```
package KlausurII;

public class Quadrat                          {

	private double seite;

	public Quadrat (double seite){
		this.seite = seite;
	}









}
```

```java
package KlausurII;

public class GeometrieTest {

	public static void main(String[] args) {

		Berechnen figuren[] = new Berechnen[2];
		double ergebnis;

		figuren[0] = new Kreis(5);
		figuren[1] = new Quadrat(3);

		ergebnis = figuren[0].berechneFlaeche();
		System.out.println(ergebnis);

		ergebnis = figuren[1].berechneUmfang();
		System.out.println(ergebnis);
	}

}
```

Aufgabenblock 2: Exception Handling

Aufgabe 3 (4 Punkte)
Erläutern Sie das Grundprinzip der Ausnahmebehandlung in Java! Geben Sie die wesentlichen Befehle (Schlüsselworte) an!

Aufgabe 4 (9 Punkte)
Schreibe Sie die Konsolenausgabe auf, die sich bei Ausführung der main Methode aus der Klasse „Schokobox“ ergeben würde. Jede richtige Zeile (Reihenfolge zählt mit!) gibt einen Punkt – jede falsche Zeile gibt einen Punkt Abzug.

```
package Nervennahrung;

public class SchokoLeer extends Exception {

	public SchokoLeer(int rest, int angefragt){
		super("Achtung: Es werden " + angefragt +
		" Tafeln gebraucht, aber es sind nur noch " + rest +
		" Tafeln in der Box.");
	}
}
package Nervennahrung;

public class Schokobox {
	private int tafeln = 5;

	public void essen(int tafeln) throws SchokoLeer{
		if(tafeln <= this.tafeln){
			this.tafeln -= tafeln;
			System.out.println("Es sind noch " + this.tafeln +
					" Tafeln in der Box.");
		}else{
			throw new SchokoLeer(this.tafeln, tafeln);
		}
	}

	public static void main (String[] args){

		System.out.println("Klausuranfang!");
		Schokobox Klausurnervennahrung = new Schokobox();
		System.out.println("Jetzt erst mal was essen...");
		try {
			Klausurnervennahrung.essen(2);
			System.out.println("Solange Schokolade da ist,
					ist die Klausur leicht!");
			Klausurnervennahrung.essen(1);
			Klausurnervennahrung.essen(3);
			System.out.println("Puh... ist Java kompliziert!");
			Klausurnervennahrung.essen(4);
		} catch (SchokoLeer e) {
			System.out.println(e.getMessage());
			System.out.println("Geh Schokolade kaufen!");
		} catch (Exception e) {
			System.out.println(e.toString());
		}
		finally{
			System.out.println("Ohne Schokolade kann man nicht arbeiten!");
		}

		System.out.println("Klausurende!");
	}
}
```

Konsolenausgabe:

Aufgabenblock 3: Collection Framework

Aufgabe 5 (6 Punkte)
Was müssen Sie beim Überschreiben der equals()-Methode für direkte und indirekte Sub-Klassen der Klasse „Object“ beachten?

Aufgabe 6 (14 Punkte)
Gegeben ist folgendes Coding, bei dem vier Objekte der Klasse „Fahrzeug“ in ein Objekt der Klasse „Vector“ eingefügt werden. Nach dem Einfügen sollen die Objekte innerhalb des Vectors sortiert werden. Bitte implementieren Sie dazu die Klasse „FahrzeugComparator“ so, dass die Objekte erst nach der PS-Zahl, bei gleicher PS-Zahl nach der Marke und bei gleicher Marke nach dem KFZ-Kennzeichen sortiert werden. Rufen Sie die entsprechende Funktion auf, um den Vector zu sortieren. Geben Sie die Elemente des Vectors über einen Iterator aus.

```
import java.util.*;

public class Fuhrpark {
      public static void main(String[] args) {
            Vector fuhrpark = new Vector();
            Fahrzeug kfz1 = new Fahrzeug("Audi","HD-XX 246",180);
            Fahrzeug kfz2 = new Fahrzeug("BMW","MA-LU 845",170);
            Fahrzeug kfz3 = new Fahrzeug("VW","MA-BA 563",170);
            Fahrzeug kfz4 = new Fahrzeug("Saab","MOS-FK 74",90);

            fuhrpark.add(kfz1);
            fuhrpark.add(kfz2);
            fuhrpark.add(kfz3);
            fuhrpark.add(kfz4);









            }
      }
}
```

```java
public class Fahrzeug {
	private String marke, kfzKz;
	private int ps;

	public Fahrzeug(String marke, String kfzKz, int ps) {
		this.marke = marke;
		this.kfzKz = kfzKz;
		this.ps = ps;
	}

	public int getPs() {
		return ps;
	}
	public String getKfzKz() {
		return kfzKz;
	}
	public String getMarke() {
		return marke;
	}
}

import java.util.Comparator;

public class FahrzeugComparator                                    {









}
```

Aufgabenblock 3: Swing

Aufgabe 7 (4 Punkte)
Erläutern Sie die wesentlichen Unterschiede zwischen AWT- und Swing-Komponenten!

Aufgabe 8 (12 Punkte)
Murvin hat sein „Bizepstagebuch“ umprogrammiert! Jetzt können zwei Spieler ihren Bizepsumfang vergleichen. Der untenstehende Quellcode erzeugt das UI.

a) Malen Sie eine Skizze von dem UI, das durch den Quellcode erzeugt wird.
b) Im Quellcode fehlt noch die Interaktion mit dem Anwender. Erweitern Sie den Quellcode, um einen „ActionListener“ und weiteres benötigtes Coding. Bei einem Klick auf den Button „Vergleichen“ sollen die Werte der beiden JComboBoxen miteinander verglichen werden. Bei einem Klick auf „Schließen“ soll das Programm beendet werden. Bei dem Vergleich sollen folgende Fälle berücksichtig werden:
    a. Sind die Bizepswerte gleich, soll auf der Konsole ausgegeben werden „`<Spieler1> und <Spieler2> müssen trainieren!`“ (Wobei <Spieler1> und <Spieler2> gegen die Namen aus den Eingabefeldern ersetzt werden sollen).
    b. Unterscheiden sich die Werte, soll auf der Konsole ausgegeben werden „`<Spieler> ist der Babo!`“ (Wobei <Spieler> gegen den Wert aus dem Eingabefeld ersetzt wird, wer den größeren Bizeps hat).

Quellcode Aufgabe 8:

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class BizepsVergleich extends JFrame {

	public static final String ACTION_VERGLEICHEN = "VERGLEICHEN";
	public static final String ACTION_SCHLIESSEN = "SCHLIESSEN";

	private BizepsVergleich(){
		super("Bizeps Vergleich");

		this.setLayout(new GridLayout(2, 3));

		JLabel lblSpieler1 = new JLabel("Spieler 1");
		JLabel lblSpieler2 = new JLabel("Spieler 2");

		final JTextField tfSpieler1 = new JTextField("Murvin");
		tfSpieler1.setEnabled(false);
		final JTextField tfSpieler2 = new JTextField();


		Integer[] bWerte = {30, 32, 34, 36, 38, 40, 42, 44, 46, 48};
		final JComboBox bizepsBox1 = new JComboBox(bWerte);
		final JComboBox bizepsBox2 = new JComboBox(bWerte);
```

```
        JButton btnVergleichen = new JButton("Vergleichen");
        JButton btnSchliessen = new JButton("Schließen");









        this.add(lblSpieler1);
        this.add(tfSpieler1);
        this.add(bizepsBox1);
        this.add(btnVergleichen);

        this.add(lblSpieler2);
        this.add(tfSpieler2);
        this.add(bizepsBox2);
        this.add(btnSchliessen);

        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        this.pack();
        this.setVisible(true);
}


public static void main(String[] args) {
        new BizepsVergleich();

}
```