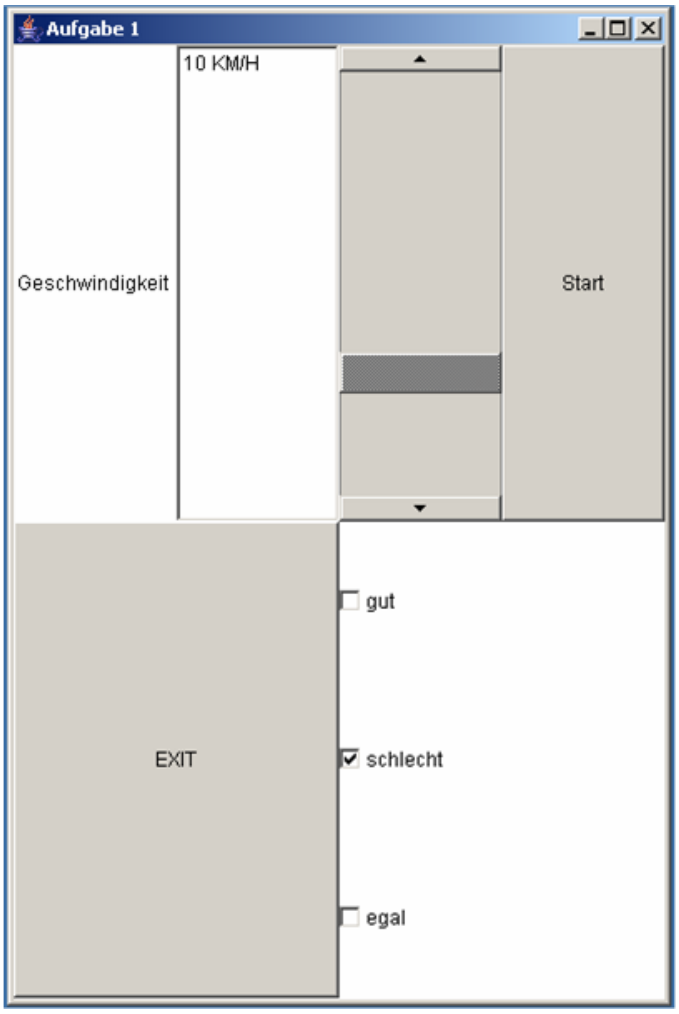
# Programmierkurs Java

**Dr. Dietrich Boles**

## Aufgaben zu UE22-NutzungVonKlassen
**(Stand 28.09.2012)**

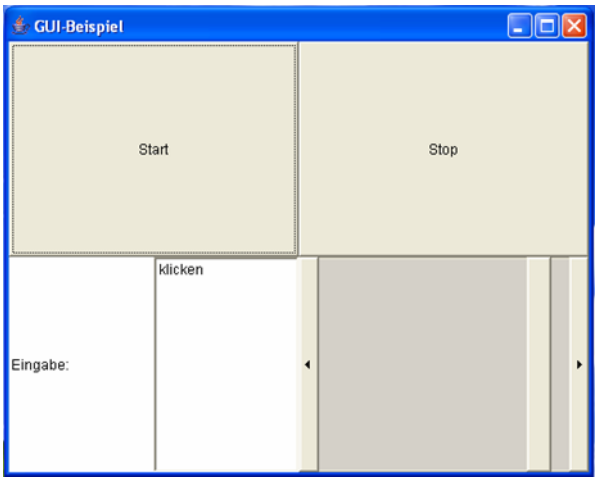
### Aufgabe 1:

Entwickeln Sie in Eclipse auf der Basis der vorgestellten Java-GUI-Klassen ein Java-Programm, das folgende GUI erzeugt:



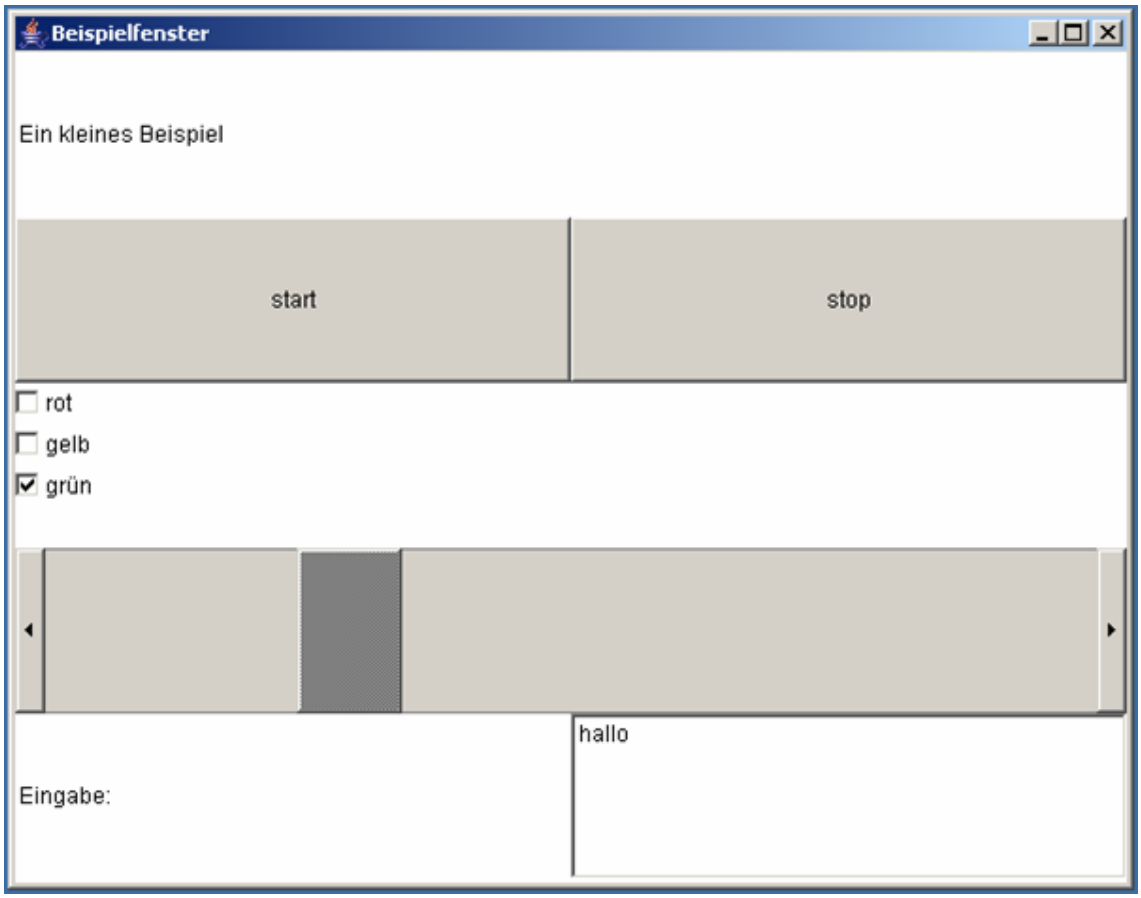

### Aufgabe 2:

Entwickeln Sie in Eclipse auf der Basis der vorgestellten Java-GUI-Klassen ein Java-Programm, das folgende GUI erzeugt:



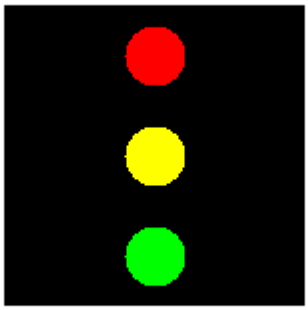

**Aufgabe 3:**

Entwickeln Sie in Eclipse auf der Basis der vorgestellten Java-GUI-Klassen ein Java-Programm, das folgende GUI erzeugt:




**Aufgabe 4:**

Laden Sie die Datei „Shapes.java“ in Eclipse. Ändern Sie die main-Prozedur so ab, dass eine rudimentäre Ampel der folgenden Form auf den Bildschirm gezeichnet wird.



Ampel zu Aufgabe 4

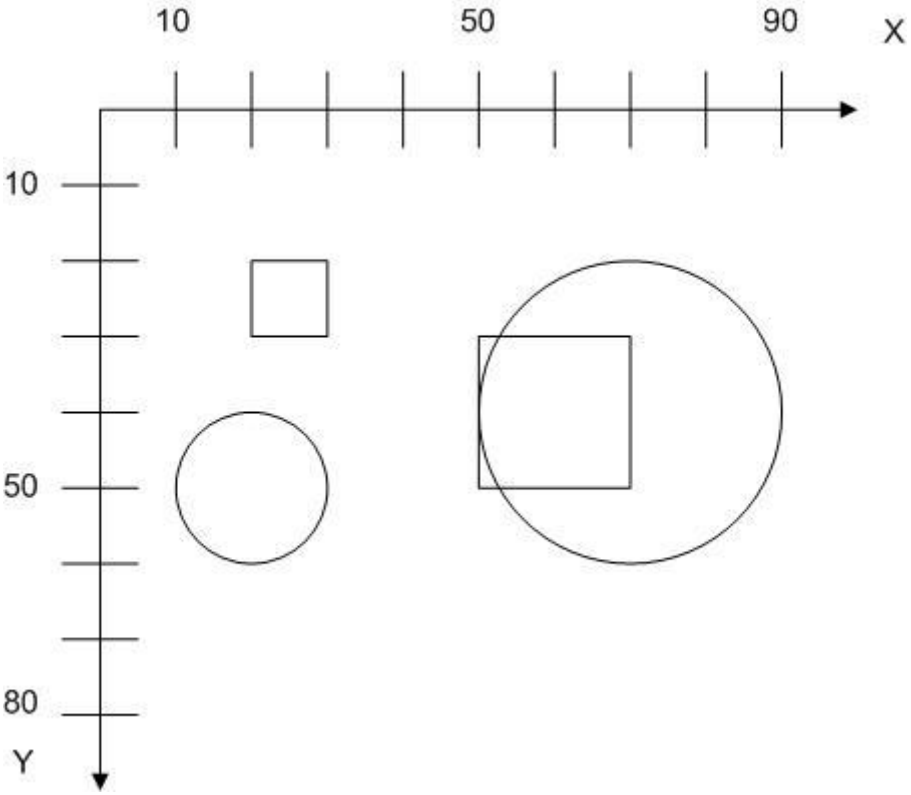
**Aufgabe 5:**

Schauen Sie sich die folgenden beiden Klassen an. Sie repräsentieren Kreise bzw. Quadrate, die auf einen Bildschirm mit einem rechtwinkligen Koordinatensystem gezeichnet werden können

```
class Kreis {

	// zeichnet einen Kreis mit dem Mittelpunkt bei Punkt (x=0/y=0) und
	// einem Radius von 10 Einheiten
	Kreis()

	// verschiebt den Kreis um den als Parameter uebergebenen Wert in der
	// Horizontalen
	void verschiebeHorizontal(int einheiten)

	// verschiebt den Kreis um eine Einheit in der Vertikalen
	void verschiebeVertikal()

	// vergroessert den Radius des Kreises um den als Parameter
	// uebergebenen Wert
	void vergroessere(int einheiten)

	// liefert die x-Koordinate des Kreises
	int getX()

	// liefert die y-Koordinate des Kreises
	int getY()
}

class Quadrat {

	// zeichnet ein Quadrat mit der linken oberen Ecke bei Punkt
	// (x=0/y=0) und einer Hoehe und Breite von 50 Einheiten
	Quadrat()

	// verschiebt das Quadrat so, dass dessen linke obere Ecke
	// anschliessend auf den Mittelpunkt des als Parameter
	//uebergebenen Kreises faellt
	void verschiebeNach(Kreis kreis)

	// vergroessert die Hoehe und Breite des Quadrates um den als
```

```
      // Parameter uebergebenen Wert
      void vergroessere(int einheiten)
}
```

**Aufgabe**: Schreiben Sie auf der Basis dieser Klassen ein Programm, dass die auf der folgenden Seite skizzierte Bildschirmausgabe erzeugt:




**Aufgabe 6:**

Schauen Sie sich die beiden folgenden Klassen an:

```
class Kreis {

      double PI = 3.1415;

      double radius;

      Kreis(double r) {
            radius = r;
      }

      void vergroessern(double wert) {
            radius += wert;
      }

      double liefereFlaeche() {
            return radius * radius * PI;
      }
}
```

```
// realisiert ein Quadrat mit einem internen die Raender des
// Quadrats berührenden Kreis
class QuadratMitKreis {

	Kreis kreis;

	double groesse;

	QuadratMitKreis(double g) {
		kreis = new Kreis(g / 2);
		groesse = g;
	}

	void vergroessern(double wert) {
		kreis.vergroessern(wert / 2);
		groesse += wert;
	}

	double liefereFlaeche() {
		return groesse * groesse;
	}

	double liefereFlaecheOhneKreis() {
		return liefereFlaeche() - kreis.liefereFlaeche();
	}
}
```

**Aufgabe**: Schreiben Sie durch Benutzung dieser Klassen (sowie der bekannten Klasse IO) ein Java-Programm, das folgendes tut:

- Zunächst wird ein QuadratMitKreis-Objekt mit einer Größe erzeugt, die vom Benutzer abgefragt wird. Achten Sie darauf, dass der Nutzer "ordentliche" Werte eingibt.
- Anschließend werden in einer Schleife, solange die Fläche des Quadrats kleiner als 100 ist, jeweils die Fläche des Quadrates und die Fläche des Quadrates abzüglich der Kreisfläche ausgegeben sowie das Quadrat um den Wert 1 vergrößert.

Im Folgenden wird ein Beispiel für eine mögliche Ausgabe des Programms gegeben (in <> die Eingaben des Benutzers):

```
Quadratgroesse: <8>
Quadratflaeche = 64.0
Quadratflaeche ohne Kreisflaeche = 13.736
Quadratflaeche = 81.0
Quadratflaeche ohne Kreisflaeche = 17.385
```

**Aufgabe 7:**

In dieser Aufgabe geht es darum, die Größe der Oberflächen zweier volumenmäßig ungefähr gleichgroßer 3-dimensionaler Behälterobjekte miteinander zu vergleichen. Schauen Sie sich dazu die folgenden Klassen an:

```
class Wuerfel {
	double kantenLaenge;

	Wuerfel(double k) {
		kantenLaenge = k;
	}

	double getOberflaeche() {
		return 6.0 * kantenLaenge * kantenLaenge;
	}

	double getVolumen() {
		return kantenLaenge * kantenLaenge * kantenLaenge;
	}

	void veraendern(double inkrement) {
		kantenLaenge += inkrement;
	}
}

class Kugel {
	double PI = 3.1415;

	double radius;

	Kugel(double r) {
		radius = r;
	}

	double getOberflaeche() {
		return 4.0 * PI * radius * radius;
	}

	double getVolumen() {
		return 4.0 / 3.0 * PI * radius * radius * radius;
	}

	void veraendern(double inkrement) {
		radius += inkrement;
	}
}
```

**Aufgabe**: Schreiben Sie durch Benutzung dieser Klassen (sowie der bekannten Klasse IO) ein Java-Programm, das folgendes tut:

- Zunächst werden ein Wuerfel- und ein Kugel-Objekt mit jeweils einer Größe (Kantenlänge bzw. Radius) erzeugt, die zuvor vom Benutzer abgefragt wird. Achten Sie darauf, dass der Nutzer "ordentliche" Werte eingibt.

- Anschließend wird das Volumen der beiden Objekte auf den Bildschirm ausgegeben

- Danach wird ein Inkrementwert *inkrement* vom Benutzer abgefragt.

- Anschließend werden in einer Schleife die Volumen der beiden Objekte "angeglichen", und zwar auf folgende Art und Weise: Das anfangs volumenmäßig größere Objekt wird in jedem Schleifendurchlauf um den Inkrementwert verkleinert (Methode *veraendern*) und das anfangs volumenmäßig kleinere Objekt wird um den Inkrementwert vergrößert. Die Schleife wird solange durchlaufen, bis das anfangs volumenmäßig größere Objekt volumenmäßig kleiner ist als das anfangs volumenmäßig kleinere Objekt.

- Zum Schluss werden Volumen und Oberfläche der beiden Objekte auf den Bildschirm ausgegeben.

Im Folgenden wird ein Beispiel für eine mögliche Ausgabe des Programms gegeben (in <> die Eingaben des Benutzers):

```
Kantenlaenge: <10.0>
Radius: <9.0>
Wuerfel-Volumen (Anfang): 1000.0
Kugel-Volumen (Anfang): 3053.5
Inkrement: <0.001>
Wuerfel-Volumen: 1612.3
Wuerfel-Oberfläche: 824.9
Kugel-Volumen: 1612.1
Kugel-Oberflaeche: 664.8
```

**Aufgabe 8:**

Gegeben ist folgende Klasse `Zahl`:

```
class Zahl {

     int wert;

     Zahl() {
          wert = (new java.util.Random()).nextInt(1000);
     }

     void inkr() {
          wert++;
     }

     void dekr() {
```

```
        wert--;
    }

    boolean equals(Zahl zahl) {
        return wert == zahl.wert;
    }

    int compareTo(Zahl zahl) {
        return wert - zahl.wert;
    }

    String nachString() {
        return "" + wert;
    }

}
```

**Aufgabe**: Schreiben Sie ein Programm, das zwei Zahl-Objekte erzeugt. Anschließend soll das größere Objekt solange verkleinert werden (`dekr`) und das kleiner Objekt solange vergrößert werden (`inkr`), bis das zunächst größere Objekt kleiner oder gleich dem zunächst kleineren Objekt ist.

**Aufgabe 9:**

Gegeben seien die beiden folgenden Klassen zur Darstellung und Bearbeitung von runden Glasböden und Trinkgläsern, die ein Trinkglas durch jeweils einen Glasboden und durch eine Füllstands-Angabe darstellen

```
class Glasboden {
    double PI = 3.1415;

    double radius;

    Glasboden(double r) {
        radius = r;
    }

    void verkleinern(double x) {
        // verkleinert den Radius des Glasboden-Objekts um x
        radius = radius - x;
    }

    double flaeche() {
        // liefert die Flaeche des Glasboden-Objekts
        return PI * radius * radius;
    }

    double umfang() {
        // liefert den Umfang der Glasboden-Objekts
        return 2 * PI * radius;
```

```
    }

    String nachString() {
        // liefert die String-Darstellung des Glasboden-Obj.
        return "B(r=" + radius + ")";
    }
}

class TrinkGlas {
    Glasboden boden;

    double fuellStand;

    TrinkGlas(double fuellStand, Glasboden boden) {
        this.fuellStand = fuellStand;
        this.boden = boden;
    }

    void verkleinern(double x) {
        boden.verkleinern(x);
        fuellStand = fuellStand - x;
    }

    double innenFlaeche() {
        return boden.umfang() * fuellStand + boden.flaeche();
    }

    double fuellMenge() {
        return boden.flaeche() * fuellStand;
    }

    String nachString() {
        return "G(b=" + boden + ",s=" + fuellStand + ")";
    }

}
```

**Aufgabe**: Implementieren Sie auf der Grundlage dieser Klassen ein Java-Programm *TesteTrinkGlas*, in dem zunächst ein Trinkglas aus einem Glasboden mit vom Benutzer eingelesenen Radius- und Füllstandstandwerten erzeugt werden soll. Danach sollen in einer Schleife das Trinkglas jeweils um den Wert 5 verkleinert werden und das aktuelle Trinkglas, seine bedeckte Innenfläche und seine Füllmenge ausgegeben werden. Die Schleife soll nur durchlaufen werden, falls bzw. solange für die Innenfläche I und die Füllmenge M des Trinkglases gilt: I < M/8. Testen Sie das Programm bspw. mit einem Radius von 100 und einem Füllstand von 50.

**Aufgabe 10:**

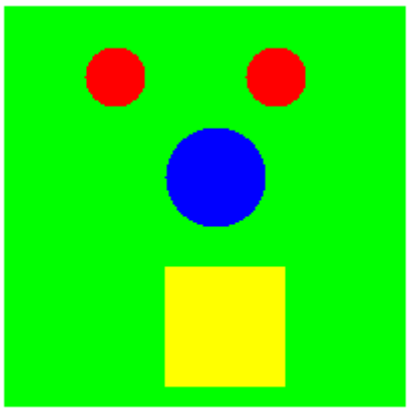
Wir haben in der Vorlesung gelernt, dass Objekte im Umfeld der objektorientierten Programmierung Gegenstände, Sachverhalte, Lebewesen, Dinge, etc. repräsentieren. Sie besitzen Eigenschaften und Verhaltensweisen. Klassen stellen Baupläne für Objekte dar. Überlegen Sie sich **fünf** verschiedene Objekte aus ihrem

tagtäglichen Umfeld und skizzieren Sie eine entsprechende Klasse. Erzeugen Sie Objekte und greifen Sie auf die Attribute und Methoden zu. Beispiel: Bücher

```
public class Buch {
  int isbn;
  String titel, autor;
  float preis;

  public Buch(String titel, String autor) {}
  public void veroeffentlichen(int isbn) {}
  public void anbieten(float preis) {}
  public void kaufen(int anzahl) {}
  public float preisErmitteln(int anzahl) {}
  public void verleihen(String person) {}
}

Buch hamster =
  new Buch("Programmieren spielend gelernt", "Dietrich Boles");
hamster.veroeffentlichen(3519022974);
hamster.anbieten(39,90);
hamster.kaufen(8);
```

**Aufgabe 11:**

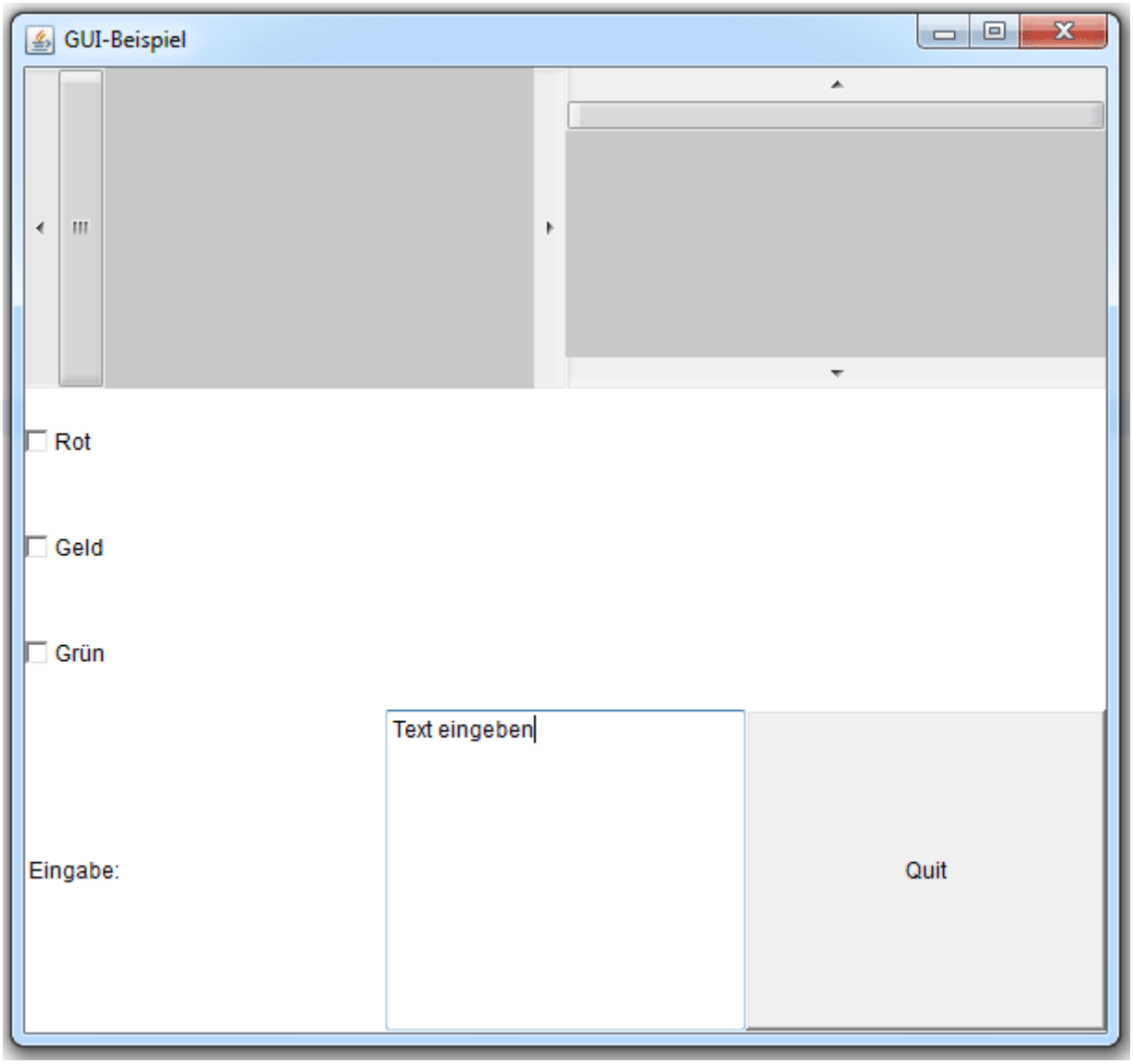
Laden Sie die Datei „Shapes.java“ in Eclipse. Ändern Sie die main-Prozedur so ab, dass ein Gesicht der folgenden Form auf den Bildschirm gezeichnet wird.



Gesicht zu Aufgabe 11

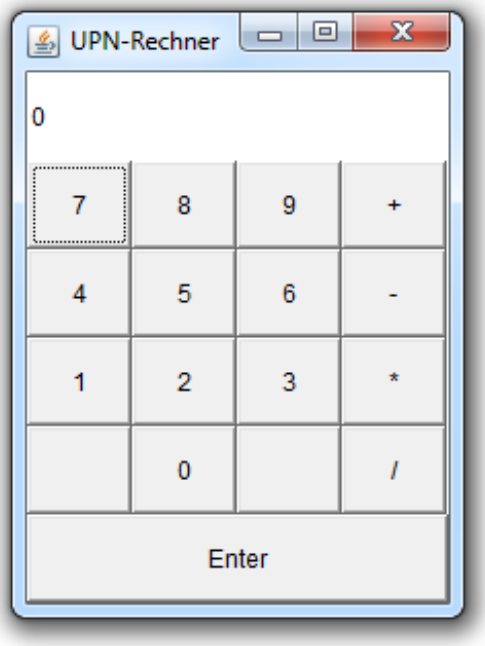
**Aufgabe 12:**

Entwickeln Sie in Eclipse auf der Basis der vorgestellten Java-GUI-Klassen ein Java-Programm, das folgende GUI erzeugt:




**Aufgabe 13:**

Bilden Sie mit den Ihnen bekannten Java-AWT-Klassen exakt folgende Taschenrechner-GUI nach:


UPN-Rechner
0
7
8
9
+
4
5
6
-
1
2
3
*
0
/
Enter