Objektorientierte Programmierung (OOP)

Aufgabenblatt
Geometrische Figuren

Geometrische Figuren modellieren

Aufgabe 1.1

Modellieren Sie folgende geometrische Figuren in UML und in C#:

- **Rechteck, Quadrat, Kreis**
  (Zusatzaufgabe: Gleichschenkliges Dreieck, Trapez)
- Jede Klasse erhält die Methode

  berechneFlaecheninhalt():double
- Erzeugen Sie eine Testklasse **TestGeoFigur**,
  - die zu jeder Klasse zwei Objekte erzeugt,
  - den gemeinsamen Flächeninhalt aller Objekte ermittelt.

Beliebig viele geometrische Figuren

Aufgabe 1.2

Erstellen Sie ein Array von 100 Kreisen, deren Größe zufällig gewählt wird und berechnen Sie den Flächeninhalt aller Kreise im Array.

Aufgabe 1.3

Wie Aufgabe 1.2, allerdings gibt der Benutzer vorher ein, wie viele Kreise erzeugt werden sollen.

Klassen verschönern

Aufgabe 1.4

Ergänzen Sie Ihre Klassen um folgende Methoden

- **Konstruktoren** zur komfortableren Erstellen von Objekten, z.B.

  Rechteck r1 = new Rechteck(100,200);

- **Getter- und Setter**-Methoden für alle Attribute, z.B.

  r1.SetBreite(500);

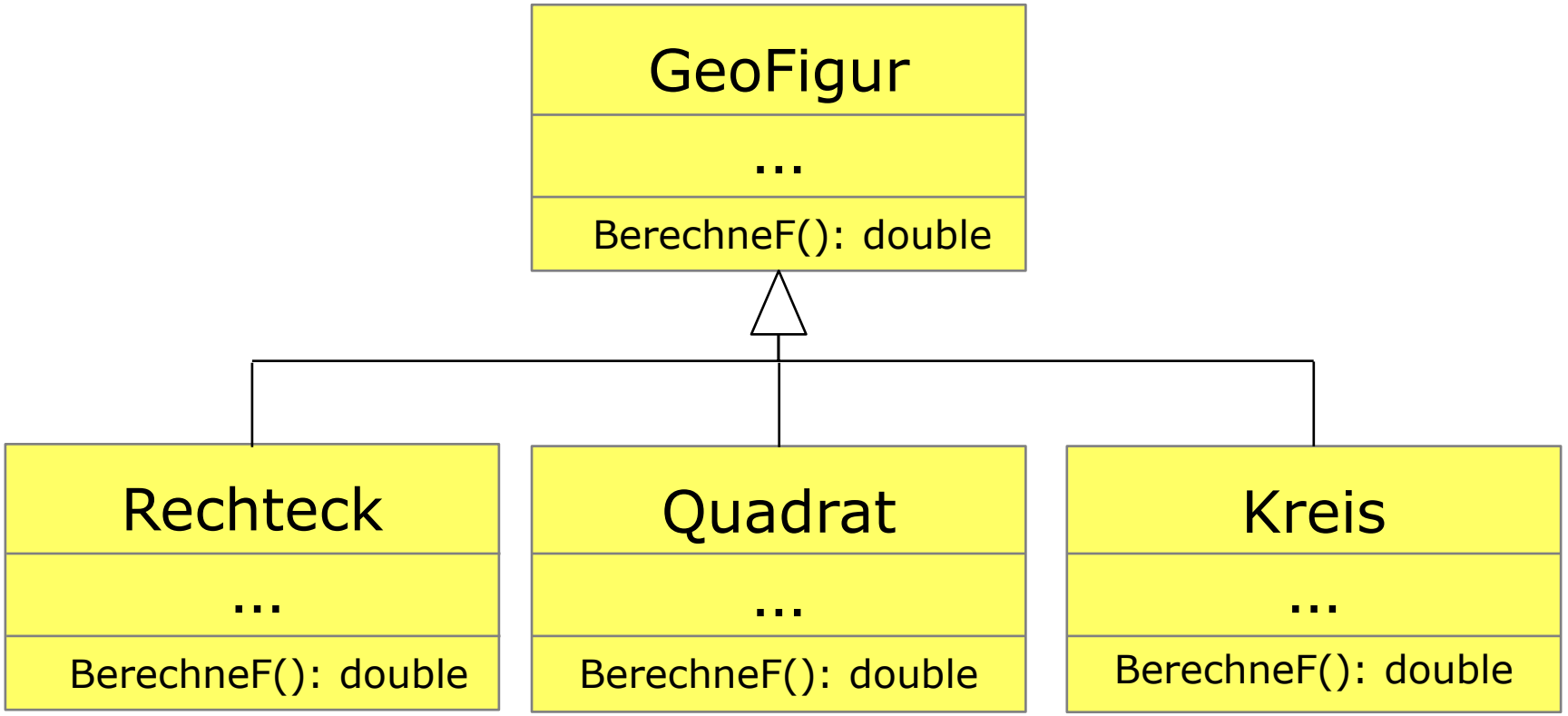
- Eine **ToString():String**-Methode die einen lesbaren Namen des Objektes ausgibt, z.B.

  CW(r1); -> "Rechteck(100,200)"

Vererbung: Klasse GeoFigur

Aufgabe 2.1

Erstellen Sie eine Klasse *GeoFigur*, und lassen Sie die Klassen Rechteck, Quadrat und Kreis von ihr erben:



Polymorphie

Aufgabe 2.2

Ähnlich wie Aufgabe 1.2

> *Erstellen Sie ein Array von 100 Kreisen, deren Größe zufällig gewählt wird und berechnen Sie den Flächeninhalt aller Kreise im Array.*

allerdings enthält das Array jetzt **verschiedene geometischen Figuren** (also Kreis, Quadrat, Rechteck).

GeoFigurPaint1
Zeichnen von geometrischen Figuren

Aufgabe 3.1

Beginnen Sie ein neues C#-Projekt mit dem Namen *GeoFigurPaint1* und kopieren Sie alle Klassen von GeoFigur in dieses Projekt. Erweitern Sie Ihre Klassen folgendermaßen:

- Jede geometrische Figur hat eine **x- und y-Koordinate**, die ihre Position angibt (getter, setter schreiben).
- Jede geometrische Figur braucht eine Methode

```
public void Paint(Graphics g){
    g.<richtigeZeichenmethode>(…);
}
```

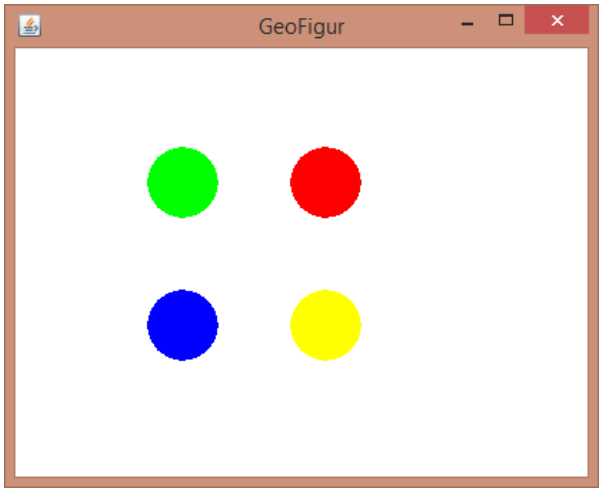
Suchen sie die passende Methode aus der Dokumentation der Klasse java.awt.Graphics heraus und implementieren Sie diese.

GeoFigurPaint1
Zeichnen von geometrischen Figuren

Erstellen Sie eine Window Forms- oder Window WPF-Klasse um folgende Kreise darzustellen.

Zeichnen Sie auch andere geometrische Figuren.

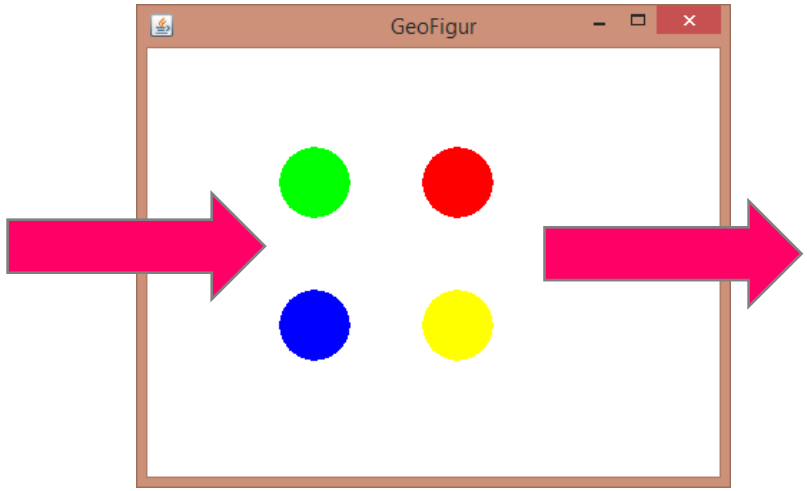
Recherchieren Sie alle notwendigen Befehle die Sie dafür benötigen



GeoFigurPaint3
Animieren von geometrischen Figuren

Aufgabe 3.3

Ergänzen Sie alle Klassen so dass die Kreise sich von links nach rechts bewegen. Erreichen die Objekte den rechten Rand, so sollen sie wieder links von Neuem erscheinen.

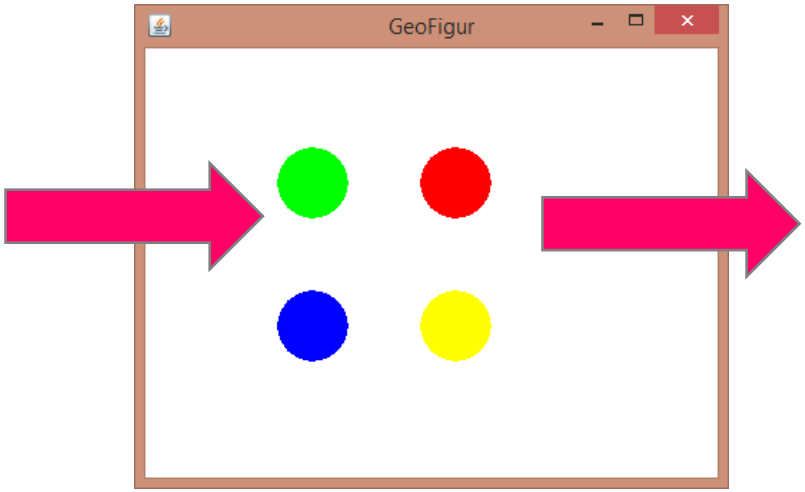


GeoFigurPaint3Abstract
Animieren von geometrischen Figuren

Aufgabe 4

Beginnen Sie ein neues C#-Projekt mit dem Namen *GeoFigurPaint3Abstract* und kopieren Sie alle Klassen von GeoFigurPaint3 in dieses Projekt.

Definieren Sie die Klasse GeoFigur als abstrakte Klasse mit einer **abstrakten Paint-Methode**. Ändern Sie die übrigen Klassen so ab, dass die abstrakte Klasse richtig verwendet wird.



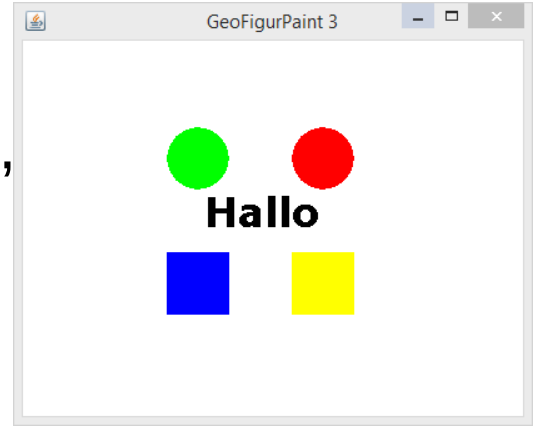
GeoFigurPaint3Interface
Animieren von geometrischen Figuren

Aufgabe 5

Beginnen Sie ein neues C#-Projekt mit dem Namen GeoFigurPaint3Interface und kopieren Sie alle Klassen von GeoFigurPaint3 in dieses Projekt.

Erstellen Sie ein Interface IPaintable, das die Paint-Methode verlangt. Ändern Sie die Klasse GeoFigurFrame so ab, dass nur Objekte vom Typ IPaintable gezeichnet werden.

Erstellen Sie zusätzlich eine Klasse TextString die auch IPaintable implementiert, die anstatt einer geometrischen Figur, einen Text auf der Zeichenfläche ausgibt.



GeoFigurPaint3Interface
Kollisionen erkennen

Aufgabe 6

Erstellen Sie ein Interface Intersectable, mit folgenden Methoden:

- boolean collisionWithObject(Intersectable obj2);
  - Erkennt, ob eine Kollision mit obj2 vorliegt.
- boolean collisionNow();
  - Gibt zurück, ob das Objekt gerade in einer Kollision ist.

Implementieren Sie das Interface für die Klasse GeoFigur und schreiben Sie die speziellen Methoden für die Unterklassen.

Kollidieren Objekte, so ändern sie für die Zeit ihre Farbe.