

# Übung 6

## Abstrakte Klassen und Interfaces in Java, Entwurfsmuster „Strategie“

Anhand der Figuren in einem Schachspiel soll das Entwurfsmuster „Strategie“ nachvollzogen werden.

Als Basisklasse soll die folgende verwendet werden:

```
public abstract class Figur {
    ZugMoeglichkeit zugMoeglichkeit;
    public void zieheFigur() { zugMoeglichkeit.ziehe(); }
}
```

1. Tippen Sie die Basisklasse `Figur.java` ab und schreiben Sie das Interface `ZugMoeglichkeit`, das die virtuelle Methode `ziehe()` bereitstellt.
2. In einem Schachspiel gibt es die Figuren `Bauer`, `Springer`, `Laeufer`, `Turm`, `Dame` und `Koenig`, die als Subklasse von `Figur` implementiert werden sollen. Schreiben Sie jede dieser von `Figur` abgeleiteten Klasse in eine entsprechende JAVA-Datei.
3. Als `Zugmoeglichkeit` verwenden wir vorerst einfach eine Bildschirmausgabe (wir wollen es nicht übertreiben), also z.B. ein `System.out.println("zwei-vor-eins-zur-seite");` für die Figur `Springer`. Wenn Sie die Züge der Schachfiguren nicht kennen, nehmen Sie als Text einfach `"Springerzug"` oder ähnliches. Die zu einer Figur gehörenden Zugmöglichkeiten sollen in der Methode `ziehe()` berücksichtigt werden (ähnlich, wie wir es in der Vorlesung am Beispiel von Bewegungsmöglichkeiten der Tierarten durchgespielt hatten).
4. Welche Vor- und Nachteile hat dieses Entwurfsmuster bezüglich des Problems „Beschreibung von Schachfiguren“? (Überlegen Sie, was getan werden müsste, wenn eine neue Figur mit anderen Zugmöglichkeiten hinzukommt).
5. Wenn Sie möchten, implementieren Sie ein Testprogramm, das ein paar Figuren ziehen lässt. In einem Schachspiel gibt es übrigens von einigen Figuren mehrere, so dass man vom code reuse z.B. beim `weißerLaeufer` und `schwarzerLaeufer` profitieren könnte.
6. Stellen Sie die programmierten Klassen und Interfaces mit `argouml` als Klassendiagramm dar.