# Auftrag 3: Methoden vererben und überschreiben, dynamische Bindung

## Aufgabe 1: Methoden vererben und dynamische Bindung (Alternativ kann auch die Aufgabe 2 gelöst werden, oder natürlich beide.)

*Einleitung:*

K 3.2

Wir wenden uns nun den *Methoden* für das Beispielprogramm Personen aus den vorhergehenden Arbeitsblättern zu. Wie Sie wissen implementieren die Methoden einer Klasse das *Verhalten* der Objekte dieser Klasse. Man kann auch sagen, die Methoden legen die möglichen *Operationen* fest, die mit den Objekten durchgeführt werden können.

*Aufgabe:*

1. Schauen Sie sich den Screencast „Methoden überschreiben.mp4“ und lösen Sie analog die neuen Anforderungen des Personenprogramms.
2. Ergänzen Sie Ihre Basisklasse (Person) im Personen-Programm um die Methode print(). Wie der Name sagt, soll diese Methode vom Objekt, bei dem Sie aufgerufen wird, den Namen, den Vornamen und die Personalnummer ausdrucken. Beim Ausdrucken eines Chefs soll auch die Abteilung und beim Lernenden das Lehrjahr ausgegeben werden.

Bauen Sie Ihr Testprogramm aus: Rufen Sie die Methode print() für Fachangestellte-, Chef- und Lernender-Objekte auf.

1. Ergänzen Sie nun Ihr Programm um eine Methode, mit der der Ferienanspruch eines Arbeitnehmers in Abhängigkeit seines Alters ausgerechnet werden kann. Fachangestellte haben einen Anspruch von 4 Wochen Ferien. Ab dem 50-sten Altersjahr haben sie eine zusätzliche Woche. Eine weitere Zusatzwoche erhalten sie nach dem 60-sten Altersjahr. Bei den Chefs findet der Wechsel zu den Zusatzferien jeweils 5 Jahre früher statt. Lernende haben 5 Wochen Ferien egal in welchem Alter.

* Implementieren Sie in der Basisklasse die abstrakte Methode:   
   **public** **abstract** **int** berechneFerien(**int** alter);
* Nehmen Sie alle daraufhin nötigen Änderungen vor und implementieren Sie auch in jeder abgeleiteten Klasse die Methode.
* Ändern Sie Ihr Testprogramm so ab, dass Sie nun ein Personen-Array erstellen mit mindestem einem Chef, einem Fachangestellten und einem Lernenden.

Person[] angestellte = **new** Person[4];

angestellte[0] = **new** Chef("Sattler","Beatrice",25,"Verkauf");

…

* Geben Sie in einer Schleife die Personaldaten mit der print()-Methode aus

p.print(); //For-Each-Schleife

oder:

angestellte[i].print(); //For-Schleife

und den Ferienanspruch. Geben Sie der Einfachheit halber einen festen Wert für das Alter beim Aufruf der Methode berechneFerien()mit. Wählen Sie ein Alter, das einen unterschiedlichen Ferienanspruch generiert.

System.***out***.println("Der Ferienanspruch beträgt: " + p.berechneFerien(46));

## Aufgabe 2: Methoden vererben im Programm Grafikeditor

*Einleitung:*

K 3.2

Wenden Sie sich nun den *Methoden* der grafischen Figuren zu.

*Aufgabe:*

Schreiben Sie in Ihrer Klasse Figur die Methode move(int deltaX, int delatY). Wie der Name sagt, soll diese Methode das Objekt, bei dem Sie aufgerufen wird um eine bestimmte Strecke verschieben.

Passen Sie Ihr Testprogramm an: Rufen Sie die Methode move(…) für Rechteck- und Linienobjekte auf. Stimmt das Resultat? (Überprüfung durch schrittweises Ausführen im Debugger).

Ergänzen Sie Ihre Klasse Linie um eine eigene Methode move(int deltaX, int delatY). Achten Sie darauf, dass Sie keinen redundanten Code produzieren! Überprüfen Sie das Resultat nochmals.

Wie heisst der Fachbegriff für das, was Sie mit der Methode move(…) in der Klasse Linie soeben gemacht haben (Buch **Seite 48**)?

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

## Aufgabe 3: Dynamische Bindung im Programm Grafikeditor

*Einleitung:*

Bleiben Sie bei den Methoden und untersuchen Sie den Zusammenhang mit *Referenzen*. Sie haben ja bereits mehrere Male gehört, und wohl auch erfahren, dass der Zugriff auf Objekte ausschliesslich über Referenzen geschieht.

*Aufgabe:*

Übernehmen Sie folgende Zeilen in Ihr Testprogramm:

Linie l = new Linie(10, 20, 30, 40);

l.move(10, 10);

Figur f = new Linie(10, 20, 30, 40);

f.move(10, 10);

Führen Sie diese Zeilen schrittweise im Debugger aus.

Was Stellen Sie fest? Begründen Sie.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

## Vertiefungs- und Ergänzungsaufgaben

Die folgenden Aufgaben dienen Ihrer individuellen Vertiefung. Sie können diese nach Belieben bearbeiten und dadurch auch Zusatzkompetenzen erlangen. Die Aufgaben sind bewusst offen formuliert und Sie werden für die Lösung im Web recherchieren müssen.

1. Implementieren Sie weitere Methoden für Ihre Figuren, z.B. strecken (Figur wird um einen bestimmten Faktor vergrössert oder verkleinert), spiegeln (horizontal oder vertikal), drehen (in 90 Grad Schritten oder feiner), etc. Überlegen Sie sich, auf welcher Ebene der Vererbungshierarchie die Methoden deklariert, implementiert und allenfalls überschrieben werden müssen.
2. Ein richtiger Grafikeditor bietet die Möglichkeit Zeichnungen in einer Datei zu speichern und auch wieder aus einer Datei zu laden. Hierzu braucht man ein geeignetes Dateiformat. Sie kennen zum Beispiel aus dem Modul 114 das SVG-Format. Andere Formate basieren auf XML, etc. Auch binäre Formate sind möglich.

K 3.5

Um die Aufgabe möglichst einfach zu halten sollen Sie ein eigenes, textbasiertes Format definieren, welches Ihnen erlaubt die Attribute Ihrer Rechtecke, Kreise, etc. auf einfache Weise in einer Textdatei zu speichern. Notieren Sie nachfolgend, welchen Inhalt die Textdatei erhalten soll, wenn Sie folgende Figur-Objekte speichern wollen:

Linie l = new Linie(10, 20, 30, 40);  
Rechteck r = new Rechteck(10, 20, 30, 40);  
Kreis k = new Kreis(10, 20, 30);

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Implementieren Sie nun eine Klasse FigurSaver mit einer save-Methode, welche beliebige Figur-Objekte in eine Textdatei speichern kann und dabei das von Ihnen definierte Format anwendet.  
  
Anschliessend können Sie noch die Klasse FigurLoader mit einer load-Methode implementieren. Wenn Sie alles richtig gemacht haben, können Sie nun mehrere Figuren in einer Datei speichern und anschliessend wieder von dieser Datei einlesen.

Weitere Informationen zum Schreiben und Lesen von Textdateien finden Sie im Buch, Abschnitt 8.7.