

Situation

Im Rahmen des Projektes implementieren Sie in Gruppen von jeweils vier Personen eine Java-Version des Spiels *Sokoban*. Die Aufgabenstellung stellt zunächst das zugrundeliegende Spiel vor.

- 5 Die Aufgabe kann in zwei „Ausbaustufen“ bearbeitet werden, die jeweils einen unterschiedlichen Funktionsumfang abdecken. Die minimale Ausbaustufe muss von allen Gruppen mindestens implementiert werden.

Das Spiel „Sokoban“

- 10 Sokoban (倉庫番 Sōkoban, japanisch „Lagerhausverwalter“) ist ein Computerspiel, das von Hiroyuki Imabayashi entwickelt und 1982 erstmals für verschiedene Computersysteme veröffentlicht wurde.


- 15 In einem einfachen Spielprinzip gilt es, mit einer Spielfigur alle Objekte – meistens sind es Kisten – nacheinander auf die dafür vorgesehenen Zielfelder zu bewegen, wobei es üblicherweise keine Vorgabe gibt, welches Objekt auf welches Zielfeld bewegt werden soll. Die Kisten können von der Spielfigur nur geschoben und nicht gezogen werden, ein Verschieben mehrerer Kisten zugleich ist nicht möglich. Üblicherweise sind die einzig möglichen Bewegungsrichtungen der Spielfigur nach oben, unten, rechts und links, sodass keine diagonalen Züge möglich sind.¹

20



Abbildung 1: Beispiel einer grafischen Umsetzung des Spiels Sokoban

¹ Vgl. <https://de.wikipedia.org/wiki/Sokoban>

Herr Gast	Kooperation von Friedrich-List-Schule und Heinrich-Emanuel-Merck-Schule in der Ausbildung der IT-Berufe	
Lernfeld 6: Programmiertechnik		

Ihre Aufgabe

- 25 Implementieren Sie eine lauffähige Java-Version des Spiels Sokoban, die mindestens der „minimalen Ausbaustufe“ entspricht.
Das fertige Spiel muss Ihrem Lehrer in Form eines Testats vorgestellt werden. Dazu muss die Dokumentation (etwas 2 DIN A4-Seiten) sowie der Source-Code und alle zum Übersetzen notwendigen Bibliotheken und Dateien rechtzeitig vor Ablauf der Einreichung **von einem**
- 30 **Gruppenmitglied bei Moodle hochgeladen werden.**
Das Testat besteht aus den folgenden zwei Bestandteilen:

Live-Test

- Ihr Lehrer startet das Spiel und testet, ob alles gemäß der Spezifikation funktioniert. Dazu
- 35 werden bestimmte, von Ihrem Lehrer vorgegebene Szenarien durchgespielt, aber auch zufällig „herumgespielt“.

Code-Review


- Ihr Lehrer sieht sich Ihren Quellcode sowie die Dokumentation an und stellt Fragen dazu.
- 40

Ablauf des Code-Reviews

- Im Hinblick auf das Code-Review sollen Sie auf gut verständlichen und dokumentierten Code achten.
- Ihr Lehrer wird einzelne Gruppenmitglieder seiner Wahl zu Teilen des Quelltextes befragen.
- 45 Daher sollte sich jedes Gruppenmitglied mit allen Codeteilen auskennen – Ihr Lehrer wählt aus, zu welchem Thema eine Frage gestellt wird und wer die Frage beantworten soll.

Dokumentation

- Neben dem Quelltext ist auch eine kurze Dokumentation abzugeben (etwa 2 DIN A4-Seiten).
- 50 Diese sollte auf die Fragen in den Aufgaben eingehen und kurz die in Ihrer Gruppe aufgetretenen Probleme beinhalten sowie Feedback zur Aufgabenstellung liefern.
Bitte reichen Sie Ihre Dokumentation **als eine PDF-Datei** ein.

Herr Gast	Kooperation von Friedrich-List-Schule und Heinrich-Emanuel-Merck-Schule in der Ausbildung der IT-Berufe	
Lernfeld 6: Programmiertechnik		

Minimale Ausbaustufe

- 55 Um das Projekt bestehen zu können, müssen **alle** nachfolgenden genannten Leistungen erbracht werden.

Pünktliche Abnahme – 0 Punkte

Der Quelltext mit Dokumentation wird rechtzeitig in Moodle hochgeladen.

60

Kompilierbares Java – 0 Punkte

Der Quelltext ist komplett in Java implementiert und kann separat ohne Fehlermeldungen neu kompiliert werden.

Nur eigener Code – 0 Punkte

Der Quelltext enthält **keinen** oder **nur genehmigte** fremde Codeteile, insbesondere keinen Code von anderen Projektgruppen oder aus dem Internet. Die Nutzung von fremdem Code kann nur durch Ihren Lehrer individuell für konkrete Codeteile oder Bibliotheken genehmigt werden.

70

Grafische Benutzerschnittstelle – 3 Punkte

Es gibt eine grafische Benutzerschnittstelle („GUI“) mit der man *Sokoban* spielen kann.

Spielmenü – 5 Punkte

- 75 Das Spiel *Sokoban* muss mit einem Spielmenü beginnen. Das Menü muss mindestens einen Button „*Neues Spiel*“ und „*Beenden*“ mit entsprechender Semantik enthalten.

Informieren Sie sich, wie Sie auf einem `JPanel` neue `JButtons` anlegen und anzeigen können.

Neues Spiel starten – 5 Punkte

Der Spieler soll aus dem Hauptmenü über das Klicken auf den Button „Neues Spiel“ ein neues Spiel starten können.

Informieren Sie sich, wie Sie mit Hilfe eines `ActionListener` Interface auf das Klicken eines `JButton` reagieren können.

85

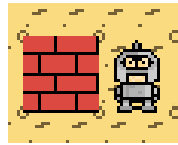
Spiel Beenden – 5 Punkte

Der Spieler soll aus dem Hauptmenü über das Klicken auf den Button „Beenden“ das ganze Programm beenden können.

Lernfeld 6: Programmieretechnik**90 Implementieren des Klasse Actor – 5 Punkte**

Die Klasse `Actor` stellt in unserem Spiel die Basisklasse für alle spielrelevanten Objekte da, die später im Spiel gezeichnet werden.

Die Klasse selbst umfasst mehrere Getter- und Setter-Methoden und fünf Methoden, die überprüfen ob eine Kollision mit einem anderen Actor entstehen könnte.



95

Abbildung 2: Beispiel einer möglichen Kollision

Ein Aufruf der Methode `isLeftCollision(Actor actor)` des Spielers (der Roboter) würde `true` zurückgeben.

100

`isLeftCollision(Actor actor)` Prüft ob es links von der Spielfigur eine Kollision gibt

`isRightCollision(Actor actor)` Prüft ob es rechts von der Spielfigur eine Kollision gibt

`isTopCollision(Actor actor)` Prüft ob es oberhalb von der Spielfigur eine Kollision gibt

`isDownCollision(Actor actor)` Prüft ob es unterhalb von der Spielfigur eine Kollision gibt

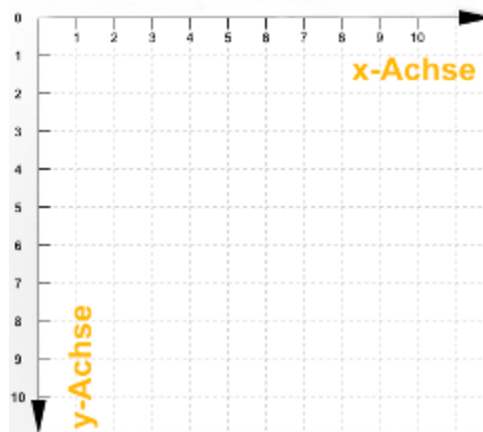
`isOnTopOfEachOther(Actor actor)` Prüft ob es auf der Spielfigur zu eine Kollision kommt

105

Das Attribut `SIZE` vom Type `int` gibt Größe der Sprites (Bilder im Spiel) an und kann von Ihnen auf 25 gesetzt werden, da alle Sprites eine Größe von 25x25 Pixeln besitzen.

Tipp:

Ein Koordinaten-System beim Programmieren beginnt immer oben Links in der Ecke.



110

Abbildung 3: Koordinatensystem beim Programmieren**Implementieren des Klasse Floor / Wall / Spot – 5 Punkte**

Die Klasse `Floor / Wall / Spot` erbt ihre Attribute und Methoden von der Elternklasse `Actor` und stellt den Boden/ die Wände / das Zielfeld in der Spielwelt dar.

115

Die Klasse besitzt eine private Methode `init()` welche im Konstruktor der Klasse aufgerufen wird und das entsprechende Sprite des Bodes lädt und als Attribut setzt. Informieren Sie sich hierbei wie Sie mittels der Klasse `ImageIcon` ein `.png` laden und dieses in den Datentyp `Image` überführen.

120 Implementieren des Klasse Box – 5 Punkte

Die Klasse `Box` erbt ihre Attribute und Methoden von der Elternklasse `Actor` und stellt in der Spielwelt die Kisten dar, die von der Spielfigur verschoben werden können.

Die Klasse besitzt drei private Attribute.

125 `boolean isHighlighted` gibt an ob die Kiste auf einem Spot (Zielfeld) steht

`Image highlighted` ist das Sprite das angezeigt wird, wenn die Kiste auf einem Spot steht

`Image notHighlighted` ist das Sprite das angezeigt wird, wenn die Kiste **nicht** auf einem Spot steht

130 Des Weiteren besitzt die Klasse vier öffentliche und eine private Methode. Die private Methode `init()` wird über den Konstruktor der Klasse aufgerufen und lädt die entsprechenden Sprites und setzt diese als Attribute in der Klasse.

Die öffentlichen Methoden implementieren die folgenden Funktionen:

135

`move(int x, int y)` verschiebt die Kiste von der aktuellen Position aus um die übergebenen Parameter

`boolean isHighlighted()` gibt `true` zurück wenn die Kiste auf einem Spot steht, sonst `false`

`setHighlighted()` setzt das Attribut `isHighlighted` auf `true` und tauscht das Sprite aus

140

`unHighlighted()` setzt das Attribut `isHighlighted` auf `false` und tausch das Sprite aus

Implementierung der Klasse Player - 5 Punkte

Die Klasse `Player` erbt ihre Attribute und Methoden von der Elternklasse `Actor` und stellt in der Spielwelt die Spielfigur dar.

145 Die Klasse besitzt vier private Attribute.

`int TOP = 0` steht für die Richtung nach oben

`int RIGHT = 1` steht für die Richtung nach rechts

`int DOWN = 2` steht für die Richtung herunter

150

`int LEFT = 3` steht für die Richtung links

Die private Methode `init()` wird über den Konstruktor der Klasse aufgerufen und lädt das Sprite der Spielfigur. Die beiden öffentlichen Methoden implementieren die folgende Funktionalität.

155

`move(int x, int y)` verschiebt die Spielfigur von der aktuellen Position aus um die übergebenen Parameter

`setPlayerDirection(int direction)` lädt und setzt entsprechend des übergebenen Wertes das richtige Sprite für die Blickrichtung der Spielfigur. Wird z.B. der Wert 1 der Methode übergeben, wird das Sprite geladen und gesetzt, in dem die Spielfigur nach rechts blickt.

160

Informationen zu den weiteren Aufgaben

Die nachfolgenden Aufgaben beziehen sich alle auf die Klasse `GameBoard`.

165 Interfaces in Java – 5 Punkte

Informieren Sie sich welche Funktion und Zweck ein Interface in Java einnimmt und beschreiben Sie dies in Ihrer Dokumentation.

Parsen deiner Karte – 10 Punkte

170 Die Methode `initGameBoard()` übernimmt in unserer Anwendung das Parsen² einer Karte. Eine Karte wird in Sokoban als String mit den folgenden Symbolen dargestellt:

	#	Wall
	@	Player
175	*	Spot
	.	Box
	Leerzeichen	Floor
	\n	Zeilenumbruch

180 Beispiel einer Karte als String-Repräsentation in Java:

```

private String level =
    "#####\n" +
    "#           #\n" +
    185 "#  .  #  #\n" +
    "#  **  #\n" +
    "#      .  #\n" +
    "#@      #\n" +
    "#####";

```

190

Die Methode geht beim Einlesen eines Stings vollgendermaßen vor:

1. Initialisierung einer `ArrayList<>()` für die jeweiligen Datentypen `Wall`, `Box`, `Spot`, `Floor`
2. Durchlaufen einer Schleife, die genauso lange läuft wie der String lang ist
 - 195 a. Auslesen eines Zeichens aus dem String³
 - b. Entscheiden welches Zeichen gerade ausgelesen wurde
 - c. Erzeugen eines entsprechenden Objektes
 - d. Hinzufügen des Objektes in die entsprechende `ArrayList<>()`

² Ein Parser ist ein Computerprogramm, das in der Informatik für die Zerlegung und Umwandlung einer Eingabe, in ein für die Weiterverarbeitung geeigneteres Format, zuständig ist. Quelle: <https://de.wikipedia.org/wiki/Parser>

³ Tipp :Schauen Sie sich `charAt()` für Strings an

200 **Zeichnen der Spielwelt – 10 Punkte**

Die Methode `paintWorld(Graphics g)` implementiert im Projekt das Zeichnen der gesamten Spielwelt und wird über die geerbte⁴ Methode `paintComponent(Graphics g)` aufgerufen.

205 Die vorgegebenen Methode `paintComponent(Graphics g)` wird immer dann automatisch aufgerufen, wenn Sie die Methode `repaint()` in Ihrem Quellcode aufrufen.

Implementierung der Methode `paintComponent(Graphics g)`:

```
210  @Override // überschreiben der Elternmethode  
    public void paintComponent(Graphics g){  
        super.paintComponent(g); // Aufruf der Elternmethode  
        paintWorld(g); // Aufruf von paintWorld(Graphics g)  
    }
```

215 Bevor Sie mit der Implementierung der Methode beginnen, informieren Sie sich und begründen Sie in Ihrer Dokumentation wie der folgende Quellcode funktioniert und welche Funktion die Methode `drawImage()` der Klasse `Graphics` implementiert:

```
220  ArrayList<Actor> eineListe = new ArrayList<>();  
    eineListe.add(new Wall(0,0));  
    eineListe.add(new Floor(25,0));  
    eineListe.add(new Floor(50,0));  
    eineListe.add(new Player(25,25));  
    eineListe.add(new Box(25,50));
```

225 Gehen Sie bei Ihrer Begründung darauf ein, welche Rolle die Vererbung in Java bei dem Quellcodebeispiel einnimmt.

Die konkrete Implementierung der Methode `paintWorld(Graphics g)`:

- 230
1. Hinzufügen aller Spielelemente in eine `ArrayList` vom Typ `Actor`
 2. Iterieren über die gesamte `ArrayList`
 - a. Zeichnen eines Objektes aus der `ArrayList`

⁴ Die Klasse `GameBoard` erbt ihre Funktionalität von der Klasse `JPanel`

Implementieren des KeyListener Interfaces – 5 Punkte

- 235 Informieren Sie sich und beschreiben Sie in Ihrer Dokumentation wie Tastatureingaben eines Spielers mithilfe des `KeyListener` Interfaces in Java implementiert werden kann.
Die Implementierung des `KeyListener` Interfaces realisiert im Spiel die Steuerung der Spielfigur.
- 240 Drückt der Spieler beispielsweise auf der Tastatur die Pfeiltaste nach oben, soll sich die Spielfigur innerhalb des Spiels ein Feld nach oben bewegen.

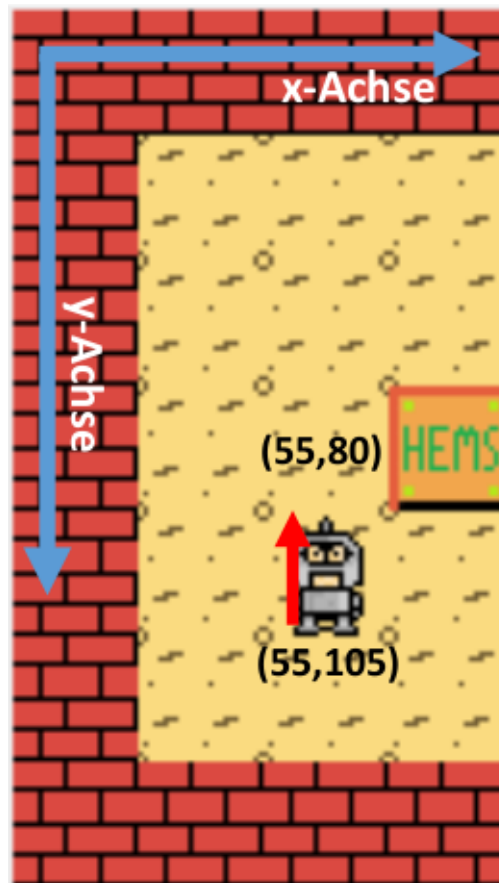


Abbildung 4: Beispiel einer Bewegung der Spielfigur

245


Erweiterte Ausbaustufe**Abfangen einer Kollision mit einer Wand – 10 Punkte**

Die Methode `boolean isWallCollision(Actor actor, int direction)` gibt `true` zurück, wenn es zu einer Kollision mit einer Wand im Spiel kommt.

250

Beispielsweise würde der Aufruf `isWallCollision(player, 2)` in *Abbildung 4* zur Rückgabe `false` führen, da eine Bewegung nach unten⁵ zu keiner Kollision mit einer Wand führt.

⁵ `int DOWN = 2` steht für die Richtung herunter

Herr Gast	Kooperation von Friedrich-List-Schule und Heinrich-Emanuel-Merck-Schule in der Ausbildung der IT-Berufe	
Lernfeld 6: Programmiertechnik		

Abfangen einer Kollision mit einer Kiste – 10 Punkte

255 Mit Hilfe der Methode `boolean isBoxCollision(int direction)` wird sichergestellt,
dass wenn ein Spieler mit einer Kiste kollidiert, die Kiste in die entsprechende Laufrichtung
des Spielers verschoben werden kann. Hierfür muss überprüft werden, dass die Kiste in ihrer
Bewegung nicht in eine Wand geschoben wird oder in eine andere Kiste.
Die Methode gibt `true` zurück, wenn es zu einer ungültigen Kollision kommt, andernfalls gibt
260 sie `false` zurück.

Korrekte Darstellung der Sprites – 5 Punkte

Die Sprites der Spielfigur werden korrekt bei der Bewegung der Spielfigur wiedergegeben.
Dies bedeutet, läuft die Spielfigur nach links, zeigt das Gesicht der Spielfigur in dieselbe
265 Richtung.

Überprüfen ob ein Spiel gewonnen wurde – 10 Punkte

Implementieren Sie einen Algorithmus, der nach jedem Zug der Spielfigur überprüft ob das
Spiel vom Spieler gewonnen wurde.
270 Ein Spiel ist genau dann gewonnen, wenn alle Zielfelder eines Levels mit Kisten abgedeckt
sind. Daraufhin erhält der Spieler eine Meldung, dass er das Spiel gewonnen hat.

Ideen für mögliche Erweiterungen

- Neustart eines Levels
- 275 • Implementierung eines Menüs nachdem ein Spiel gewonnen wurde
- Das Einlesen von Karten aus einer Datei
- Karteneditor
- Überraschen Sie mich mit Ihren Ideen