Karlsruher Institut für Technologie

Prof. Dr.-Ing. Anne Koziolek
Institute of Information Security and Dependability (KASTEL)
https://sdq.kastel.kit.edu/wiki/Programmieren
programmieren-vorlesung@cs.kit.edu

Programmieren – Wintersemester 2025/26

Übungsblatt 3
Version 1.0

20 Punkte

Ausgabe:	03.12.2025, ca. 12:00 Uhr
Abgabestart:	10.12.2025, 12:00 Uhr
Abgabefrist:	18.12.2025, 06:00 Uhr

Plagiarismus

Es werden nur selbstständig angefertigte Lösungen akzeptiert. Das Einreichen fremder Lösungen, seien es auch nur teilweise Lösungen von Dritten, aus Büchern, dem Internet oder anderen Quellen, ist ein Täuschungsversuch und führt jederzeit (auch nachträglich) zur Bewertung „nicht bestanden“. Ausdrücklich ausgenommen hiervon sind Quelltextschnipsel von den Vorlesungsfolien und aus den Lösungsvorschlägen des Übungsbetriebes in diesem Semester. Alle benutzten Hilfsmittel müssen vollständig und genau angegeben werden. Alles, was aus Arbeiten anderer unverändert oder mit Abänderungen entnommen wurde, muss deutlich kenntlich gemacht werden. Beachten Sie darüber hinaus die Richtlinien der Fakultät zum Verwenden von Generativer KI [1].

Studierende, die den ordnungsgemäßen Ablauf einer Erfolgskontrolle stören, können von der Erbringung der Erfolgskontrolle ausgeschlossen werden. Ebenso stellt unter anderem die Weitergabe von Teilen von Testfällen oder Lösungen bereits eine Störung des ordnungsgemäßen Ablaufs dar. Auch diese Art von Störungen können ausdrücklich jederzeit zum Ausschluss der Erfolgskontrolle führen. Dies bedeutet ausdrücklich, dass auch nachträglich die Punktzahl reduziert werden kann.

Kommunikation und aktuelle Informationen

In unseren *FAQs*[2] finden Sie einen Überblick über häufig gestellte Fragen und die entsprechenden Antworten zum Modul „Programmieren“. Bitte lesen Sie diese sorgfältig durch, noch bevor Sie Fragen stellen, und überprüfen Sie diese regelmäßig und eigenverantwortlich auf Änderungen. Beachten Sie zudem die Hinweise im Wiki[3].

In den *ILIAS-Foren* oder auf *Artemis* veröffentlichen wir gelegentlich wichtige Neuigkeiten. Eventuelle Korrekturen von Aufgabenstellungen werden ebenso auf diesem Weg bekannt gemacht. Das aktive Beobachten der Foren wird daher vorausgesetzt.

[1] https://www.informatik.kit.edu/faq-wiki/doku.php?id=generative_ki
[2] https://sdq.kastel.kit.edu/wiki/Programmieren/FAQ
[3] https://sdq.kastel.kit.edu/programmieren/

Überprüfen Sie das Postfach Ihrer *KIT-Mailadresse* regelmäßig auf neue E-Mails. Sie erhalten unter anderem eine Zusammenfassung der Korrektur per E-Mail an diese Adresse. Alle Anmerkungen können Sie anschließend im Online-Einreichungssystem[4] einsehen.

Bearbeitungshinweise

Bitte beachten Sie, dass das erfolgreiche Bestehen der verpflichtenden Tests für eine erfolgreiche Abgabe von Übungsblatt 3 notwendig ist. Ihre Abgabe wird automatisch mit null Punkten bewertet, falls eine der nachfolgenden Regeln verletzt ist. Sie müssen zuerst die verpflichtenden Tests bestehen, bevor die anderen Tests ausgewertet werden können. Planen Sie entsprechend Zeit für Ihren ersten Abgabeversuch ein.

- Achten Sie auf fehlerfrei kompilierenden Programmcode.
- Verwenden Sie ausschließlich *Java SE 21*.
- Sofern in einer Aufgabe nicht ausdrücklich anders angegeben, verwenden Sie keine Elemente der Java-Bibliotheken. Ausgenommen ist die Klasse `java.util.Scanner` und alle Elemente aus den folgenden Paketen: `java.lang`, `java.io`, `java.util`, `java.util.regex` und `java.nio.file`.
- Achten Sie darauf, nicht zu lange Zeilen, Methoden und Dateien zu erstellen. Sie müssen bei Ihren Lösungen eine maximale Zeilenbreite von 140 Zeichen einhalten.
- Halten Sie alle Whitespace-Regeln ein.
- Halten Sie alle Regeln zu Variablen-, Methoden- und Paketbenennung ein.
- Wählen Sie geeignete Sichtbarkeiten für Ihre Klassen, Methoden und Attribute.
- Nutzen Sie nicht das `default`-Package.
- `System.exit()`, `Runtime.exit()` oder ähnliches dürfen nicht verwendet werden.
- Halten Sie die Regeln zur Javadoc-Dokumentation ein.
- Halten Sie auch alle anderen Checkstyle-Regeln ein.

Diese folgenden Bearbeitungshinweise sind relevant für die Bewertung Ihrer Abgabe. Dennoch wird Ihre Abgabe durch das Abgabesystem *nicht* automatisch mit null Punkten bewertet, falls eine der nachfolgenden Regeln verletzt ist. Orientieren Sie sich zudem an den Bewertungskriterien im Wiki.

- Fügen Sie außer Ihrem u-Kürzel keine weiteren persönlichen Daten zu Ihren Abgaben hinzu.
- Beachten Sie, dass Ihre Abgaben sowohl in Bezug auf objektorientierte Modellierung als auch Funktionalität bewertet werden. Halten Sie die Hinweise zur Modellierung im Wiki ein.
- Programmcode muss in englischer Sprache verfasst sein.
- Kommentieren Sie Ihren Code angemessen: So viel wie nötig, so wenig wie möglich.

[4] `https://artemis.cs.kit.edu/`

- Die Kommentare sollen einheitlich in englischer oder deutscher Sprache verfasst werden.
- Geben Sie im Javadoc-Autoren-Tag nur Ihr u-Kürzel an.
- Wählen Sie aussagekräftige Namen für alle Ihre Bezeichner.

Checkstyle

Das Online-Einreichungssystem überprüft Ihre Quelltexte während der Abgabe automatisiert auf die Einhaltung der Checkstyle-Regeln. Es gibt speziell markierte Regeln, bei denen das Online-Einreichungssystem die Abgabe mit null Punkten bewertet, da diese Regeln verpflichtend einzuhalten sind. Andere Regelverletzungen können zu Punktabzug führen. Sie können und sollten Ihre Quelltexte bereits während der Entwicklung auf die Regeleinhaltung überprüfen. Das Programmieren-Wiki beschreibt, wie Checkstyle verwendet werden kann.

Abgabehinweise

Die Abgabe im Online-Einreichungssystem wird am 10.12.2025, 12:00 Uhr, freigeschaltet. Achten Sie unbedingt darauf, Ihre Dateien im Einreichungssystem bei der richtigen Aufgabe vor Ablauf der Abgabefrist am 18.12.2025, 06:00 Uhr, hochzuladen. Beginnen Sie frühzeitig mit dem Einreichen, um Ihre Lösung dahingehend zu testen, und verwenden Sie das Forum, um eventuelle Unklarheiten zu klären. Falls Sie mit Git abgeben, *muss immer* auf den `main`-Branch gepusht werden.

- Geben Sie online Ihre `*.java`-Dateien zur Aufgabe A in Einzelarbeit mit der entsprechenden Ordnerstruktur im zugehörigen Verzeichnis ab.
- Geben Sie online Ihre `*.java`-Dateien zur Aufgabe B in Einzelarbeit mit der entsprechenden Ordnerstruktur im zugehörigen Verzeichnis ab.

Wiederverwendung von Lösungen

Falls Sie für die Bearbeitung der Abschlussaufgaben oder Übungsblätter Beispiellösungen aus diesem Semester wiederverwenden, *müssen* Sie in die entsprechenden Klassen "Programmieren-Team" ins Autor-Tag eintragen. Dies ist nötig, um die Checkstyle-Kriterien zu erfüllen.

Aufgabe A: Vier bis zwei gewinnt (14 Punkte)

Vier gewinnt[5] (englisch: Connect Four oder Captain's mistress) ist ein Strategiespiel mit dem Ziel, als erster Spieler vier der eigenen Spielsteine in eine Linie zu bringen. Vier gewinnt wurde von Howard Wexler mit Ideen von Ned Strongin entwickelt und 1974 veröffentlicht.

Das Spiel wird auf einem senkrecht stehenden hohlen Spielbrett gespielt, in das die Spieler abwechselnd einen ihrer Spielsteine fallen lassen. Das Spielbrett besteht aus sieben Spalten (senkrecht) und sechs Reihen (waagerecht). Wenn ein Spieler einen Spielstein in eine Spalte fallen lässt, besetzt dieser den untersten freien Platz der Spalte. Gewinner ist, wer vier oder mehr seiner Spielsteine ununterbrochen in eine waagerechte, senkrechte oder diagonale Linie bringt. Das Spiel endet unentschieden, wenn das Spielbrett komplett gefüllt ist, ohne dass ein Spieler eine Viererlinie gebildet hat.

Abbildung A.1 zeigt ein Beispiel für ein vier gewinnt Spielfeld. Während Spieler 2 hier durch eine horizontale Viererlinie gewonnen hat, kann ein Spieler auch durch vertikale oder diagonale Viererlinien gewinnen.

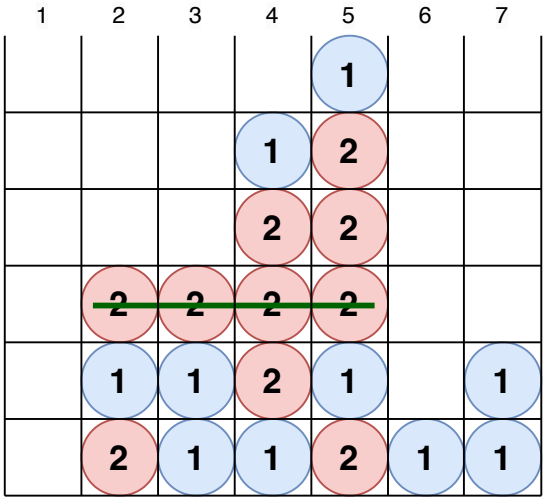

Abbildung A.1: Vier gewinnt Spielfeld nach Spielende durch den Sieg von Spieler 2.

In dieser Aufgabe sind alle Spalten des Spielfeldes eindeutig von 1 bis 7 von links nach rechts durchnummeriert, so wie in Abbildung A.1 dargestellt ist.

Im Gegensatz zur klassischen Variante, bei der nur zwei Spieler möglich sind, implementieren wir in dieser Aufgabe eine Variante, bei der 1 bis 6 Spieler möglich sind. Hierbei lassen alle Spieler der Reihe nach ihre Spielsteine fallen, die Siegbedingung hängt jedoch von der Spielerzahl ab. Dabei variiert die Mindestanzahl von Steinen, die in eine Linie gelegt werden müssen, um zu gewinnen. Für n Spieler bestimmt sich die Steinanzahl s wie folgt:

$$s = \min(4, \lceil \frac{9}{n} \rceil)$$

Dabei ist 4 die Obergrenze und 9 der Siegbedingungs-Dividend.

[5] `https://de.wikipedia.org/wiki/Vier_gewinnt`

A.1 Programmablauf und Interaktion

Im Folgenden wird der Spielablauf erklärt. Zudem wird erläutert, wie Spieler mit dem Programm interagieren.

A.1.1 Ausgabe des Spielfeldes

Das Spielfeld wird immer zeilenweise ausgegeben. Jede Zeile wird durch einen Zeilenumbruch getrennt. Besetzte Felder werden durch das Zeichen des Spielers dargestellt, leere Felder durch Leerzeichen. Die Spielerzeichen sind frei wählbare Zeichen vom Typ `char`. Alle Felder haben jeweils rechts und links ein Trennzeichen, welches immer der durchgehende[6] senkrechte Strich `'|'` ist. Folglich enthält jede Zeile des Spielfeldes 8 Trennzeichen.

Das leere Spielfeld besteht aus den folgenden 6 Zeilen (links ohne und rechts mit sichtbaren Leerzeichen):

```
1 | | | | | | | | |
2 | | | | | | | | |
3 | | | | | | | | |
4 | | | | | | | | |
5 | | | | | | | | |
6 | | | | | | | | |
```

```
1 |␣|␣|␣|␣|␣|␣|␣|␣|
2 |␣|␣|␣|␣|␣|␣|␣|␣|
3 |␣|␣|␣|␣|␣|␣|␣|␣|
4 |␣|␣|␣|␣|␣|␣|␣|␣|
5 |␣|␣|␣|␣|␣|␣|␣|␣|
6 |␣|␣|␣|␣|␣|␣|␣|␣|
```

Das Spielfeld in Abbildung A.1 besteht aus den folgenden 6 Zeilen (links ohne und rechts mit sichtbaren Leerzeichen):

```
1 | | | | |1| | |
2 | | | |1|2| | |
3 | | | |2|2| | |
4 | |2|2|2|2| | |
5 | |1|1|2|1| |1|
6 | |2|1|1|2|1|1|
```

```
1 |␣|␣|␣|␣|1|␣|␣|
2 |␣|␣|␣|1|2|␣|␣|
3 |␣|␣|␣|2|2|␣|␣|
4 |␣|2|2|2|2|␣|␣|
5 |␣|1|1|2|1|␣|1|
6 |␣|2|1|1|2|1|1|
```

A.1.2 Spielstart

Das Spiel wird über eine `main`-Methode gestartet. Die Spielerzeichen werden über die Argumente der `main`-Methode übergeben. Die Anzahl der Spielerzeichen bestimmt zudem die Anzahl der Spieler. Die Reihenfolge der Argumente bestimmt zudem die Nummerierung der Spieler. Werden keine Argumente übergeben, wird ein Standardspiel mit zwei Spielern (Spieler 1 mit `'x'`, Spieler 2 mit `'o'`) gestartet.

In folgenden Fällen wird eine Fehlermeldung ausgegeben:

[6] `https://de.wikipedia.org/wiki/Senkrechter_Strich`

- Falls mehr als 6 Argumente übergeben werden.
- Falls ein Argument mehr als ein einzelnes Zeichen enthält.
- Falls zwei oder mehr Spielerzeichen identisch sind.

Fehlermeldungen beginnen immer mit der Zeichenkette `"ERROR: "` (Leerzeichen beachten!). Jede Fehlermeldungen sollte eine aussagekräftige Beschreibung enthalten. Nach einer Fehlermeldung wird das Programm direkt beendet.

Bevor der erste Spielzug beginnt, wird das leere Spielfeld ausgeben. Ihr Programm nimmt per Kommandozeileneingabe abwechselnd die Züge der Spieler entgegen. Es beginnt immer zuerst der Spieler 1.

A.1.3 Spielzug

Die Züge sind beginnend mit 1 lückenlos und fortlaufend durchnummeriert. Ein Spielzug enthält die folgenden Schritte (in dieser Reihenfolge):

1. Zu Beginn eines Spielzugs wird zuerst die Zugnummer und die Spielernummer ausgegeben, also zum Beispiel für Zug 3 und Spieler 1 wie folgt: `"3. Zug, Spieler 1:"`.
2. Danach wird der Zug des Spielers eingelesen. Der Spieler gibt über die Kommandozeile eine Ganzzahl zwischen 1 und 7 an, die die Spalte referenziert (siehe Abbildung A.1), in welche er seinen Spielstein einwerfen will. Ist diese Spalte voll oder die Eingabe keine Ganzzahl zwischen 1 und 7, wird eine Fehlermeldung ausgegeben und erneut Schritt 1 und 2 ausgeführt. Eine Ausnahme bildet hierbei die Eingabe der Zeichenkette `"quit"`, welche das Programm direkt und ohne weitere Ausgaben beendet[7].
3. Der Spielstein des Spielers fällt in der gewählten Spalte auf das unterste freie Feld.
4. Zuletzt wird das aktuelle Spielfeld ausgegeben. Hierbei soll der soeben platzierte Spielstein bereits Teil des Spielfeldes sein.

A.1.4 Spielende

Wenn ein Spieler passend zur Siegbedingung genug Steine in einer Linie platziert hat, endet das Spiel. Auf der Kommandozeile wird `Sieger: Spieler <NR>` ausgegeben, wobei `<NR>` ein Platzhalter für die Spielernummer des Siegers ist. Ist das Spielfeld voll und kein Spieler hat gewonnen, endet das Spiel und `Unentschieden!` wird ausgegeben.

A.2 Umsetzung

Im Folgenden wird auf die Umsetzung des Spiels eingegangen. Die folgenden Vorgaben lassen Ihnen absichtlich Freiheiten bei der genauen Umsetzung. Sie können frei nach eigenem Ermessen Klassen, Methoden und Felder hinzufügen, um zum Beispiel Spieler, Spielsteine oder andere Konzepte zu modellieren.

[7] Verwenden Sie hierzu *nicht* `System.exit()`.

A.2.1 Spielfeld

Erstellen Sie eine Klasse für das Spielfeld. Wie bereits erklärt, enthält das Spielfeld alle Felder in einem rechteckigen Spielfeld. Stellen Sie sicher, dass diese Klasse erweiterbar ist und nicht nur für die spezifischen Dimensionen verwendbar ist. Die Klasse sollte mindestens die folgenden Funktionalitäten bereitstellen:

- Eine Methode, welche überprüft, ob ein Feld frei oder besetzt ist.
- Eine Methode, welche überprüft, ob ein Spieler gewonnen hat.
- Eine Methode, welche überprüft, ob das Spielfeld voll ist.
- Eine Methode, welche den Spielstein eines Spielers in eine Spalte einwirft.

A.2.2 Ein- und Ausgabe

Erstellen Sie eine Klasse für die Ein- und Ausgabe. Stellen Sie sicher, dass alle Ein- oder Ausgaben in dieser Klasse stattfinden. Abwechselnd sollen, wie in Abschnitt A.1 erwähnt, die Eingaben der Spieler eingelesen werden und wie bereits beschrieben verarbeitet werden. Anschließend sollen die nötigen Ausgaben erfolgen.

Dazu soll mit `System.out`[8] die Standardausgabe verwendet werden. Nutzen Sie für die Eingabe wie bisher die Klasse `Scanner`[9]. Um Eingaben einfach von der Konsole zu lesen, verwenden Sie die `nextLine`-Methode, welche eine durch die Eingabe eines Zeilenumbruchs beendete Textzeile von der Konsole einliest. Nachdem keine Textzeilen mehr eingelesen werden sollen, muss der Scanner immer mit der `close`-Methode geschlossen werden.

A.2.3 Beispielablauf

Der Beispielablauf zeigt zwei verschiedene Spiele nach Start des Programmes. Die Zeilennummern und die Trennlinie sind kein Bestandteil der Benutzerschnittstelle, sie dienen lediglich zur Orientierung für die gegebene Beispielinteraktion. Nutzereingaben sind mit dem Zeichen `>` gekennzeichnet, die Eingabe ist jedoch nur eine Ganzzahl. Die Zeichen `%>` (Prozent-Zeichen und Größer-Zeichen gefolgt von einem Leerzeichen) stellt die Kommandozeile dar. Der Name des Programms dient nur als Beispiel und ist nicht vorgeschrieben.

[8] `https://docs.oracle.com/en/java/javase/21/docs/api/java.base/java/lang/System.html#out`
[9] `https://docs.oracle.com/en/java/javase/21/docs/api/java.base/java/util/Scanner.html`

Beispielinteraktion

```
%> java ConnectFour #
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
1. Zug, Spieler 1:
> 1
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
|#| | | | | | |
2. Zug, Spieler 1:
> 2
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
|#|#| | | | | |
3. Zug, Spieler 1:
> 3
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
|#|#|#| | | | |
4. Zug, Spieler 1:
> 4
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
|#|#|#|#| | | |
Sieger: Spieler 1
```

Beispielinteraktion

```
%> java ConnectFour
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
1. Zug, Spieler 1:
> 4
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | |x| | | |
2. Zug, Spieler 2:
> 3
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | |o|x| | | |
3. Zug, Spieler 1:
> 5
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | |o|x|x| | |
4. Zug, Spieler 2:
> 4
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | |o| | | |
| | |o|x|x| | |
5. Zug, Spieler 1:
> quit
```

Aufgabe B: Wörterbuch (6 Punkte)

Ziel dieser Aufgabe ist es, eine geeignete Datenstruktur zu finden und zu implementieren, welche ein Wörterbuch abspeichern und verwalten kann. Dem Programm wird als Kommandozeilenargument ein Pfad zu einer Textdatei übergeben, welche eine Liste von Wörterpaaren enthält, welche in Abschnitt B.1 beschrieben sind. Diese Liste wird eingelesen und soll in einer geeigneten Datenstruktur abgespeichert werden. Alle eingelesenen Wörter aus der „Startsprache“ sollen alphabetisch sortiert[10] gespeichert werden.

In dieser Aufgabe wird immer eine alphabetische Sortierung verwendet, welche mit der in deutschen Wörterbüchern, z.B. Duden oder Wahrig, vergleichbar ist. Hierbei basiert die Sortierung auf der Buchstabenfolge des deutschen Alphabets. Bei gleicher Buchstabenfolge werden Kleinbuchstaben vor Großbuchstaben gesetzt. Die Umlaute werden mit den Grundbuchstaben gleichgesetzt, z.B. sind `Ä` und `A` gleich. Nur wenn die Schreibweise ansonsten gleich ist, steht der Grundbuchbuchstabe an erster Stelle, z.B. `Buhl` vor `Bühl`. Der Eszett wird wie `ss` geordnet, nur wenn die Schreibweise ansonsten gleich ist, steht `ss` vor `ß`.

Nach dem Einlesen gibt es die Möglichkeit über eine interaktive Kommandozeilenschnittstelle Befehle, welche in Abschnitt B.2.2 beschrieben sind, auszuführen. Dazu gehören unter anderem das Hinzufügen oder Entfernen von Wörtern und Übersetzungen sowie das Übersetzen von Wörtern und einfachen Sätzen.

B.1 Wörterlistendatei

Ihr Programm nimmt als einziges Kommandozeilenargument einen Pfad auf eine Textdatei entgegen. Sie können zum Einlesen der Datei `Files.readAllLines(Path.of(fileName))` aus dem Paket `java.nio.file` nutzen. Hierbei können Sie nicht davon ausgehen, dass die gegebene Eingabe der gegebenen Spezifikationen entspricht. Fehlerhafte Textdateien haben keine weiteren Auswirkungen auf das Programm, es wird lediglich eine entsprechende Fehlermeldung angezeigt. Änderungen am Wörterbuch müssen nicht in die Wörterbuchdatei zurückübertragen werden, d.h. die Textdatei darf nicht modifiziert werden.

Die Wörterlistendatei enthält eine oder mehrere Zeilen. Jede Zeile beschreibt ein Wörterpaar. Ein Wörterpaar besteht aus exakt zwei Wörtern, die durch einen Separator voneinander getrennt sind. Jedes Wort besteht aus einer Sequenz einem oder mehreren Buchstaben des deutschen Alphabets[11]. Der Separator besteht aus den drei Zeichen mit dem ASCII-Hexadezimalcodes `20 2D 20`. Links neben dem Separator steht das Wort aus der „Startsprache“ und rechts neben dem Separator steht das Wort aus der „Zielsprache“. Einem Wort aus der Startsprache können mehrere Wörter aus der Zielsprache zugewiesen werden. Eine beispielhafte Wörterlistendatei könnte so aussehen:

[10] `https://de.wikipedia.org/wiki/Alphabetische_Sortierung`
[11] `https://de.wikipedia.org/wiki/Deutsches_Alphabet`

WordList.txt

```
1  Der - The
2  Die - The
3  Das - The
4  Haus - House
5  klein - little
6  klein - tiny
7  klein - small
8  blau - blue
9  Meer - sea
10 Meer - ocean
11 ist - is
```

B.2 Interaktive Benutzerschnittstelle

Nach dem Start nimmt das Programm über die Konsole Befehle entgegen. Nach der Verarbeitung eines Befehls wartet das Programm auf weitere Eingaben, bis es durch Eingabe des `quit`-Befehls beendet wird. Die Eingabe und die Ausführung der Befehle dürfen nicht gegen die definierten Spezifikationen verstoßen. Bei einem Verstoß muss immer eine aussagekräftige Fehlermeldung ausgegeben, anschließend wartet das Programm regulär auf weitere Eingaben. Beachten Sie, dass nicht bei jeder erfolgreichen Interaktion automatische eine Ausgabe erfolgen muss.

B.2.1 Terminalsymbole

B.2.1.1 `<word> <translated_word>` Wörter aus der Start- bzw. der Zielsprache.

B.2.1.2 `<letter>` Ein Buchstabe aus dem deutschen Alphabet.

B.2.1.3 `<sentence>` Satz bestehend aus mehreren Wörtern die durch Leerzeichen voneinander getrennt sind. Es sind keine Satzzeichen erlaubt.

B.2.2 Befehle

B.2.2.1 `add <word> <translated_word>` Fügt das eingegebene Wortpaar dem Wörterbuch hinzu, wenn dieses Wortpaar noch nicht gespeichert wurde.

B.2.2.2 `remove <word>` Entfernt ein Wort und alle zugehörigen Übersetzungen aus dem Wörterbuch, wenn dieses Wort dort existiert.

B.2.2.3 `print` Im Erfolgsfall wird jedes Wort zeilenweise zusammen mit seinen Übersetzungen ausgegeben. Das Wort aus der Startsprache wird durch den Separator von einem oder mehreren Wörtern aus der Zielsprache getrennt. Dabei werden alle Wörter aus dem Wörterbuch zusammen mit ihren Übersetzungen in alphabetischer Reihenfolge aufgelistet. Wenn ein Wort mehr als eine Übersetzung hat, werden diese Wörter alphabetisch geordnet und durch ein Komma getrennt. Wenn keine Wörter ausgegeben werden können, wird nur eine Fehlermeldung ausgegeben.

B.2.2.4 `print <letter>` Die Ausgabe dieses Befehls entspricht weitgehend der des parameterlosen `print`-Befehls. Im Erfolgsfall wird jedes Wort, das mit dem Buchstaben (Groß-/Kleinschreibung wird nicht berücksichtigt) beginnt, Zeile für Zeile zusammen mit seinen Übersetzungen ausgegeben. Alle Wörter aus dem Wörterbuch werden zusammen mit ihren Übersetzungen, die mit dem angegebenen Buchstaben beginnen, aufgelistet. Wenn es im Wörterbuch keine Wörter gibt, die mit diesem Buchstaben beginnen, wird nur eine Fehlermeldung ausgegeben.

B.2.2.5 `translate <word>` Im Erfolgsfall gibt dieser Befehl alle Übersetzungen des eingegebenen Wortes aus. Dabei wird die Groß- und Kleinschreibung der Eingabe berücksichtigt. Wenn für das Wort mehrere Übersetzungen gespeichert wurden, werden diese Wörter alphabetisch geordnet und durch ein Komma getrennt in einer Zeile ausgegeben.

B.2.2.6 `translate <sentence>` Mit diesem Befehl wird der eingegebene Satz übersetzt, indem Wort für Wort die entsprechende Übersetzung aus dem gespeicherten Wörterbuch ausgegeben wird. Im Erfolgsfall werden alle möglichen Kombinationen von Übersetzungen des Satzes zeilenweise ausgegeben. Bei der Eingabe wird zwischen Groß- und Kleinschreibung unterschieden, und die Zeilen der Ausgabe werden alphabetisch geordnet. Wenn für mindestens eines der Wörter des eingegebenen Satzes keine Übersetzung im Wörterbuch gefunden wird, wird nur eine Fehlermeldung ausgegeben.

B.2.2.7 `quit` Der Befehl beendet das Programm vollständig.

B.3 Beispielinteraktion

Die Zeilennummern und die Trennlinie sind kein Bestandteil der Benutzerschnittstelle, sie dienen lediglich zur Orientierung für die gegebene Beispielinteraktion. Die Eingabezeilen werden mit dem Größer-als-Zeichen (`>`) gefolgt von einem Leerzeichen eingeleitet, diese beiden Zeichen sind ebenfalls kein Bestandteil des eingegebenen Befehls, sondern dienen nur der Unterscheidung zwischen Ein- und Ausgabezeilen.

Für die folgende Beispielinteraktion wurde das Kommandozeilenargument zum Pfad der in Abschnitt B.1 gezeigten Textdatei angegeben, d.h. in diesem Beispiel enthält das Wörterbuch bereits die dort elf angegebenen Wortpaare.

Beispielinteraktion

```
1  > print d
2  Das - The
3  Der - The
4  Die - The
5  > translate Der
6  The
7  > translate klein
8  little,small,tiny
9  > add Haus Home
10 > translate Haus
11 Home,House
12 > remove Haus
13 > print
14 blau - blue
15 Das - The
16 Der - The
17 Die - The
18 ist - is
19 klein - little,small,tiny
20 Meer - ocean,sea
21 > translate Das Meer ist blau
22 The ocean is blue
23 The sea is blue
24 > quit
```