

## Aufgabe 3: Wortsuche

### 3.1 Lösungsidee

#### Schwierigkeitsstufen

Bevor wir eine Wortsuche erzeugen können, benötigen wir Kriterien, welche sich eignen, die Rätsel intuitiv voneinander zu unterscheiden. In der Aufgabenstellung finden sich bereits zwei Beispiele, von denen wir einige Kriterien ableiten können:

- **Richtung:** Es ist intuitiv leichter, Wörter zu finden, welche nur in Zeilen oder nur in Spalten stehen, als Wörter, die diagonal geschrieben sind.
- **Leserichtung:** Vorwärts eingefügte Wörter sind einfacher zu finden als rückwärts geschriebene Wörter.
- **Größe:** Kleine Wortsuchen sind einfacher zu lösen als große.
- **Auffüllen:** Eine Wortsuche, welche mit komplett zufälligen Buchstaben aufgefüllt wurde, ist einfacher zu lösen, als eine Wortsuche, welche mit Fragmenten der eingefügten Wörter aufgefüllt wurde.

Auf diesen Kriterien basieren wir die Definition der Schwierigkeitsstufen. Dabei gehen wir davon aus, dass wie in den Beispielen die Größe der Wortsuche festgelegt ist:

#### (1) einfach

- Alle Wörter werden vorwärts eingefügt.
- Die Richtungen sind von links nach rechts und von oben nach unten.
- Die Leerstellen werden mit zufälligen Buchstaben aufgefüllt.

#### (2) mittel

- Die Wörter werden vorwärts oder rückwärts eingefügt.
- Alle Richtungen sind erlaubt (links nach rechts, oben nach unten und diagonal).
- Die Leerstellen werden mit zufälligen Buchstaben aufgefüllt.

#### (3) schwer

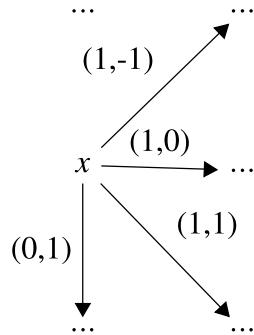
- Die Wörter werden vorwärts und rückwärts eingefügt.
- Alle Richtungen sind erlaubt (links nach rechts, oben nach unten und diagonal).
- Die Leerstellen werden mit Buchstaben aufgefüllt, welche bereits in den eingefügten Wörtern vorkommen.

#### Die Struktur einer Wortsuche

Eine Wortsuche lässt sich beschreiben als eine  $h \times b$ -Matrix, wobei  $h$  die Anzahl der Zeilen und  $b$  die Anzahl der Spalten ist. Ein eingefügtes Wort können wir dann durch einen Startindex  $(i, j)$  mit  $0 \leq i \leq h - 1$  und  $0 \leq j \leq b - 1$  sowie eine Richtung  $(d_x, d_y)$  beschreiben. Dabei beschreibt  $d_x$  die horizontale Veränderung, also die Veränderung von  $j$ , und  $d_y$  die vertikale Veränderung, also die Veränderung von  $i$ . Um eine Richtung zu definieren, müssen wir verstehen, wie sich die Indizes  $i$  und  $j$  verhalten, wenn wir über das einzufügende Wort in der

Matrix iterieren. Fügen wir zum Beispiel ein Wort von links nach rechts ein, dann bleibt der Reihenindex  $i$  unverändert und der Spaltenindex  $j$  steigt mit jedem weiteren Buchstaben. Von der Startposition aus können wir also die Position jedes weiteren Buchstabens bestimmen, indem wir  $i$  unverändert lassen und  $j + 1$  rechnen. Somit ergeben sich vier Richtungen  $(d_x, d_y)$ :

1.  $(1, 0)$  von links nach rechts
2.  $(0, 1)$  von oben nach unten
3.  $(1, -1)$  diagonal nach oben
4.  $(1, 1)$  diagonal nach unten



Alle anderen Richtungen ergeben sich durch das Rückwärts-Einfügen des Wortes.

### Das Einfügen eines Wortes

Um eine zufriedenstellend schwierige Wortsuche zu erstellen, müssen die Wörter an zufälligen Positionen (in einer zufälligen erlaubten Richtung) eingefügt werden. Dabei bietet es sich an, die Wörter zu Beginn nach Länge zu sortieren und mit dem längsten Wort anzufangen. Falls das längste Wort länger als  $\max(h, b)$  ist, so kann jetzt bereits abgebrochen werden, da es keine Richtung gibt, in der dieses Wort in die Matrix passt. Basierend auf der Wortlänge  $l$  und der Richtung ergibt sich ein Bereich, in welchem es möglich ist, das Wort einzufügen:

$$\begin{cases} 0 \leq i \leq h-l & \text{für } d_y \geq 0 \\ l-1 \leq i \leq h-1 & \text{sonst} \end{cases} \quad \text{und} \quad \begin{cases} 0 \leq j \leq b-l & \text{für } d_x = 0 \\ 0 \leq j \leq b-1 & \text{sonst} \end{cases} \quad (3.1)$$

Eine gültige Position für ein Wort ergibt sich aus den Koordinaten  $(i, j)$  innerhalb des validen Bereiches sowie einer Richtung  $(d_x, d_y)$ , sodass für jeden Buchstaben des Wortes in der Zelle  $(i, j)$  in der Matrix entweder nichts oder bereits der korrekte Buchstabe steht. Für jedes Wort in der sortierten Wortliste können nun die gültigen Positionen berechnet werden und eine zufällige ausgewählt. Falls sich für ein Wort keine gültige Position mehr ergibt, dann bricht der Algorithmus ab und beginnt von vorne. Ein Backtracking zum letzten Wort, welches mehrere gültige Positionen hat, d. h. ein Neustart von dort aus garantiert nicht notwendigerweise, dass diesmal für ein späteres Wort eine gültige Position gefunden wird, da der Fehler bereits beim zufälligen Einfügen eines früheren Wortes zustande gekommen sein kann.

### Generieren der Wortsuche

Bei komplexeren Wortsuchen (mehr Wörter, weniger Dimensionen etc.) ist nicht garantiert, dass der Algorithmus eine Lösung zurück gibt, auch wenn theoretisch eine Lösung möglich wäre. Die Laufzeit des naiven Algorithmus wird optimiert, indem

- zu Beginn geprüft wird, ob eine Lösung theoretisch möglich ist, indem die Dimensionen mit der Länge des längsten Wortes verglichen wird und
- der Bereich, über den iteriert wird, um gültige Positionen zu finden, eingeschränkt wird.

---

**Algorithmus 3** Generierung einer Wortsuche

---

```

1: procedure WORTSUCHE( $b, h, \text{wortliste}$ )
2:   if  $\max_{\text{wort} \in \text{wortliste}}(\text{länge}(\text{wort})) > \max(h, b)$  then return false
3:   end if
4:    $\text{richtungen} \leftarrow (0, 1), (1, 0), (1, 1), (1, -1)$ 
5:    $\text{sortiere}(\text{wortliste}, \text{wortlänge}, \text{absteigend})$             $\triangleright$  Fange mit dem längsten Wort an
6:   for  $\text{wort} \in \text{wortliste}$  do
7:      $\text{permutiere\_zufällig}(\text{richtungen})$ 
8:     for  $(d_x, d_y) \in \text{richtungen}$  do            $\triangleright$  Fange mit einer zufälligen Richtung an
9:       # Bestimmt den gültigen Bereich nach 3.1
10:       $i_{\text{unten}}, i_{\text{oben}}, j_{\text{unten}}, j_{\text{oben}} \leftarrow \text{gültigerBereich}(n, h, \text{länge}(\text{wort}))$ 
11:      # Iteriere über den Bereich und bestimme alle gültigen Positionen
12:      for  $i \leftarrow i_{\text{unten}}, \dots, i_{\text{oben}}$  do
13:        for  $j \leftarrow j_{\text{unten}}, \dots, j_{\text{oben}}$  do
14:           $\text{koordinaten} \leftarrow \text{validePositionen}(i, j, d_x, d_y)$ 
15:        end for
16:      end for
17:      if  $\text{koordinaten}$  then            $\triangleright$  Falls eine gültige Position gefunden wurde
18:        # Füge an einer zufälligen gültigen Position ein
19:         $\text{permutiere\_zufällig}(\text{koordinaten})$ 
20:         $\text{fügeWortEin}(\text{wort}, \text{koordinaten}, d_x, d_y)$ 
21:        break
22:      end if
23:    end for
24:  end for
25:   $\text{fülleMitBuchstabenAuf}()$             $\triangleright$  Alle Wörter sind eingefügt
26:  return true
27: end procedure

```

---

Der Algorithmus wird maximal 10 mal ausgeführt, danach wird davon ausgegangen, dass keine Lösung existiert. Für die bereitgestellten Beispiele benötigt der Algorithmus meist nur einen Versuch, um alle Wörter einzufügen.

Für jedes Wort gibt es höchstens  $b \cdot h$  mögliche Startpositionen und 4 mögliche Richtungen. Das Wort einzufügen bzw. zu überprüfen, ob das Wort so eingefügt werden kann, benötigt  $\mathcal{O}(\text{länge}(\text{wort}))$  Anweisungen. Ist  $w$  die Anzahl der Wörter und  $l$  die Länge des längsten Wortes, so lässt sich die Laufzeit daher durch  $\mathcal{O}(w \cdot l \cdot b \cdot h)$  nach oben abschätzen.

Die Änderung der Schwierigkeitsstufen ist der Übersicht halber nicht im Pseudocode enthalten. Im Falle von *einfach* und *mittel* wird die Schwierigkeitsstufe durch eine entsprechende Auswahl aus *richtungen* erzwungen. Für Schwierigkeit *schwer* soll die Wortsuche nicht mit zufälligen Buchstaben aufgefüllt werden, sondern mit Buchstaben oder Fragmenten, welche in den Wörtern aus der Wortliste vorkommen. Dies führt im Beispiel *worte5.txt* (wo nur **DAS** in eine  $30 \times 30$ -Matrix eingefügt werden soll) dazu, dass mit sehr hoher Wahrscheinlichkeit mehrfach das Wort **DAS** entsteht. Dies ist zwar laut Aufgabenstellung nicht explizit verboten, führt allerdings dazu, dass nach den obigen Definitionen der Schwierigkeitsgrad *schwer* sehr viel einfacher wäre als *einfach*. Dies sollte auffallen und zum Beispiel gelöst werden, indem beim Auffüllen der Matrix mit den zufälligen Buchstaben getestet wird, ob ein neues Wort entsteht, welches bereits in der Wortliste enthalten ist.

### 3.2 Beispiele

Angewandt auf die Eingaben auf der BWINF-Website liefert das Programm folgende Wortsuchen. Da zu erwarten ist, dass alle Wortsuchen voneinander abweichen, sind hier nur ein paar repräsentative Beispiele gezeigt.

```
worte2.txt:
LEVEL 1:
J I W O P B F T J F
K N H G C I T A M E
Z B Q W O L G S G S
M A U S M D A T J T
U L Y W P S F A W P
M N Z F U C A T A L
V U S B T H A U D A
W R K T E I M R D T
S B H W R R H N M T
F B Y Z K M C Y N E

worte1.txt:
LEVEL 1:
O I N F O O
U B D D A G
G Q E I N J
L E C U N D
E R P R N D
R X X S P U

LEVEL 2:
Y N E D S W
F A U U L O
D Q N S F K
R D X N U Q
P E I I C M
H P R E J P

LEVEL 3:
D A N U I N
F U F E N A
E F I A F N
A N E D O O
D E R U N E
F R D D N U

SUCHE:
TORF, VOR,
RAD, EVA

SUCHE:
INFO, EIN,
UND, DA,
ER, DU

SUCHE:
BILDSCHIRM,
FESTPLATTE,
TASTATUR, COMPUTER,
MAUS, USB

LEVEL 1:
U S B R E F G M N Q
M P A U R E J R O C
C S Z T M S Z I Y B
O U O A D T F H X U
M S U T V P J C A M
P S E S K L Z S B H
U W D A U A M D S V
T D W T X T B L P V
E W P V Z T W I W Q
R D H S Z E S B I I

LEVEL 2:
P E B L E F A U M M
A T I S U A M A M C
M T L U M A M M O T
U A D M S E E M A A
M L S A M B P M U S
F P C P E U P P E T
L T H F T U M P E A
P S I E A A A A A T
U E R F A L U U E U
M F M E L P M L A R

LEVEL 3:
P E B L E F A U M M
A T I S U A M A M C
M T L U M A M M O T
U A D M S E E M A A
M L S A M B P M U S
F P C P E U P P E T
L T H F T U M P E A
P S I E A A A A A T
U E R F A L U U E U
M F M E L P M L A R
```

worte4.txt:

LEVEL 1:

F O R D K T R G I Y R U C E Q D A R C H I V C J J J W B M J N S  
 N R G A Q E F A B R U F D A T U M L Y P A F U Y W Q J B S N F I  
 Z K G E R X S T H ö H E N M J S V I S D Z O Y B F A V K T M U O  
 C O R Q H T V B X A C H T U N G K T O Q E W N E G H N L V O S A  
 O Y E W Y H G I K Q Z F Q M U K D E R M G K F G G N I S U U S U  
 M R Z V F A R B L E G E N D E D I R T X E F A R N Y A Q N U B V  
 M A I U F Y N R N F T E Z Q C C U A J F O X U I M D J N K V A W  
 O F L H A P O E L W E S A U T T R T L E Q H D F A O E A C Q L U  
 N A S V C X K C Z X M L N K Y M C U C P U M K F U I N V E Y L C  
 S D Y F E G C O O R D I N A T E A R U S E W F S T G O I Z Z D D  
 C Y F E X A M R B N E T R C Y K N U D J L P B K O Y C G R U A H  
 A D N H I U Q D I N T E R N E T Q U E L L E D L A W O A E J T X  
 T K T H R S I P C O M M O N S W K B E L E G E Ä R I M T E X E J  
 P G G I G Z J U F Y A R C H I V B O T Q U J C R C K M I D E N U  
 ö S T E R R E I C H B E Z O G E N E K M E O W U H T O O K Y C G  
 P O S I T I O N S K A R T E W I K I D A T A Z N I I N N P A E F  
 L V B A B E L C R L O Y L B Z Z I T A T I O N G V O S S M B D M  
 A N S U P I N G D I T M T J I N F O B T A J D S H N W L E O B U  
 A F E R P L S B S M I L E Y H E T N E Y B S T H X A B E D O N L  
 W J T N S S C C S L U M N L B X E L G B T Z I I S R C I A L E T  
 J I N F O R M A T I O N F L J G R A R F I M L N S Y F S I A E I  
 P I W E B A R C H I V N V I J Y Y N I L D E U W G Z O T L N S L  
 S D I S K U S S I O N S S E I T E G F I R W D E S Y L E L D T I  
 A C W M H I X X R B E N U T Z E R S F Z F A U I A H G K E P Q N  
 O B T T J C F C B A U S T E L L E H S E W P K S M T E S N M B G  
 E G R O X O S O Q J C V T A I G E K K N I P M P B N N S S R A U  
 T R S K U O D G R H W Y W K B T Q J L Z K E U E K W L O P G L A  
 A W A A A R P O K C H A R T S F R A Ä U I N S R A O E H I A L L  
 U U S T I D A R C H I V I E R U N G R M S R I S L O I O E B M I  
 V A K E Q I P C S Z J M P G G Y O Q U S O E K O E Z S A G S U E  
 V K J G V N Q E E R L E D I G T J W N T U C C N N X T Z E A S C  
 Q Z K O Y A D N R Y N A V F R A M E G E R H H E D A E F L T I I  
 R J T R Y T W T T J O E L D D B J L X L C T A N E I X F Q Z C M  
 P I C I V E H E A O R C R O M C N P J L E F R L R R I S A C T D  
 H I G E K M P R X B K A S T E N R T Q U J H T E S W A P X F P B  
 Q K Z G A A S I O B O E O A U R F F J N Z V S I T V K K G I H Y  
 H Q Q R L P I P B U J A Q Q T A R A E G O L L S I Z V P V L T N  
 L H Y A O T P C O C O L Q R K E J Z S B E L F T L Y P R O M F G  
 P Z V P V Q A I X E Q A Y G A N G N M R X D T E H N S D Q C X R  
 V S C H I N N O K L M W B L W V R A W I P Y H G I J X S P O R H

SUCHE:

BEGRIFFSKLÄRUNGSINWEIS , NAVIGATIONSLEISTE , öSTERREICHBEZOGEN ,  
 DISKUSSIONSSSEITE , LIZENZUMSTELLUNG , MEDAILLENSPIEGEL ,  
 BEGRIFFSKLÄRUNG , INTERNETQUELLE , KATEGORIEGRAPH ,  
 PERSONENLEISTE , POSITIONSKARTE , COORDINATEMAP , FUSSBALLDATEN ,

ARCHIVIERUNG , FOLGENLEISTE , KALENDERSTIL , MULTILINGUAL ,  
 FARBLEGENDE , INFORMATION , MUSIKCHARTS , WAPPENRECHT , ABRUFDATUM ,  
 AUTOARCHIV , COMMONSCAT , COORDINATE , WIKISOURCE , WIKTIONARY ,  
 ARCHIVBOT , BAUSTELLE , BIBRECORD , GEOQUELLE , LITERATUR ,  
 NOCOMMONS , WEBARCHIV , ALLMUSIC , BENUTZER , ERLEDIGT , NAVFRAME ,  
 WIKIDATA , ZITATION , ACHTUNG , BOOLAND , COMMONS , TAXOBOX , ABSATZ ,  
 ARCHIV , BELEGE , CENTER , CHARTS , KASTEN , SMILEY , BABEL , BBKL ,  
 FILM , GNIS , HÖHE , IMDB , INFO , LANG , PING , SORT , TEXT , AUS ,  
 AUT , BEL , BGR , CAN , COL , DDB , DEU , DOI , FNZ , FRA , GER , IPA ,  
 PRO , FN

worte5.txt:

LEVEL 1:

```

U I O R U J Q K Q L Y S P B B E Y K I D S Q L V M B S U Q A
H K L J A E X M F N Y A F A X B Q S D Y K U C Q I R F N B B
L K T Z O Z F V R C F H H C O M F D I V X D G E P H I K Z B
M D V T O D S E B W V S L J F A E A K A B F M Q M O Y N M V
N P A H A O U O C M O P K Q S L Q K M D W J B I B Y H V K L
H Y L D G F L Y C S X C R G Q J Y I M R V Z M L Z W K P P M
O P V W H C U T U L X S Y T Z K N X J V J C M H S I L K O E
G Q B V Z L Y O J C Z X S X Z P Z A O R S Y U L B M X D D X
K H W B I H D I G S A A N L O F A W L I R C W J R M N N J C
Q M Z I P Y S G N W K P B P G D O V P S H F X W S Y X U K W
R O O M X B C S E K G C A U F C U N M I W A N X N U T E E L
F V S P N F J L K W P D F N R O W N A S R U F L R A S A I J
V T E V D Z H M U N M A P F X H W A M J D U C C O J X T B I
H J G P L P V C Y X X M T Y J Q Z C B F Q T S R U Z T B P A
A K R Y U V I E S N M U D R Q G D Y P Y H C B Z J X M Z E R
S L U U N H Y W M V K L X K Y N D D A S K W Z H X B D T I W
Y E J E X C V J K C S B K G O I H O R B D L B T O X P X W H
U W D K A E H D Y X M B X L C Z A R Y O G M N X S A T X V Q
V S K M I H C D L Y Z C N B N I U F J M R G J S G F L X C A
E W M X T I Y U G R W W L E G X R E K T U T X P K W R J O B
D E I N X F W P Q D D N T C J T V L H U I C W D Y Z G D M N
Y P I H E S N Q F D Y N I P M R A W R Y I Z Z T E Y F E R T
O N V H D Q O K T W Z F A Z T D X Y B D L K I K J T N N B Q
C H U T V E P R F R O V S L G T W E Z Z H V T V C E C R D L
H W S J H N G I M E N P R E B I Y V P D I N T H X S I R E Q
Z R T X K V A Z E C I E R M Y P A F C H E J P W U H L N B R
N J Q N P M O H I Q W I Z G N J Z M Y P J V J U Y W U O R F
D M D F L D B S A A R M E S T G A A C L H O A W O U K D A M
Z C T U U W L V V D P Q V W G M Y I X H L D S A C X C Z X V
A V G X B H M F S S E U N W I Q T P V M W Y E Y Q R I V S U

```

LEVEL 3:

D S S S S S D U S D S S D A A A A S D O D A D A U D A E D S  
S O D O S A S D D S U D U U U D D S U U S S S O A O O O U O  
O O S S A U S A S U S A S O A S S D D S D S U O A D O A A S  
S A U U A D X G S U S U S D A L A O O A O U O U D S O A A S  
F U O O S T U A A S S S D A D A M D D S U U A S S S U S C U S  
O S O O S O S D A M D A X D S A S S A A O S O A A S U D S U  
A A D O O O A A D A D U S O U U O U O Y H S U S S O O S U S  
D A O S A U U U A U U S A S D O D U A D A O O O A D S O O D  
A A U D D D U O O A U A O A S O S S O B S U D D O S D D U O  
H A A U O O U O D S S O D U A S U S A S U S U U D D U S A U  
D U D D A U D O S O D U S U S D D D A R A D A U D A J O O P  
A U U A O A A A D U O O O D S O D S A O D Q D U D O M A D O  
I A O O D S O S U A S D A D A A D S D O D U A O A O U U U S  
O O A A D O A O O U D A D S C D O U D S O O U A J D O A U A  
O A U U D U O U D D A U Q U U O U O A U A D D D U O A A O S  
U A D A E S A U S D T D S D D D S A E A S O A O A D O A O  
O D O D O S D O O D A U A A O O D S S U O O U U U S D Q A S  
O U A O O A A A D D U D V U S A S U S S O O O D S U O A O S  
O U O B U O A O D A O U O S D S S S D S A U U S S D A O A S  
S D A T S O A D O U A A A D O U U A S O O O O O U A A S D  
A A D S U U D U S O O Z U A D U A S S A A S D O A O O Q S D  
W O O S D U D A A O A D D G O U D U D O S S S U A S O O O O  
A D D U S D S U O O O A D O O A O A O A D S S D A W D U A S  
A D A A A A A D D U K D A D D D O U O S U D A U D D S D A A  
S O F U L U S G A U S S U A D A U U O D D D S U O A S F U S  
S O A D D O D U A U A D O D U O O S U O O S D D U H X O A S  
U D U A D O D U D A A O O U U U S S D O S A O U D A A Q U S  
S U A O A S A U U A O O A A O D A O A S O U A A S A S U O U  
D S U Q D O O O A U U S O O D O U A A A O A A O S A G U A O  
D O O A D O O S D O U U A S U A A W O U A A S O O O S O D A

SUCHE :

DAS

### 3.3 Bewertungskriterien

Die Bewertungskriterien vom Bewertungsbogen werden hier erläutert (Punktabzug in []).

- [-1] **Lösungsverfahren fehlerhaft**

Bei einer gültigen Lösung müssen alle Worte aus der Wortliste korrekt eingefügt worden sein. Wenn beim Auffüllen der Leerstellen zufällig Worte aus der Wortliste entstehen, dann führt dies nicht zu Punktabzug. In üblichen Wortsuchen sollte das zwar nicht vorkommen, aber die Aufgabenstellung war in diesem Punkt nicht klar.

- [-1] **Kriterien ungeeignet definiert**

Die Aufgabe fordert, dass Kriterien definiert werden, welche die Schwierigkeit von Wortsuchen intuitiv voneinander unterscheiden. Diese Kriterien sollten einerseits eine Klassifizierung von gegebenen Rätseln ermöglichen und andererseits als Basis zur Erzeugung neuer Rätsel geeignet sein. Spätestens für die Definition der Schwierigkeitsstufen sollten Kriterien erkennbar sein.

- [-1] **Schwierigkeitsgrade ungeeignet definiert**

Die Aufgabe fordert, dass  $\geq 3$  Schwierigkeitsgrade definiert werden, welche auf den vorher definierten Kriterien basieren. Die Anwendung der Schwierigkeitsgrade auf eine Wortliste sollte zu merklich unterschiedlichen Wortsuchen führen. Es muss betrachtet werden, dass mit Regeln, die auf dem zufälligen Einfügen von Fragmenten basieren, weitere Worte entstehen können. Dies kann dazu führen, dass Wortsuchen, welche mit schwierigeren Regeln generiert wurden, einfacher zu lösen sind (z. B. `worte5.txt`).

- [-1] **Verfahren bzw. Implementierung unnötig aufwendig / ineffizient**

Es sollte sinnvolle Bedingungen geben, die den Algorithmus abbrechen lassen, falls nach mehrfachen Versuchen keine Wortsuche erzeugt werden konnte. Ein Verfahren ist akzeptabel, wenn innerhalb weniger Sekunden eine Wortsuche generiert wird.

- [-1] **Beispiele fehlerhaft bzw. zu wenige oder ungeeignete Beispiele**

Die Dokumentation soll Ergebnisse zu mindestens 3 der vorgegebenen Beispiele (`worte1.txt` bis `worte5.txt`) enthalten. Zu jedem dokumentierten Beispiel sollten idealerweise Wortsuchen mit jeweils drei Schwierigkeitsgraden angegeben werden. Sollten es weniger sein, ist das nur akzeptabel, wenn die Beispiele insgesamt einen guten Überblick vermitteln.