# 1 Grundlagen

#### 1.1 Aufgabe

Implementiere eine Funktion showTriesLeft. Dieser wird die Anzahl der übrigen Versuche als Integer übergeben. Sie gibt einen String zurück, in dem steht wie viele Versuche der Spieler noch hat.

```
showTriesLeft(3)
```

3 Versuche übrig

```
showTriesLeft(1)
```

1 Versuch übrig

Beachte das e in Versuche.

#### 1.2 Aufgabe

Implementiere eine Funktion contains. Diese prüft, ob ein Buchstabe in einem Wort enthalten ist..

```
contains("GHSE", 'G')

true
contains("Informatik", 'y')
```

false

#### 1.3 Aufgabe

Implementiere eine Funktion contains. Diese prüft, ob ein Buchstabe in einer Liste von Buchstaben enthalten ist.

```
contains(listOf('G', 'H', 'e', 'S', 'E'), 'G')
true
contains(listOf('I', 'n', 'f', 'o', 'r', 'm', 'a', 't', 'i', 'k'), 'y')
false
```



#### 1.4 Aufgabe

Implementiere eine Funktion compTriesLeft. Dieser werden die Anzahl der übrigen Versuche vor dieser Runde, das Wort, das erraten werden soll und der zuletzt geratene Buchstabe übergeben. Sie gibt die Anzahl der übrigen Versuche nach dieser Runde zurück.

```
compTriesLeft("GHSE", 'G', 3)

compTriesLeft("Informatik", 'y', 10)

Nutze contains
```

#### 1.5 Aufgabe

Implementiere eine Funktion showGuessedCharsAndBars. Dieser werden das Wort, das erraten werden soll und eine Liste mit Buchstaben übergeben. Sie gibt einen String zurück. In diesem wurden alle Buchstaben aus dem übergebenen Wort, die nicht in der Liste sind durch Bindestriche ersetzt. Alle anderen Buchstaben wurden unverändert übernommen.

```
showGuessedCharsAndBars("GHSE", listOf('G', 'H', 'e', 'S', 'E'))

GHSE

showGuessedCharsAndBars("Informatik", listOf('I', 'a', 'y'))

I----a---

Nutze contains
```

#### 1.6 Aufgabe

Implementiere eine Funktion checkGuessedWholeWord. Dieser werden das Wort, das erraten werden soll und eine Liste mit Buchstaben übergeben. Sie gibt zurück, ob alle Buchstaben aus dem Wort in der Liste enthalten sind.

```
checkGuessedWholeWord("GHSE", listOf('G', 'H', 'e', 'S', 'E'))

true
```



checkGuessedWholeWord("Informatik", listOf('I', 'a', 'y'))

false

Nutze showGuessedCharsAndBars

#### 1.7 Aufgabe

Implementiere eine Funktion showHangmanAsciiArt. Dieser wird die Anzahl der übrigen Versuche übergeben. Sie gibt die entsprechende Zeichnung des Spielers und des Galgens zurück. Bei 6 übrigen Versuchen ist der Spieler noch nicht zu sehen. Bei 0 Versuchen ist der Spieler komplett zu sehen.

showHangmanAsciiArt(6)



showHangmanAsciiArt(5)



showHangmanAsciiArt(1)



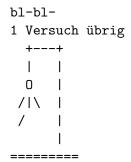


showHangmanAsciiArt(0)



#### 1.8 Aufgabe

Implementiere eine Funktion showHangmanState. Dieser werden das Wort, das erraten werden soll, die Liste der erratenen Buchstaben und die Anzahl der übrigen Versuche übergeben. Sie gibt einen String zurück, in dem der gesamte Zustand des Spiels inklusive Ascii-Art dargestellt ist.



Nutze showHangmanAsciiArt, showGuessedCharsAndBars und showTriesLeft! Erzeuge mit "\n" einen Zeilenumbruch.



#### 1.9 Aufgabe

Implementiere eine Funktion showGameResult. Dieser werden das Wort, das erraten werden soll und die Liste der erratenen Buchstaben übergeben. Sie gibt einen String zurück. Wenn das gesamte Wort erraten wurde, steht in dem String, dass das Spiel gewonnen wurde. Ansonsten steht in dem String, dass das Spiel verloren wurde.

```
showGameResult("GHSE", listOf('G', 'H', 'x', 'S', 'E'))
Du hast gewonnen!
showGameResult("blabla", listOf('b', 'a', 'S', 'E'))
Du hast verloren!
  Nutze guessedWholeWord!
```

#### 1.10 Aufgabe

Implementiere eine Funktion showHangmanStateAndGameResult. Dieser werden das Wort, das erraten werden soll, die Liste der erratenen Buchstaben und die Anzahl der übrigen

```
Versuche übergeben. Sie gibt einen String zurück, in dem der Zustand des Spiels und das
Ergebnis des Spiels stehen.
showHangmanStateAndGameResult("GHSE", listOf('G', 'H', 'e', 'S', 'E'), 1)
GHSE
1 Versuch übrig
  0
 /1\
Du hast gewonnen!
showHangmanStateAndGameResult("blabla", listOf('b', 'l', 'E'), 0)
bl-bl-
O Versuche übrig
```





Du hast verloren!

 $Nutze \verb| showHangmanState| und \verb| showGameResult!$ 

## 1.11 Aufgabe

Implementiere eine Funktion gameFinished. Dieser werden das Wort, das erraten werden soll, die Liste der erratenen Buchstaben und die Anzahl der übrigen Versuche übergeben. Sie gibt zurück, ob das Spiel zu Ende ist.

```
gameFinished("GHSE", listOf('G', 'H'), 1)

false

gameFinished("GHSE", listOf('G', 'H', 'e', 'S', 'E'), 1)

true

gameFinished("blabla", listOf('b', 'l', 'E'), 0)

true
```

Nutze checkGuessedWholeWord! Das Spiel ist zu Ende, wenn das Wort erraten wurde oder der Spieler keine Versuche mehr übrig hat.

#### 1.12 Aufgabe

Definiere zwei globale Variablen mit den folgenden Inhalten:

questionWordToGuess

Welches Wort soll erraten werden?

questionGuessChar

Rate einen Buchstaben!

#### 1.13 Aufgabe

Implementiere eine Funktion askChar. Diese fragt die Benutzerin, welchen Buchstaben sie rät. Der eingegebene Buchstabe wird anschließend als Char zurückgegeben.



# 2 Fertigstellung des Spiels

Ein Durchlauf des Spiels kann folgendermaßen aussehen.

```
Welches Wort soll erraten werden?
   GHSE
   6 Versuche übrig
10
11
   Rate einen Buchstaben!
12
13
   G---
   6 Versuche übrig
15
16
17
18
19
20
21
22
   Rate einen Buchstaben!
23
24
   У
   5 Versuche übrig
27
      28
      0
29
30
32
33
   Rate einen Buchstaben!
34
   Η
35
   GH--
36
   5 Versuche übrig
     +---+
38
      39
```

```
41
42
43
44
    Rate einen Buchstaben!
45
    S
    GHS-
    5 Versuche übrig
48
49
50
      0
51
53
54
55
   Rate einen Buchstaben!
56
   Ε
57
    GHSE
    5 Versuche übrig
59
60
61
      0
62
63
64
65
66
    GHSE
67
    5 Versuche übrig
68
70
      0
71
72
73
74
    =======
   Du hast gewonnen!
```

#### 2.1 Aufgabe

Implementiere eine Funktion playHangmanHelper. Dieser wird ein Wort als String übergeben. Dieses muss anschließend erraten werden. Ein Aufruf der Funktion ist oben ab Zeile 3 zu sehen.

 $Nutze \verb| showHangmanState|, gameFinished|, askChar|, compTriesLeft und \verb| showHangmanStateAndGameResual askChar|, compTriesLeft und showHangmanStateAndGameResual askChar|, compTriesLeft un$ 



## 2.2 Aufgabe

Implementiere eine Funktion askWordToGuess. Diese fragt die Benutzerin, welches Wort erraten werden soll. Das eingegebene Wort wird anschließend zurückgegeben.

Nutze die Variable questionWordToGuess!

### 2.3 Aufgabe

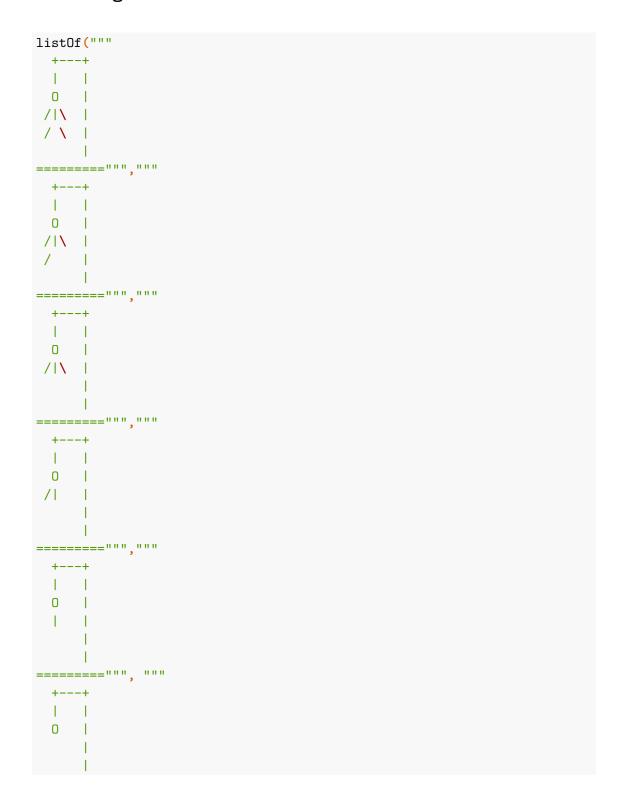
Implementiere eine Funktion playHangman. Diese hat keine Parameter. Sie fragt zunächst, welches Wort erraten werden soll. Anschließend wird mit diesem Wort eine Runde Galgenmännchen gespielt.

### 2.4 Aufgabe

Überspringe mit for-Schleifen, dem Kontrollzeichen "\n" und println Zeilen um das Spiel spielbar zu machen!



# 3 Anhang



======="","""	
++	
1 1	
======""")	