

Würfelsimulation

Dickbauer Y., Moser P., Perner M.

PS Computergestützte Modellierung, WS 2016/17

November 29, 2016

Outline

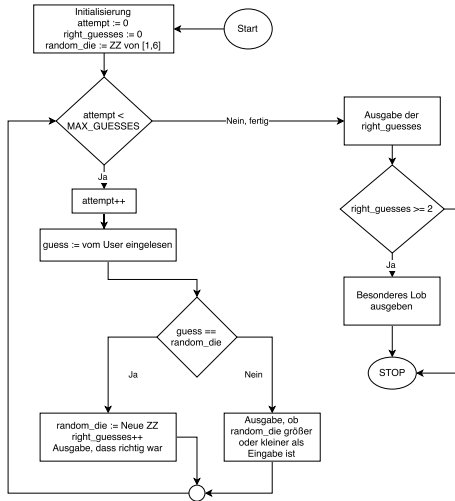
- 1 Aufgabenstellung
- 2 Flow Chart
- 3 Flow Chart
- 4 Programmcode
 - Main Funktion
 - Verwendete Funktionen
- 5 Beispiel

Aufgabenstellung

Der Benutzer darf sechsmal eine Zahl raten. Wenn die Zahl größer bzw. kleiner ist als die vom Rechner gewürfelte Zahl, dann wird dies dem Benutzer jeweils mitgeteilt. Bei richtiger Eingabe der Zahl gratuliert der Rechner dem Benutzer. Hat der Benutzer mindestens zweimal richtig oder nie richtig geraten, gibt es spezielle Glückwünsche bzw. Bedauern des Rechners.

- Eingabe: geratene Zahl (insgesamt 6x)
- Output: Rückmeldung je Durchgang, Endergebnis

Flow Chart



Main Funktion - Programmeinstieg

```
1 def main():
2     attempt = 0
3     right_guesses = 0
4     random_die = int(random_number_from_interval(0, 6)+1)
5
6     while attempt < MAX_NUMBER_OF_GUESSES:
7         attempt += 1
8         guess = int(input('Rate: '))
9         if guess == random_die:
10             print('Richtig, du bekommst einen neuen Wuerfel')
11             right_guesses += 1
12             random_die = int(random_number_from_interval(0, 6)+1)
13         else:
14             less_or_greater = 'kleiner' if random_die < guess else 'groesser'
15             print('Nein, gesuchte Zahl ist {} als {}'.format(less_or_greater, guess))
```



Main Funktion - Ausgabe der Ergebnisse

```
1  # print result
2  print('{}mal_richtig_geraten'.format(right_guesses))
3  if right_guesses == 0:
4      print('Kein_einziges_Mal_richtig_bei_{}_Versuchen_ist_halt_kein_guter_Sch
5  elif right_guesses >= 2:
6      print('Super,_mindestens_zwei_mal_richtig,_gut_gemacht!')
```



Funktion `random_number_from_interval(..)`

- Diese Funktion verlangt zwei Eingabeparameter *lower* und *upper*
- Gibt eine (pseudo)Zufallszahl (*float*) im Intervall [*lower*, *upper*) zurück
- `random.random()` ist eine Funktion der Python Standardbibliothek, welche eine Zufallszahl (*float*) im Intervall [*lower*, *upper*) zurück gibt
- Mersenne Twister Methode wird als Generator der ZZ verwendet^{1 2}

```
1 def random_number_from_interval(lower, upper):  
2     val = random.random()  
3     return lower + (upper - lower) * val
```



¹<https://docs.python.org/3.5/library/random.html>

²https://en.wikipedia.org/wiki/Mersenne_Twister

Beispiel anhand fixer Zufallszahlen

- Annahme folgender ZZ und Eingaben:

Iteration	0	1	2	3	4	5
ZZ vor iteration	3			1		
Usereingabe guess	1	2	3	4	5	6

Iteration 0:

- User gibt 1 ein, ZZ ist 3, Programm meldet 'zu niedrig'
- attempt := 1, right_guesses := 0

Iteration 1:

- User gibt 2 ein, ZZ ist 3, Programm meldet 'zu niedrig'
- attempt := 2, right_guesses := 0

Iteration 2:

- User gibt 3 ein, ZZ ist 3, Programm meldet 'passt'
- attempt := 1, right_guesses := 0
- Neuberechnung random_die := 1

Anhang: Modifikation des Source Codes um Demo Beispiel zu erhalten

```
1  # Aendere random_number_from_interval() in lib.py wie folgt:
2  i = -1
3  ZZ = [3,1] + list(range(1,10))
4  def random_number_from_interval(lower, upper):
5      global i; i += 1;
6      return ZZ[i]-1
```

