Sortierung von Zufallszahlen

Dickbauer Y., Moser P., Perner M.

PS Computergestützte Modellierung, WS 2016/17

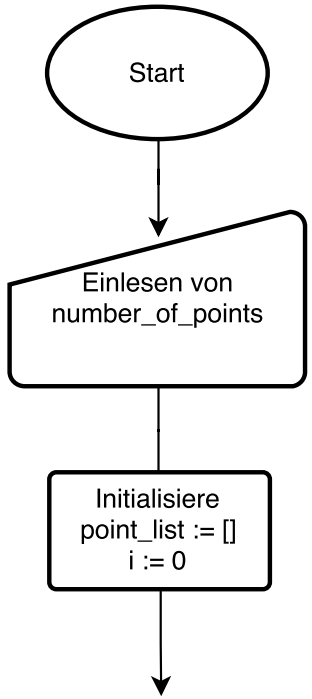
November 28, 2016

Outline

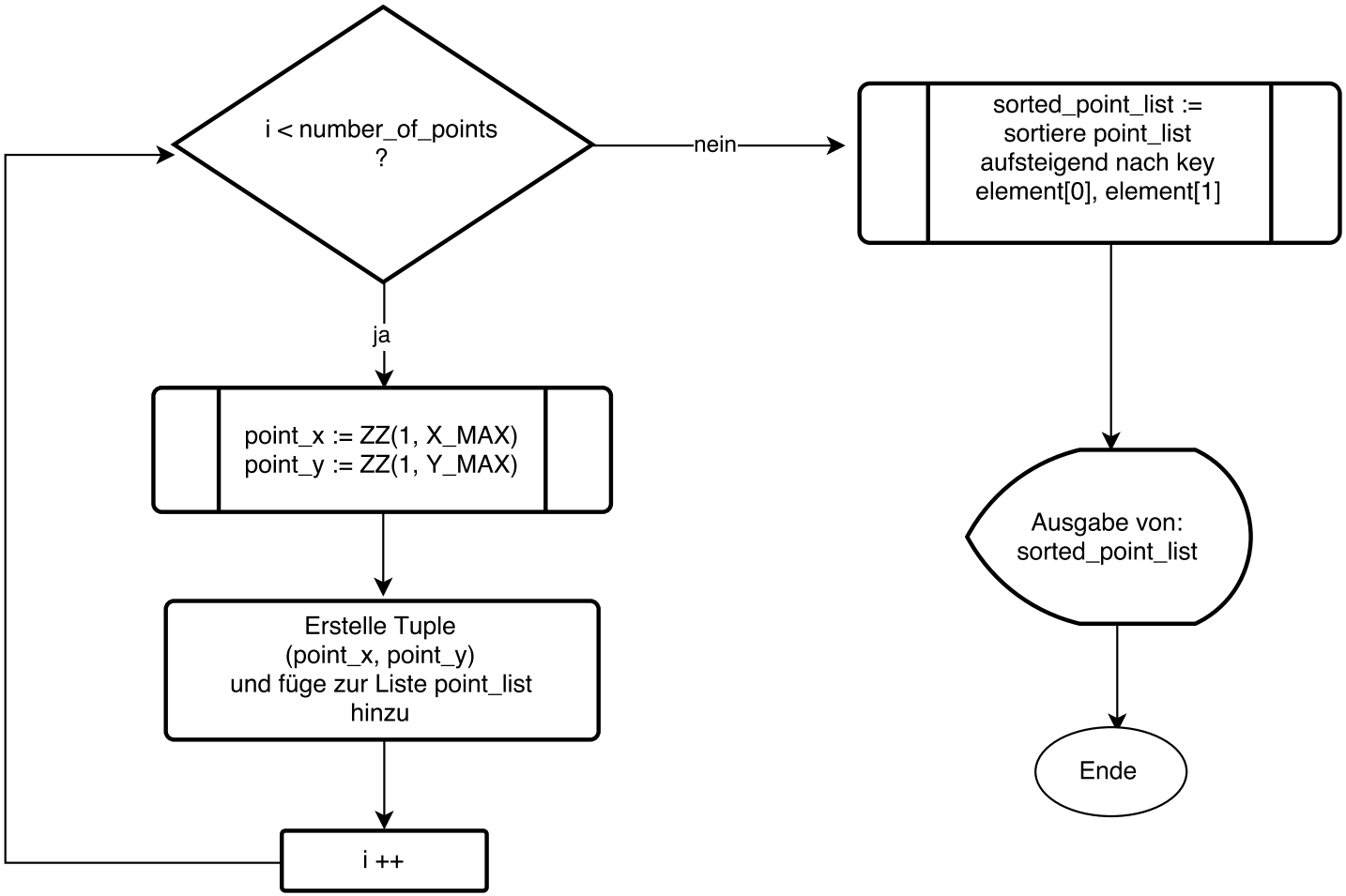
Aufgabenstellung

- Eingabe: Anzahl an n Zufallszahlen
- Erstellung von n Zufallszahlen
- Erstellung einer Liste dieser generierten Zufallszahlen
- Sortierung dieser Liste Aufsteigend. 1. Priorität nach X, 2. Priorität: Y
- Ausgabe der Liste

Flow Chart


Start
Einlesen von
number_of_points
Initialisiere
point_list := []
i := 0

Flow Chart


i < number_of_points
?
nein
ja
sorted_point_list :=
sortiere point_list
aufsteigend nach key
element[0], element[1]
point_x := ZZ(1, X_MAX)
point_y := ZZ(1, Y_MAX)
Ausgabe von:
sorted_point_list
Erstelle Tuple
(point_x, point_y)
und füge zur Liste point_list
hinzu
Ende
i ++

Main Funktion - Programmeinstieg

```
1   def main():
2       # user input:
3       inp = user_input((
4           ('Number␣of␣points', int, 10),), DEBUG)
5       number_of_points = inp[0]
6
7       # generate points:
8       point_list = []
9       for i in range(number_of_points):
10          point_x = random_number_from_interval(1, X_MAX)
11          point_y = random_number_from_interval(1, Y_MAX)
12          point = (point_x, point_y)
13          point_list.append(point)
14
15      # sort
16      sorted_point_list = sorted(point_list, key=sort_key_function)
17
18      # output
19      for x, y in sorted_point_list:
20          print('x:␣{:5f}\ty:␣{:5f}'.format(round(x, 2), round(y, 2)))
```

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 ein 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
[2] https://en.wikipedia.org/wiki/Mersenne_Twister

Funktion user_input(input_vars, [use_defaults])

- Diese Funktion verlang vom User die geforderten Eingabeparameter und gibt diese als von der Programmiererin gewünschten Datentyp wieder zurück
- Funktion verlangt als ersten Eingabeparameter die Liste *input_vars*
- Falls *use_defaults* == *True* wird der User nicht nach Eingabe gefragt (Dient zum Testen)
- Diese Liste besteht wiederrum aus Listen mit je Länge = 3:
  - 0: Text, welcher dem User ausgegeben wird
  - 1: Datentyp (int/float/str)
  - 2: Default value: Dieser Wert wird zurueckgegeben, falls *use_defaults* == *True*

```
1   x, y = user_input((
2       ('Geben Sie einen X Wert ein', int, 10),
3       ('Geben Sie einen Y Wert ein', int, 5), False):
```

Beispiel anhand fixer Zufallszahlen

- X_MAX = 10, Y_MAX = 10
- Usereingabe: number_of_points := 8
- Annahme der Zufallszahlen wie folgt:

i	0	1	2	3	4	5	6	7	8
ZZ fuer X	5	2	5	3	3	9	1	7	3
ZZ fuer Y	9	7	8	4	3	1	7	2	1

Beispiel anhand fixer Zufallszahlen

For-Schleife zum Erzeugen der point_list:

i := 0

- point_x := 5, point_x := 9
- point_list := [(5,9)]

i := 1

- point_x := 2, point_x := 7
- point_list := [(5,9), (2,7)]

i := 2

- point_x := 5, point_x := 8
- point_list := [(5,9), (2,7), (5,8)]

...

i := 8

- point_x := 3, point_x := 1
- point_list := [(5,9),(2,7),(5,8),(3,4),(3,3),(9,1),(1,7),(7,2),(3,1)]

Beispiel anhand fixer Zufallszahlen

Sortieren der point_list:

- point_list hat nun die Länge 8 mit je (x,y) Tuples als Eintrag
- sorted() ist eine built-in Funktion, welche einen *key* benötigt, um zu wissen, wie sie die Listeneinträge sortieren soll
  - Als Key wird nun einfach (x,y) gegeben, das Sortieren geht nun automatisch aufsteigend zuerst nach x, danach nach y (comperator gibt es in python ab Version 3 nicht mehr)[3]
  - Alternativlösung: Zuerst nach key y sortieren, danach nach x
- sorted_point_list :=
  - [(1,7), (2,7), (3,3), (3,4), (5,8), (5,9), (7,2), (9,1)]

[3] `https://docs.python.org/3.5/library/functions.html#sorted`