ulm university universität
uulm

Institut für Numerische Mathematik

Dr. Andreas F. Borchert
Moritz Carmesin

22. November 2019
Blatt 5

Systemnahe Software I (WS 2019/2020)

Abgabe bis zum 29. November 2018, 14:00 Uhr

Lernziele:

- Einfache Datenstrukturen in C
- Einlesen von Datensätzen aus einer Datei
- *Reservoir-Sampling*-Algorithmus

Aufgabe 7: Städte sortieren

Im Rahmen dieser Aufgabe ist ein Spiel zu programmieren, bei dem der Spieler die Aufgabe erhält, Städte richtig nach ihrer Einwohnerzahl zu sortieren. Für jeden korrekten Zug gibt es einen Punkt. Bei einem Fehler wird das Spiel beendet und die bislang erreichte Punktzahl ausgegeben.

Hier ist ein beispielhafter Spielverlauf:

```
theon$ ./sortthings
Current state:
  1:
Still to be sorted:
  1: Heilbronn
  2: Offenbach am Main
  3: Dortmund
  4: Freiburg im Breisgau
  5: Wiesbaden
  6: Reutlingen
  7: Siegen
  8: Essen
What is to be inserted where? 3 1
Current state:
  1:
     Dortmund
```

```
  2:
Still to be sorted:
  1: Heilbronn
  2: Offenbach am Main
  4: Freiburg im Breisgau
  5: Wiesbaden
  6: Reutlingen
  7: Siegen
  8: Essen
What is to be inserted where? 7 1
Current state:
  1:
     Siegen
  2:
     Dortmund
  3:
Still to be sorted:
  1: Heilbronn
  2: Offenbach am Main
  4: Freiburg im Breisgau
  5: Wiesbaden
  6: Reutlingen
  8: Essen
What is to be inserted where? 6 2
Current state:
  1:
     Siegen
  2:
     Reutlingen
  3:
     Dortmund
  4:
Still to be sorted:
  1: Heilbronn
  2: Offenbach am Main
  4: Freiburg im Breisgau
  5: Wiesbaden
  8: Essen
What is to be inserted where? 4 3
Current state:
  1:
     Siegen
  2:
     Reutlingen
  3:
     Freiburg im Breisgau
  4:
     Dortmund
  5:
```

```
Still to be sorted:
  1: Heilbronn
  2: Offenbach am Main
  5: Wiesbaden
  8: Essen
What is to be inserted where? 1 3
Current state:
  1:
     Siegen
  2:
     Reutlingen
  3:
     Heilbronn
  4:
     Freiburg im Breisgau
  5:
     Dortmund
  6:
Still to be sorted:
  2: Offenbach am Main
  5: Wiesbaden
  8: Essen
What is to be inserted where? 5 5
Current state:
  1:
     Siegen
  2:
     Reutlingen
  3:
     Heilbronn
  4:
     Freiburg im Breisgau
  5:
     Wiesbaden
  6:
     Dortmund
  7:
Still to be sorted:
  2: Offenbach am Main
  8: Essen
What is to be inserted where? 8 7
Sorry, then it is no longer sorted:
     102355 Siegen
     114310 Reutlingen
     122567 Heilbronn
     226393 Freiburg im Breisgau
     276218 Wiesbaden
     586181 Dortmund
-->  582624 Essen
```

```
Bye!
You got 6 points.
theon$
```

Die Städte und ihre Einwohnerzahlen werden Ihnen in der Datei *staedte* auf der Website zur Verfügung gestellt. Diese Daten wurden extrahiert aus dem entsprechenden Datensatz des Statistischen Bundesamts mit dem Stand vom 29. August 2019, wobei für das Spiel nur Städte mit mehr als 100.000 Einwohner ausgesucht wurden.

In jeder Zeile der Datei steht der Name einer Stadt und ihre Einwohnerzahl. Getrennt werden diese beiden Angaben durch einen Doppelpunkt.

Für ein Spiel werden jeweils nur acht zufällig auszuwählende Einträge aus dem Datenbestand benötigt. Diese sind in einer selbst definierten Datenstruktur zu speichern. Dabei sind der Name der Stadt und die Einwohlerzahl in einem gemeinsamen Datentyp zu integrieren.

Die zufällige Auswahl der acht Einträge sollte mit Hilfe des *Reservoir-Sampling*-Algorithmus erfolgen. Der auf Alan G. Waterman zurückgehende Algorithmus (auch Algorithm R genannt) wird von Knuth in The *Art of Computer Programming*, Vol. 2, im Abschnitt 3.4.2. beschrieben. Das Buch finden Sie in der Uni-Bibliothek. Natürlich gibt es auch im Internet Beschreibungen, u.a. auch in der Wikipedia.

Wenn n Einträge von einer Datei unbekannter Länge zufällig auszuwählen sind, werden die folgenden Variablen für den Algorithmus benötigt:

- $R[n]$, ein Array mit n Elementen, in dem die ausgewählten Einträge gespeichert werden (das Reservoir), ab 0 indiziert.
- t, die Anzahl der bereits von der Datei gelesenen Einträge, zu Beginn 0.

Reservoir-Sampling-Algorithmus:

1. Öffnen Sie die Datei und lesen Sie n Einträge ein und speichern Sie sie in R ab. Wenn es weniger Einträge gibt, brechen Sie an der Stelle ab. Ansonsten setzen Sie t auf n.
2. Lesen Sie den nächsten Eintrag aus der Datei ein. Wenn es keine weiteren Einträge mehr gibt, dann sind Sie fertig.
3. Erhöhen sie t um 1 und erzeugen Sie eine Zufallszahl M zwischen 0 und $t - 1$. Wenn $M \geq n$ ist, geht es mit Punkt 2 weiter.
4. Kopieren Sie den aktuellen Eintrag nach $R[M]$ (dabei wird ein früherer Eintrag überschrieben). Weiter geht es mit Punkt 2.

Hinweise: Die Datei können Sie mit *FILE* fp = fopen*("staedte", "r"); zum Lesen öffnen. Die Operation ist erfolgreich, wenn *fp* nicht der Nullzeiger ist. Der Dateiname darf in diesem Fall ausnahmsweise fest einprogrammiert werden. Danach können Sie die Datei zeilenweise einlesen mit *fgets*(*buf*, *buflen*, *fp*), wobei *fgets* beim Dateiende einen Nullzeiger zurückliefert. Wenn Sie alles gelesen haben, können Sie die Dateiverbindung mit *fclose*(*fp*) schließen.

Die Aufgabe eignet sich besonders gut für die Bearbeitung als Team. Sie lässt sich gut in zwei Teilaufgaben aufteilen, die sich jeweils durch eine Funktion umsetzen lassen und die von unterschiedlichen Personen bearbeitet werden können:

- Einlesen und zufällige Auswahl der acht Einträge mit dem *Reservoir-Sampling*-Algorithmus
- Durchführung des Spiels

Sie müssen sich zuvor nur auf die gemeinsame Datenstruktur und die Aufrufsyntax der beiden Funktionen einigen.
Reichen Sie bitte Ihre Lösung mit folgendem Kommando ein:

```
theon$ submit ss1 7 sortthings.c
```

Viel Erfolg!