Programmieren C: Statische Variablen: Zufallszahlen-Funktion

Klaus Kusche

Die einfachste (und jahrelang allgemein übliche) Implementierung von **`rand()`**
hat in Linux wie folgt funktioniert:

- Die **`rand`**-Funktion enthält eine *interne Variable* vom Typ **`int`**,
  die ihren *Wert zwischen zwei Aufrufen behält*.
- Diese Variable wird *beim Programmstart auf 1 initialisiert*
  (oder mittels **`srand`** gesetzt) und bei jedem Aufruf von **`rand`** neu berechnet:

  *Neuer Wert = alter Wert * 1103515245 + 12345* (der *Überlauf* wird *ignoriert*)

  (Microsoft und viele andere Firmen haben dasselbe Prinzip,
  aber etwas andere Konstanten verwendet, siehe
  https://en.wikipedia.org/wiki/Linear_congruential_generator)
- Dann berechnet man aus dieser Variable die Ergebnis-Zufallszahl, indem man
  ihren Wert mit Restrechnung (**`%`**) auf den gewünschen Zahlenbereich reduziert.

Anmerkung: Die erzeugten Zufallszahlen sind ziemlich schlecht.
Heute kennt man *viel bessere* (aber auch viel aufwändigere) Verfahren für Zufallszahlen.

Aufgabe:

- Schreib eine *Funktion* **`myRand`** mit *zwei* **`int`**-*Parametern* **`from`** und **`to`**,
  die eine nach der oben beschriebenen Methode erzeugte ganze Zufallszahl **`z`**
  auf den *Bereich* **`from`** kleinergleich **`z`** kleinergleich **`to`** reduziert
  und als *Ergebnis* zurückliefert.
- Schreib dazu ein **`main`**, das mit dieser Funktion *30 Zahlen zwischen 1 und 6*
  "würfelt" und diese Zahlen sowie deren *Mittelwert* (als *Kommazahl*!) ausgibt.

Hinweise:

- Nimm **`unsigned int`** statt **`int`** für deine Variablen, Parameter und Returnwerte:
  Unsere Zufallszahlen sind immer positiv, und die interne Berechnung
  liefert sonst falsche (weniger gut zufällige) Zahlen.

Zusatzaufgabe:

Da die interne Statusvariable ja *fix initialisiert* ist, liefert die Funktion
bei jedem Programmlauf *dieselben* Zufallszahlen. Wir wollen unser **`myRand`** so umbauen,
dass es sich mittels **`time(NULL)`** *automatisch auf verschiedene Anfangswerte initialisiert*
(so, wie man das beim Standard-**`rand`** mittels **`srand`** macht).

Realisiere dazu folgende Idee:

- **`myRand`** bekommt eine *zweite "ständig lebende" interne Variable*, und zwar
  vom Typ **`bool`**, die anzeigt, ob die Funktion *schon jemals aufgerufen* wurde
  (auf **`false`** initialisieren!).
- Wenn die Variable **`false`** ist (also *beim ersten Aufruf*), wird **`time(NULL)`**
  aufgerufen, das Ergebnis als Startwert in der internen Zufalls-Variable gespeichert,
  und unsere **`bool`**-Variable auf **`true`** gesetzt.