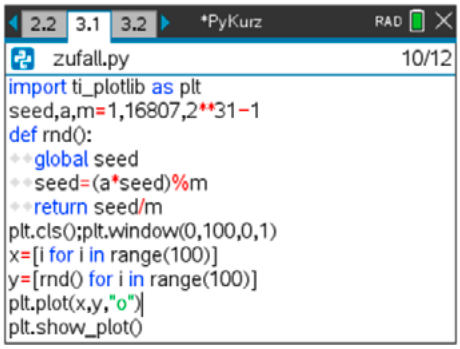
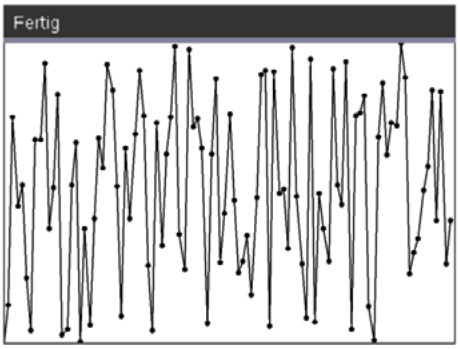
Zufallszahlen

zufall.py

In einer Programmiersprache muss man oft zufällige Zahlen erzeugen. In Python macht dies das Modul random. Aber wir können auch selbst einen Zufallsgenerator erzeugen. Der einfachste ist der *Lehmer-Zufallszahlengerator*. Dazu wählt man einen *seed* oder Startwert x_0. Mit diesem seed erzeugt man die folgende Zahlenfolge: $x_n = a \cdot x_{n-1} \bmod m$. Dabei muss m eine große Primzahl (oder eine hohe Potenz einer Primzahl) sein und a ist eine geschickt gewählte Konstante. Damit ist a ein Generator der multiplikativen Gruppe $\mathbb{Z}_m$.

Hier sind ein paar Beispiele: Auf dem Sinclair ZX81 (einer der ersten Homecomputer) wurde der Zufallsgenerator mit $m = 2^{16} + 1$ und $a = 75$ verwendet. In der Programmiersprache C++ wird der MINST Zufallsgenerator eingesetzt. Da nimmt man die Mersenne-Primzahl (siehe auch 2.2.14) $2^{31} - 1$ und $a = 48271$. IBM hat in der Vergangenheit RANDU verwendet mit $m = 2^{31} - 1$ und $a = 65539$. Dieser Zufallsgenerator erwies sich aber als sehr schlecht und führte zu vielen Problemen, wenn RANDU für Simulationen eingesetzt wurde.





- Wir starten die Implementation in Python mit dem Laden des Grafikmoduls und beginnen mit dem *seed* = 1. Für a und m nehmen wir die Werte von MINST.
- Die Prozedur rnd() weist der globalen Variablen seed jedes Mal einen neuen Wert durch die Formel (a*seed)%m zu. Schließlich wird ein skalierter Wert seed/m zwischen 0 und 1 zurückgegeben.
- Zur Illustration richten wir ein entsprechendes Grafikfenster ein, berechnen eine Folge von 100 Zufallszahlen und stellen diese mit plt.plot als Punktfolge dar. Wir können sehen, dass es sich tatsächlich um eine Zufallsfolge handelt.

Versuche auch die anderen angegebenen Algorithmen.

Bemerkung: global findest du nicht in den Menüs und leider gibt es auch keine nähere Erklärung in der Dokumentation außer dem Hinweis, dass global ein Schlüsselwort ist.