Dokumentation zur großen IHK-Abschlussprüfung Sommer 2020

Entwicklung eines Softwaresystems

Von Felix Warschewski, aixITem GmbH

Inhaltsverzeichnis

[1 Änderungen zu Tag 1 3](#_Toc42761748)

[2 Neufassung der Dokumentation von Tag 1 3](#_Toc42761749)

[3 Diskussion der Testbeispiele 3](#_Toc42761750)

[4 Benutzeranleitung 3](#_Toc42761751)

[4.1 Laufzeitumgebung 3](#_Toc42761752)

[4.2 Installation 3](#_Toc42761753)

[4.3 Dateistruktur 3](#_Toc42761754)

[4.4 Dateiformat 3](#_Toc42761755)

[4.5 Aufrufbefehle 3](#_Toc42761756)

[5 Zusammenfassung und Ausblick 4](#_Toc42761757)

[6 Programmcode im Anhang 4](#_Toc42761758)

# Eigenhändigkeitserklärung



# Änderungen zu Tag 1

Sämtliche Variabeln und Eigenschaften mit dem Datentyp long wurden in double umgeändert. Dies hat den Vorteil, dass sämtliche mathematische Funktionen nicht umgewandelt werden müssen.

Zur Speicherung der Beispiel-LCGs verwende ich eine Konstanten Klasse. Diese beinhaltet eine Liste von Double-Arrays benannt nach dem Beispiel-Verfahren.

Darüber hinaus habe ich anstatt eine einfache Verteilungsklasse eine Verteilungsschnittstelle implementiert, welche zwei Unterklassen hat. Diese Unterklassen besitzen eine Funktion transformiere und in der transformiere-Funktion der Standardnormalverteilt-Klasse wird die Polarmethode angewendet.

# Beschreibung der Zufallszahlengeneratoren und Verfahren

Zu implementieren waren folgende Generatoren und Testverfahren:

* Linearer Kongruenz-Generator mit 6 verschiedenen Einstellungsmöglichkeiten
* Umwandlung gleichverteilter in normalverteilte Zufallszahlen mithilfe der Polar-Methode
* Eigener Zufallszahlengenerator
* Güte-Testverfahren Serielle Autokorrelation
* Güte-Testverfahren Sequenz-Up-Down-Test
* Eigenes Güte-Testverfahren

Der Lineare Kongruenz-Generator(LCG) errechnet Zufallszahlen auf Basis der Kongruenzrechnung. Hierfür sind 4 Eingabeparameter wichtig. Das Modul m, der Multiplikator a, das Inkrement c und der Startwert x0.

# Diskussion der Testbeispiele

Beispiel 1: Test zum Zweck mit den Eingaben und der erwarteten Ausgabe bzw. Fehlermeldungen

Bsp.2: ….

# Benutzeranleitung

## Laufzeitumgebung

Das verwendete Betriebssystem ist Windows 10 64-bit Version. Die Programmiersprache ist C# (Version C# 7.3) und es wurde in der Entwicklungsumgebung Visual Studio 2017 Enterprise Edition geschrieben. Das Zielframework der Konsolen-Applikation ist das .NET Core 2.1.

## Installation

Die Installation erfolgt über das extrahieren der vorhandenen .zip-Datei(Name angeben)

Der Speicherort der extrahierten Dateien ist nun der Ordner in der die Dateien gelesen und gespeichert werden….

## Dateistruktur

## Dateiformat

## Aufrufbefehle

# Vergleich und Interpretation der Ergebnisse

Erörterung des Einflusses der verwendeten Parameter der LCGs oder der Sequenzlänge bezüglich der Güte.

# Zusammenfassung und Ausblick

In weiteren Schritten könnte man die Klassenbibliothek um weitere LCG-Schemas erweitern. Damit stellt man eine breitere Funktionalität des LCG-Verfahrens bereit. Darüber hinaus kann man mehr Verteilungen an die Verteilungs-Schnittstelle anbinden. Dies ermöglicht die Transformation von Koordinaten oder Zahlen in andere Verteilungen. Dafür müsste man lediglich eine neue Klasse anlegen mit der dazugehörigen „Transformiere()“-Methode der Verteilungsschnittstelle. Des Weiteren kann man Klassen an die GüteTests-Schnittstelle anklemmen. Man kann mehr Testverfahren benutzen um die Güte der verschiedenen Generatoren, welche erzeugt wurden, zu bestimmen. Dies kann man alles mit beliebigen Parametern für die Anzahl der zufallsgenerierten Element (Sequenzlänge) und die Ordnung k der Funktion ausführen, indem die Berechne()-Methode aus dem Interface „GüteTests“ verwendet wird. Die drei Schnittstellen sind Hauptbestandteile um diverse Funktionalitäten zu erweitern. Neue Tests einzuführen, in andere Verteilungen umzuwandeln oder komplett andere Zufallszahlen zu erstellen.

# Programmcode im Anhang