TThema, Ziele: CRC Berechnung und C++ Implementation

Aufgabe 1: CRC-8 Berechnungsbeispiel (Papierübung)

- Berechnen Sie die Checksumme für den unten abgebildeten Bytestream. Verwenden Sie dafür das Generatorpolynom $G = x^8 + x^2 + x^1 + 1$ (entspricht CRC-8 CCITT).

```
Message:     0 0 1 0 1 1 0 1 = 0x2D = ´-´
Polynomial:  _________________ = 0x__
CRC:         _________________ = 0x__
```

Berechnung:

Aufgabe 2: Verifikation empfangener Daten mittels CRC

Nach einer seriellen Übertragung wurden die folgenden drei Messages empfangen. Dabei entspricht das erste Byte den Daten und das zweite der CRC-Prüfsumme (MSB first). Das verwendete Generatorpolynom ist dasselbe wie in Aufgabe 1.

- Überprüfen Sie mittels CRC, welche der drei fehlerfrei übertragen wurden und welche nicht.

Message 1:	1 1 0 1 1 0 0 1	0 0 0 0 0 0 0 1	0xD901	□ korrekt □ falsch
Message 2:	0 1 0 1 0 0 0 1	0 1 1 1 0 1 1 1	0x5177	□ korrekt □ falsch
Message 3:	0 1 1 0 0 0 0 1	0 0 1 0 0 0 0 0	0x6120	□ korrekt □ falsch

Aufgabe 3: C++ Implementation

- Implementieren Sie eine Klasse `Crc`, welche die Prüfsumme CRC-8 CCITT mit Schiebeoperationen auf Bit-Ebene berechnet.
- Um den Algorithmus zu überprüfen, soll ein Testprogramm geschrieben werden, welches entweder die Prüfsumme der Datenbytes aus Aufgabe 2 berechnet, oder die Prüfsumme von einer Datei ausgibt. Der Dateiname soll der Applikation als Argument übergeben werden.

Aufgabe 4: C++ Implementation mit einer Lookup Tabelle

- Implementieren Sie eine Klasse `Crc`, welche die Prüfsumme CRC-8 CCITT mit Hilfe einer 8 Bit breiten Lookup Tabelle berechnet. Die Lookup Tabelle kann im Voraus berechnet werden oder sogar konstant im ROM abgelegt werden.
- Um den Algorithmus zu überprüfen soll ein Testprogramm geschrieben werden, welches entweder die Prüfsumme der Datenbytes aus Aufgabe 2 berechnet, oder die Prüfsumme von einer Datei ausgibt. Der Dateiname soll der Applikation als Argument übergeben werden.
- Variieren Sie die Dateigrösse und vergleichen Sie die Berechnungszeit mit der Variante aus der Aufgabe 3.

Hinweis File I/O:

```
#include <fstream>

…

f.open(argv[1], ios::in | ios::binary);

// determine file size
f.seekg(0, ios::end);
len = f.tellg();
f.seekg(0, ios::beg);

// Allocate byte buffer
buf = new uint8_t[len];

// read file content into byte buffer
f.read((char*)buf, len);
f.close();

…

delete[] buf;  // free allocated memory
```