Aufgabe 4g: Modulare Programmierung

1 Lernziele

Anwenden der modularen Programmierung sowie Vertiefen der Kenntnisse über die Gültigkeit und Sichtbarkeit von Variablen. Sie können erste Erfahrungen sammeln im gemeinsamen Erstellen von Software: Man einigt sich auf eine Schnittstelle und jeder Beteiligte löst unabhängig vom anderen einen Teil der Aufgabe.

Die Aufteilung, wer aus der Zweiergruppe welches Modul implementiert, bleibt Ihnen überlassen. Am Ende des Praktikums müssen beide Module zusammen kompilierbar sein und das Programm die gewünschte Funktionalität aufweisen.

2 Aufgabe

Erstellen Sie ein Testprogramm für eine Datenstruktur `polynom_t`, die die folgenden Funktionen bereitstellt:

```
polynom_t *createPolynom(void);
void setCoeff(polynom_t *p, int order, double coeff);
double getValue(polynom_t *p, double x);
polynom_t *getDerivation(polynom_t *p);
double getIntegralValue(polynom_t *p, double left, double right);
void destroyPolynom(polynom_t *p);
```

Schreiben Sie ein Modul `polynom.c`, das die obige Schnittstelle implementiert. Stellen Sie sicher, das auf die Komponenten der Struktur nur innerhalb des Moduls zugegriffen werden kann. Überlegen Sie sich, welche Komponenten die Struktur `polynom_t` haben muss und definieren Sie mittels `typedef` einen Datentyp.

Schreiben Sie ein Modul `main.c`, das die obige Polynom-Implementierung testet. Dabei soll jede Funktion der Schnittstelle getestet werden. Der Test soll vollautomatisch ablaufen, d.h. es wird eine Übersicht über erfolgreiche und fehlgeschlagene Teiltests erstellt.

3 Testat

Voraussetzung ist jeweils ein fehlerfreies, korrekt formatiertes Programm. Der korrekte Programmlauf muss anhand einer Beispieleingabe nachgewiesen werden. Sie müssen in der Lage sein, Ihr Programm zu erklären.