C++ WS 03/04

Aufgabe : Polynomarithmetik und Linearfaktorzerlegung (86 Punkte)

Gegeben seien zwei Polynome p und q von ganzzahligem, nichtnegativem Grad $n := \text{grad}(p)$ bzw. $m := \text{grad}(q)$ mit

$$p(x) := \sum_{i=0}^{n} p_i x^i = p_0 + p_1 x + \ldots + p_n x^n \quad \text{mit } p_0, \ldots, p_n \in \mathbb{R}, p_n \neq 0,$$

$$q(x) := \sum_{i=0}^{m} q_i x^i = q_0 + q_1 x + \ldots + q_m x^m \quad \text{mit } q_0, \ldots, q_m \in \mathbb{R}, q_m \neq 0.$$

Das Polynom $s := p + q$ vom Grad l, das sich als die Summe der Polynome p und q ergibt, ist definiert durch

$$s(x) := p(x) + q(x) := \sum_{i=0}^{l} (p_i + q_i) x^i \quad \text{mit } p_l + q_l \neq 0,$$

wobei

$$p_i := 0 \quad \text{für } i > n,$$
$$q_i := 0 \quad \text{für } i > m$$

mit 0 auffüllen

zu beachten ist.
Der Grad l des Polynoms s ist für $n \neq m$ gegeben durch $l = \max\{n, m\}$. Im Fall $n = m$ kann $l < \max\{n, m\} = n$ gelten. Dieser Fall tritt ein, falls $p_n + q_n = 0$ gilt. Der Grad l ergibt sich dann als der maximale Index j mit $p_j + q_j \neq 0$.

Mittels des Hornerschemas kann zu gegebenem $x_0 \in \mathbb{R}$ ein Polynom r vom Grad $n - 1$ mit

$$r(x) := \sum_{i=0}^{n-1} r_i x^i$$

bestimmt werden, für das die Beziehung

$$p(x) = r(x) \cdot (x - x_0) + p(x_0)$$

gilt. Dieses Verfahren kann zusätzlich zur Bestimmung des Funktionswertes $p(x_0)$ benutzt werden. Die Koeffizienten r_i, $i = 0, \ldots, n-1$, des Polynoms r und der Funktionswert $p(x_0)$ ergeben sich aus

(H)
$r_{n-1} := p_n;$
Für $i = n-2, n-3, \ldots, 0$
$\quad r_i := r_{i+1} x_0 + p_{i+1};$
$p(x_0) := r_0 x_0 + p_0;$

Ist x_0 insbesondere eine Nullstelle von p, d.h. es gilt $p(x_0) = 0$, so kann der Linearfaktor $(x - x_0)$ in der Form

$$p(x) = r(x) \cdot (x - x_0)$$

abgespalten werden. Um weitere Nullstellen von p zu finden genügt es, das Polynom r zu betrachten. Im Weiteren soll der folgende Algorithmus verwendet werden, um die (bekannten) Nullstellen eines Polynoms abzuspalten:

(L)

Solange grad$(p) > 1$
  Wiederhole
    Wähle $x_0 \in \mathbb{R}$; (Einlesen: x_0)
    Bestimme ein Polynom r mit
    $p(x) = r(x) \cdot (x - x_0) + p(x_0)$;
    Falls $p(x_0) = 0$
      $p = r$; (Ausgeben: r, x_0)
  solange $p(x_0) \neq 0$;

Schreiben Sie ein C++-Programm, welches das Rechnen mit Polynomen ermöglicht. Das Ziel ist es, ein gegebenes Polynom in seine Linearfaktoren zu zerlegen. (Bemerkung: Diese reelle Zerlegung ist nicht für alle Polynome möglich.)

Die Polynome sollen mittels einer Struktur `struct polynom` dargestellt werden. Der Speicher für die Polynomkoeffizienten ist dynamisch zu allokieren. Für das Rechnen mit Polynomen sind die nachfolgenden Operatoren und Funktionen zu vereinbaren.

Gehen Sie wie folgt vor:

a) Definieren Sie eine Struktur `struct polynom` mit einer Komponenten `int grad` für den Grad des Polynoms und einer Komponenten `double* koeff` als Zeiger auf ein dynamisches Feld mit den Koeffizienten des Polynoms.

b) Erstellen Sie eine Funktion `polynom createpoly(int n)`, die eine Größe vom Typ `polynom` für ein Polynom vom Grad `n` vereinbart. Belegen Sie die Komponente `grad` entsprechend und allokieren Sie dynamischen Speicherplatz für die Koeffizienten des Polynoms. Weisen Sie den Koeffizienten den Wert Null zu.

c) Schreiben Sie eine Funktion `void deletepoly(polynom p)`, die den dynamisch allokierten Speicherplatz für die Koeffizienten des Polynoms `p` wieder freigibt.

d) Überladen Sie den Eingabeoperator `>>` für den Typ `polynom` zum Einlesen der Koeffizienten eines Polynoms von einem `istream`. Hierbei ist davon auszugehen, dass die entsprechende Größe vom Typ `polynom` bereits mit Hilfe der Funktion `createpoly` erzeugt wurde. Speichern Sie die Koeffizienten des Polynoms in aufsteigender Reihenfolge beginnend mit dem Index 0 ab. Versehen Sie die Eingabe mit einem erläuternden Text.

e) Überladen Sie den Ausgabeoperator << für die Ausgabe eines Polynoms auf einen `ostream`. Koeffizienten mit dem Wert Null sollen ignoriert werden. Gestalten Sie die Ausgabe derart, dass z.B. das Polynom $x^3 + 2x + 1$ in der Form

```
p(x) = 1*x^3 + 2*x + 1
```

ausgegeben wird. Achten Sie auf das Setzen der Zeichen + und ^.

f) Überladen Sie den Operator + für die Addition zweier Größen `p` und `q` vom Typ `polynom`. Bestimmen Sie zunächst den Grad der Summe `p+q`. Beachten Sie insbesondere die Erläuterungen hierzu im einführenden Text. Vereinbaren Sie dann ein Polynom entsprechenden Grades für die Summe. Addieren Sie in der Summe zunächst die Größe `p` und dann die Größe `q`. Achten Sie auf die Feldgrenzen!

g) Schreiben Sie eine Funktion
`void hornerdiv(polynom p, polynom& r, double x0, double& px0)`,
die mittels des Hornerschemas (H) das Polynom `p` vom Grad `n` bezüglich des Linearfaktors $(x - x_0)$ zerlegt. Das Polynom `r` vom Grad `n-1` mit

$$p(x) = r(x) \cdot (x - x_0) + p(x_0).$$

und der Funktionswert `px0` $= p(x_0)$ sind als Referenzargumente zurückzugeben. Hierbei ist davon auszugehen, dass die Größe `r` vom Typ `polynom` bereits zuvor mittels der Funktion `createpoly` erzeugt wurde. Ist der Grad des übergebenen Polynoms `r` ungleich `n-1`, so muss der entsprechende Speicherplatz zuerst mittels `deletepoly` freigegeben und dann neu allokiert werden.

h) Erstellen Sie eine Funktion `void copy(polynom& p, polynom q)`, welche die Koeffizienten des Polynoms `p` mit den Koeffizienten des Polynoms `q` belegt. Hierbei ist davon auszugehen, dass die Größe `p` vom Typ `polynom` bereits zuvor mittels der Funktion `createpoly` erzeugt wurde. Falls sich der Grad von `p` von dem Grad von `q` unterscheidet, muss der Speicherplatz für die Koeffizienten von `p` zunächst freigegeben und neu allokiert werden.

i) Verfassen Sie ein Hauptprogramm mit den folgenden Elementen:

- Lesen Sie den Grad eines gegebenen Polynoms ein. Stellen Sie sicher, dass nur nichtnegative Werte akzeptiert werden.
- Vereinbaren Sie ein Polynom `p` entsprechenden Grades und lesen Sie die Koeffizienten ein.
- Geben Sie das Polynom `p` wieder auf dem Bildschirm aus.
- Zerlegen Sie das Polynom `p` mittels des Algorithmus (L) in seine Linearfaktoren. Geben Sie jeweils die Zwischenergebnisse `r` und `x0` mit einem begleitenden Text aus.
- Geben Sie sämtlichen dynamisch allokierten Speicherplatz wieder frei.

Vergessen Sie nicht alle benötigten Variablen zu deklarieren. Verwenden Sie die in den Teilen b)-e) und g)-h) definierten Funktionen und Operatoren.