Familienname:

Vorname:

Matrikelnummer:

Aufgabe 1 (3 Punkte):
Aufgabe 2 (2 Punkte):
Aufgabe 3 (1 Punkte):
Aufgabe 4 (3 Punkte):
Aufgabe 5 (4 Punkte):
Aufgabe 6 (5 Punkte):
Aufgabe 7 (6 Punkte):
Aufgabe 8 (2 Punkte):
Aufgabe 9 (2 Punkte):
Aufgabe 10 (2 Punkte):

Gesamtpunktzahl:

Schriftlicher Nachtest zu C++ (Bearbeitungszeit: 90 Minuten)
VU Einführung ins Programmieren für TM

01. März 2012

Aufgabe 1 (3 Punkte). Schreiben Sie die Klassendefinition einer Klasse `Polynomial` zur Speicherung von Polynomen p beliebigen Grades $n \in \mathbb{N}$. Dabei sollen die Koeffizienten a_j in der Monomdarstellung $p(x) = \sum_{j=0}^{n} a_j x^j$ als `vector<double>` gespeichert werden. Der Grad n soll hierbei nur implizit über die Länge des Koeffizientenvektors gespeichert sein. Die Klasse soll zudem über die Methoden `getPolDeg`, `getPolCoeff`, `setPolCoeff`, `addPolCoeff`, sowie einen Konstruktor verfügen. Außerdem soll es die Möglichkeit geben, zwei Polynome r und q mittels $p = r * q$ zu multiplizieren, und das Ergebnis wieder als Polynom abzuspeichern. Achten Sie darauf, die Zugriffsspezifizierer sinnvoll zu verwenden.
Hinweis: In dieser Aufgabe ist nur die Klassendefinition gefragt. Es soll noch keine Funktionalität implementiert werden.

Aufgabe 2 (2 Punkte). Implementieren Sie einen Konstruktor für `Polynomial`, der zu gegebenem $n \in \mathbb{N}$ das Nullpolynom $p(x) = 0$ als Polynom vom Grad n erzeugt.

Aufgabe 3 (1 Punkt). Implementieren Sie die Methode `getPolDeg`, die den Grad des Polynoms p zurückgibt.

Aufgabe 4 (3 Punkte). Schreiben Sie eine Funktion `getPolCoeff`, die für einen gegebenen Index j den Koeffizienten a_j zurückgibt. Beachten Sie explizit den Fall, dass j größer ist als der (intern gespeicherte) Grad $n \in \mathbb{N}$ des Polynoms. In diesem Fall werde 0 zurückgegeben.

Aufgabe 5 (4 Punkte). Schreiben Sie eine Funktion `setPolCoeff`, die für gegebenen Index j und Wert b dem Koeffizienten a_j des Polynoms den Wert b zuweist. Beachten Sie den Fall, dass j größer ist als der (intern gespeicherte) Grad $n \in \mathbb{N}$. In diesem Fall und für $b \neq 0$ muss der Koeffizientenvektor entsprechend verlängert und initialisiert werden.

Aufgabe 6 (5 Punkte). Was macht die folgende Funktion bei Übergabe des Polynoms $p(x) = 2 + 3x + 7x^2 + 3x^3$ und $t = 2$? Geben Sie tabellarisch wieder, welchen Wert die Variablen zu den angegebenen Zeitpunkten haben. Welche Funktionalität wird durch die Funktion bereitgestellt?

```
double function(Polynomial p, double t) {
  int n = p.getPolDeg();
  double result = 0;
  double tmp = 1;

  for (int j=1; j<=n; ++j) {
    result = result + j*tmp*p.getPolCoeff(j);
    tmp = tmp * t;
    /* Wert der Variablen zu diesem Zeitpunkt */
  }

  /* Wert der Variablen zu diesem Zeitpunkt */
  return result;
}
```

Verlängern Sie die folgende Tabelle geeignet und füllen Sie sie aus:

j	t	result	tmp

Aufgabe 7 (6 Punkte). Das Produkt $r = pq$ zweier Polynome $p(x) = \sum_{j=0}^{m} a_j x^j$ und $q(x) = \sum_{k=0}^{n} b_k x^k$ ist wieder ein Polynom. Schreiben Sie eine Methode, die das Polynom r berechnet und als Objekt der Klasse `Polynomial` speichert. Sie dürfen dazu die zusätzliche Methode

```
void addPolCoeff(int j, double value)
```

der Klasse `Polynomial` verwenden, die den aktuellen Koeffizienten c_j des Polynoms um `value` erhöht (d.h. $c_j \leftarrow c_j +$ `value`). Die Multiplikation von zwei Polynomen soll zudem mittels $*$ möglich sein, d.h. der Befehl $r = p * q$ liefert tatsächlich das Zielpolynom und speichert dieses in r ab.
Hinweis. Überlegen Sie sich zunächst, welchen Grad das Produkt r hat, und wie sich die Koeffizienten c_ℓ von r berechnen.

Aufgabe 8 (2 Punkte). Was sind die Unterschiede zwischen einer imperativen Programmiersprache wie C und einer objektorientierten Programmiersprache wie C++.

Aufgabe 9 (2 Punkte). Was bedeuten die Zugriffsspezifizierer `private`, `protected` und `public`? Was sind die Vorteile Ihrer Verwendung?

Aufgabe 10 (2 Punkte). Schreiben Sie die Klassendefinition der Klasse `complexPoly`, die Sie öffentlich von `Polynomial` ableiten. In dieser Klasse sollen Polynome mit komplexen Koeffizienten gespeichert werden können. Zusätzlich zum Koeffizientenvektor für die Realteile, soll also ein weiterer Vektor für die Imaginärteile der Koeffizienten gespeichert werden. Außerdem soll die Klasse zusätzlich über die Methoden `getImagCoeff` und `setImagCoeff` verfügen um auf die Imaginärteile der Koeffizienten zugreifen zu können.
Hinweis: Analog zu Aufgabe 1 ist hier nur die Klassendefinition gefragt.