Familienname:

Vorname:

Matrikelnummer:

Aufgabe 1 (3 Punkte):
Aufgabe 2 (3 Punkte):
Aufgabe 3 (1 Punkt):
Aufgabe 4 (3 Punkte):
Aufgabe 5 (1 Punkt):
Aufgabe 6 (3 Punkte):
Aufgabe 7 (5 Punkte):
Aufgabe 8 (2 Punkte):
Aufgabe 9 (5 Punkte):
Aufgabe 10 (5 Punkte):
Aufgabe 11 (5 Punkte):
Aufgabe 12 (4 Punkte):

Gesamtpunkte (40 Punkte):

Schriftlicher Test (120 Minuten)
VU Einführung ins Programmieren für TM

28. Februar 2017

Aufgabe 1 (3 Punkte). Schreiben Sie eine rekursive C++ Funktion `binomial`, die mit Hilfe des Additionstheorems

$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1} \quad \text{für } k, n \in \mathbb{N} \text{ mit } 2 \leq k < n$$

den Binomialkoeffizienten berechnet. Beachten Sie dabei

$$\binom{n}{1} = n \quad \text{und} \quad \binom{n}{n} = 1 \quad \text{für alle } n \geq 1.$$

Stellen Sie mittels `assert` sicher, dass für die Input-Parameter $1 \leq k \leq n$ gilt.

Hinweis. In den folgenden Aufgaben betrachten wir Vektoren $x \in \mathbb{R}^n$ als Objekte der C++ Klasse `Vector`. Neben Konstruktor, Kopierkonstruktor, Destruktor und Zuweisungsoperator, gibt es eine Methode, um die Dimension n auszulesen (`size`), das Maximum zu bestimmen (`max`), den Vektor zu sortieren (`sort`), mehrfache Einträge zu streichen (`unique`) und alle Einträge außerhalb eines Intervalls $[C_{\min}, C_{\max}]$ zu streichen (`cut`). Die Koeffizienten x_j erhält man mittels `x(j)`, wobei die Indizes $j = 1, \ldots, n$ im mathematisch üblichen Sinn verwendet werden.

```
class Vector {
private:
  int n;
  double* coeff;
public:
  Vector(int, double = 0);
  Vector(const Vector&);
  ~Vector();
  Vector& operator=(const Vector&);

  int size() const;
  double& operator()(int);
  const double& operator()(int) const;

  double max() const;
  void sort();
  void unique();
  void cut(double Cmin, double Cmax);
};
```

Aufgabe 2 (3 Punkte). Schreiben Sie den Konstruktor der Klasse `Vector`, der einen Vektor $x \in \mathbb{R}^n$ anlegt, wobei der optionale Parameter `init` der Initialisierungswert für die Koeffizienten sei (d.h. $x_j =$ `init` für alle $j = 1, \ldots, n$). Stellen Sie mittels `assert` sicher, dass die Dimension $n \geq 0$ ist. Für $n = 0$ werde der leere Vektor angelegt (d.h. es wird kein Speicher allokiert).

Lösung zu Aufgabe 2.

Aufgabe 3 (1 Punkt). Schreiben Sie den Destruktor der Klasse `Vector`.

Lösung zu Aufgabe 3.

Aufgabe 4 (3 Punkte). Schreiben Sie den Zuweisungsoperator der Klasse `Vector`.

Lösung zu Aufgabe 4.

Aufgabe 5 (1 Punkt). Schreiben Sie den Zugriffsoperator `( )` für konstante Objekte der Klasse `Vector` für den Koeffizientenzugriff auf x_j mittels `x(j)`. Stellen Sie mittels `assert` sicher, dass die Koeffizienten im zulässigen Bereich sind, d.h. $j \in \{1, \ldots, n\}$ für $x \in \mathbb{R}^n$. Beachten Sie, dass der Speichervektor in C++ intern mit $\ell = 0, \ldots, n-1$ indiziert wird.

Lösung zu Aufgabe 5.

Aufgabe 6 (3 Punkte). Schreiben Sie die Methode `max` der Klasse `Vector`, die das Maximum $\max_{j=1,\ldots,n} x_j$ von $x \in \mathbb{R}^n$ zurückgibt.

Lösung zu Aufgabe 6.

Aufgabe 7 (5 Punkte). Schreiben Sie die Methode `sort` der Klasse `Vector`, die den Vektor $x \in \mathbb{R}^n$ aufsteigend sortiert.

Beispiel. $x = (1, 7, 2, 5, 6, 5, 9, 6)$ wird durch Aufruf von `sort()` zu $x = (1, 2, 5, 5, 6, 6, 7, 9)$.

Lösung zu Aufgabe 7.

Aufgabe 8 (2 Punkte). Welchen Aufwand hat Ihre Implementierung der Methode `sort` für $x \in \mathbb{R}^n$? Falls die Funktion für $n = 10^5$ eine Laufzeit von 16 Sekunden hat, welche Laufzeit erwarten Sie für $n = 10^6$?

Lösung zu Aufgabe 8.

Aufgabe 9 (5 Punkte). Schreiben Sie die Methode `unique` der Klasse `Vector`, die einen Vektor $x \in \mathbb{R}^n$ sortiert und alle mehrfach vorkommenden Einträge x_j streicht. Der Vektor x soll mit dem gekürzten Vektor überschrieben werden (d.h. Vektorlänge und dynamisches Koeffizientenfeld müssen ggf. angepasst werden!)

Beispiel. $x = (1, 7, 2, 5, 6, 5, 9, 6)$ wird durch Aufruf von `unique()` zu $x = (1, 2, 5, 6, 7, 9)$.

Lösung zu Aufgabe 9.

Aufgabe 10 (5 Punkte). Schreiben Sie die Methode `cut` der Klasse `Vector`, die aus einem Vektor $x \in \mathbb{R}^n$ alle Einträge x_j mit $x_j < C_{\min}$ oder $x_j > C_{\max}$ streicht. Der Vektor x soll mit dem gekürzten Vektor überschrieben werden (d.h. Vektorlänge und dynamisches Koeffizientenfeld müssen ggf. angepasst werden!)

Beispiel. $x = (1, 7, 2, 5, 6, 5, 9, 6)$ wird durch Aufruf von `cut(2,6)` zu $x = (2, 5, 6, 5, 6)$.

Lösung zu Aufgabe 10.

Aufgabe 11 (5 Punkte). Schreiben Sie eine C++ Funktion `eratosthenes`, die für gegebenes $n \geq 2$ den Vektor aller Primzahlen $\leq n$ zurückgibt. Gehen Sie dazu wie folgt vor:

- Stellen Sie mittels `assert` sicher, dass $n \geq 2$ gilt.
- Legen Sie einen Vektor $x = (1, 2, 3, \ldots, n)$ an.
- Setzen Sie alle Vielfachen von $j = 2, 3, \ldots$ im Vektor x auf Null.
- Verwenden Sie `cut(2,n)`, um alle Nicht-Primzahlen aus dem Vektor zu eliminieren.

Beispiel. `Vector x = eratosthenes(25);` soll den Vektor $x = (2, 3, 5, 7, 11, 13, 17, 19, 23)$ generieren.

Lösung zu Aufgabe 11.

Aufgabe 12 (4 Punkte). Was ist der Shell-Output der folgenden Funktion bei Aufruf `Vector y = foo(12)`? Was ist die mathematische Funktionalität der Funktion `foo`?

```
Vector foo(int n) {
  Vector x = eratosthenes(n);
  Vector y(x.size());
  for (int j=1; j<=x.size(); ++j) {
    while (n/(int) x(j)*x(j) == n) {
      n = n/x(j);
      y(j) = y(j) + 1;
      cout << n << ":";
      for (int k=1; k<=x.size(); ++k) {
        cout << " " << y(k);
      }
      cout << endl;
    }
  }
  return y;
}
```

Lösung zu Aufgabe 12.