
Familienname:

Vorname:

Matrikelnummer:

Aufgabe 1 (2 Punkte):

Aufgabe 2 (1 Punkt):

Aufgabe 3 (1 Punkt):

Aufgabe 4 (1 Punkt):

Aufgabe 5 (2 Punkte):

Aufgabe 6 (2 Punkte):

Aufgabe 7 (3 Punkte):

Aufgabe 8 (3 Punkte):

Aufgabe 9 (3 Punkte):

Aufgabe 10 (1 Punkt):

Aufgabe 11 (2 Punkte):

Aufgabe 12 (4 Punkte):

Aufgabe 13 (4 Punkte):

Aufgabe 14 (6 Punkte):

Aufgabe 15 (2 Punkte):

Aufgabe 16 (3 Punkte):

Gesamtpunkte (40 Punkte):

Schriftlicher Test (120 Minuten)

VU Einführung ins Programmieren für TM

22. Januar 2018

Hinweis. In den folgenden Aufgaben sollen Sie zunächst Teile der Funktionalität der Klasse **Complex** implementieren, die den elementaren Umgang mit komplexen Zahlen $z = a + bi \in \mathbb{C}$ realisiert. Dabei sollen Realteil $\text{Re}(z) := a \in \mathbb{R}$ und Imaginärteil $\text{Im}(z) := b \in \mathbb{R}$ als **double** gespeichert werden.

```
class Complex {
private:
    double re;
    double im;
public:
    Complex(double real=0, double imag=0);
    double real() const;
    double imag() const;
    double abs() const;
    const Complex operator-() const;
    const Complex operator~() const;
};

const Complex operator+(const Complex&, const Complex&);
const Complex operator-(const Complex&, const Complex&);
const Complex operator*(const Complex&, const Complex&);
const Complex operator/(const Complex&, const Complex&);
const bool operator<(const Complex&, const Complex&);
```

Aufgabe 1 (2 Punkte). Schreiben Sie den Konstruktor der Klasse `Complex`, dem optional Real- und Imaginärteil von $z = a + bi$ übergeben werden können.

Lösung zu Aufgabe 1.

Aufgabe 2 (1 Punkt). Schreiben Sie die Methode `real` der Klasse `Complex`, die den Realteil $\operatorname{Re}(z)$ von $z \in \mathbb{C}$ zurückgibt.

Lösung zu Aufgabe 2.

Aufgabe 3 (1 Punkt). Schreiben Sie die Methode `abs` der Klasse `Complex`, die den Absolutbetrag $|z| := \sqrt{a^2 + b^2}$ von $z = a + bi \in \mathbb{C}$ zurückgibt.

Hinweis. Zur Berechnung der Wurzel können Sie die Funktion `sqrt` verwenden.

Lösung zu Aufgabe 3.

Aufgabe 4 (1 Punkt). Schreiben Sie den Operator `~` der Klasse `Complex`, der für $z = a + bi \in \mathbb{C}$ die komplex konjugierte Zahl $\bar{z} = a - bi \in \mathbb{C}$ zurückgibt.

Lösung zu Aufgabe 4.

Aufgabe 5 (2 Punkte). Warum dürfen bei der Klasse `Complex` Kopierkonstruktor, Destruktor und Zuweisungsoperator fehlen? Was ist die Funktionalität des Zuweisungsoperators, wenn er nicht explizit geschrieben wird?

Lösung zu Aufgabe 5.

Aufgabe 6 (2 Punkte). Schreiben Sie den Code für die Multiplikation $w \cdot z$ zweier komplexer Zahlen $w, z \in \mathbb{C}$. Überladen Sie dazu den `*` Operator. Um die entsprechende Formel herzuleiten, beachten Sie, dass $i^2 = -1$!

Lösung zu Aufgabe 6.

Aufgabe 7 (3 Punkte). Schreiben Sie den Code für die Division w/z zweier komplexer Zahlen $w, z \in \mathbb{C}$. Überladen Sie dazu den `/` Operator. Stellen Sie mittels `assert` sicher, dass $z \neq 0$ gilt. Um die entsprechende Formel herzuleiten, erweitern Sie den Bruch mit der komplex konjugierten Zahl \bar{z} .

Lösung zu Aufgabe 7.

Aufgabe 8 (3 Punkte). Überladen Sie den Vergleichsoperator $<$ für komplexe Zahlen. Für alle $z, w \in \mathbb{C}$ definieren wir

$$z < w \iff \begin{cases} |z| < |w| & \text{oder} \\ (|z| = |w| \text{ und } \operatorname{Re}(z) < \operatorname{Re}(w)) & \text{oder} \\ (|z| = |w| \text{ und } \operatorname{Re}(z) = \operatorname{Re}(w) \text{ und } \operatorname{Im}(z) < \operatorname{Im}(w)). \end{cases}$$

Falls eine der drei Bedingungen zutrifft, soll $z < w$ als Ergebnis **true** zurückgeben. Anderenfalls werde **false** zurückgegeben.

Lösung zu Aufgabe 8.

Hinweis. In den folgenden Aufgaben sollen Sie Teile der Funktionalität der Klasse **Vector** implementieren, die für die Speicherung von Vektoren $x \in \mathbb{C}^n$ verwendet werden kann.

```
class Vector {
private:
    int n;
    Complex* entry;
public:
    Vector(int n=0);
    ~Vector();
    Vector(const Vector&);
    const Vector& operator=(const Vector&);
    const Complex& operator()(int) const;
    Complex& operator()(int);
    int size() const;
    int find(const Complex& z) const;
    void sort();
};
```

Aufgabe 9 (3 Punkte). Schreiben Sie den Konstruktor der Klasse **Vector**, der den Nullvektor $x = (0, \dots, 0) \in \mathbb{C}^n$ anlegt. Stellen Sie mittels **assert** sicher, dass $n \geq 0$ gilt. Für $n = 0$ soll der leere Vektor generiert werden, d.h. es wird kein Speicher allokiert.

Lösung zu Aufgabe 9.

Aufgabe 10 (1 Punkt). Schreiben Sie den Destruktor der Klasse `Vector`.

Lösung zu Aufgabe 10.

Aufgabe 11 (2 Punkte). Schreiben Sie den Zugriffsoperator (`()`) für nicht-konstante Objekte der Klasse `Vector` für den Koeffizientenzugriff auf $x_j \in \mathbb{C}$ mittels `x(j)`. Stellen Sie mittels `assert` sicher, dass die Koeffizienten im zulässigen Bereich sind, d.h. $0 \leq j \leq n - 1$.

Lösung zu Aufgabe 11.

Aufgabe 12 (4 Punkte). Schreiben Sie den Zuweisungsoperator = der Klasse `Vector`.

Hinweis. Beachten Sie den Fall, dass der gegebene Vektor leer ist.

Lösung zu Aufgabe 12.

Aufgabe 13 (4 Punkte). Schreiben Sie die Methode `find` der Klasse `Vector`, die für den Vector $x \in \mathbb{C}^n$ und eine gegebene Zahl $z \in \mathbb{C}$ den größten Index $j \in \{0, \dots, n-1\}$ zurückgibt, sodass gilt

$$|x_j - z| = \min_{k=0, \dots, n-1} |x_k - z|.$$

Lösung zu Aufgabe 13.

Aufgabe 14 (6 Punkte). Schreiben Sie die Methode `sort` der Klasse `Vector`, die den Vector $x \in \mathbb{C}^n$ aufsteigend sortiert (gemäß dem Vergleichsoperator $<$ aus Aufgabe 8). Der gegebene Vektor x soll einfach überschrieben werden.

Beispiel. Der Vektor $x = (3, 2, 1, i, 1+i, 1-i, 0)$ wird durch $x = (0, i, 1, 1-i, 1+i, 2, 3)$ überschrieben.

Lösung zu Aufgabe 14.

Aufgabe 15 (2 Punkte). Bestimmen Sie den Aufwand Ihrer Funktion aus Aufgabe 14 für einen Vektor der Länge n . Falls die Funktion für $n = 10^3$ eine Laufzeit von 1,5 Sekunden hat, welche Laufzeit erwarten Sie für $n = 5 \cdot 10^3$? Begründen Sie Ihre Antwort!

Lösung zu Aufgabe 15.

Aufgabe 16 (3 Punkte). Was ist der Output des folgenden Programms?

```
#include <iostream>
using std::cout;
using std::endl;

class A {
private:
    int data;
public:
    A(int data = 0) {
        this->data = data;
        cout << "+A: " << data << endl;
    };
    ~A() {
        cout << "-A" << endl;
    }
};

class B {
private:
    A data;
public:
    B(int data = 1) {
        cout << "+B: " << data << endl;
    }
    ~B() {
        cout << "-B" << endl;
    }
};

class C : public A {
private:
    int data;
public:
    C(int first , int second) : A(first) {
        data = second;
        cout << "+C: " << first << ", " << second << endl;
    }
    ~C() {
        cout << "-C" << endl;
    }
};

int main() {
    A x(2);
    cout << "***" << endl;
    B y(4);
    cout << "***" << endl;
    C z(1,2);
    return 0;
}
```

Lösung zu Aufgabe 16.

