Dirk Praetorius,
Michele Ruggeri

Wintersemester 2017/18
11./12.01.2018

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 10

Aufgabe 10.1. Was ist der Output des folgenden Programms? Erklären Sie warum! Was ist der Unterschied zwischen den verschiedenen verwendeten Variablentypen?

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

const int proc(int & input){input = input*2; return input;}
int proc(const int & input){ int output = input; return output;}

void swap(int& x, int& y){
int tmp;
tmp = x;
x = y;
y = tmp;
}

void swap(const int& x,const int& y){;}

int main() {

int var1 = 1;
int var2 = 2;
int var3 = proc(var1);
int var4 = proc(var2);
const int var5 = proc(var1);
const int var6 = proc(var2);
int var7 = proc(proc(var1));
int var8 = proc(proc(var2));
int& var9 = var1;
int& var10 = var2;
const int& var11 = proc(var1);
const int& var12 = proc(var2);

swap(var3,var4);
swap(var5,var6);
swap(var7,var8);
swap(var9,var10);
swap(var11,var12);

cout << "var1 = " << var1 << " var2 = " << var2 << endl;
cout << "var3 = " << var3 << " var4 = " << var4 << endl;
cout << "var5 = " << var5 << " var6 = " << var6 << endl;
cout << "var7 = " << var7 << " var8 = " << var8 << endl;
cout << "var9 = " << var9 << " var10 = " << var10 << endl;
cout << "var10 = " << var11 << " var11 = " << var12 << endl;
```

```
return 0;
}
```

Aufgabe 10.2. Erweitern Sie die Klasse `Fraction` von Folie 230 und Beispiel 9.1 um

- den Standardkonstruktor (ohne Parameter), der $p = 0$ und $q = 1$ setzt,
- einen Konstruktor, der $p, q \in \mathbb{Z}$ mit $q \neq 0$ als Input übernimmt und den Bruch speichert,
- den Kopierkonstruktor,
- den Zuweisungsoperator und
- den Destruktor.

Stellen Sie mittels `assert` sicher, dass die Übergabeparameter zulässig sind, d.h. $q \neq 0$. Beachten Sie den Fall $q < 0$, bei dem intern $(-p)/|q|$ gespeichert wird. Testen Sie ihre Implementierung geeignet! Speichern Sie den Source-Code unter `fraction.{hpp/cpp}` in das Verzeichnis `serie10`.

Aufgabe 10.3. Überladen Sie den Vorzeichenoperator $-$, der zum Bruch x im Format `Fraction` aus Aufgabe 10.2 den Bruch $-x$ liefert, und den `<<`-Operator, um einen Bruch $x := p/q$ im Format `Fraction` aus Aufgabe 10.2 in der Form `p/q` ausgeben zu können (siehe Folie 290 für ein Beispiel mit der Klasse `Complex` aus der Vorlesung). Testen Sie Ihre Implementierung geeignet!

Aufgabe 10.4. Überladen Sie die Operatoren $+$, $-$, $*$ und / um die Summe, die Differenz, das Produkt und den Quotienten zweier Brüche im Format `Fraction` aus Aufgabe 10.2 zu berechnen. Stellen Sie bei / mittels `assert` sicher, dass Sie nicht durch 0 dividieren. Das Ergebnis soll in allen Fällen gekürzte Form haben. Testen Sie ihre Implementierung geeignet!

Aufgabe 10.5. Schreiben Sie die Klassendefinition zu einer Klasse `Polynomial` zur Speicherung von Polynomen vom Grad $n \in \mathbb{N}$, die bezüglich der Monombasis dargestellt sind, d.h.

$$p(x) = \sum_{j=0}^{n} a_j x^j.$$

In der Klasse soll neben dem dynamischen Vektor $(a_0, \ldots, a_n) \in \mathbb{R}^{n+1}$ der Koeffizienten `(double*)` auch der Grad $n \in \mathbb{N}$ gespeichert werden. Implementieren Sie die folgenden Funktionalitäten:

- Destruktor, Konstruktor zum Allokieren des Null-Polynoms mit Grad n, Kopierkonstruktor,
- Zuweisungsoperator,
- Zugriff auf die Koeffizienten des Polynoms mittels `[ ]`, d.h. für $0 \leq j \leq n$ liefert `p[j]` dann a_j und
- die Möglichkeit ein Polynom `p` mit `cout << p` in der Monombasis ausgeben zu können.

Beachten Sie, dass Sie im Kopierkonstruktor neuen dynamischen Speicher für den Koeffizientenvektor des Outputs anlegen müssen (Stichwort: Deep Copy). Erklären Sie warum! Schreiben Sie auch ein main-Programm, in welchem Sie die Implementierung testen.

Aufgabe 10.6. Die Summe zweier Polynome ist wieder ein Polynom. Implementieren Sie für die Klasse `Polynomial` aus Aufgabe 10.5 die nötige Funktionalität, um zwei Polynome p und q mittels `r=p+q` zu addieren. Implementieren Sie hinaus darüber die Möglichkeit, für eine skalare Größe $a \in \mathbb{R}$ im Format `double` bzw. `int` und für ein Polynom p die Operationen `r=a+p` oder `r=p+a` in sinnvoller Weise ausführen zu können. Schreiben Sie auch ein main-Programm, in welchem Sie die Implementierung testen.

Aufgabe 10.7. Das Produkt zweier Polynome ist wieder ein Polynom. Implementieren Sie für die Klasse `Polynomial` aus Aufgabe 10.5 die nötige Funktionalität, um zwei Polynome p und q mittels `r=p*q` zu multiplizieren. Implementieren Sie darüber hinaus die Möglichkeit, für eine skalare Größe $a \in \mathbb{R}$ im Format `double` bzw. `int` und für ein Polynom p die Operationen `r=a*p` und `r=p*a` in sinnvoller Weise ausführen zu können. Schreiben Sie auch ein main-Programm, in welchem Sie die Implementierungen testen.

Aufgabe 10.8. Das Taylor-Polynom vom Grad n einer Funktion $f \in C^n(\mathbb{R})$ an einer Stelle $x_0 \in \mathbb{R}$ ist gegeben durch

$$T^{(n)} f(x) = \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k.$$

Schreiben Sie einen Konstruktor für die Klasse `Polynomial` aus Aufgabe 10.5 der für ein $n \geq 0$ wahlweise das Taylor-Polynom vom Grad n der Funktionen `sin`, `cos` oder `exp` in $x_0 = 0$ erzeugt. Verwenden Sie für die Fallunterscheidung einen string. Testen Sie Ihren Code entsprechend!