Anmerkungen

- Sichern Sie ihre Berechnungen, so weit wie möglich gegen fehlerhafte Eingaben ab.
- Verwenden Sie sprechende Variablennamen und formatieren Sie Ihren Programmtext ordentlich.
- Erstellen Sie Inputprompts und Ausgaben, sodass das Programm für BenutzerInnen angenehm zu bedienen ist.
- Schreiben Sie Ihre Programm so, dass mehrere Berechnungen ausgeführt werden können, ohne das Programm immer wieder neu starten zu müssen.
- Testen Sie Ihre Programme und unterziehen Sie die Ergebnisse einer Prüfung (Schätzung des erwarteten Ergebnisses, bzw. im Notfall mit dem Taschenrechner prüfen).
- Mit * markierte Beispiele sind etwas anspruchsvoller

Aufgabe 1 *

Stellen Sie Ihren Rechner aus der Übungseinheit fertig und versuchen Sie sich an einigen Erweiterungen

Aufgabe 2

Schreiben Sie eine Klasse, deren Objekte ein Sparschwein repräsentieren. In dieses Sparschwein können nur Münzen im Wert von 1 Cent, 5 Cent und 10 Cent geworfen werden. Implementieren Sie zumindest die Methoden ''leeren'' (Setzt den Inhalt des Sparschweins auf 0), ''einwerfen'' (ermöglicht es, Münzen in das Sparschwein einzuwerfen) und ''print'' (gibt den Inhalt des Sparschweins in der Form z.B.: "2*1Cent + 1*5Cent+3*10Cent = 37Cent" aus)

Schreiben Sie:

- die nötigen Konstruktoren, damit folgende Definitionen von Objekten möglich sind (unter der Annahme, dass die Klasse den Namen Sparschwein hat):

```
Sparschwein a; // erzeugt ein leeres Sparschwein
Sparschwein b{10}; // erzeugt ein Sparschwein mit 10 Eincentmünzen
                   // als Inhalt
Sparschwein c{2,3,4}; // erzeugt ein Sparschwein mit 2 Eincent-,
                      // 3 Fünfcent- und 4 Zehncentmünzen als Inhalt
```

- Die Vergleichsoperatoren <, >, ==, !=, die jeweils den Inhalt zweier Sparschweine vergleichen

Schreiben Sie ein Hauptprogramm, das mindestens zwei Instanzen Ihrer Klasse verwendet und die Verwendung der verschiedenen Methoden illustriert.

Aufgabe 3

Schreiben Sie eine Klasse, deren Objekte eine Ampel repräsentieren. Die Ampel hat die Zustände grün, grün blinkend, gelb, rot, rot+gelb in der üblichen Reihenfolge. Zusätzlich gibt es noch den Zustand gelb blinkend. Implementieren Sie zumindest die Methoden ''anschalten'' (versetzt die Ampel in den Zustand gelb blinkend), ''weiter'' (geht in den nächstfolgenden Zustand über; dabei ist der Folgezustand von gelb blinkend gleich rot) und ''print'' (gibt den aktuellen Zustand der Ampel aus).

Schreiben Sie

- die nötigen Konstruktoren, damit folgende Definitionen von Objekten möglich sind (unter der Annahme, dass die Klasse den Namen Ampel hat):

```
Ampel a; // erzeugt eine Ampel im Zustand gelb blinkend
Ampel b {AktZustand}; // erzeugt eine Ampel in dem Zustand, der durch
                      // AktZustand vorgegeben wird
```

- Einen Operator ++, der das Fortschalten der Ampel bewirkt

Schreiben Sie ein Hauptprogramm, das mindestens zwei Instanzen Ihrer Klasse verwendet und die Verwendung der verschiedenen Methoden illustriert.

Anmerkung: Der Operator ++ kann in C++ sowohl in Postfix- (a++), als auch in Prefixform (++a) verwendet werden. Bei der Definition der Methoden zum Überladen des Operators ++ gibt es daher zwei verschiedene Möglichkeiten:

''void operator++()'' definiert die Prefixform

''void operator++(int)'' definiert die Postfixform

Der zusätzliche Parameter bei der Postfixform unterscheidet nur zwischen den beiden Möglichkeiten und hat sonst keinerlei Verwendung. Für die Lösung des Beispiels brauchen Sie nur eine der beiden Möglichkeiten zu implementieren. Statt void kann bei Bedarf natürlich auch ein passender Typ für das Ergebnis der Operation gewählt werden.

Aufgabe 4

Schreiben Sie eine Klasse, deren Objekte Spielkarten repräsentieren. Die möglichen Kartenwerte sind (in aufsteigender Reihenfolge) 10, Bube, Dame, König und As. Die möglichen Farben (ebenfalls aufsteigend) Pik, Treff, Karo und Herz. Eine Karte sticht, wenn entweder ihr Farbenwert höher ist, oder bei gleichen Farben ihr Kartenwert höher ist. Schreiben Sie

- die nötigen Konstruktoren, damit folgende Definitionen von Objekten möglich sind (unter der Annahme, dass die Klasse den Namen Karte hat):

```
Karte a; // erzeugt Herz As
Karte b = Bube; // erzeugt Herz Bube
Karte c(Zehn, Herz);
```

- Die Vergleichsoperatoren <, >, ==, !=, die jeweils zwei Karten vergleichen. Dabei ist a>b, fall a b nach obigen Regeln sticht.

Schreiben Sie ein Hauptprogramm, das mindestens zwei Instanzen Ihrer Klasse verwendet und die Verwendung der verschiedenen Methoden illustriert.

Aufgabe 5

Schreiben Sie eine Klasse, deren Objekte rationale Zahlen repräsentieren. Dabei sollen die Zahlen nicht in der üblichen Kommaschreibweise gespeichert werden (z.B.: 0.33333), sondern in der Form Zähler/Nenner (also etwa 1/3). Implementieren Sie zumindest die Methoden ''init'' (setzt den Wert der rationalen Zahl, indem Zähler und Nenner gesetzt werden), ''kuerzen'' (kürzt den dargestellten Wert auf den kleinstmöglichen Nenner, so dass etwa aus der Darstellung 3/9 die gleichwertige Darstellung 1/3 wird), ''wert'' (retourniert den aktuellen Wert des Bruches als double Wert) und ''print'' (gibt den Bruch in der Form z.B.: "3/9" aus)

Schreiben Sie

- die nötigen Konstruktoren, damit folgende Definitionen von Objekten möglich sind (unter der Annahme, dass die Klasse den Namen Q hat):

```
Q a; // erzeugt eine rationale Zahl mit dem Wert Null
Q b=10; // erzeugt eine rationale Zahl mit dem Wert 10
Q c(1,2); // erzeugt eine rationale Zahl mit dem Wert 0,5
```

Die Operatoren +, -, * und /, die jeweils rationale Zahlen addieren, subtrahieren, multiplizieren und dividieren

Schreiben Sie ein Hauptprogramm, das mindestens zwei Instanzen Ihrer Klasse verwendet und die Verwendung der verschiedenen Methoden illustriert.