Tag 6a: Objekte und C++

Aufgabe 1: Modellierung mit Objekten

(a) Überlegt Euch in Partner- oder Gruppenarbeit Beispiele für Objekte, die sinnvoll sich zu Klassen zusammenfassen lassen. Verwendet dazu Sachverhalte/Gegenstände aus Eurer Alltagswelt. Durch welche Attribute und Methoden lassen sich diese modellieren? Gebt insgesamt drei verschiedene Beispiele an.

(b) Gelingt es Euch, Beispiele für Klassen zu finden, die so viele Gemeinsamkeiten haben, dass es sich lohnen würde, dafür eine gemeinsame Elternklasse zu definieren?

Aufgabe 2: Objekte in C++

Beantworte folgende Fragen (auch durch Ausprobieren am Rechner!):

(a) Erläutere Unterschiede zwischen Strukturen (`struct`) und Klassen (`class`).

(b) Besitzen Strukturen `struct` einen Zeiger `this`?

(c) Darf man Konstruktoren bzw. Destruktoren überladen? Wie viele Konstruktoren und wie viele Destruktoren darf eine Klasse haben?

(d) Erläutere die Bedeutung der Zugriffsrechte `public`, `protected` und `private` (ggf. an einem Beispiel).

(e) Bisher waren alle Konstruktoren öffentlich. Kannst Du ein Beispiel dafür geben, indem ein privater Konstruktor sinnvoll ist?

Aufgabe 3: Rationale Zahlen als Objekte

(a) Schreibe eine Klasse `Bruch`, die eine rationale Zahl repräsentieren soll. Als Attribute soll diese zwei `long`-Variablen `zaehler` und `nenner` besitzen. Der Konstruktor sei:

```
Bruch(long z, long n);
```

Außerdem soll es folgende Methoden geben:

```
long Zaehler() const;
long Nenner() const;
double Wert() const;
```

Diese liefern Zähler und Nenner sowie den Wert des Bruches als Gleitkommazahl zurück. Implementiere diese vier Methoden!

(b) Überlege, welche Zugriffsrechte (`public`, `protected`, `private`) für die o.g. Attribute und Methoden sinnvoll sind.

(c) Schreibe ein Programm, welches eine `Bruch`-Variable definiert und mit dem Wert 47/11 initialisiert. Danach sollen Zähler, Nenner und der resultierende Wert ausgegeben werden.

(d) Erweitere die Klasse `Bruch` um einen Konstruktor `Bruch(long g)` mit dem der Bruch mit einer ganzen Zahl initialisiert werden kann.

(e) Erweitere die Klasse `Bruch` um eine Methode `kuerzen()`. In dieser sollen Zähler und Nenner durch ihren größten gemeinsamen Teiler (ggT) dividiert werden. (Hinweis: Der ggT kann entweder naiv über ausprobieren oder über den sog. *Euklidischen Algorithmus* berechnet werden.) Teste die Methode für den Bruch $-12/48$.

Aufgabe 4: Türme von Hanoi

Wir kehren noch einmal zur prozeduralen Programmierung zurück: In der Vorlesung und im Skript (Seite 78ff) wird das Problem der Türme von Hanoi betrachtet. Die Lösung erfolgt mittels eines rekursiven Algorithmus.

(a) Schreibe eine (rekursive) Funktion `void bewege_scheiben(int n, char start, char ziel, char hilf)`, welche `n` Scheiben von `start` nach `ziel` verschiebt und dabei den Hilfsstapel `hilf` verwendet. Dabei sollen alle Zwischenschritte ausgegeben werden.

(b) Schreibe ein Programm, welches die Anzahl der Scheiben `n` erfragt und dann die rekursive Funktion aufruft.

(c) Theoretisch wurde gezeigt, dass für n Scheiben $2^n - 1$ Bewegungen der Scheiben nötig sind. Überprüfe das am Programm. Bis zu welchem Wert von n arbeitet Dein Programm?

(d) Die Türme von Hanoi beziehen sich auf eine Legende, nach der ein Mönch in einem Kloster in Hanoi 64 Scheiben umschichten sollte. Angenommen, er benötigt pro Scheibe 1 Sekunde. Wie lange wäre er beschäftigt?

Viel Erfolg!