5.3.A Einführung und Klassen/Konstruktor und Destruktor – Arbeitsblatt

Aufgabe 1: Konstruktoren für `punkttyp`

Bei der Klasse `punkttyp` soll die Methode `init()` durch Konstruktoren ersetzt werden (`punkttyp3.cpp`): Zuerst soll ein Konstruktor erstellt werden, der keine Parameter hat und beim Anlegen eines Punktes (also eines Objekts der Klasse `punkttyp`) beide Attribute, x und y, einfach auf den Wert `0.0` setzt.

a) Wie nennt man einen Konstruktor wie diesen, der keine Parameter hat, mit einem Fachbegriff?

- _____

b) Welchen Namen trägt die Konstruktor-Methode in unserem Fall?

- _____

c) Welchen Rückgabetyp hat jeder Konstruktor?

- _____

d) Schreiben Sie bitte den beschriebenen Konstruktor!

Nun soll ein neuer Konstruktor hinzukommen, der zwei Parameter aufnimmt. Der erste Parameter soll den Wert für x initialisieren, der zweite Parameter den Wert für y.

e) Der neue Konstruktor hat denselben Namen wie der erste. Wie nennt man die Möglichkeit, mehrere Funktionen (oder Methoden) mit demselben Namen in einem Programm (oder einer Klasse) zu definieren, vorausgesetzt, sie haben verschiedene Parameterlisten?

- _____

f) Schreiben Sie bitte den neuen Konstruktor!

g) Ändern Sie bitte Ihr Hauptprogramm so, dass der Aufruf der `init()`-Methode unnötig wird, weil der Punkt schon beim Anlegen durch den Konstruktor seine Koordinaten bekommt!

Aufgabe 2: Konstruktoren für `tageszeittyp`

Bei der Klasse `tageszeittyp` gibt es bereits zwei `init()`-Methoden. Sie sollen durch zwei geeignete Konstruktoren ersetzt werden (`tageszeittyp3.cpp`):

a) Was sind die Vorteile von Konstruktoren gegenüber den `init()`-Methoden? Nennen Sie zwei!

- _____
- _____

b) Ersetzen Sie die Methode `void init(void)` durch den passenden Konstruktor!

c) Ersetzen Sie die Methode `void init(unsigned char stunde)` durch den passenden Konstruktor!

d) Ändern Sie das Hauptprogramm so, dass die `init()`-Methoden nicht mehr benutzt werden, die Funktionalität aber gleich bleibt!

Aufgabe 3: Konstruktoren für `udouble`

Die Klasse `udouble` braucht ebenfalls zwei Konstruktoren: Einen ohne Parameter und einen mit einem `double`-Parameter.

a) Wann wird der Konstruktor aufgerufen, der keinen Parameter hat?

- _____

b) Erstellen Sie diesen Konstruktor!

c) Der zweite genannte Konstruktor mit einem `double`-Parameter ermöglicht das Anlegen und Initialisieren einer Variablen nach folgendem Muster:

```
1  udouble x(3.0);
```

Wie sieht die andere Form dieser Initialisierung aus, die genau das Gleiche bewirkt?

- _____

d) Erstellen Sie diesen zweiten Konstruktor!

e) Erweitern Sie das Hauptprogramm so, dass drei Objekte der Klasse `udouble` angelegt und mit `print()` auf den Bildschirm ausgegeben werden. Das erste Objekt soll mit dem ersten Konstruktor angelegt werden, das nächste mit dem zweiten Konstruktor in der oben angegeben Form (`udouble x(3.0);`) und das dritte auch mit dem zweiten Konstruktor, aber in der anderen Form.

Aufgabe 4: Konstruktoren für `bruch`

Ergänzen Sie `bruch` um Konstruktoren, so dass man Objekte auf die folgende Weise anlegen kann (`bruch3.cpp`):

a) `bruch a;` – ergibt $\frac{0}{1} = 0$

b) `bruch b(7);` – ergibt $\frac{7}{1} = 7$

c) `bruch c(3,4);` – ergibt $\frac{3}{4} = 0,75$

Aufgabe 5: Konstruktoren für `datum`

Ergänzen Sie `datum` um Konstruktoren, so dass man Objekte auf die folgende Weise anlegen kann (`datum3.cpp`):

a) `datum a;` – heutiges Datum

b) `datum b(17);` – der 17. dieses Monats

c) `datum c(17,6);` – der 17. Juni dieses Jahres

d) `datum d(17,6,1953);` – der 17. Juni 1953

e) `datum e("17.06.1953");` – der 17. Juni 1953

Aufgabe 6: Konstruktoren für `person`

Ergänzen Sie `person` um Konstruktoren, so dass man Objekte auf die folgende Weise anlegen kann (`person3.cpp`):

a) `person a;` – Vorname und Nachname leer, Geburtstag heute

b) `person b("Paul", "Meier");` – Paul Meier, Geburtstag heute

c) `person c("Paul", "Meier", 19901003);` – Paul Meier, Geburtstag 3. Oktober 1990