Übung 6

Vererbung, virtuelle Funktionen, dynamisches Binden

Aufgabe 6.1

Es soll ein Programm zur Verwaltung einer ***einfach verketteten Liste*** entwickelt werden.

1. Definieren Sie zunächst eine Klasse **`cell`**, die als Basisklasse für die Elemente einer Liste fungiert. Als Datenelement besitzt die Klasse einen Zeiger vom **`cell*`** zur unidirektionalen Verkettung der Elemente.

   Von der Klasse **`cell`** sollen **keine** Objekte erzeugt werden. Sowohl der Zeiger als auch der Default-Konstruktor sind daher als **`protected`** zu deklarieren.

   Als **`public`** wird nur eine **virtuelle** Funktion **`print()`** zur Verfügung gestellt. Die noch zu definierende Klasse `Liste` ist ein **`friend`** der Klasse.

2. Definieren Sie dann eine Klasse **`list`**, die zwei Zeiger vom Typ **`cell *`** besitzt. Ein Zeiger soll auf das **erste** Listenelement, der andere Zeiger auf das **letzte** Listenelement verweisen.

   Außerdem ist ein **Default-Konstruktor** zu deklarieren, der beide Zeiger auf **`NULL`** setzt.

   Die Methode **`append()`** hängt ein Element als letztes in die Liste ein. Als Argument erhält **`append()`** einen Zeiger vom Typ **`cell *`** auf das einzuhängende Element.

   Die Methode **`getNext()`** entnimmt das **erste Element** der Liste. Sie liefert einen **`cell`**-Zeiger auf das entnommene Element bzw. den **`Null`**-Zeiger, wenn die Liste **leer** ist.

   Die Methode **`displayAll()`** zeigt mit Hilfe der virtuellen Methode **`print()`** alle Elemente der Liste am Bildschirm an.

3. Die abgeleitete Klasse **`measurement`**, soll aus der Klasse **`cell`** durch **`public`**-Vererbung abgeleitet werden. Sie besitzt als Datenelement einen Messwert vom Typ **`long`**.

   (*Beachte*: Neben einem Konstruktor, dem ein Messwert als Argument übergeben wird, ist die Methode **`print()`** zu definieren, die den gespeicherten Messwert ausgibt !).

Testen Sie die Klasse **`list`**, indem Sie »Messwerte« von der Tastatur einlesen lassen:

- Für jeden Messwert wird dynamisch ein Objekt vom Typ **`measurement`** erzeugt und mit der Methode **`append()`** in eine Liste eingefügt.
- Anschließend lassen Sie die gesamte Liste anzeigen.
- Entnehmen Sie dann mit **`getNext()`** alle Elemente der Liste, wobei das entnommene Element angezeigt wird.

Aufgabe 6.2

- Sei

```
class Y {
        private: int ydat;
        public: int f();
};
class X: public Y {
            ....
};
 X *x; Y y;
```

Beurteilen Sie folgende Zuweisungen

`x = &y;`	o korrekt	o nicht korrekt
`y = *x;`	o korrekt	o nicht korrekt
`x = y.f();`	o korrekt	o nicht korrekt

- In Unterklassen können Datenelemente der Oberklasse wie private Datenelemente der Unterklasse verwendet werden o korrekt o nicht korrekt

- In Oberklassen können Datenelemente der Unterklasse wie private Datenelemente der Unterklasse verwendet werden o korrekt o nicht korrekt

- Jede Funktion einer Oberklasse ist in einer abgeleiteten Klasse immer sichtbar
  o korrekt o nicht korrekt