Programmiertechnik – erste Übung zu Klassen: die Klasse Zeit

Aufgabe 1:

Erstellen Sie eine **Klasse „Zeit“** mit folgenden (privaten) **Eigenschaften**:

- Stunde : int
- Minute : int
- Sekunde : int

und den (öffentlichen / public) **Methoden**:

+ setTime(int stunde, int minute, int sekunde) : void
(setzt die Zeit auf den übergebenen Wert)

+ getTime() : string
(liefert einen String der gespeicherten Zeit im Format „Stunde : Minute : Sekunde“)

+ vorstellen(int stunde, int minute, int sekunde) : bool
(addiert das übergebene Zeitintervall zur gespeicherten Zeit und liefert „true“ falls ein Überlauf auf den nächsten Tag aufgetreten ist)

sowie zwei **Konstruktoren**:

- Einen eigenen Standardkonstruktor, der die Zeit auf 00:00:00 setzt
- Einen weiteren Konstruktor, der die Zeit sogleich auf einen übergebenen Wert (Stunde, Minute, Sekunde) setzt.

Schreiben Sie im main-Programm einige Testaufrufe, indem Sie auf unterschiedliche Weise zwei „Alarmzeiten“ setzen, Zeiten dazu addieren und nach jeder Addition die Zeit abrufen und am Bildschirm ausgeben. Beispiel:

```
/* ------------------------ Hauptprogramm ------------------------------- */
int main() {
    Zeit Alarmzeit1;
    cout << "Alarmzeit1: " << Alarmzeit1.getTime() << endl;

    Alarmzeit1.setTime(9,9,9);
    cout << "Alarmzeit1: " << Alarmzeit1.getTime() << endl;

    Zeit* Alarmzeit2 = new Zeit(12,12,12);
    cout << "Alarmzeit2: " << Alarmzeit2->getTime() << endl;

    if (Alarmzeit1.vorstellen(12,55,55))
        cout << "Bitte stellen Sie auch das Datum! ";
    else
        cout << "Datum ist weiterhin korrekt! ";
    cout << "Alarmzeit1: " << Alarmzeit1.getTime() << endl;

    if (Alarmzeit2->vorstellen(12,55,55))
        cout << "Bitte stellen Sie auch das Datum! ";
    else
        cout << "Datum ist weiterhin korrekt! ";
    cout << "Alarmzeit1: " << Alarmzeit2->getTime() << endl;

    delete Alarmzeit2; // der Vollständigkeit halber...

    system("pause");
    return 0;
}
```

Aufgabe 2:

Erweitern Sie die Klasse Zeit um folgende Eigenschaften und Methoden (und sehen Sie im Testprogramm entsprechende zusätzliche Testaufrufe vor). Ersetzen sie die drei ??? durch passende Datentypen:

- Format: ??? (um die Stringausgabe im 12-(AM/PM) oder 24-Stunden-Format zu erzeugen)

+ void setFormat(???); (setzt das Ausgabeformat auf 12- oder 24-Stunden-Anzeige)

+ int stellen(int stunde, int minute, int sekunde);
(verstellt die Zeit in beide Richtungen und gibt zurück, um wie viele Tage ein Datum geändert werden müsste im Falle, dass ein (auch mehrtägiger) Überlauf aufgetreten ist.)

Alle Methoden sollen außerdem eine beliebige Anzahl (bis zu drei) Parameter entgegennehmen können. Wird kein Parameter für Stunde, Minute oder Sekunde übergeben, so ist der entsprechende Wert als Wert 0 anzunehmen (Standardparameter…).

Aufgabe 3:

Erweitern Sie die Klasse Zeit um eine public-Methode „vergleiche(int stunde, int minute, int sekunde)“. Diese soll die (positive oder negative) Differenz zwischen gespeicherter Zeit und der übergebenen Zeit in Sekunden zurückgeben (mit Testaufruf…).

Überladen Sie die Methode, damit bei zusätzlicher Übergabe eines bool-Wertes true statt der Rückgabe der Zeitdifferenz gleich die Zeitdifferenz in Stunden, Minuten und Sekunden als String zurückgegeben wird und zur Anzeige auf dem Bildschirm verwendet werden kann.

Erweitern Sie das main-Programm um entsprechende Aufrufe.