Hochschule Furtwangen
Fakultät Informatik
Prof. Dr. Lothar Piepmeyer

Praktikum ‚Objektorientierte Programmierung'

Aufgabenblatt 8

Denken Sie nach wie vor an die Trennung von Deklaration und Implementierung. Definieren Sie einen eigenen Namespace. Zu allen Aufgaben gehören umfassende Tests.

Aufgabe 1:
a. Ändern Sie die Klasse `Movie` aus Aufgabenblatt 5 so, dass sie bei unveränderter Schnittstelle der Nullregel genügt. Die Tests müssen *unverändert* bestanden werden. Zusätzlich muss der folgende Test bestanden werden:

```
hfu::Person director("Donald", "Duck");
int scores[] = {4, 7, 1, 1};
hfu::Movie scored_movie("Modern Times", 90, director, scores, 4);
auto copy = hfu::Movie ("Modern Times", 90, director);
copy=scored_movie;
scored_movie.setScore(3,42);
assert(copy.getScore(3) == 1);
```

Welche Funktionalität prüft der Test?

b. Ergänzen und testen Sie die Vergleichsoperatoren == und `!=`.

Aufgabe 2*:
a. Zu einer Liste `nums` mit lauter ganzen Zahlen ist eine weitere ganze Zahl `n` gegeben. Gesucht sind die Indexe der beiden Zahlen aus der Liste, deren Summe `n` ist. Gehen Sie davon aus, dass es immer eine Lösung gibt. Die Lösung darf keinen Index zweimal enthalten.
Ihre Funktion muss die folgende Signatur haben:

```
vector<int> sum(const vector<int>& nums, int n)
```

Beispiel:
`nums = {2,7,11,15}, n = 9`
Wegen `nums[0] + nums[1] == 9` ist das Ergebnis `{0, 1}`.
Entwickeln Sie weitere Tests. Dieser Aufgabenteil kann sehr leicht gelöst werden.

b. Aufgabenteil a. hat eine sehr einfache Lösung. In *diesem* Aufgabenteil ist eine Lösung gesucht, die die Datenstruktur `map` – oder noch besser `unordered_map` – geeignet nutzt, um quadratische Laufzeiten zu vermeiden. Das ist etwas gedankenintensiver. Um die Aufgabe *noch* interessanter zu gestalten, sind für die Lösung maximal 15 *ordentliche*, *klare* Zeilen zulässig.