Hochschule Furtwangen
Fakultät Informatik
Prof. Dr. Lothar Piepmeyer

Praktikum ‚Objektorientierte Programmierung'

Aufgabenblatt 9

Wie in der Vorlesung besprochen, empfehle ich die Compiler-Flags `-Wall` und `-Wextra`. Um möglichst im Einklang mit der Vorlesung zu sein, sollten Sie dafür sorgen, dass Ihr Compiler den C++14-Standard unterstützt. In keiner der Aufgaben darf der *globale* Namensraum genutzt werden. Definieren Sie einen eigenen Namensraum. Die Anweisung `using namespace` darf nicht genutzt werden.

.

Aufgabe 1:

a. Sie haben in einer früheren Aufgabe die Funktion `sort` für den Datentyp `Position` entwickelt. Ändern Sie die Funktion in ein Template.

b. Stellen Sie sicher, dass die Tests noch funktionieren, die Sie seinerzeit entwickelt haben.

c. Stellen Sie sicher, dass bei der Sortierung von Daten vom Typ `char`, die Groß- und Kleinschreibung *keine Rolle* spielt. Nur bei gleichen Buchstaben gilt: Kleinschreibung vor Großschreibung. Der folgende Test muss also bestanden werden:

```
char chars[4]{'B','a','A','x'};
char expected[4]{'a','A','B','x'};
sort(chars,4);
for (int i = 0; i < 4; ++i) {
    assert(chars[i]==expected[i]);
}
```

Achten Sie auf eine redundanzarme Implementierung.

d. Können Sie das Template auch für den Typ `City` nutzen? Erläutern Sie Ihre Antwort.

Nein, weil für City keine < und > Operatoren definiert sind

Aufgabe 2:

a. Ändern Sie die Klasse `City` so, dass die Sehenswürdigkeiten nicht mehr in einem einfachen Array abgelegt werden, sondern in einem Attribut vom Datentyp `std::unique_ptr<std::string[]>`. Die Schnittstelle und somit auch die Tests können also unverändert bleiben. Innerhalb der Klasse müssen jedoch einige Änderungen durchgeführt werden. Die Aufgabe ist schwieriger, als sie auf den ersten Blick aussieht.

b. Sie haben mehrere Konstruktoren in der Klasse `City`. Nur einer der Konstruktoren darf Attribute in einer Liste initialisieren. Die anderen Konstruktoren müssen mit Konstruktorverkettung arbeiten.