Die Standard Template Library

Zu einem der wesentlichen positiven Merkmale und Entwicklungen von C++ zählt die Standardisierung nach ANSI und ISO. Dazu gehört auch die Festlegung der Standard Template Library (STL). In ihr sind eine Reihe von nützlichen Klassen und Funktionen festgelegt.

Container:
Das sind Objekte, die andere Objekte (Elemente) verwalten können. Sie repräsentieren verschiedene abstrakte Datenstrukturen und können mit Iteratoren die jeweiligen Elemente einfach bearbeiten.

Iterator:
Dies ist eine Abstraktion (Verallgemeinerung) von Zeigern und dient dem Zugriff auf Container-Elemente und/ oder arrays und wird von Algorithmen benutzt.

Algorithmen:
Sie sind Container unabhängig und können nur über Iteratoren auf die Datenstruktur des jeweiligen Containers zugreifen. Da Algorithmen mit jeder Art von Iteratoren zusammenarbeiten können, muss jeder Algorithmus in nur einer Form vorhanden sein – dadurch werden Inkonsistenzen ausgeschlossen und Verwaltungsprobleme minimiert

1. Arten von Containern

In der STL sind eine Reihe von Containern und Container-Adaptern (Container, welche andere Container enthalten können) vordefiniert. Container gliedert man dabei in verschiedene Gruppen auf.

1.1 Listen

Listen gehören zu den sequenziellen Containern. Das heißt, die gespeicherten Elemente sind streng geordnet. L>isten speichern ihre Daten in Knoten Ein Knoten enthält neben den Daten (Elementen) zusätzliche Verweise auf weitere Knoten. Man unterscheidet zwischen:

- **Einfach verketteten Listen:** Ein Knoten enthält nur den Verweis auf den nächsten Knoten. Hier kann man den Iterator nur in einer Richtung zum nächsten Knoten bewegen.
- **Doppelt verkettete Listen:** Ein Knoten enthält Verweise auf den vorherigen und den folgenden Knoten. Die doppelt verkettete Liste kann mit Iteratoren in beide Richtungen durchlaufen werden.

Beim Einfügen neuer Elemente sind nur die Verweise nächster und vorherigen anzupassen. Dadurch können Elemente sehr leicht hinzugefügt werden.

Beispiele aus der STL:

- **deque**: sollte benutzt werden, wenn häufiges Einfügen am Anfang und am Ende der Liste erwartet wird.
- **List**: ist auf das Einfügen von Daten in der Mitte des Containers optimiert
- **vector**: sollte standardmäßig verwendet werden.

1.2 Sets

Set zählen zu den assoziativen Containern. In ihnen werden Elemente mit Schlüsseln gespeichert wobei die Elemente selbst den Schlüssel darstellen. Sets unterstützen Abbildungen von Mengen in einem Computer. Das Einfügen neuer Elemente verläuft etwas langsamer als bei den Listen Dafür sind diese Container auf der Suchen von Elementen über den Schlüssel optimiert.

Beispiele aus der STL:

- **set**: beim Set dürfen keine doppelten Elemente (Elemente mit dem gleichen Schlüssel) vorhanden sein
- **multiset**: im Multiset kann es gleiche Schlüssel geben, es erfolgt ein schneller Zugriff auf die Elemente über Schlüssel.

1.3 Maps

Maps (auch als hash-tables bezeichnet) gehören zu den assoziativen Containern und arbeiten mit Schlüsseln (keys). Die Schlüssel und die dazugehörigen Daten können von unterschiedlichem Datentyp sein. Elemente einer Map setzen sich aus einem Schlüssel-Daten-Paar zusammen. Der schnelle Zugriff auf die Daten erfolgt über den Schlüssel.

Beispiele aus der STL:

- **map**: ein Schlüssel kann nur auf ein enthaltenes Element der Map verweisen
- **multimap**: ein Schlüssel kann auf mehrere Elemente verweisen

1.4 Container-Adapter

Container-Adapter der STL setzen auf andere sequentielle Container auf. Sie implementieren besondere Arten des Einfügens und Herausnehmens von Elementen aus einem Container. Man unterscheidet zwischen:

- Queue: Ein Queue arbeitet nach dem fifo Prinzip (first in first out). Elemente, die zuerst abgelegt werden, können auch als erstes wieder aus dem Queue herausgenommen werden.
- Stack: Ein Stack (Stapelspeicher) funktioniert nach dem Prinzip first in last out (filo). Elemente, die zuletzt abgelegt werden, sind zuerst wieder zu entnehmen.

2. Die Verwendung von Algorithmen

   Algorithmen sind fertige Funktionstemplates, mit denen der Inhalt von Containern bzw. arrays bearbeitet werden kann. Zum Verständnis für die Funktionsweise von Algorithmen muss zunächst ein kleines Testprogramm mit zwei arrays geschrieben werden.
   Um die Algorithmen nutzen zu können, ist die Headerdatei algorithm einzubinden.

   Zufallszahlen.cpp

3. Die Verwendung von Containern

Container sind Behälter, die Daten eines Typs speichern (vergleichbar mit einem Array). Spe passen ihren Speicherbedarf aber automatisch dem Inhalt an. Auf die Inhalte eines Containers kann nur über Iteratoren zugegriffen werden. Einen Iterator kann man sich dabei als Zeiger auf die Speicheradresse eines Elements des Containers vorstellen.

Zum Testen der Funktionalität dient folgendes Programm, in dem verschiedene Werte vom Typ char in einem vector- Container gespeichert und bearbeitet werden: