String-Streams

Die Headerdatei <sstream> definiert die Typen

- `std::istringstream` für die **Eingabe aus** C++-Strings
- `std::ostringstream` für die **Ausgabe auf** C++-Strings
- `std::stringstream` für beides.

Die Eingabe und Ausgabe erfolgt wie gewohnt mit den C++Operatoren << und >>. Auch die IO-Manipulatoren lassen sich wie gewohnt verwenden.

Verwendung von Stringstreams:

```
std::ostringstream oss;          // legt einen leeren Ausgabe-Stringstream an
oss << "Der Wert von a ist " << a; // schreibt in den String
```

`oss.str()`: der Return-Wert dieser Methode ist der auf `oss` geschriebene Text als `std::string`

```
// Das folgende Funktions-Template wandelt jeden beliebigen Typ in einen String um,
// der eine Streamausgabe besitzt.
// Ein entsprechendes Utility gibt es ab C++11 für die eingebauten Typen und heißt
// std::to_string()

auto toString(const auto& x)
{    std::ostringstream oss;  // leeren ostringstream anlegen
     oss << x;                // den Wert von x hineinschreiben
     return oss.str();        // den String zurückgeben
}

// Das folgende Template liest einen beliebigen Typ T aus einem C++String

void fromString(std::string s, auto& x)            // liest x aus s
{    std::istringstream{std::move(s)} >> x; }      // nur 1 Zeile !
```

`std::istringstream{std::move(s)}`
erzeugt einen iStringstream ohne Namen (dann ist dieser temporär und wird gleich nach Gebrauch vernichtet). Dieser wird mit `std::move(s)` initialisiert und mit `>> x` ausgelesen. Fertig!

Da wir den String `s` als Kopie übergeben haben, wird er danach nicht mehr benötigt. Man kann ihn deshalb per `std::move()` an den Konstruktor übergeben, was Zeit sparen kann.