FH-OÖ Hagenberg/HSD
Software Entwurf 2, SS 2013
Übung 6

Name: Abgabetermin: KW 19

Mat.Nr: Punkte:

Übungsgruppe: korrigiert:

Aufwand in h:

Beispiel 1 (16 Punkte) Aktienkurse: Schreiben Sie mit Hilfe des Scanners ein C++ Programm, das die Startkurse (in Euro) von Aktien einliest und in einer entsprechenden Datenstruktur speichert. Die Eingabe enthält eine beliebige Liste von Aktien mit dem dazugehörigen Start-Kurswert.

Eingabebeispiel:

```
"Andritz AG": 68.00
"Zumtobel": 24.61
"VOEST Alpine": 33.88
"Telekom Austria": 10.34
"OMV": 31.90
...
```

Simulieren Sie nun den täglichen Aktienhandel an der Börse, indem Sie mit Hilfe des Zufallszahlengenerators auf der Kommunikationsplattform einen Prozentsatz ermitteln, mit dem ein neuer Aktienwert pro Börsentag berechnet wird. Der Prozentsatz soll eine Genauigkeit von zwei Stellen haben und im Bereich zwischen -5% und +5% liegen. Speichern Sie die neu berechneten Kurse ebenfalls in der Datenstruktur. Zur Simulation implementieren Sie eine Funktion `SimulateStocks`, der die einzelnen Aktien und die Anzahl der Handelstage übergeben werden. Simulieren Sie die Kurse für ein Jahr und geben Sie anschließend die Ergebnisse aus.

Die Ausgabe soll (am Ende des simulierten Jahres) nach dem Namen sortiert in der folgenden Form erfolgen:

```
-------------------------------------------------------------------
Aktien          (+/-)Proz   Aktuell    Vortag      Hoch       Tief
-------------------------------------------------------------------
Andritz AG         -1.33%     55.64     56.40     72.58     50.50
OMV                +2.44%     12.59     12.29     34.31     10.45
Telekom Austria    -0.91%      6.65      6.71     11.47      5.81
VOEST Alpine       +3.64%     41.51     40.05     46.24     27.04
Zumtobel           +2.69%     21.50     20.94     28.84     15.39
...
```

Pro Aktie werden folgende Informationen ausgegeben: Name der Aktie, prozentuelle Änderung zum Vortag, aktueller Aktienwert, Aktienwert am Vortag, höchster Aktienwert, niedrigster Aktienwert. Die Spaltenbreite für den Aktiennamen richtet sich nach dem längsten Namen. Die Spaltenbreite für die Prozentangabe und für die einzelnen Aktienwerte soll 10 Zeichen betragen. Ihre Ausgabe soll exakt auf die gleiche Art und Weise formatiert sein. Definieren Sie sich dafür geeignete Ausgabemanipulatoren.

Definieren Sie zusätzlich folgende Prädikate und testen Sie diese:

- Sortierung nach dem höchsten aktuellen Aktienwert absteigend.
- Sortierung nach dem höchsten erreichten Aktienwert absteigend.
- Sortierung nach der maximalen Differenz zwischen höchstem und niedrigstem erreichten Aktienwert absteigend.

Hinweis: Der STL-Algorithmus `max_element(begin,end)` liefert einen Iterator auf das größte Element im angegebenen Bereich `(begin,end)`. Das Gegenstück ist der Algorithmus `min_element(begin,end)`.

Verwenden Sie in Ihrer Implementierung Typdefinitionen, sodass sich der verwendete Container und der Elementtyp leicht durch andere Typen ersetzen lassen.

Beispiel 2 (8 Punkte) Iteratoren und Prädikate: Erweitern sie das Modul aus Beispiel 1 um eine Funktion `Extract`, die als Eingangsparameter zwei Iteratoren aufnimmt, die den Anfang und das Ende eines Containers bezeichnen, in dem die zuvor eingelesenen und simulierten Aktien abgespeichert sind.

`Extract` soll für alle Aktien jene Aktienwerte, die über dem Startwert liegen, herausfiltern. Der ursprüngliche Aktien-Container darf dabei nicht verändert werden.

`Extract` retourniert als Funktionswert einen Container mit allen Aktien, von denen mindestens ein Wert extrahiert wurde. Jede Aktie in diesem Container soll nur die extrahierten Werte enthalten.

Schreiben Sie eine Funktion, die den von `Extract` gelieferten Container auf einen beliebigen Ausgabestrom ausgibt. Geben Sie den Namen der Aktie und die Anzahl der extrahierten Werte aus. Der

Container soll vor der Ausgabe aufsteigend nach der Anzahl der Werte sortiert werden.

Allgemeine Hinweise: Legen Sie bei der Erstellung Ihrer Übung großen Wert auf eine **saubere Strukturierung** und auf eine **sorgfältige Ausarbeitung!** Verwenden Sie immer **Module**, um den Testtreiber und die eigentliche Implementierung zu trennen! Dokumentieren Sie alle Schnittstellen und versehen Sie Ihre Algorithmen an entscheidenden Stellen ausführlich mit **Kommentaren! Testen** Sie ihre Implementierungen ausführlich! Geben Sie **Lösungsideen** an!