ulm university universität
uulm

Institut für Numerische Mathematik
Dr. Andreas F. Borchert und Dr. Michael C. Lehn

2. Februar 2017
Blatt 13

Objektorientierte Programmierung mit C++ (WS 2016/2017)

Abgabe bis zum 9. Februar 2017, 16:00 Uhr

Lernziele:

- Verwendung assoziativer Container-Klassen aus der STL

Aufgabe 16: Schauspieler/Film-Raten

Dieses Ratespiel, das die Filmkenntnisse auf eine harte Probe stellt, beginnt mit der Nennung eines beliebigen bekannten Schauspielers. Dann muss ein Film benannt werden, in dem dieser mitgespielt hat, dann ein anderer Schauspieler aus dem Film. Dann ein anderer Film, in dem der zuletzt genannte Schauspieler teilgenommen hat. Und so weiter. Dabei dürfen aber weder Schauspieler noch Filme mehrfach genannt werden. Ein Spiel könnte etwa so ablaufen:

```
thales$ movies data/movies.latin1
228563 actors and 382726 movies loaded.
*** Actors/Movies Game ***
Actor: Matt Damon
Movie: The Martian
Actor: Jeff Daniels
Movie: 2 Days in the Valley
Actor: Teri Hatcher
Movie: Tomorrow Never Dies
Actor: Pierce Brosnan
Movie: Mars Attacks!
Actor: Jack Nicholson
Movie: As Good as It Gets
Actor: Helen Hunt
Movie: The Silence of the Lambs
Sorry, but Helen Hunt did not participate in 'The Silence of the Lambs'
Well done, you have scored 5 points!
thales$
```

Damit das Spiel durchgeführt werden kann, werden natürlich die Daten benötigt, die sämtliche Filme und die darin teilnehmenden Schauspieler nennen. Glücklicherweise stellt die *Internet Movie Database* (IMDB) dieses Material zur Verfügung. Dieses wurde für den Zweck dieser Übungsaufgabe aufbereitet und unter den Dateien

/home/www/htdocs/numerik/cpp/ws16/uebungen/13/movies.latin1

und

/home/www/htdocs/numerik/cpp/ws16/uebungen/13/movies.utf8

bzw. unter den URLs

```
http://www.mathematik.uni-ulm.de/numerik/cpp/ws16/uebungen/13/
                          movies.latin1
```

und

```
http://www.mathematik.uni-ulm.de/numerik/cpp/ws16/uebungen/13/
                           movies.utf8
```

zur Verfügung gestellt. Sie können sich die passende Datei aussuchen je nach der bevorzugten Kodierung (ISO-8859-1 oder UTF-8). Bitte kopieren Sie nicht diese Dateien auf unseren Rechnern, da sie 112 Megabyte groß ist. Arbeiten Sie stattdessen besser mit einem symbolischen Verweis:

```
thales$ ln -s /home/www/htdocs/numerik/cpp/ws16/uebungen/13/movies.latin1 .
```

In jeder Zeile der Datei wird ein Film und seine Schauspieler benannt. Die einzelnen Teile einer Zeile sind durch Tilden getrennt („~“). Zuerst kommt der Name des Films, dann werden die einzelnen Schauspieler benannt. So sieht ein (relativ kurzer) Eintrag aus der Datei aus:

```
Angara~Aasia~Allauddin~Sabiha Khanum~Sultan Rahi~Yusuf Khan
```

Das ist der Film *Angara*, an dem die Schauspieler Aasia, Allauddin, Sabiha Khanum, Sultan Rahi und Yusuf Khan teilgenommen haben.

Im Rahmen der Übungsaufgabe sind die Klassen *Actor*, *Movie* und *Database* zu erstellen. Auf dieser Basis ist dann das Ratespiel zu implementieren. Sie werden zu der Implementierung assoziative Container-Klassen der STL benötigen. Es steht Ihnen dabei frei, ob Sie geordnete oder ungeordnete Container verwenden.

Beachten Sie dabei, dass der Speicheraufwand mit 382.726 Filmen und 228.563 Schauspielern gewaltig ist. Deswegen sollten die Container mit Zeigern auf die jeweiligen Objekte zeigen und nicht die Objekte duplizieren. Da Sie dann mehrere assoziative Container haben, die auf die gleichen Objekte verweisen, sollten sie entsprechend von geeigneten *smart pointers* Gebrauch machen, damit die Aufräumarbeiten automatisiert werden.

Sie werden sowohl für die Schauspieler als auch all die Filme jeweils einen assoziativen Container benötigen. Zusätzlich ist die Beziehung zwischen Schauspielern und Filmen umzusetzen. Die Beziehung lässt sich entweder dadurch realisieren, dass entweder jeder Schauspieler „weiß“, an welchen Filmen er teilgenommen hat oder bei jedem Film steht, welche Schauspieler daran beteiligt waren. Empfohlen sei hier die erstgenannte Möglichkeit, denn wenn bei den Schauspielern jeweils ein assoziativer Container mit den Filmtiteln untergebracht wird, reduziert das die Gesamtzahl der assoziativen Container, da die Zahl der Schauspieler geringer als die der Filme ist.

Hinweise: Zum Einlesen aus einer Datei empfiehlt sich *std*::*ifstream*. Für das zeilenweises Einlesen erscheint *std::getline* recht geschickt. Beim Zerlegen der Zeile lohnt es sich mit einem *std::istringstream* zu arbeiten, einem Stream, der aus einem *std::string*-Objekt liest. Aus diesem können Sie dann wiederum mit *std::getline* lesen, wobei als dritter Parameter auch das Trennzeichen angegeben werden kann.

Sie können wie üblich Ihre Lösung wieder einreichen:

```
thales$ tar cvf game.tar *.?pp [mM]akefile
thales$ submit cpp 16 game.tar
```

Viel Erfolg!