PR1 VU Programmierung 1	Abschlussklausur	26.06.2018

Implementieren Sie die Klassen `Gossipgirl` und `Clique`:

Ein `Gossipgirl` hat einen Vornamen (string), Zunamen (string) und einen Ortsteil. Der Ortsteil ist ein Wert aus der vordefinierten Enumeration `Ortsteil(Ortsteil::Hudson, Ortsteil::Queens, Ortsteil::Bronx, Ortsteil::Brooklyn, Ortsteil::Manhattan)`. Für die Klasse Gossipgirl sind folgende Methoden zu implementieren:

- Konstruktor(en) mit 2 bzw. 3 Parametern. Vorname, Zuname und Ortsteil in dieser Reihenfolge. Der Ortsteil ist optional und per Default Manhattan. Bei Ortsteil Hudson oder leerem Vor- bzw. Zunamen, ist ein Fehler vom Typ `runtime_error` zu werfen.
- `bool uebersiedeln (const Ortsteil&)`: Setzt den Ortsteil des `this`-Objekts auf den als Parameter erhaltenen Ortsteil. Bei einer Änderung des Ortsteils soll `true` zurück gegeben werden, ansonsten `false`. Beim Übersiedeln ist `Ortsteil::Hudson` erlaubt.
- `int bekanntheitsgrad() const`: Berechnet den Bekanntheitsgrad eines Gossipgirls. Der Bekanntheitsgrad besteht aus der Summe der Länge des Zunamens und des Werts des Ortsteils. Der Wert des Ortsteils wird durch seine Position in der enum Definition bestimmt. (Hudson=0, Queens=1, etc.)
- `bool operator==(const Gossipgirl&) const`: Liefert `true` zurück, falls Vorname, Zuname und Ortsteil gleich sind, ansonsten `false`.
- `operator<<`: Ein `Gossipgirl`-Objekt muss in der Form [*Vorname*, *Zuname*, *Ortsteil*, *Bekanntheitsgrad*] ausgegeben werden. Der vordefinierte Vektor `ortsteilnamen` kann für die Ausgabe der Enumerationswerte verwendet werden, z.B.: [Serena, van der Woodsen, Manhattan, 19].

Eine Clique hat einen Namen (string) und eine Mitgliedsliste (`vector<Gossipgirl>`). Für die Klasse `Clique` sind folgende Methoden zu implementieren:

- Konstruktor mit 2 Parametern. Name und Vektor von `Gossipgirl`-Objekten in dieser Reihenfolge. Die Elemente aus dem Parametervektor müssen unter Beibehaltung der Reihenfolge in die Mitgliedsliste der Clique übernommen werden. Bei leerem Namen oder leerer Mitgliedsliste ist ein Fehler vom Typ `runtime_error` zu werfen.
- `bool check(const Gossipgirl&) const`: Retourniert `true`, wenn das als Parameter erhaltene Gossipgirl in der Mitgliedsliste dieser Clique enthalten ist, `false` sonst.
- `double durchschnitt_bekanntheit() const`: Berechnet den durchschnittlichen Bekanntheitsgrad aller Mitglieder dieser Clique.
- `bool hinzufuegen(const Gossipgirl&)`: Falls das als Parameter erhaltene Gossipgirl bereits Mitglied dieser Clique ist, wird `false` retourniert, andernfalls wird es am Ende der Mitgliedsliste eingefügt und `true` retourniert.
- `operator<<`: Die Ausgabe eines Objekts des Typs `Clique` muss in der Form [*Name*, {*Mitgliedsliste*}] erfolgen, z.B.: [UpperEastSiders, {[Serena, van der Woodsen, Manhattan, 19], [Daniel, Humphrey, Brooklyn, 11]}].
- Zusatz für 10 Punkte: Erweitern Sie die Klasse `Clique` um folgende Methode:
  `void entfernen()`: Entfernt aus der Mitgliedsliste der Clique alle Gossipgirls, deren Bekanntheitsgrad unter dem (am Beginn der Operation ermittelten) durchschnittlichen Bekanntheitsgrad der Clique liegt. Die relative Reihenfolge in der Liste soll dabei nicht verändert werden.
- Zusatz für 15 Punkte: Erweitern Sie die Klasse `Clique` um folgende Methode:
  `void aufnehmen(const vector<Gossipgirl>& v, bool fame)`: Die Gossipgirls aus v, die noch nicht Mitglieder sind, sollen aufgenommen werden (sie sind unter Beibehaltung der relativen Reihenfolge am Ende der Mitgliedsliste einzufügen). Ist der Parameter `fame true`, werden nur Gossipgirls aufgenommen, die mindestens den gleichen Bekanntheitsgrad haben wie der Durchschnitt der Clique am Beginn des Methodenaufrufs. Ist `fame false`, ist der Bekanntheitsgrad irrelevant.

Implementieren Sie die Klassen `GossipGirl` und `Clique` mit den notwendigen Konstruktoren, Methoden und Operatoren, sodass jedenfalls das Rahmenprogramm kompiliert und ausgeführt werden kann und die gewünschten Ergebnisse liefert. Achten Sie in Ihren Konstruktoren darauf, dass nur gültige Objekte erstellt werden können. Werfen Sie gegebenenfalls eine Exception vom Typ `runtime_error`.

Für Ihr Programm dürfen Sie **nur** die im vorgegebenen Rahmenprogramm angeführten include-Dateien verwenden!

Instanzvariablen sind `private` zu definieren und die Verwendung globaler Variablen ist (abgesehen von im Rahmenprogramm eventuell bereits definierten) nicht erlaubt! Die Datenkapselung darf nicht durchbrochen werden. Es ist daher unter anderem nicht erlaubt, Referenzen oder Pointer auf private Instanzvariablen einer Klasse nach außen zu vermitteln, `friend`-Deklarationen (mit Ausnahme bei Operatorfunktionen) zu verwenden, oder setter-Methoden zu implementieren, die die Integrität der Daten nicht gewährleisten. Interpretationsspielraum in der Angabe können Sie zu Ihren Gunsten nutzen.

Die Teilaufgaben, bei denen keine Punkteanzahl angegeben ist, gelten als Basisfunktionalität. Für eine positive Beurteilung ist zumindest die Basisfunktionalität zu implementieren. Diese wird mit 30 Punkten bewertet. Die übrigen Teilaufgaben müssen nicht unbedingt implementiert werden, führen aber im Falle einer korrekten Implementierung zu einer entsprechenden Erhöhung der Punkteanzahl.