PR1 VU Programmierung 1	Abschlussklausur	25.06.2019

Boutique

Implementieren Sie die Klassen `Garment` (= Kleidungsstück) und `Boutique`: Ein `Garment`-Objekt hat die Instanzvariablen `description` (`string`, Bezeichnung, nicht leer), `stock` (`int`, Lagerstand nicht negativ), `designer` (ein Wert aus der vordefinierten Enumeration `Designer` `Designer::Prada`, `Designer::Chanel`, `Designer::Dior` und `Designer::Unknown`) und `reserved` (`vector<string>`, eventuell leere Liste der Namen jener KundInnen, die eine Reservierung für das Kleidungsstück haben). Der Einfachheit halber wird davon ausgegangen, dass sowohl die Bezeichnungen der Kleidungsstücke, als auch die Namen der KundInnen jeweils eindeutig sind. Für die Klasse `Garment` sind folgende Methoden und Funktionen zu implementieren:

- Konstruktor(en) mit 2 oder 3 Parametern: Bezeichnung, Lagerstand und Designer in dieser Reihenfolge. Designer ist optional mit dem Defaultwert `Designer::Unknown`. Sollte einer der Parameter nicht die Voraussetzungen erfüllen (z. B. Bezeichnung ist leer oder Lagerstand ist negativ), ist eine Exception vom Typ `runtime_error` zu werfen. Die Liste der Reservierungen ist für ein neues `Garment`-Objekt leer.
- `bool by_designer(const vector<Designer>& vd) const`: Retourniert `true`, wenn das `this`-Objekt von einem der Designer in der Liste `vd` stammt, `false` sonst.
- `int available() const`: Retourniert die Anzahl der noch verfügbaren Stücke des Kleidungsstücks (`this`-Objekts). Dazu ist die Anzahl auf Lager (`stock`) um die Anzahl der Reservierungen zu vermindern.
- `bool reserve(const string& name)`: Wenn der Parameter `name` leer ist, muss eine Exception vom Typ `runtime_error` geworfen werden. Sonst retourniert die Methode `false`, wenn nicht zumindest ein Stück noch verfügbar ist oder bereits eine Reservierung für denselben Namen vorliegt. Andernfalls wird der KundInnen-Name (`name`) am Ende der Reservierungsliste eingefügt und `true` retourniert.
- `operator<<`: Kleidungsstücke müssen in der Form [*description*: *designer*, *available*, {*Liste der Reservierungen*}] ausgegeben werden. Dabei ist *available* die Anzahl der verfügbaren Stücke, z. B. **[Exquisite Shoe: Dior, 4, {Herbert, Susanne}]**. Der vordefinierte Vektor `designer_names` kann für die Ausgabe der Enumerationswerte verwendet werden.

Ein `Boutique`-Objekt hat nur eine Instanzvariable `offer` (nicht leere Liste der Kleidungsstücke im Angebot) Für die Klasse `Boutique` sind folgende Methoden und Funktionen zu implementieren:

- Konstruktor mit einem Parameter Angebot. Sollte der Parameter die Voraussetzungen nicht erfüllen (also Liste der Kleidungsstücke im Angebot ist leer), ist eine Exception vom Typ `runtime_error` zu werfen.
- `vector<Garment> selection(const vector<Designer>& vd) const`: Retourniert eine Liste aller Kleidungsstücke, die von einem der Designer in `vd` stammen und von denen mindestens ein Stück verfügbar ist. Die relative Reihenfolge der Kleidungsstücke in der retournierten Liste muss jener in der Angebotsliste der Boutique entsprechen.
- `operator<<`: Die Ausgabe eines Objekts vom Typ `Boutique` muss in der Form [{*Liste der Kleidungsstücke im Angebot*}] erfolgen. In der Liste der Kleidunsstücke werden jene, die nicht verfügbar sind, durch einen Stern ('*') davor und danach gekennzeichnet, z. B.:
  [{[Exquisite Shoe: Dior, 4, {Herbert, Susanne}], *[Skirt: Prada, 0, {Maria}]*, [T-Shirt: Unknown, 10, {}]}].
- Zusatz für 10 Punkte: Erweitern Sie die Klasse `Garment` um die Methode `bool buy(const string& name)`:
  Wenn eine Reservierung auf den Namen `name` existiert, so wird diese gelöscht, der Bestand (`stock`) wird um eins reduziert und `true` wird retourniert. Andernfalls ist zu prüfen, ob noch mindestens ein Stück verfügbar ist. Wenn dem so ist, wird der Bestand um eins reduziert und `true` zurückgeliefert sonst wird `false` retourniert.
- Zusatz für 15 Punkte: Erweitern Sie die Klasse `Boutique` um die Methode `vector<Garment> sale(Designer d)`:
  Diese entfernt alle Kleidungsstücke des Designers `d`, für die es nicht zumindest eine Reservierung gibt, aus dem Bestand. Zu retournieren ist die Liste aller Kleidungsstücke des Designers, die wegen bestehender Reservierungen nicht entfernt werden konnten.

Implementieren Sie die Klassen `Garment` und `Boutique` 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.