**Name:**

**Matrikelnummer:**

**Aufgabe 1.** Gegeben seien die Klassen *Pruefer*, *Pruefung* und *Pruefungskorrektur*.

Die Klasse *Pruefer* verfügt über die Methode

*public boolean gutGelaunt()*,

die genau dann *true* zurückgibt, wenn der Prüfer gut gelaunt ist.

Die Klasse *Pruefung* verfügt über die Methode

*public int gibMaximalePunkteanzahl()*,

die die maximal mögliche Punkteanzahl für die Prüfung zurückgibt, sowie über eine weitere Methode

*public void beurteile(String b)*,

die die Prüfung mit *b* beurteilt.

Die Klasse *Pruefungskorrektur* verfügt über einen parameterlosen Konstruktor sowie über die Methode

*public int gibErreichtePunkteanzahl(Pruefung einePruefung)*,

die die (objektiv) erreichte Punkteanzahl von *einePruefung* zurückgibt.

Schreiben Sie eine Klasse *Notenvergabe*, die von *Pruefungskorrektur* erbt. Die Klasse soll dabei über einen Konstruktor

*public Notenvergabe(Pruefer der Pruefer)*

verfügen. Überschreiben Sie außerdem die Methode *gibErreichtePunkteanzahl* so, dass zum Wert aus der Methode der Oberklasse bei guter Laune des Prüfers ein zufälliger Wert addiert wird, der höchstens ein Zehntel der maximal erreichbaren Punkteanzahl ausmacht. Achten Sie darauf, dass dabei die maximal mögliche Punkteanzahl nicht überschritten wird.

Weiters soll die Klasse *Notenvergabe* über eine Methode

*public void beurteile(Pruefung pruefung)*

verfügen, die entsprechend der Anzahl der erreichten Punkte eine Beurteilung nach dem folgenden Schema vergibt: Wurden $\leq 50\%$ der maximal möglichen Punkte erreicht, wird die Prüfung mit "nicht bestanden" beurteilt. Bei mehr als 50% soll die Prüfung entweder mit "bestanden" beurteilt werden, falls $\leq 80\%$ erreicht wurden, oder aber mit "mit Auszeichnung bestanden", falls mehr als 80% der maximal möglichen Punkte erreicht wurden.

**Aufgabe 2.** Gegeben seien die Klassen *Aktie* und *Aktienkurse*. In der Klasse *Aktie* ist die Methode *equals* aus *Object* überschrieben. Die Klasse *Aktienkurse* verfügt über eine statische Methode

*public static Map<Aktie, Double[]> gibKurse()*,

die für jede gehandelte Aktie den Kursverlauf seit Ausgabe der Aktie in einem Array zurückgibt.

Schreiben Sie eine Klasse *Aktiendepot* mit Konstruktor

*public Aktiendepot(Set<Aktie> dieAktien)*,

der ein Aktiendepot mit entsprechenden Aktien anlegt.

Schreiben Sie weiters eine Methode

*public void kaufenVerkaufen(int rein, int raus)*,

der sämtliche Aktien aus dem Depot entfernt, deren Kursverlauf in den letzten *raus* Einträgen fallend ist. Weiters soll diese Methode all jene gehandelten Aktien ins Depot aufnehmen, deren Kursverlauf in den letzten *rein* Einträgen steigend ist, vorausgesetzt diese Aktie ist nicht bereits im Depot vorhanden.

**Aufgabe 3.** Gegeben sei eine Klasse *Dominostein*, die über die Methode

*public int[] gibPunkte()*

verfügt, die die beiden Punkteanzahlen des Dominosteins in einem Array der Größe 2 zurückgibt.

Schreiben Sie eine Klasse *Dominospiel* mit einer Methode

*public boolean istKorrekt(ArrayList<Dominostein> dieSteine)*,

die genau dann `true` zurückgibt, wenn *dieSteine* aus richtig aneinandergelegten Dominosteinen besteht, d.h. für jeden Stein ist eine der beiden Punkteanzahlen identisch mit einer Punkteanzahl des vorhergehenden Steins, die andere Punkteanzahl ist identisch mit einer Punkteanzahl des nachfolgenden Steins. (Sie können davon ausgehen, dass die Dominosteine in *dieSteine* paarweise verschieden sind, wobei ein Stein {j,k} derselbe ist wie {k,j}. )

**Beispiel:** Die Listen {{4,3},{4,6},{1,6}} und {{3,4},{3,2},{4,2},{4,4}} sind richtig aneinandergelegt, nicht aber die Liste {{3,4},{3,2},{3,1}}.