Klassen und Objekte

Artikelverwaltung

Ein Artikel besteht aus einer Artikelnummer (artikelNr), einem Einzelpreis (einzelPreis) und einer Menge die speichert, wie viele Stück pro Einkauf verkauft wurden (Array menge).

Schreiben Sie jeweils eine Methode zum Lesen und zum Ausgeben eines Artikels:

```
static Article readArticle(String input) {…}
static void printArticle(Article a) {…}
```

wobei `Article` eine Klasse ist, die die Daten eines Artikels enthält.

Der String Input besteht aus den Informationen die in einem Artikel gespeichert werden sollten. zB

```
102700          999         1 3 1 2 4
102701          3250        2 13 4 1 1
102702          1190        2 1
```

Die Ausgaben sollen zB so aussehen:

102700	10989
102701	68250
102702	3570

Schneiden von Rechtecken

Schreiben Sie eine Methoden `intersection(r1, r2)`, die das Schnittrechteck der beiden Rechtecke r1 und r2 berechnet und zurückgibt.

Rechtecke sollen durch eine Klasse

```
class Rectangle {

  int x, y;
  int width;
  int height;

}
```

beschrieben werden. Wenn sich r1 und r2 nicht schneiden, soll `intersection` als Ergebnis null liefern.

Datumsberechnung

Implementieren Sie eine Methode `dayDiff(d1, d2)`, die zwei Datumsangaben d1 und d2 der Klasse `Date` als Parameter bekommt und ihre Differenz in Tagen zurückgibt. Ignorieren Sie Schaltjahre.

Die Klasse Date ist ebenfalls zu implementieren und besteht auf 3 int-Feldern (day, month, year).

Kundenkartei

Die Kundenkartei einer Firma enthält pro Kunden einen Datensatz, der den Namen des Kunden, seine Kundennummer, seine Privat- und Geschäftsadresse sowie eine Liste der im laufenden Jahr erfolgend Bestellungen enthält. Jede Bestellung besteht aus einer Bestellmenge, einer Artikelnummer und einem Artikelpreis. Adressen bestehen aus Straße, Hausnummer, Postleizahl und Ort.

Modellieren Sie diese Kartei mittels Arrays und Klassen. Erstellen Sie mindestens drei Beispiel-Kundenkarteien und initialisieren sie die Karteien.

Zahlenmengen

Implementieren Sie eine Klasse Set, die Mengen von Zahlen zwischen 0 und 31 verwaltet. Benutzen Sie als Datenstruktur eine int-Zahl, die als Bitfolge interpretiert wird. Die Zahl 44 hat zum Beispiel die Binärdarstellung 00000000000000000000000000101100 und kann als Menge {2, 3, 5} aufgefasst werden, weil darin die Bits 2, 3 und 5 gesetzt sind. Implementieren Sie Operationen zum Einfügen und Entfernen von Elementen sowie zum Vereinigen von Mengen und zur Berechnung der Schnittmenge zweier Mengen. Hinweis: die Bitoperationen können mit den Operatoren &, | und ~ implementiert werden.

Vektoren

Ein Vektor im zweidimensionalen Raum kann durch zwei reelle Zahlen dargestellt werden. Implementieren Sie eine Klasse Vector, die Operationen für die Addition von Vektoren, die Multiplikation eines Vektors mit einem Skalar sowie für das Skalarprodukt zweier Vektoren (die Summe der Produkte der Komponenten) anbietet. Sehen Sie auch einen geeigneten Konstruktur vor. Zur Berechnung der Wurzel einer Zahl x kann Math.sqrt(x) verwendet werden.

Komplexe Zahlen

Eine komplexe Zahl besteht aus einem reellen und einem imaginären Teil, beide vom Typ float. Implementieren Sie eine Klasse Complex, die komplexe Zahlen darstellt. Als Operationen sollen die vier Grundrechnungsarten (+, -, *, /) sowie geeignete Kontrukturen angeboten werden.

Prioritätenschlange

Implementieren Sie eine Klasse PriorityQueue, die Elemente nach Prioritäten verwaltet. Jedes Element hat zusätzlich zu seinen Daten eine Priorität zwischen 0 und 9. Die Operation put(x, prio) fügt das Element x gemäß seiner Priorität in die Schlange ein. Die Operation get() liefert aus der Schlange das Element mit der höchsten Priorität. Enthält die Schlange mehrere Elemente gleicher Priorität, so sollen sie in der Reihenfolge geliefert werden, in der sie in die Schlange gestellt wurden.

Wörterbuch

Implementieren Sie ein Wörterbuch als Klasse Dictionary. Das Wörterbuch soll paare von Worten enthalten (z.B. deutsches Wort und seine englische Übersetzung), wobei das erste Wort als Schlüssel dient und das zweite als Wert. Sehen Sie zumindest folgende beiden Operationen vor:

```
class Dictionary {

  void insert(String key, String value) { … }
  String lookup(String key) { … }

}
```

Insert fügt key und value in das Wörterbuch ein. Lookup sucht key im Wörterbuch und liefert das entsprechende value oder null, falls key nicht gefunden wird.

Ihr Wörterbuch kann maximal 100 Wörter speichern.

Uhrzeiten

Implementieren Sie eine Klasse Time zur Speicherung von Uhrzeiten. Jedes Time Objekt soll eine Zeit in Form von Stunden und Minuten speichern. Es soll folgende Methoden geben:

- Sinnvoller Konstruktor
- Addieren einer Zeit zu einer anderen (z.B. 5h 42min + 3h 27min = 9h 9min)
- Berechnung der Differenz zwischen zwei Zeiten in Minuten (z.B. 5h 20min – 3h 1- min = 130min)
- Umrechnung einer Zeit in Minuten (z.B. 3h 20min = 200min)