Java

Datenstrukturen

Aufgabe - Mengenlehre

Gegeben sind drei Zahlenmengen A, B, C - realisiert als Arrays vom Typ Integer:

A = { 49, 30, 14, 47, 13, 2, 12, 29, 19, 11, 15, 39, 43, 45, 34 }

B = { 39, 33, 38, 14, 4, 32, 40, 25, 17, 46, 35, 6, 2, 12, 49 }

C = { 41, 12, 5, 35, 42, 28, 47, 20, 26, 24, 50, 40, 14, 17, 10 }

Berechne möglichst effizient die Vereinigungsmenge von A ∪ B ∪ C sowie alle möglichen Schnittmengen und Differenzmengen der drei Zahlenmengen. Du kannst davon ausgehen, dass alle Mengen gleich groß sind. Gestalte deine Methoden mit dem **fluent-interface**. Das heißt, dass alle Methoden zur Berechnung dieser Verknüpfungen den selben return Typ haben sollen wie die Parameter.

```
getUnionList(a, getIntersectionList(b,c)); // das sollte funktionieren und heißt A ∪ (B ∩ C)
```

Optional Erweitere deine Methoden so, dass getUnionList und getIntersectionList mit 1-n parametern aufgerufen werden können. Das Keyword für die Google-Suche lautet "params"^^.

Java

Data Structures

Exercise - Set Theory

Three sets of integers A, B, C are given, realized as Java Arrays: A = { 49, 30, 14, 47, 13, 2, 12, 29, 19, 11, 15, 39, 43, 45, 34 } B = { 39, 33, 38, 14, 4, 32, 40, 25, 17, 46, 35, 6, 2, 12, 49 } C = { 41, 12, 5, 35, 42, 28, 47, 20, 26, 24, 50, 40, 14, 17, 10 }

Calculate the union as well as all possible difference and intersections of these three sets with focus on efficiency. You can assume that all collections have the same size. Create your methods with the **fluent-interface** in mind. That means that every method should return the same type as it takes in through parameters.

```
getUnionList(a, getIntersectionList(b,c)); // this should work and means A ∪ (B ∩ C)
```

Optional Expand your methods getUnionList and getIntersectionList so that they can be called with 1-n parameters. The keyword for googling is "params"^^.

Java

Datenstrukturen

Aufgabe - Wörterbuch: Deutsch - Englisch

Erstelle ein Programm welches ein Wörterbuch zum Übersetzen von Wörtern zwischen Englisch und Deutsch implementiert (bi-direktional). Es soll folgende Funktionalitäten umfassen: - Hinzufügen eines neuen Wort-Paars (Englisch und Deutsch) - Entfernen eines Wort-Paars aus dem Wörterbuch

- Übersetzen eines Wortes (Deutsch oder Englisch) in die jeweilig andere Sprache

Java

Data Structures

Exercise - Dictionary: German - English

Create a Java program realizing a dictionary for the languages English and German (bi-directional). Following functionalities have to be implemented: - Add a new word-pair (English and German) to the dictionary - Remove a word-pair from the dictionary - Translate a Word (Geman or English) to the corresponding other "target" language. [8:45 AM]

Java

Datenstrukturen

Aufgabe - Liste sortieren

Wähle einen der bereits mit Arrays implementierten Sortieralgorithmus und ändere die Implementation derart, dass statt Arrays Listen verwendet werden. Wichtig: Es soll NICHT die eingebaute Sortierfunktion des Listen Interface, sondern euer eigener Algorithmus verwendet werden.

Java

Data Structures

Exercise - Sort a list

Choose any prior implemented sorting algorithm (working with arrays) and change it to work with List instead. Important: Do NOT use the built-in sort method from the List interface.