Übung 04: Brüche als Objekte

Vorbereitung

Laden Sie die im Repository bereitgestellten Dateien `Rational.java` und `Test.java` auf Ihren PC und fügen sie diese Dateien einem IntelliJ-Projekt hinzu. Hierfür gibt es 2 Möglichkeiten.

- Legen Sie zunächst ein neues leeres Projekt an und kopieren Sie die beiden Dateien dann direkt über das Dateisystem (z.B. Windows-Explorer) in das `src`-Verzeichnis des IntelliJ-Projektes.
- Erzeugen Sie sich ein neues Projekt aus bestehenden Source-Dateien durch „*File – New - Project from existing sources*“ bzw. „*Import Project*“).

Aufgabe 1: Klasse Rational

Implementieren Sie die Klasse `Rational`! Jede Instanz der Klasse soll einen Bruch bestehend aus Zähler und Nenner repräsentieren. Verwenden Sie den vorgegebenen Code und vervollständigen Sie alle mit `..` gekennzeichneten Stellen.

```
public class Rational {
   private long numerator;                          // Zähler
   private long denominator;                        // Nenner

   private void norm() { .. }                       // kürzen, Nenner positiv

   // Konstruktoren
   public Rational() { .. }                         // erzeugt Standard-Bruch: 0/1
   public Rational(long num, long den) { .. }
   public Rational(long val) { .. }                 // erzeugt Bruch: “val”/1
   public Rational(double val) { .. }
   public Rational(String val) { .. }

   // get-Methoden
   public long getNumerator() {..}
   public long getDenominator() {..}

   // einfache Umwandlungen
   public double doubleValue() { .. }               // Division Zähler/Nenner
   public String toString() { .. }                  // Darstellung "a/b"
   public Rational negate() { .. }                  // Multiplikation mit -1
   public Rational invert() { .. }                  // Kehrbruch (aus a/b wird b/a)

   // Bruchrechnen
   public Rational add(Rational val) { .. }
   public Rational subtract(Rational val) {..}      // addiere negierten Bruch
   public Rational multiply(Rational val) { .. }
   public Rational divide(Rational val) { .. }      // multipliziere Kehrbruch
}
```

Hinweise:

- Beachten Sie, dass es verschiedene Möglichkeiten gibt, einen Bruch darzustellen. Zum Beispiel handelt es sich bei 1/2 und 2/4 mathematisch um den gleichen Wert. Intern soll ein Bruch deshalb immer im ***vollständig gekürzten*** Zustand gespeichert sein (also 1/2). Ferner soll der Nenner ***immer positiv*** sein. Dies wird durch die Methode `norm()` erreicht.
- Verwenden Sie für ***norm()*** die bereits vorgegebene Methode `long gcd(long a, long b)`, die den größten gemeinsamen Teiler aus zwei Zahlen a und b berechnet.
- Die Klasse `Rational` soll **unveränderlich („immutable“) sein.** Die Methoden `negate, invert, add, subtract, multiply, divide` lassen die Aufrufparameter unverändert und erzeugen als Ergebnis ***immer neue Objekte***.

(umblättern)

- Verwenden Sie im Konstruktor `Rational(String val)` die Klasse `Scanner` mit dem *Delimiter* „/“, um aus einem String „`x/y`“ Zähler `x` und Nenner `y` zu ermitteln: https://docs.oracle.com/javase/8/docs/api/java/util/Scanner.html
- Verwenden Sie soweit als möglich ***bestehende Methoden***. Beispielsweise lässt sich eine Division von Brüchen auf eine Multiplikation von Brüchen zurückführen.
- Verwenden sie zur Fehlersuche ggfs. ***Debugging.***

Aufgabe 2: Test

Testen Sie Ihr Programm mit der `main`-Methode von `Test.java`. Innerhalb dieser Methode werden einige einfache Brüche erzeugt bzw. einfache Berechnungen durchgeführt. Sollte eine Berechnung falsch sein, so gibt es auf der Standardausgabe eine entsprechende Fehlermeldung.

Hinweis: In späteren Übungen wird für solche Tests ***JUnit*** eingesetzt.

Aufgabe 3 (Zusatzaufgabe)

Implementieren Sie einige Test mit ***JUnit***.