Übung 03

Korrektur: Sebastian Pritz

Name: Wolfgang Eder

**Punkte: 64**

Allgemeine Anmerkungen:

# Beispiel 1: Operatoren überladen 64 von 100

Lösungsidee: 15/22

|  |  |
| --- | --- |
| **Anmerkung**  Normalize nicht beschrieben (nur erwähnt)  Wofür werden die „friend“ Deklarationen gebraucht? | **Abzug**  -5  -2 |

Quellcode: 24/45

|  |  |
| --- | --- |
| **Anmerkung**  Die Implementierungen von den Rechnungsarten bei Brüchen sind leider falsch ☹ Bei Addition/Subtraktion, muss man auf den kleinsten gemeinsamen Nenner bringen vorher. Multiplizieren ist Nenner1 \* Nenner2 und Zähler1 \* Zähler2. Dividieren ist ähnlich zum Multiplizieren, aber mit dem Kehrwert des zweiten Bruchs.  Normalize: Ist etwas merkwürdig den Nenner 0 ungefragt auf 1 zu ändern, falls dieser 0 ist. Für das ist ja die Exception da. | **Abzug**  **-20**  **-1** |

Testfälle: 25/33

|  |  |
| --- | --- |
| **Anmerkung**  Testfall „not same“ ist etwas verwirrend, weil „same“ ausgegeben wird, obwohl sie ungleich sind.  Ergebnisse der Testfälle großteils Falsch (z.B. dass zwei positive Brüche dividert nichts negatives ergeben können) , wegen den Fehlern bei den Grundrechnungsarten oben, und kein Kommentar dazu.  Durch den „Fehler“ in Normalize, ist der Testfall mit „DivideByZero“ leider hinfällig 🡪 Denominator wird nie 0.  Abseits davon, aber alle Testfälle zumindest abgedeckt.  Bei Comparisons sollte man beide Seiten testen: z.B. bei smaller than: 1x größer(gleich) und 1x kleiner. | **Abzug**  **Hinweis**  **-5**  **-2**  **-1** |