Mathehelfer Bruchrechnen

Ein Hackathon für Kinder und Jugendliche

Norbert Seulberger

September 2022

Inhaltsverzeichnis

1 Die Aufgabenstellung

Der Mathehelfer Bruchrechnen kann in der Grundversion Brüche und Kommazahlen ineinander umrechnen. Außerdem können Brüche gekürzt werden.

Die Darstellung der Umrechnungen erfolgt in einem Browser. Dabei werden die Brüche grafisch schön mit Zähler, Nenner und Bruchstrich dargestellt wie in einem Mathematikbuch.

Es sind also beispielsweise folgende Operationen möglich:

$$\frac{6}{8} = \frac{3}{4}$$

$$\frac{3}{4} = 0.75$$

$$0.8 = \frac{4}{5}$$

Dabei geben wir die linke Seite fest vor und berechnen die rechte Seite jeweils mit einem Programm. Wir schreiben jeweils eine eigene Funktion, also drei Funktionen, die

- einen Bruch kürzt,
- einen Bruch in eine Kommazahl umrechnet oder
- eine Kommazahl in einen Bruch umwandelt.

Als Programmiersprache verwenden wir Python.

Die Formelzeile tragen wir automatisch in eine kleine Web-Seite ein, die wir uns im Browser ansehen. Dabei beschreiben wir die Formel in einer speziellen HTML-Sprache für mathematische Formeln: MathML.

Die Sprache MathML ist sehr einfach. Allerdings muss man sehr viel Text eingeben, um eine Formel mit MathML zu beschreiben. Daher schreiben wir ein Programm, das die Formel automatisch in MathML ausdrückt.

2 MathML

MathML ist eine einfache Markup-Sprache, die in einem HTML-Dokument dazu verwendet werden kann, um mathematische Formeln darzustellen.

Genauso wie in HTML werden die Daten in sogenannte Tags eingeschlossen, d.h. ein Element beginnt mit einen Tag, dann stehen die Daten und ein schließendes Tag beendet den Ausdruck.

2.1 Operanden und Operatoren

Für die verschiedenen Teile einer mathematischen Formel gibt es unterschiedliche Elemente.

- `<mn>42</mn>` stellt die Zahl 42 dar.
- `<mo>=</mo>` beschreibt ein Gleichheitszeichen.
- `<mo>+</mo>` schreibt ein Pluszeichen.

Die Formel $2 + 3 = 5$ lautet also in MathML:

```
1 <mn>2</mn>
2 <mo>+</mo>
3 <mn>3</mn>
4 <mo>=</mo>
5 <mn>5</mn>
```

Listing 1: Eine einfache Addition

2.1 Tipp

Der Browser ignoriert die Zeilenumbrüche in MathML. Wir können die Formel also auch in eine Zeile schreiben.

```
1 <mn>2</mn><mo>+</mo><mn>3</mn><mo>=</mo><mn>5</mn>
```

Listing 2: Die Formel in einer Zeile

2.2 Brüche

2.3 Eine Formelzeile

Eine Formel wird wie folgt geschrieben: `<math>` Hier steht die Formel. `</math>`

Damit die Formel in eine eigene Zeile geschrieben wird, verwenden wir das Absatz-Tag von HTML: `<p>` ... `</p>`

Also zusammen:

```
<p>
  <math>
    Hier steht die Formel.
  </math>
</p>
```

Listing 3: Eine Formelzeile

3 Aufgabenblatt MathML

3.1 Aufgabe: Anzeige der HTML-Datei

Schau dir im Firefox-Browser die Datei `manuell.html` an. Die Anzeige sollte etwa so aussehen:

$$\frac{6}{8} = \frac{3}{4}$$

Hier kannst du die Formeln für das Umrechnen eines Bruchs in eine Dezimalzahl und umgekehrt einfügen.

3.2 Aufgabe: Ansicht der HTML-Datei im Editor

Schau dir jetzt dieselbe Datei im Editor der Entwicklungsumgebung an. Der Text sieht etwa so aus:

```
<!doctype html>
<html lang="de">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Bruchrechnen</title>
  </head>
  <body>
    <p>
      <math>
        <mfrac>
          <mi>6</mi>
          <mi>8</mi>
        </mfrac>
        <mo>=</mo>
        <mfrac>
          <mi>3</mi>
          <mi>4</mi>
        </mfrac>
      </math>
    </p>
    <p>
      Hier kannst du die Formeln für das Umrechnen ...
    </p>
  </body>
```

```
26 </html>
```

Listing 4: Die Datei manuell.html

Kannst du die Element entdecken, die für die Darstellung des Kürzens der beiden Brüche stehen?

3.3 Aufgabe: Die Beschreibung der beiden Umrechungen

Füge in die Datei `manuell.html` den Text in der MathML-Sprache ein, so dass die beiden Umrechungen dargestellt werden (zusätzlich zur Formel für das Kürzen). Der Text „Hier kannst du ...“ darf überschrieben werden.

$$\frac{6}{8} = \frac{3}{4}$$

$$\frac{3}{4} = 0.75$$

$$0.8 = \frac{4}{5}$$

Überprüfe das Ergebnis im Firefox-Browser.

4 Beispiel: Die Darstellung eines Bruches in MathML

Lies dieses Beispiel sorgfältig durch. Es dient als Vorlage für die Aufgaben, die du danach selbstständig lösen sollst.

4.1 Der Algorithmus

Einen Bruch in MathML zu schreiben, ist nicht schwierig, aber mühsam. Daher möchten wir eine Funktion ein Python programmieren, die das automatisch erledigt. Der Algorithmus lautet so:

- Nimm einen Bruch entgegen.
- Wandle den Zähler des Bruchs in eine Zeichenkette.
- Wandle den Nenner des Bruchs in eine Zeichenkette.
- Gib der Zeichenkette, die den Bruch in MathML beschreibt, einen Namen.
- Baue die Zeichenketteaus Einzelteilen zusammen.
- Beginne die Zeichenkette mit `<mfrac>`.
- Rahme die Zeichenkette, die den Zähler beschreibt, mit den Tags `<mi>` und `</mi>` ein.
- Verfahre ebenso mit der Zeichenkette für den Nenner.
- Beende die Zeichenkette mit `</mfrac>`.
- Gib die Zeichenkette zurück.

4.2 Die Funktion

Die Funktion sieht dann so aus:

```
1 def schreibeBruch(bruch: Bruch) -> str:
2   zaehlerAlsString = str(bruch.zaehler)
3   nennerAlsString = str(bruch.nenner)
4   ergebnis = "<mfrac>"
5   ergebnis += "<mi>" + zaehlerAlsString + "</mi>"
6   ergebnis += "<mi>" + nennerAlsString + "</mi>"
7   ergebnis += "</mfrac>"
8   return ergebnis
```

Listing 5: Darstellung eines Bruchs in MathML

4.3 Das Überprüfen des Ergebnisses

In der Funktion `schreibeMathML()` ergänzen wir nun die Zeilen für die Darstellung des Bruchs.

```
1  def schreibeMathML() -> str:
2    inhalt = "<p><math><mi>2</mi><mo>+</mo><mi>3</mi>"
3          + "<mo>=</mo><mi>5</mi></math></p>"
4    bruch: Bruch = Bruch(3,4)
5    inhalt += "\n\t\t<p>"
6    inhalt += "<math>" + schreibeBruch(bruch) + "</math>"
7    inhalt += "</p>"
8    return inhalt
```

Listing 6: Schreibe die HTML-Datei

Danach führen wir die Python-Datei `htmlErzeugung.py` aus und betrachten im Firefox-Browser die Datei `index.html`

5 Algorithmen

Ein Algorithmus ist die eindeutige Beschreibung eines Verfahrens, um eine bestimmte Aufgabe zu lösen.

Beispielsweise lautet ein möglicher Algorithmus, um den Mittelwert zweier ganzer Zahlen zu berechnen:

- Nimm zwei Zahlen entgegen.
- Gib den beiden Zahlen Namen, z.B. a und b.
- Bilde die Summe der beiden Zahlen.
- Teile die Summe durch 2.
- Gib das Ergebnis der Division zurück.

6 Funktionen

Eine Funktion ist ein kleines Teilprogramm, das eine spezifische Aufgabe löst. Für bestimmte Eingabewerte erhält man immer dasselbe Ergebnis.

6.1 Beispiel: Eine Funktion zur Mittelwertberechnung zweier Zahlen

Der Algorithmus lautet:

7 Beispiel: Das Kürzen von Brüchen

Lies dieses Beispiel sorgfältig durch. Es dient als Vorlage für die Aufgaben, die du danach selbstständig lösen sollst.

7.1 Der Algorithmus

Das Kürzen eines Bruches funktioniert so:

- Nimm einen Bruch.
- Merke dir den Zähler und den Nenner des Bruchs.
- Berechne den größten gemeinsamen Teiler von Zähler und Bruch. (Es gibt dafür eine fertige Funktion.)
- Berechne den gekürzten Zähler.
- Berechne den gekürzten Nenner.
- Erstelle einen neuen Bruch aus dem gekürzten Zähler und dem gekürzten Nenner.

7.2 Die Funktion zum Kürzen von Brüchen

Die Funktion lautet dann:

```
1 def kuerzeBruch(bruch: Bruch) -> Bruch:
2   alterZaehler = bruch.zaehler
3   alterNenner = bruch.nenner
4   kuerzungsFaktor = ggT(alterZaehler, alterNenner)
5   neuerZaehler = alterZaehler // kuerzungsFaktor
6   neuerNenner = alterNenner // kuerzungsFaktor
7   neuerBruch = Bruch(neuerZaehler, neuerNenner)
8   return neuerBruch
```

Listing 7: Funktion zum Kürzen von Brüchen

7.3 Die Funktion zur Darstellung der Rechnung zum Kürzen

Der Algorithmus lautet:

- Nimm einen Bruch.
- Erzeuge aus diesem Bruch einen gekürzten Bruch. Verwende dazu die Funktion, die du eben gesehen hast.

- Erzeuge aus dem ursprünglichen Bruch die Beschreibung in MathML. Verwende dazu die Funktion, die du im vorherigen Beispiel gesehen hast.
- Erzeuge aus dem gekürzten Bruch die Beschreibung in MathML. Verwende dazu ebenfalls die Funktion aus dem vorherigen Beispiel.
- Setze jetzt den MathML-Text aus den einzelnen Angaben zusammen.
- Die Zeichenkette beginnt mit <math>.
- Es folgt der MathML-Text für den ungekürzten Bruch.
- Dann kommt das Gleichheitszeichen.
- Es folge der MathML-Text für den gekürzten Bruch.
- Die Zeichenkette beginnt mit </math>

Die Funktion lautet dann:

```
1 def schreibeKuerzen(bruch: Bruch) -> str:
2   bruchGekuerzt = kuerzeBruch(bruch)
3   textUngekuerzt = schreibeBruch(bruch)
4   textGekuerzt = schreibeBruch(bruchGekuerzt)
5   return "<math>" + textUngekuerzt + "<mo>=</mo>" + textGekuerzt + "
      </math>"
```

Listing 8: MathML für das Kürzen eines Bruchs

7.4 Überprüfung des Ergebnisses

Wir überprüfen das Ergebnis analog zum Vorgehen im Beispiel für die Darstellung eines Bruchs. Dazu ergänzen wir in der Funktion `schreibeMathML()` die Zeilen

```
1 bruchUngekuerzt: Bruch = Bruch(6,8)
2 inhalt += "\n\t\t<p>"
3 inhalt += schreibeKuerzen(bruchUngekuerzt)
4 inhalt += "</p>"
```

Listing 9: Integration des Kürzens

8 Aufgabenblatt: Umwandlung eines Bruchs in eine Dezimalzahl

9 Aufgabe: Umwandlung einer Dezimalzahl in einen Bruch

10 Aufgabe: Programmiere die Grundrechenarten für Brüche

Diese Aufgabe ist etwas schwieriger.

11 Aufgabe: Gemischte Zahlen

Diese Aufgabe ist schwierig.

12 Aufgabe: Objektorientierte Programmierung

Diese Aufgabe ist sehr schwierig. Es werden gute Kenntnisse der Programmierung in Python vorausgesetzt.