1. Übung	Berechenbare Analysis	SS 2025
	Besprechung: Mittwoch, 30.04.2025	

Aufgabe 1:

Unter `moodle.uni-trier.de` finden Sie mehrere kleine Python-Programme für Implementierungen berechenbarer reller Zahlen. Implementieren Sie nun analog den Teil (5) von Satz 3.1 wie folgt; in Moodle ist ein passender VPL-Rahmen dazu vorgegeben:

- Erstellen Sie eine zu 3.1(5) passende Klasse `REAL` in der Datei `decidableset.py`.
- Der Konstruktor der Klasse soll dabei natürlich die Idee von 3.1(5) wiederspiegeln und über zwei Parameter also eine ganze Zahl und auf geeignete Art eine entscheidbare Menge $A \subseteq \mathbb{N}$ verwenden.
- Wie können dann zum Beispiel die reellen Zahlen $1/3$ oder $8+1/5$ konstruiert werden?
- Als kleines Anwendungsbeispiel der Klasse soll diese auch eine Methode `asString(self,n)` enthalten, die ähnlich zu dem Python-Beispiel `bsp_1.py` bei einer Zahl $x = z + x_A$ die Annäherung $z + \sum_{i \in A, i<n} 2^{-i-1}$ in binärer Form anzeigt.

  Als Test sollte bei der Zahl $x = 41/5 (= 8 + 1/5)$ der String `x.asString(10)` wie folgt formatiert sein:

  `1000.0011001100`
- Entwickeln Sie einen Algorithmus, der für beliebige (feste) Nenner funktionieren würde...

Aufgabe 2:

Modifizieren Sie Ihre Lösung der obigen Aufgabe so, dass aus einer rationalen Zahl als Parameter eine reelle Zahl erzeugt werden kann, etwa in der Form `x = REAL(mpq(a,b))`. Die Ausgabe aus der obigen Aufgabe sollte also auch durch folgendes Programm erzeugt werden können:

```
1 from decidableset import REAL
2 from gmpy2 import mpq
3 a = 41
4 b = 5
5 x = REAL( mpq(a,b) )
6 print ( x.asString(10) )
```

Wieder ist ein passender VPL-Rahmen in Moodle vorhanden. Die Methode `asString(self,n)` soll unverändert bleiben!

Tipp: Es gibt hier etliche Wege zu einer Lösung, etwa mit Subklassen, mit optionalen Argumenten bei Konstruktoren, mit Hilfe von Lambda-Funktionen...; außerdem helfen Integer-Operationen wie Division `//` oder Rest `%`...