Humboldt University of Berlin

Einführung in das wissenschaftliche Rechnen

Documentation of CLI Fraction Calculator

Christian Parpart & Kei Thoma

May 17, 2019

Contents

1 Introduction

2 Euclidean Algorithm Library

In this module, we first implemented the well known euclidean algorithm which finds the greatest common divisor given two integers. From there, we use the result of the above mentioned algorithm to calculate the least common multiple.

2.1 euclidean_algorithm(a, b)

Arguments

1. `first_number` (int): the first integer, a; negative values are accepted
2. `second_number` (int): the second integer, b; negative values are accepted

Returns

- (int): the greatest common divisor found via the recursive euclidean algorithm

Description

Given two integers, she finds the greatest common divisor via the recursively implemented euclidean algorithm.

The algorithm itself starts with two integers a and b. If $b = 0$ then a is returned and the recursive loop stops. In any other case, this function is called again, but the arguments are modified in the following manner

$$a \mapsto (a \mod b) \qquad\qquad b \mapsto a,$$

or if one prefers to read the statement in code

```
return euclidean_algorithm(b, a % b)
```

Worked Example of the Algorithm

Let $a = 195$ and $b = 1287$. Following the algorithm above, we have

$$\begin{array}{llll}
\text{Step 0} & a_0 = 195 & b_0 = 1287 & \\
\text{Step 1} & a_1 = 1287 & b_1 = 195 \mod 1287 & = 195 \\
\text{Step 2} & a_2 = 195 & b_2 = 1287 \mod 195 & = 117 \\
\text{Step 3} & a_3 = 117 & b_3 = 195 \mod 117 & = 78 \\
\text{Step 4} & a_4 = 78 & b_4 = 117 \mod 78 & = 39 \\
\text{Step 4} & a_5 = 39 & b_5 = 78 \mod 39 & = 0
\end{array}$$

Since $b = 0$, the algorithm is broken and $a_5 = 39$ is returned.

2.2 least_common_multiple(a, b)

Arguments

1. `first_number` (int): the first integer, a; negative values are accepted
2. `second_number` (int): the second integer, b; negative values are accepted

Returns

- (int): the least common multiple calculated with the help of the euclidean algorithm and the formula

Description

This finds the least common multiple.

Example

2.3 main()

3 Fraction API

The Fraction class in `fraction.py` implements a fraction, i.e. concepts such as $\frac{1}{2}$ or $-\frac{2}{3}$, mathematically correctly. For this endeavor, Fraction saves three pseudo-private attributes representing the unsigned numerator, the unsigned denominator and finally the sign of the fraction.

After an instance of Fraction is initialized, it is automatically reduced properly to the most minimal form, e.g. $\frac{8}{12}$ naturally becomes $\frac{2}{3}$, with the help of the euclidean algorithm.

Finally, to allow some easy way to handle this class, few build-in operators such as the absolute function and binary addition were overloaded.

3.1 __init__(numerator, denominator)

Arguments

1. `numerator` (int): the numerator; negative values are allowed, but is then saved as a positive integer at `numerator_`
2. `denominator` (int): the denominator; negative values are allowed, but is then saved as a positive integer at `denominator_`; if no argument is passed, it defaults to 1

Note that even though `numerator_` and `denominator_` are always positive, the sign of the Fraction is determined at the point of initalization and is saved under the boolean attribute `sign_`.

Raises

- `ZeroDivisionError`: if 0 is passed as the parameter for the denominator

Description

3.2 Get Attribute Functions

Fortunataly or unfortunataly depending on one's perspective about dynamic languages, Python does not allow private attributes or methods. However, we don't want that the three attributes, `numerator_`, `denominator_`, and `sign_`, are modifiable from the outside of the Fraction class. Therefore, this class provides three methods, `get_numerator()`, `get_denominator()`, and `get_sign()`, which simply returns the respective attribute.

3.3 __pos__()

Return

- (self): returns the unchanged self

Description

Overloading the unitary plus operator is not very exciting. The fraction object is unchanged and returned immediately.

3.4 __neg__()

3.5 __abs__()

3.6 __add__()

3.7 __sub__()

3.8 __mul__()

3.9 __truediv__()

3.10 __str__()

3.11 __main__()

4 Fraction Calculator CLI

5 Improvement Horizon

5.1 Project Complexity One

5.2