

# Fraction

<b>COLLABORATORS</b>
----------------------

	<i>TITLE :</i> Fraction	
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>
WRITTEN BY		July 10, 2022

<b>REVISION HISTORY</b>
-------------------------

NUMBER	DATE	DESCRIPTION	NAME

# Contents

<b>1</b>	<b>Fraction</b>	<b>1</b>
1.1	Fractional numbers implementation for AmigaTalk© 1998: . . . . .	1

---

# Chapter 1

## Fraction

### 1.1 Fractional numbers implementation for AmigaTalk© 1998:

The Class Fraction is an implementation of rational numbers (i.e., ones with a numerator & denominator) for the AmigaTalk system.

The parent Class is Magnitude.

The methods defined for Class Fraction are:

`new`

Initialize a new instance of Class Fraction.

`numerator`

Return the Integer that represents the numerator.

`denominator`

Return the Integer that represents the denominator.

`fraction`

Return the Float that represents the Fraction.

`asFloat`

Return the Fraction as a floating point number. Report an error if the denominator is zero.

`coerce: aNumber`

Transform aNumber to a Fraction.

`numerator: newNum`

Change the numerator of the receiver to newNum.

`denominator: newDenom`

Change the denominator of the receiver to newDenom.

`reciprocal`

Invert the Fraction & report an error if the old numerator was zero, resulting in an improper Fraction. Return the floating point representation of the Fraction.

+ aNumber

Add aNumber to the receiver Fraction. If aNumber is not a Fraction, transform it into one first.

Return the floating-point representation of the result.

- aNumber

Subtract aNumber to the receiver Fraction. If aNumber is not a Fraction, transform it into one first.

Return the floating-point representation of the result.

\* aNumber

Multiply aNumber to the receiver Fraction. If aNumber is not a Fraction, transform it into one first.

Return the floating-point representation of the result.

/ aNumber

Divide aNumber to the receiver Fraction. If aNumber is not a Fraction, transform it into one first.

Return the floating-point representation of the result.

printString

Print the receiver as a String.

== aNumber

Test whether the receiver is equal to aNumber.

~= aNumber

Test whether the receiver is NOT equal to aNumber.

< aNumber

Test whether the receiver is less than aNumber.

> aNumber

Test whether the receiver is greater than aNumber.

<= aNumber

Test whether the receiver is less than or equal to aNumber.

>= aNumber

Test whether the receiver is greater than or equal to aNumber.

---