

Complex

COLLABORATORS

	<i>TITLE :</i> Complex		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY		August 26, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	Complex	1
1.1	Complex number Class implementation for AmigaTalk© 1998:	1

Chapter 1

Complex

1.1 Complex number Class implementation for AmigaTalk© 1998:

The Complex Class is an implementation of complex numbers for the AmigaTalk system. Its parent Class is Magnitude.

Methods available for the Complex Class are:

`new`

Initialize a new instance of the Class Complex.

`realpart`

Return the real number portion of the Complex number.

`imagpart`

Return the imaginary number portion of the Complex number.

`magpart`

Return the magnitude of the Complex number.

WARNING: There is no cross-checking to see if the magnitude is correct, use `computeMag` or `computeMagPhase` first!

`phasepart`

Return the phase of the Complex number.

WARNING: There is no cross-checking to see if the phase is correct, use `computeMagPhase` first!

`computeMag`

Determine the magnitude of the Complex number from the real & imaginary portions.

`computeMagPhase`

Determine the magnitude & phase of the Complex number from the real & imaginary portions. If the imaginary part is zero, an error will be reported!

`realpart: newReal`

Change the real number portion of the Complex number.

imagpart: newImag

Change the imaginary number portion of the Complex number.

magpart: newMag

Change the magnitude (class instance variable) of the Complex number.

phasepart: newPhase

Change the phase (class instance variable) of the Complex number.

coerce: aNumber

Transform aNumber to an instance of Class Complex.

conjugate

Compute the complex conjugate of the Receiver.

~

Compute the complex conjugate of the Receiver.

+ aNumber

Add a number to the Complex receiver. The number will be transformed to a Complex if it's not one already!

- aNumber

Subtract a number from the Complex receiver. The number will be transformed to a Complex if it's not one already!

* aNumber

Multiply a number by the Complex receiver. The number will be transformed to a Complex if it's not one already!

/ aNumber

Divide a number into the Complex receiver. The number will be transformed to a Complex if it's not one already!

An error message is returned if aNumber is equal to zero.

printString

Print the Complex number as a String.

== aNumber

Test whether the Receiver is equal to aNumber.

< aNumber

Test whether the magnitude of the receiver is less than aNumber.

> aNumber

Test whether the magnitude of the receiver is greater than aNumber.

<= aNumber

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

`>= aNumber`

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

`~= aNumber`

Test whether the receiver is NOT equal to aNumber.
