Testing Document for Project Three

Kameron Damaska

3/28/2016

Contents

1 ComplexN Class

1.1 Constructor

Takes two double parameters

- The first parameter initializes the real part of the complex number.
- The second parameter initializes the imaginary part of the complex number.

1.2 getRealPart

This method must return a double value that represents the real part of the complex number.

The testing must cover:

- the return value equaling the real part from the constructor.

1.3 getImaginaryPart

This method must return a double value that represents the imaginary part of the complex number.

The testing must cover:

- the return value equaling the real part from the constructor.

1.4 toString

This method must return a String value for the complex number in the form of "x + yi", with x being the real part and y being the imaginary part.

The testing must cover:

- non-negative
- negative
- zero

values for both the real and imaginary parts of the complex number.

1.5 equals

This method must return a boolean that represents whether the object parameter is equal to this ComplexN.

Return true when:

- both objects are instances of ComplexN
- the parameters real and imaginary parts equal

Return false otherwise.

The test must cover:

- a ComplexN paramer with equal real and imaginary parts
- a ComplexN parameter with unequal real parts
- a ComplexN parameter with unequal imaginary parts
- an Object that is not a ComplexN

2 RealN

2.1 Constructor

Takes one double parameter that is the real number

Assigns the imaginary part of the number to 0

2.2 getRealPart

This method must return a double value equal to the real number.

The testing must cover:

- the return value equaling the real number set in the constructor.

2.3 getImaginaryPart

This method must return 0.0.

2.4 toString

This method should return a String value that represents the real number.

The test must cover:

- a String whos value is equal to the real number parameter

2.5 equals

This method must return a boolean that represents whether the object parameter is equal to this RealN.

Return true when:

- both objects are instances of RealN
- the real number parameters are equal

Return false otherwise.

The test must cover:

- a RealN parameter with equal real numbers
- a RealN parameter with unequal real numbers
- an Object that is not a RealN

3 RationalN

3.1 Constructor

Take two int parameters.

- First int parameter sets the numerator.
- Second int parameter sets the denominator
- Real part set the numerator divided by the denominator
- An ArithmeticError is thrown when the denominator is equal to 0.

The test must cover:

- non-zero denominator
- denominator equal to 0.

3.2 getRealPart

This method must return a double value that is equal to the decimal equivalent of the numerator over the denonimator.

The test must cover:

- the return value equaling the numerator divided by the denominator.

3.3 getImaginaryPart

This method must equal 0.0.

3.4 getNumerator

This method must equal the numerator initialized in the constructor.

The test must cover:

- the return value equaling the numerator set in the constructor.

3.5 getDenominator

This method must equal the denominator initialized in the constructor.

The test must cover:

- the return value equaling the denominator set in the construcotr.

3.6 toString

This method must return a String value that represents the rational number.

When the denominator equals 1, the String should be the numerator.

When the denominato is not 1, the String should be the number in rational form.

The test must cover:

- a rational number with a numerator of 1. The returned String should be the numerator
- a rational number with a numerator that doesn't equal 1. The returned String should be the numerator, followed by a division symbol, followed by the denominator.

3.7 equals

This method must return a boolean that represents whether the object parameter is equal to this RationalN.

Return true when:

- both objects are instances of RationalN
- The numerator divided by the denominator are equal for each RationalN

Return false otherwise.

The test must cover:

- a RationalN parameter with equal numerator and denominator to the RationalN
- a RationalN parameter whos ratio between the numerator and denominator is equal to the RationalN.
- a RationalN parameter whos ratio between the numerator and denominator is unequal to the RationalN.
- an Object that is not a RationalN.

4 IntegerN

Takes one int parameter.

- The int parameter sets the numerator.
- The denominator is set to 1.

4.1 getRealPart

This method must return a double value that is equal to the integer set in the constructor.

The test must cover:

- the return value equaling the integer as a double.

4.2 getImaginaryPart

This method must return 0.0.

4.3 getNumerator

This method must return an int value that is equal to the integer set in the constuctor.

The test must cover:

- the return value equaling the integer.

4.4 getDenominator

This method must return 1.

4.5 toString

This method must return a String value that represents the integer value.

The test must cover:

- The returned String representing the integer parameter set in the constructor.

4.6 equals

This method must return a boolean that represents whether the object parameter is equal to this IntegerN.

Return true when:

- both objects are instances of IntegerN.
- The integer values are equal for both IntegerN.

Return false otherwise.

The test must cover:

- an IntegerN parameter with an equal integer to the IntegerN
- an IntegerN parameter with an unequal integer to the IntegerN.
- an Object that is not an IntegerN.

5 NaturalN

Takes one int parameter that sets the natural number.

Throws an ArithmeticError if int parameter is negative.

The test must include:

- negative int parameter
- non-negative int parameter

5.1 getRealPart

This method must return a double value that is equal to the natural number set in the constructor.

The test must cover:

- the return value equaling the natural number as a double.

5.2 getImaginaryPart

This method must return 0.0.

5.3 getNumerator

This method must return an int value that is equal to the natural number set in the constructor.

The test must cover:

- the return value equaling the natural number as a double.

5.4 getDenominator

This method must return 1.

5.5 toString

This method must return a String value that represents the natural number value.

The test must cover:

- The returned String representing the natural number parameter set in the constructor.

5.6 equals

This method must return a boolean that represents whether the object parameter is equal to this NaturalN.

Return true when:

- both objects are instances of NaturalN.
- The natural numbers are equal for both NatualN.

Return false otherwise.

The test must cover:

- a NaturalN parameter with an equal natual number to the NaturalN
- a NaturalN parameter with an unequal natual number to the NaturalN.
- an Object that is not an NaturalN.

6 Arithmetic

6.1 add

This method takes two of these types and returns the sum of an appropriate type.

Return NaturalN when:

- both parameters are NaturalN

Return IntegerN when:

- both parameters are IntegerN or
- one parameter is narrower than IntegerN

Return RationalN when:

- both parameters are RationalN or
- one parameter is narrower than RationalN

Return RealN when:

- both parameters are RealN or
- one parameter is narrower than RealN

Return ComplexN when:

- both parameters are ComplexN or
- one parameter is narrower than ComplexN

The test must cover:

- two NaturalN parameters
- two IntegerN parameters
  - both are typecasted to IntegerN
  - one narrower than IntegerN
- two RationalN parameters
  - both are typecasted to RationalN
  - one narrower than RationalN
- two RealN parameters
  - both are typecasted to RealN
  - one narrower than RealN
- two ComplexN parameters
  - both are typecasted to ComplexN
  - one narrower than RealN

6.2 subtract

This method takes two of these types and returns the difference of the two as an appropriate type.

Return NaturalN when:

- both parameters are NaturalN
- throws ArithmeticError when the second parameter is larger than the first parameter

Return IntegerN when:

- both parameters are IntegerN or
- one parameter is narrower than IntegerN

Return RationalN when:

- both parameters are RationalN or
- one parameter is narrower than RationalN

Return RealN when:

- both parameters are RealN or
- one parameter is narrower than RealN

Return ComplexN when:

- both parameters are ComplexN or
- one parameter is narrower than ComplexN

The test must cover:

- two NaturalN parameters
  - second NaturalN parameter larger than first
- two IntegerN parameters
  - both are typecasted to IntegerN
  - one narrower than IntegerN
- two RationalN parameters
  - both are typecasted to RationalN
  - one narrower than RationalN
- two RealN parameters
  - both are typecasted to RealN
  - one narrower than RealN
- two ComplexN parameters
  - both are typecasted to ComplexN
  - one narrower than RealN

6.3 multiply

This method takes two of these types and returns the product of the two as an appropriate type.

Return NaturalN when:

- both parameters are NaturalN

Return IntegerN when:

- both parameters are IntegerN or
- one parameter is narrower than IntegerN

Return RationalN when:

- both parameters are RationalN or
- one parameter is narrower than RationalN

Return RealN when:

- both parameters are RealN or
- one parameter is narrower than RealN

Return ComplexN when:

- both parameters are ComplexN or
- one parameter is narrower than ComplexN

The test must cover:

- two NaturalN parameters
- two IntegerN parameters
  - both are typecasted to IntegerN
  - one narrower than IntegerN
- two RationalN parameters
  - both are typecasted to RationalN
  - one narrower than RationalN
- two RealN parameters
  - both are typecasted to RealN
  - one narrower than RealN
- two ComplexN parameters
  - both are typecasted to ComplexN
  - one narrower than RealN

6.4 division

This method takes two of these types and returns the product of the two as an appropriate type.

An ArithmeticError error is thrown whenever the second parameter is equal to 0.

Return NaturalN when:

- both parameters are NaturalN

Return IntegerN when:

- both parameters are IntegerN
- one is narrower than IntegerN

Return RationalN when:

- both parameters are RationalN
- one is narrower than RationalN

Return RealN when:

- both parameters are RealN
- one is narrower than RealN

Return ComplexN when:

- both parameters are ComplexN
- one is narrower than ComplexN

The test must cover:

- two NaturalN parameters
- two IntegerN parameters
  - both are typecasted to IntegerN
  - one narrower than IntegerN
- two RationalN parameters
  - both are typecasted to RationalN
  - one narrower than RationalN
- two RealN parameters
  - both are typecasted to RealN
  - one narrower than RealN
- two ComplexN parameters
  - both are typecasted to ComplexN
  - one narrower than ComplexN
- A zero value in the second parameter of each type.