Assignment 1

Generated by Doxygen 1.8.13

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 4

Class Documentation

4.1 complex_adt.ComplexT Class Reference

An ADT for complex numbers.

Public Member Functions

- def __init__ (self, x, y)

  Constructor for ComplexT class.
- def real (self)

  Returns the real value of the complex number.
- def imag (self)

  Returns the real value of the complex number.
- def get_r (self)

  Returns radius of the complex number.
- def get_phi (self)

  Returns angle of the complex number.
- def equal (self, obj)

  Checks if two numbers are equal.
- def conj (self)

  Calculates the conjugate of the complex number.
- def add (self, obj)

  Adds two complex numbers.
- def sub (self, obj)

  Subtracts two complex numbers.
- def mult (self, obj)

  Multiplies two complex numbers.
- def recip (self)

  Reciprocal function.
- def div (self, obj)

  Divides two complex numbers.
- def sqrt (self)

  Square root of the complex number.

Static Public Attributes

- **x**
- **y**

4.1.1 Detailed Description

An ADT for complex numbers.

This class represents a complex number composed of real and imaginary components

4.1.2 Constructor & Destructor Documentation

4.1.2.1 __init__()

```
def complex_adt.ComplexT.__init__ (
              self,
              x,
              y )
```

Constructor for ComplexT class.

This constructor creates an object which represents a complex number in the form a + bi.

Parameters

x	The real value of the complex number
y	The imaginary value of the complex number

4.1.3 Member Function Documentation

4.1.3.1 add()

```
def complex_adt.ComplexT.add (
              self,
              obj )
```

Adds two complex numbers.

This function returns the sum of two complex numbers by by creating a new complex number with the real and imaginary components respectively summed

Returns

The sum of the complex numbers

4.1.3.2 conj()

```
def complex_adt.ComplexT.conj (
              self )
```

Calculates the conjugate of the complex number.

This function returns the reciprocal of the function by duplicating the current ComplexT, but with a negative imaginary

Returns

The reciprocal of the complex number

4.1.3.3 div()

```
def complex_adt.ComplexT.div (
              self,
              obj )
```

Divides two complex numbers.

This function returns the quotient of two complex numbers by by creating a new complex number that is the result of the complex number multiplied by the reciprocal of the input

Returns

The quotient of the complex numbers

Parameters

obj	The divisor

4.1.3.4 equal()

```
def complex_adt.ComplexT.equal (
              self,
              obj )
```

Checks if two numbers are equal.

This function determines if two complex numbers are equal by checking their respective real and imaginary values against eachother

Returns

True if the two numbers are equivalent

Parameters

obj	Complex number being checked against

4.1.3.5 get_phi()

```
def complex_adt.ComplexT.get_phi (
              self )
```

Returns angle of the complex number.

This function returns the angle of the complex number

Returns

The angle of the complex number

Exceptions

ZeroDivisionError	if the denominator comes out to zero (such as in the case of 0 + 0i)

4.1.3.6 get_r()

```
def complex_adt.ComplexT.get_r (
              self )
```

Returns radius of the complex number.

This function returns the polar length of the complex number

Returns

The radius of the complex number

4.1.3.7 imag()

```
def complex_adt.ComplexT.imag (
              self )
```

Returns the real value of the complex number.

This function returns the stored imaginary value of the complex number.

Returns

The imaginary value of the number

4.1.3.8 mult()

```
def complex_adt.ComplexT.mult (
              self,
              obj )
```

Multiplies two complex numbers.

This function returns the product of two complex numbers by by creating a new complex number with the respective components multiplied by expansion.

Returns

The product of the complex numbers

Parameters

obj	The second factor of the multiplication

4.1.3.9 real()

```
def complex_adt.ComplexT.real (
              self )
```

Returns the real value of the complex number.

This function returns the stored real value of the complex number.

Returns

The real value of the number

4.1.3.10 recip()

```
def complex_adt.ComplexT.recip (
              self )
```

Reciprocal function.

This function returns the reciprocal of the current complex number.

Returns

The reciprocal of the complex number

4.1.3.11 sqrt()

```
def complex_adt.ComplexT.sqrt (
              self )
```

Square root of the complex number.

This function returns the square root of the complex number by computing the value of each respective component.

Returns

The square root of the complex number

4.1.3.12 sub()

```
def complex_adt.ComplexT.sub (
              self,
              obj )
```

Subtracts two complex numbers.

This function returns the difference of two complex numbers by by creating a new complex number with the real and imaginary components respectively subtracted

Returns

The difference of the complex numbers

The documentation for this class was generated from the following file:

- src/complex_adt.py

4.2 triangle_adt.TriangleT Class Reference

Triangle defined by 3 side lengths.

Public Member Functions

- def __init__ (self, a, b, c)
  Constructor for TriangleT.
- def get_sides (self)
  Returns the side lengths of the triangle.
- def equal (self, obj)
  Compares the current triangle and a given triangle.
- def perim (self)
  Sums the side lengths of all 3 sides.
- def area (self)
  Computes the area of the triangle.
- def is_valid (self)
  Determines whether the given triangle is possible in Euclidian space.
- def tri_type (self)
  Determines the type of the triangle.

Static Public Attributes

- **a**
- **b**
- **c**

4.2.1 Detailed Description

Triangle defined by 3 side lengths.

An ADT for a triangle represented by 3 side lengths

4.2.2 Constructor & Destructor Documentation

4.2.2.1 __init__()

```
def triangle_adt.TriangleT.__init__ (
              self,
              a,
              b,
              c )
```

Constructor for TriangleT.

This constructor creates a triangle from 3 given side lengths

Parameters

a	An integer representing the length of the first side
b	An integer representing the length of the second side
c	An integer representing the length of the third side

4.2.3 Member Function Documentation

4.2.3.1 area()

```
def triangle_adt.TriangleT.area (
              self )
```

Computes the area of the triangle.

Computes the area of the triangle using Heron's theorem

Returns

The area of the triangle

4.2.3.2 equal()

```
def triangle_adt.TriangleT.equal (
              self,
              obj )
```

Compares the current triangle and a given triangle.

This function checks if two side triangles are equivalent by checking if the sorted sets of their side lengths are equal

Returns

True if the triangles are equal

Parameters

obj	The triangle being compared to

4.2.3.3 get_sides()

```
def triangle_adt.TriangleT.get_sides (
              self )
```

Returns the side lengths of the triangle.

This function returns the three side lengths as a tuple

Returns

The three side lengths in a tuple

4.2.3.4 is_valid()

```
def triangle_adt.TriangleT.is_valid (
              self )
```

Determines whether the given triangle is possible in Euclidian space.

This fuction tests if all the side lengths are greater than 0, and that no side length is greater than the sum of the other two

Returns

True if the triangle is physically possible, False if otherwise

4.2.3.5 perim()

```
def triangle_adt.TriangleT.perim (
              self )
```

Sums the side lengths of all 3 sides.

This function returns the sum of all of the side lengths

Returns

The perimeter of the triangle

4.2.3.6 tri_type()

```
def triangle_adt.TriangleT.tri_type (
              self )
```

Determines the type of the triangle.

This function tests what type of triangle the current object is. If the triangle is not a possible triangle, it returns a None type. Due to only being able to return a single value, right triangles are prioritized over isosclese and scalene triangles in the case that it happens to be both.

Returns

A TriType value representing the type of triangle

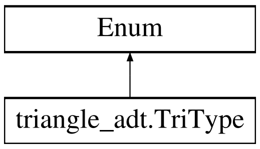
The documentation for this class was generated from the following file:

- src/triangle_adt.py

4.3 triangle_adt.TriType Class Reference

TriType enumerate object type.

Inheritance diagram for triangle_adt.TriType:



Static Public Attributes

- int **equilat** = 1
- int **isosceles** = 2
- int **scalene** = 3
- int **right** = 4

4.3.1 Detailed Description

TriType enumerate object type.

This class is an enumerated list which represents one of four different types of triangles.

The documentation for this class was generated from the following file:

- src/triangle_adt.py

Chapter 5

File Documentation

5.1 src/complex_adt.py File Reference

Contains a class for representing a complex number.

Classes

- class complex_adt.ComplexT

  An ADT for complex numbers.

5.1.1 Detailed Description

Contains a class for representing a complex number.

Author

scotta30

Date

2021-01-13

5.2 src/triangle_adt.py File Reference

Contains a class which represents a given triangle.

Classes

- class triangle_adt.TriangleT

  Triangle defined by 3 side lengths.
- class triangle_adt.TriType

  TriType enumerate object type.

5.2.1 Detailed Description

Contains a class which represents a given triangle.

Author

Alan Scott

Date

01/18/2020

Index