```
##*************************************************************
## C.Bruni (instructor)
## CS 116 Fall 2022
## Assignment 01 Problem 1
##*************************************************************

import check
import math

def doubling_time(i):
  '''
  Returns the absolute error in rule of 72 estimation vs actual
  for calculating investment doubling given the interest rate i.

  doubling_time: Float -> Float
  requires: i > 0.0

  Example:
  doubling_time(0.08) => 0.00646834200587768
  '''
  approx =  72/(100*i)
  exact = math.log(2)/math.log(1+i)
  return abs(exact - approx)

#Examples

check.within("Example", doubling_time(0.08), 0.00646834200587768,
0.00001)

#Tests:

check.within("Test small", doubling_time(0.01), 2.339283106, 0.00001)
check.within("Test large", doubling_time(1.00), 0.28, 0.00001)
check.within("Test really large", doubling_time(10.00), 0.21706482631,
             0.00001)
check.within("Test random", doubling_time(0.25), 0.2262837195,
0.00001)


##*************************************************************
## C.Bruni (instructor)
## CS 116 Fall 2022
## Assignment 01 Problem 1
##*************************************************************

import check
import math


def d(x1, y1, x2, y2):
```

```
  '''
  Returns the distance between (x1, y1) and (x2, y2)

  d: (anyof Int Float) (anyof Int Float)
       (anyof Int Float) (anyof Int Float) -> Float
  '''
  return math.sqrt((x1-x2)**2 + (y1-y2)**2)

def fire_safety(x1, y1, x2, y2, x3, y3, hx, hy):
  '''
  Returns the distance of the fire station at
  (x1, y1) or (x2, y2) or (x3, y3) closest to (hx, hy)

  d: (anyof Int Float) (anyof Int Float)
       (anyof Int Float) (anyof Int Float)
       (anyof Int Float) (anyof Int Float) -> Float

  Examples:
     fire_safety(-3, 0, 4, 0, 4, 5, 6, 6) => 2.23606797749979...
     fire_safety(-3, 0, 4, 0, 4, 5, 0, 0) => 3.0
  '''
  return min(d(x1, y1, hx, hy),
             d(x2, y2, hx, hy),
             d(x3, y3, hx, hy))

EPSILON = 0.00001
##Examples

check.within("Ex1", fire_safety(-3, 0, 4, 0, 4, 5, 6, 6),
             2.23606797749979, EPSILON)

check.within("Ex2", fire_safety(-3, 0, 4, 0, 4, 5, 0, 0),
             3.0, EPSILON)

#Tests

check.within("Origin", fire_safety(0, 0, 0, 0, 0, 0, 0, 0),
             0.0, EPSILON)

check.within("Origin Moved house", fire_safety(0, 0, 0, 0, 0, 0, 3,
4),
             5.0, EPSILON)

check.within("Floats", fire_safety(1.2, 1.4, 0.3, -2, 0.3, 4.5, 2.1,
2.1),
             1.1401754250991383, EPSILON)

check.within("All same", fire_safety(-3.2, 0.0, 3.2, 0, 0.0, -3.2,
0.0, 0.0),
```

```
              3.2, EPSILON)

check.within("Negative coords house",
              fire_safety(-3.2, 0.0, 3.2, 0, 0.0, -3.2, -1.5, -2.2),
              1.8027756377319946, EPSILON)


##################################################################
## C.Bruni (instructor)
## CS 116 Fall 2022
## Assignment 01 Problem 1
##################################################################

import check
import math

def outer_radius(r, n):
  '''
  Returns the radius of the n outer circles mutually tangent
  to inner circle of given radius r

  outer_radius: Float Nat -> Float
  requires: r > 0.0, n >= 3

  Examples:

  outer_radius(1.0, 6) => 0.9999999999999999
  outer_radius(2.0, 4) => 4.828427124746189

  '''
  sin_inner_angle = math.sin(math.pi/n)
  return r*(sin_inner_angle/(1-sin_inner_angle))


check.within("Example 1", outer_radius(1.0, 6), 1.0, 0.00001)
check.within("Example 2", outer_radius(2.0, 4), 4.828427, 0.00001)

#Tests
check.within("Test smallest n", outer_radius(1.0, 3), 6.46410161,
0.00001)
check.within("Test large n", outer_radius(1.0, 200), 0.01595797,
0.00001)
check.within("Test small radius", outer_radius(0.0003, 6), 0.0003,
0.00001)
check.within("Test small radius 2", outer_radius(0.0003, 10),
0.000134164078,
              0.00001)
check.within("Test random", outer_radius(2.65, 7), 2.03101728,
0.00001)
```