consigne

Triangle
Determine if a triangle is equilateral, isosceles, or scalene.

An equilateral triangle has all three sides the same length.

An isosceles triangle has at least two sides the same length. (It is sometimes specified as having exactly two sides the same length, but for the purposes of this exercise we'll say at least two.)

A scalene triangle has all sides of different lengths.

Note
For a shape to be a triangle at all, all sides have to be of length > 0, and the sum of the lengths of any two sides must be greater than or equal to the length of the third side. See Triangle Inequality.

Dig Deeper
The case where the sum of the lengths of two sides equals that of the third is known as a degenerate triangle - it has zero area and looks like a single line. Feel free to add your own code/tests to check for degenerate triangles.

tests : `triangle.spec.js`

```
var Triangle = require('./triangle');

describe('Triangle', function () {
  it('equilateral triangles have equal sides', function () {
    var triangle = new Triangle(2, 2, 2);
    expect(triangle.kind()).toEqual('equilateral');
  });

  xit('larger equilateral triangles also have equal sides', function () {
    var triangle = new Triangle(10, 10, 10);
    expect(triangle.kind()).toEqual('equilateral');
  });

  xit('isosceles triangles have last two sides equal', function () {
    var triangle = new Triangle(3, 4, 4);
    expect(triangle.kind()).toEqual('isosceles');
  });
  xit('isosceles triangles have first two sides equal', function () {
    var triangle = new Triangle(2, 2, 3);
    expect(triangle.kind()).toEqual('isosceles');
  });

  xit('isosceles trianges have first and last sides equal', function () {
    var triangle = new Triangle(4, 3, 4);
    expect(triangle.kind()).toEqual('isosceles');
  });

  xit('isosceles triangles have two first sides equal', function () {
    var triangle = new Triangle(4, 4, 3);
    expect(triangle.kind()).toEqual('isosceles');
  });

  xit('isosceles triangles have in fact exactly two sides equal', function (
) {
    var triangle = new Triangle(10, 10, 2);
    expect(triangle.kind()).toEqual('isosceles');
  });

  xit('scalene triangles have no equal sides', function () {
    var triangle = new Triangle(3, 4, 5);
    expect(triangle.kind()).toEqual('scalene');
  });

  xit('scalene triangles have no equal sides at a larger scale too', functio
n () {
    var triangle = new Triangle(10, 11, 12);
    expect(triangle.kind()).toEqual('scalene');
  });

  xit('scalene triangles have no equal sides in descending order either', fu
nction () {
    var triangle = new Triangle(5, 4, 2);
    expect(triangle.kind()).toEqual('scalene');
  });

  xit('very small triangles are legal', function () {
    var triangle = new Triangle(0.4, 0.6, 0.3);
    expect(triangle.kind()).toEqual('scalene');
  });

  xit('test triangles with no size are illegal', function () {
    var triangle = new Triangle(0, 0, 0);
    expect(triangle.kind.bind(triangle)).toThrow();
  });
  xit('triangles with negative sides are illegal', function () {
    var triangle = new Triangle(3, 4, -5);
    expect(triangle.kind.bind(triangle)).toThrow();
  });

  xit('triangles violating triangle inequality are illegal', function () {
    var triangle = new Triangle(1, 1, 3);
    expect(triangle.kind.bind(triangle)).toThrow();
  });

  xit('triangles violating triangle inequality are illegal 2', function () {
    var triangle = new Triangle(7, 3, 2);
    expect(triangle.kind.bind(triangle)).toThrow();
  });

  xit('triangles violating triangle inequality are illegal 3', function () {
    var triangle = new Triangle(10, 1, 3);
    expect(triangle.kind.bind(triangle)).toThrow();
  });
});
```