KATA: Valid RGB(A)

Given an RGB(A) CSS color, determine whether its format is valid or not. Create a function that takes a string (e.g. “rgb(0, 0, 0)”, or “rgba(50,255,0,0.5)”) and return true if it's format is correct, otherwise return false. Look at the tests for specifics on functionality.

**Tests**:

Console.WriteLine(“rgb lowest valid numbers”);

Assert.Equals(true, ColourUtil.ValidRgbColour(“rgb(0,0,0)”));

Console.WriteLine(“rgb highest valid numbers”);

Assert.Equals(true, ColourUtil.ValidRgbColour(“rgb(255,255,255”));

Console.WriteLine(“rgba lowest valid numbers”);

Assert.Equals(true, ColourUtil.ValidRgbColour(“rgba(0,0,0,0)”));

Console.WriteLine(“rgba highest valid numbers”);

Assert.Equals(true, ColourUtil.ValidRgbColour(“rgba(255,255,255,1)”));

Console.WriteLine(“alpha can have many decimals”);

Assert.Equals(true, ColourUtil.ValidRgbColour(“rgba(0,0,0,0.123456789)”));

Console.WriteLine(“in alpha the number before the dot is optional”);

Assert.Equals(true, ColourUtil.ValidRgbColour(“rgba(0,0,0,.8)”));

Console.WriteLine(“whitespace is allowed around numbers (even tabs)”);

Assert.Equals(true, ColourUtil.ValidRgbColour(“rgba( 0 , 127 , 255 , 0.1 )”));

Console.WriteLine(“numbers can be percentages”);

Assert.Equals(true, ColourUtil.ValidRgbColour(“rgb(0%,50%,100%)”));

Console.WriteLine(“INVALID: missing number”);

Assert.Equals(false, ColourUtil.ValidRgbColour(“rgb(0,,0)”));

Console.WriteLine(“INVALID: whitespace before parenthesis”);

Assert.Equals(false, ColourUtil.ValidRgbColour(“rgb (0,0,0)”));

Console.WriteLine(“INVALID: rgb with 4 numbers”);

Assert.Equals(false, ColourUtil.ValidRgbColour(“rgb(0,0,0,0)”));

Console.WriteLine(“INVALID: rgba with 3 numbers”);

Assert.Equals(false, ColourUtil.ValidRgbColour(“rgba(0,0,0)”));

Console.WriteLine(“INVALID: numbers below 0”);

Assert.Equals(false, ColourUtil.ValidRgbColour(“rgb(-1,0,0)”));

Console.WriteLine(“INVALID: numbers above 255”);

Assert.Equals(false, ColourUtil.ValidRgbColour(“rgb(255,256,255)”));

Console.WriteLine(“INVALID: numbers above 100%”);

Assert.Equals(false, ColourUtil.ValidRgbColour(“rgb(100%,100%,101%)”));

Console.WriteLine(“INVALID: alpha below 0”);

Assert.Equals(false, ColourUtil.ValidRgbColour(“rgba(0,0,0,-1)”));

Console.WriteLine(“INVALID: alpha above 1”);

Assert.Equals(false, ColourUtil.ValidRgbColour(“rgba(0,0,0,1.1)”));