Covéa Technical Assessment

Create a function that can calculate a life insurance premium based on the applicant's age and the sum to be assured.

Using a test driven development (TDD) approach, create a C# or JavaScript function that accepts an age and a sum assured as positive integers.

The supported age range is 18 years to 65 years inclusive. The supported sum assured range is £25,000 to £500,000.

For each age and sum assured, there is a corresponding risk rate band as detailed below:

	Age		
Sum Assured	**<=30**	**31 – 50**	**>50**
£25,000	0.0172	0.1043	0.2677
£50,000	0.0165	0.0999	0.2565
£100,000	0.0154	0.0932	0.2393
£200,000	0.0147	0.0887	0.2285
£300,000	0.0144	0.0872	Not available
£500,000	0.0146	Not available	Not available

The maximum sum assured reduces as the age of the applicant increases (e.g. the maximum sum assured for 31-50 year olds is £300,000).

Your function should find the correct risk rate based on the applicant's age and sum assured and, where supported, calculate a monthly premium based on the following formulae:

```
Risk premium = Risk rate * (Sum assured / 1000)

Renewal commission = 3% * Risk premium

Net premium = Risk premium + Renewal commission

Initial commission = Net premium * 205%

Gross premium = Net premium + Initial commission
```

When the requested sum assured is between bands you should use the following formula to calculate the risk rate:

```
Risk rate = ((Sum assured - Lower band sum assured)/(Upper band sum
assured - Lower band sum assured) * Upper band risk rate + (Upper
band sum assured - Sum assured)/(Upper band sum assured - Lower band
sum assured) * Lower band risk rate)
```

There is a minimum gross premium of £2. Where the premium is lower than this, you should increase the sum assured by increments of £5,000 until the minimum gross premium is exceeded.

Your function should return the sum assured (adjusted where applicable), the age of the applicant and the gross premium payable. Your solution should allow for the rates to be changed without additional calculations or other changes to the code (changing an in-code variable for the rates is acceptable provided no additional processing of the rate is required).

Example Inputs

Sum Assured	Age
£25,000	18
£50,000	30
£60,000	49

General Guidance

The exercise should take around 2 hours to complete.

If you don't have the time to fully complete the exercise, don't worry! We are looking at how you approach a problem and structure your code. Please send us what you are able to do with details of what you would plan to do next.

There is no prescribed solution to the problem – it can be a console app, an API or something else but you must provide instructions as to how we can view your code and test your work along with a sample input (if applicable).

It should be possible to send a zipped solution to us but be aware that our email virus scanning may block these files and we may need you to provide access in a different way (e.g. Bitbucket).

You are free to use whichever resources that you need in order to create your solution (e.g. Google, etc.).

You can create your solution in whichever IDE you wish. It is possible to download a free community edition of Visual Studio or Visual Studio Code which can be used to create your solution. If you do not have a computer that supports this, you can use https://dotnetfiddle.net/. In dotnetfiddle, you can create and run basic tests as follows:

```
using System;
using System.Collections.Generic;
using System.Linq;

public class TestRunner
{
    public static void Main()
    {
        var tests = new MyTests();

        var failedTests = new List<string>();

        // using reflection, run every test method and record the names of those methods that fail
        foreach (var m in typeof(MyTests).GetMethods().Where(m => m.DeclaringType != typeof(object)))
        {
            if (Convert.ToBoolean(m.Invoke(tests, null)) != true)
                failedTests.Add(m.Name);
        }

        // display all the failed tests, or a message that everything passed
        if (failedTests.Any())
            Console.WriteLine("Failed Tests: \r\n\r\n{0}", string.Join("\r\n", failedTests));
        else
            Console.WriteLine("All tests passed!");
    }
}

public class MyTests
{
    public bool True_Is_True()
    {
        return true == true;
    }
}
```