For tests 1 – 5, follow the instructions provided in the Submission.cs file

Test 1 – Write an overloaded constructor

Write an overloaded constructor that accepts a string (only). Use the string to initialize the member field, `retort`. You do not need to update the value of any other member field(s).

Test 2 – Write a 'Getter' method

Write a getter for the member field, `retort`. The getter must be named `GetRetort`. The getter will return the data type that matches the member field being returned and accept no parameters.

Test 3 – Write a 'Setter' method

Write a setter for the member field, `retort`. The setter must be named `SetRetort`. The setter will not return anything and will accept a single parameter that matches the member field being updated.

Test 4 – Write a C# property

Write a property, `Hidden`. The property will be a `public int` and will access/update the member field `hidden`

Test 5 – Write a 'regular' method

Write a method named `YellAtMe` that accepts no parameters and returns the value of `retort` as all capital letters - Use the `ToUpper` method

Test 6 – Use a C# operator

```
public static int Test6(int input)
```

Given an int, input, return an int that is 4 times the input value.

Example input

```
37
```

Example output

```
148
```

Test 7 – Call a static method and cast the result

```
public static float Test7(float input)
```

Given a float, input, find the square root of the input using the Sqrt method found in the Math class. Return the result as a float.

Example input

```
98.81
```

Example output

```
9.9
```

Test 8 – Create a Random object using a seed, generate a ranged random number

```
public static int Test8(int min, int max, int seed)
```

Given three int values, min, max and seed, create a Random object passing the seed provided to the constructor for Random. Using the appropriate Next method, generate a random number between min (inclusive) and max (exclusive). Return the generated random number.

Example input

```
3, 19, 66
```

Example output

```
6
```

Test 9 – Use integer division to find a quotient

```
public static int Test9(int number1, int number2)
```

Given two int values, number1 and number2, find the integer quotient when dividing number1 by number2.

Example input

```
19, 3
```

Example output

```
6
```

Test 10 – Use integer division to find a remainder

```
public static int Test10(int number1, int number2)
```

Given two int values, number1 and number2, find the integer remainder when dividing number1 by number2.

Example input

```
47, 13
```

Example output

```
8
```

Notes:

- Tests 1-5 should NOT be defined as static.

- For Test8, the Random class provides 2 different constructors. The default constructor (the one that accepts no parameters) uses the system time/clock to generate a seed value. The overloaded constructor accepts an int to use as a seed value. A Random object (an instance of the Random class) has 3 methods that generate/return a random int. The methods are 'overloaded' (same name but accept different parameters). The first accepts no parameters and returns a 'non-negative' int value (between 0 and Int32.MaxValue - 1). The second accepts a single int parameter and returns an int between 0 and the parameter's value minus 1. The third accepts two int parameters and returns an int between the first parameter (a minimum value) and the second parameter's value minus 1.