This is Lab 4 - it is intended for you to demonstrate your understanding of selection statements (if and switch). This means you are required to use either a switch or if statement for all Test solutions (Test6 is an exception to this rule). For Test6, you are permitted to simply 'pass through' the value returned by the Equals method.
Additional reminders. In PG1 you are not allowed to use var in place of a known data type.

Note: This program will not compile/build (so it won't run) until you successfully complete the MathOperator enum for Tests 1 & 2.

Before beginning work on Tests 1 and 2, you are to create/define an enum named MathOperator with five values: Add, Subtract, Multiply, Divide and Modulo. ***The enum should not be 'nested' within a class (it should be defined in the namespace, but outside all classes).***

Test 1 – Return an enum

```
public static MathOperator Test1(int input)
```

After creating the enum, you are to return a MathOperator value based on the value of the parameter, `input`, provided:

Parameter		Return
0	=	MathOperator.Add
1	=	MathOperator.Subtract
2	=	MathOperator.Multiply
3	=	MathOperator.Divide
4	=	MathOperator.Modulo

You do not need to worry about/handle any other possible input values (the only values passed to the method will be 0, 1, 2, 3 or 4).

Example input

```
3
```

Example output

```
MathOperator.Divide
```

Test 2 – Use an enum

```
public static int Test2(int number1, int number2, MathOperator operation)
```

Given two int values, `number1` and `number2`, and a MathOperator, `operation`, perform the indicated operation on the int values provided. Return the result of the operation.

Example input

```
89, 16, MathOperator.Modulo
```

Example output

```
9
```

Test 3 – Act on a bool value

```
public static string Test3(bool input)
```

Given a bool, `input`, return a string indicating the related term. When the bool is `true` return "Proven", when the bool is `false` return "Denied". Remember all identifiers are case sensitive. For this Test, the returned string's case does not matter.

Example input

```
89.3
```

Example output

```
B
```

Test 4 – Perform unit conversion based on a condition

```
public static double Test4(double input, bool kiloToLb)
```

Given a weight in either kilograms or pounds, `input`, and a bool, `kiloToLb`, you will either convert kilograms to pounds (when the bool is `true`) or pounds to kilograms (when the bool is `false`). Return the result rounded to 3 decimal places (use `Math.Round`).
The conversions are:

1 kg = 2.20462 lb
1 lb = 0.453592 kg

Example input

```
10, true
```

Example output

```
22.046
```

Test 5 – Evaluate a situation

```
public static string Test5(int input)
```

Given an int, `input`, that represents the severity of an infraction and the following decision table, return the string representing the action to be taken.

Infraction Severity	Required Action
Level 1	Verbal Reprimand
Level 2	Formal Reprimand
Level 3	Suspension
Level 4	Termination

Keep in mind, action required must be spelled correctly (exactly as above) but is not case sensitive.

Example input

```
2
```

Example output

```
"Formal Reprimand"
```

Test 6 – Compare two objects for equality (of contents)

`public static bool Test6(Triangle t1, Triangle t2)`

Given two Triangle objects, `t1` and `t2`, use their Equals method to determine if the two objects are equal. Keep in mind, the Equals method is not the same as the equality operator (==). The Equals method evaluates the value of the contents of each object while the equality operator (==) will evaluate if both references are the same (the same object/reference).

Example input

```
t1, t2
```

Example output

```
false
```

Test 7 – Using values to analyze data

`public static bool Test7(int a, int b, int c)`

Given the three sides of a Triangle and int values, `a`, `b` and `c`, determine if the triangle is a Right Triangle. A triangle where the sum of the squares of the two shortest sides (`a` and `b`) is equal to the square of longest side (c) is a 'Right' triangle ($a^2 + b^2 = c^2$). The third parameter (`c`) will always be the longest side given.

Example input

```
4, 3, 5
```

Example output

```
true
```

Test 8 – Letter grade

`public static char Test8(double grade)`

Given the `grade` variable, which indicates a student's numeric grade, determine which letter grade they should receive. Return the proper letter (char) as the result of this test. Do not worry about rounding the grade. Use the following table to indicate which letter corresponds to provided grades.

>= 90 and <= 100	'A'
>= 80 and < 90	'B'
>= 73 and < 80	'C'
>= 70 and < 73	'D'
>= 0 and < 70	'F'
< 0 or > 100	'?'

Example input

```
89.3
```

Example output

```
B
```

Notes:

- Test6: When comparing reference types for equality, you cannot use the equality operator (==). While the code will build/run, the equality operator compares the references (the memory addresses) of the two objects (reference types); it does not compare the 'contents' for equivalence. To check reference types for 'equivalence' you need to use the Equals method for that object/reference type. To use the Equals method for an object, access it using the 'dot' operator and pass the second object's reference (variable name)
  `objectReferenceName1.Equals(objectReferenceName2);`
  The Equals method returns a `bool` – `true` if the object contents are deemed equivalent or `false` if the contents are not deemed equivalent.