You cannot use multiple return statements in any method submitted for this project. All methods must demonstrate the correct use of branching/selection statements (they all have to have either an if or a switch statement) with the exception of Test6.

Test 1 – Numeric multiple

```
public static bool Test1(int num1, int num2)
```

Given two ints, num1 and num2, determine if num1 is a multiple of num2. If it is a multiple return true, otherwise, return false.

Example input

```
57, 19
```

Example output

```
True
```

Test 2 – Same temperature

```
public static bool Test2(double fahrenheit, double celsius)
```

Given two doubles, fahrenheit and celsius, determine if they are equal temperatures. Before comparing the temperatures, convert them to the same units. The formula to convert Fahrenheit to Celsius is:
Celsius = (fahrenheit - 32.0) * (5.0/9.0)
The formula to convert Celsius to Fahrenheit to Celsius is:
Fahrenheit = (celsius * 9.0 / 5.0) + 32.0
You only need to use one of the formulas – not both.

Example input

```
122.0, 50.0
```

Example output

```
true
```

Test 3 – Letter grade

```
public static char Test3(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
```

Test 4 – Use an enum

```
public static double Test4(double number1, double number2, MathOperator op)
```

Given two doubles (`num1` and `num2`) and a `MathOperator`, an `enum` value that indicates a math operation, perform the appropriate math operation on the two numbers and return the result. The `MathOperator` enum has already been defined (so you do not need to redefine it) as follows:

```
public enum MathOperator
{
   Add,
   Subtract,
   Multiply,
   Divide
}
```

Example input

```
47.7, 0.9, MathOperator.Divide
```

Example output

```
53
```

Test 5 – Evaluate situation

```
public static int Test5(int speed)
```

Given an int, speed, and the following decision table, determine what action should be taken. Return the number (an int) that represents the action from the table

Over limit by	Action	
<= 5 MPH	Verbal warning	0
>5 MPH and <= 10 MPH	Written warning	1
>10 MPH and <= 25 MPH	Citation	2
> 25 MPH	Take into custody	3

Example input

```
7
```

Example output

```
1
```

Test 6 – Compare two objects for equality

```
public static bool Test6(Point p1, Point p2)
```

Given two Point objects, determine if the Point objects are equal. The Point class provides an 'Equals' method that can be used to determine if a Point is equal to a second point. The Equals method signature is:
bool Equals(object obj)

Example input

```
p1, p2
```

Example output

```
true;
```

Test 7 – Use operators to make decisions

```
public static bool Test7(double price, double taxRate, double cashOnHand)
```

Given three doubles that represent the cost of an item, the applicable tax rate for the item and cash on hand, determine if cash on hands will cover the cost of the item. To help eliminate floating-point errors, you should round all computations to 2 decimal places using the Round method found in the Math class. Return true if the item can be purchased, otherwise return false.

Example input

```
3.29, 0.07, 3.50
```

Example output

```
false
```

Test 8 – Find the greater distance

```
public static double Test8(double miles, double kilometers)
```

Given a distance in miles and a distance in kilometers, determine which is the greater distance. Before comparing the distances, convert them to the same units. Return the original value (miles or kilometers) of the greater/longer distance. Use the conversion factor:
1km = 0.621 miles (1 mile = 1.609km).
Perform only one of the conversions (either km to miles or miles to km), not both. Do not overwrite the original values.

Example input

```
55.0, 90.0
```

Example output

```
90.0
```