

# Documentación desafíos C# edabit.com

## Nivel: muy fácil

### Instrucciones

**Devuelve la suma de dos números**

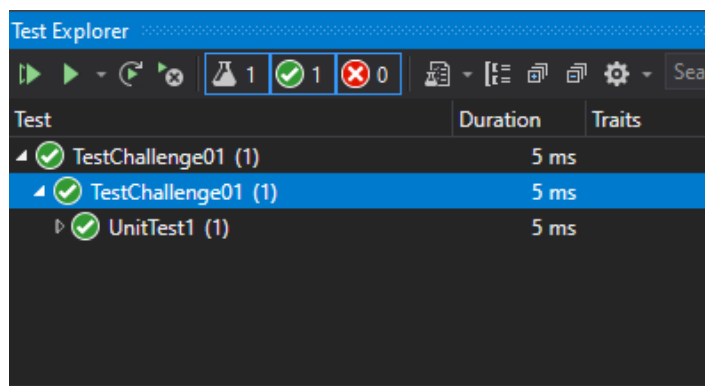
**Cree una función que tome dos números como argumentos y devuelva su suma.**

### Código de la solución

```
public static int SumaDeDosNumeros(int valorUno, int valorDos)
{
    int retorno = valorUno+ valorDos;
    return retorno;
}
```

### Casos de prueba

```
namespace TestChallenge01
{
    [TestClass]
    public class UnitTest1
    {
        [TestMethod]
        public void TestSumaDeDosNumeros()
        {
            Assert.AreEqual(2, Challenge01.Program.SumaDeDosNumeros(1,1));
        }
    }
}
```



Estimación del desafío : 5 min

Duración real: 15 min

Instrucciones

**Convertir minutos a segundos**

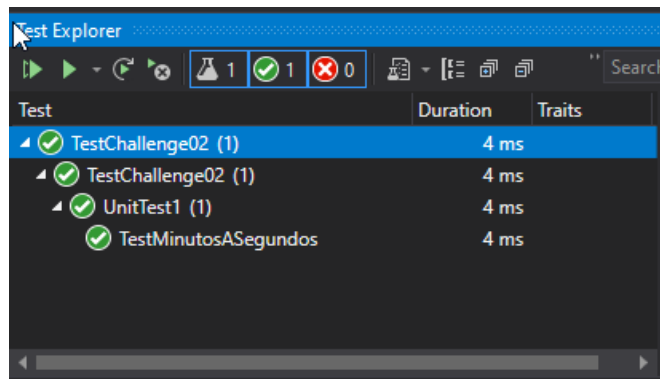
**Escribir una función que tome un número entero “minutos” y lo convierta en segundos.**

Código de la solución

```
public static int MinutosASegundos(int minutes)
{
    return ( minutes * 60);
}
```

Casos de prueba

```
[TestMethod]
public void TestMinutosASegundos()
{
    Assert.AreEqual(300, Challenge02.Program.MinutosASegundos(5));
}
```



Estimación del desafío : 5 min

Duración real: 5 min

### Instrucciones

**Devolver el próximo número del entero pasado.**

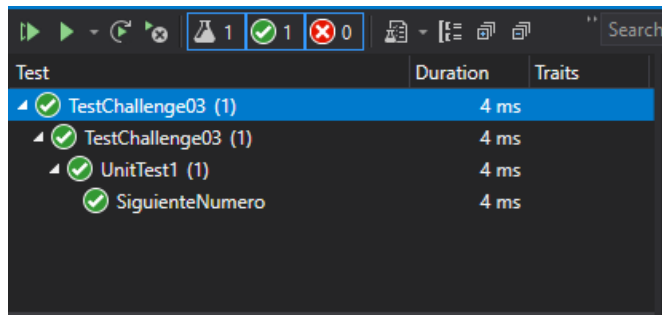
**Cree una función que tome un número como argumento, incremente el número en +1 y devuelva el resultado.**

### Código de la solución

```
public static int SiguieteNumero(int valor)
{
    return valor + 1;
}
```

### Casos de prueba

```
[TestMethod]
public void SiguieteNumero()
{
    Assert.AreEqual(1, Challenge03.Program.SiguieteNumero(0));
}
```



Test	Duration	Traits
TestChallenge03 (1)	4 ms	
TestChallenge03 (1)	4 ms	
UnitTest1 (1)	4 ms	
SiguieteNumero	4 ms	

Estimación del desafío : 5 min

Duración real: 5 min

Instrucciones

### Calculadora de potencia

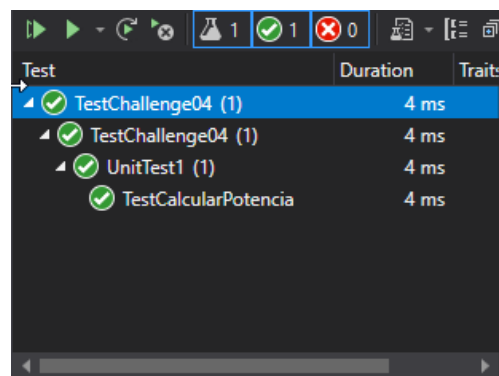
Cree una función que tome “voltage” y “current” y devuelva la portencia calculada.

Código de la solución

```
public static int CalcularPotencia(int potencia, int current )  
{  
    return potencia * current;  
}
```

Casos de prueba

```
[TestMethod]  
public void TestCalcularPotencia()  
{  
    Assert.AreEqual(2300, Challenge04.Program.CalcularPotencia(230, 10));  
}
```



Test	Duration	Trait
✓ TestChallenge04 (1)	4 ms	
✓ TestChallenge04 (1)	4 ms	
✓ UnitTest1 (1)	4 ms	
✓ TestCalcularPotencia	4 ms	

Estimación del desafío : 5 min

Duración real: 5 min

Instrucciones

### Convertir edad a días

Cree una función que tome la edad en años y devuelva la edad en días.

Código de la solución

```
public static int ConvertirAniosADias(int anios)
{
    int retorno = 0;

    if (anios >= 0)
    {
        retorno = anios * 365;
    }
    return retorno;
}
```

Casos de prueba

```
[TestMethod]
public void TestConvertirAniosADias()
{
    Assert.AreEqual(23725, Challenge05.Program.ConvertirAniosADias(65));
}
```

Estimación del desafío : 5 min

Duración real: 5 min

Instrucciones

### Área de un triángulo

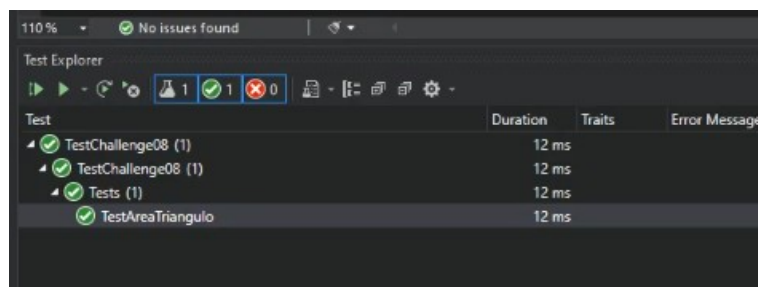
Escribe una función que tome la base y la altura de un triángulo y retornar su área.

Código de la solución

```
public static double AreaTriangulo(int _base, int _altura)
{
    return (_base * _altura)/2;
}
```

Casos de prueba

```
[Test]
public void TestAreaTriangulo()
{
    Assert.AreEqual(3, Challenge08.Program.AreaTriangulo(3, 2));
}
```



The screenshot shows the Test Explorer window with a toolbar at the top indicating 1 passed test and 0 failed tests. The test results table below shows that all tests passed successfully.

Test	Duration	Traits	Error Message
TestChallenge08 (1)	12 ms		
TestChallenge08 (1)	12 ms		
Tests (1)	12 ms		
TestAreaTriangulo	12 ms		

Estimación del desafío : 5 min

Duración real: 5 min

## Instrucciones

### Devuelve el resto de dos números

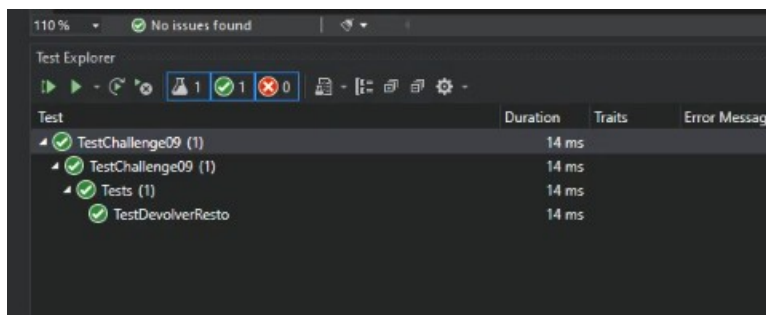
Hay un solo operador en c#, capaz de proporcionar el resto de una operación de división. Se pasan dos números como parámetros. El primer parámetro dividido por el segundo parámetro tendrá un resto, posiblemente cero. Devolver ese valor.

### Código de la solución

```
public static int DevolverResto(int valor1, int valor2)
{
    return ( valor1 % valor2);
}
```

### Casos de prueba

```
[Test]
public void TestDevolverResto()
{
    Assert.AreEqual(1, Challenge09.Program.DevolverResto(1, 3));
}
```



Estimación del desafío : 5 min

Duración real: 5 min

# Nivel Easy

## Instrucciones

### Convertir número al nombre del mes correspondiente

Cree una función que tome un número (del 1 al 12) y devuelva su nombre de mes correspondiente como una cadena. Por ejemplo, si se le proporciona 3 como entrada, su función debería devolver “marzo”, porque marzo es el tercer mes.

## Código de la solución

```
public static string ConvertirNumeroAMes(int mes){  
    List<string> meses = new List<string>() {  
        "Enero", "Febrero", "Marzo", "Abril", "Mayo",  
        "Junio", "Julio", "Agosto", "Septiembre",  
        "Octubre", "Noviembre", "Diciembre"  
    };  
  
    string retorno="";  
    if(mes >0 && mes <13)  
    {  
        retorno = meses[mes - 1];  
    }  
    return retorno;  
}
```

## Casos de prueba

```
[TestMethod]  
public void TestConvertirNumeroAMes()  
{  
    Assert.AreEqual("Marzo", Challenge06.Program.ConvertirNumeroAMes(3));  
}
```

Estimación del desafío : 10 min

Duración real: 10 min



Instrucciones

**Encuentra los números más pequeños y más grandes**

**Cree una función que tome una matriz de números y devuelva los números mínimo y máximo, en ese orden.**

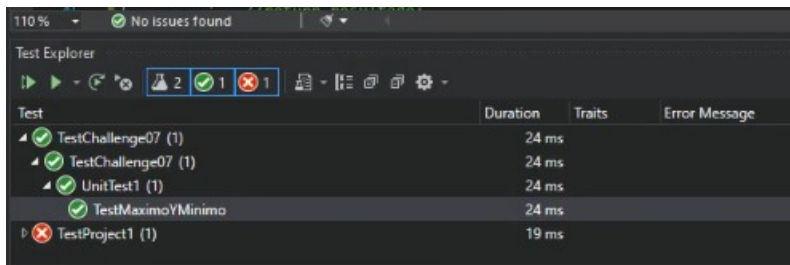
Código de la solución

```
public static int[] MaximoYMinimo(int[] matriz)
{
    return ( new int []{matriz.Min(), matriz.Max()});
}
```

Casos de prueba

```
[TestMethod]
public void TestMaximoYMinimo()
{
    int[] esperado = new int []{1,5};
    int[] enviado = new int []{ 1, 5 };

    CollectionAssert.AreEqual(esperado, Challenge07.Program.MaximoYMinimo(enviado));
}
```



Test	Duration	Traits	Error Message
TestChallenge07 (1)	24 ms		
TestChallenge07 (1)	24 ms		
UnitTest1 (1)	24 ms		
TestMaximoYMinimo	24 ms		
TestProject1 (1)	19 ms		

Estimación del desafío : 15 min

Duración real: 90 min

## Instrucciones

Es el número menor o igual a cero?

**Crear una función que tome un número como único argumento y retorne true si es menor o igual a cero y sino, false.**

## Código de la solución

```
✓ | 0 references  
public void TestMenorOIgualAcero()  
{  
    Assert.AreEqual(false, challenge10.Program.MenorOIgualAcero(5));  
}
```

## Casos de prueba

```
1 reference | ✓ 1/1 passing  
public static bool MenorOIgualAcero(int valor)  
{  
    ...  
    return (valor <= 0) ;  
}
```

Test Explorer			
Test			
		Duration	Traits
TestChallenge10 (1)		34 ms	
TestChallenge10 (1)		34 ms	
Tests (1)		34 ms	

Estimación del desafío : 5 min

Duración real: 5 min

## Instrucciones

### Suma absoluta

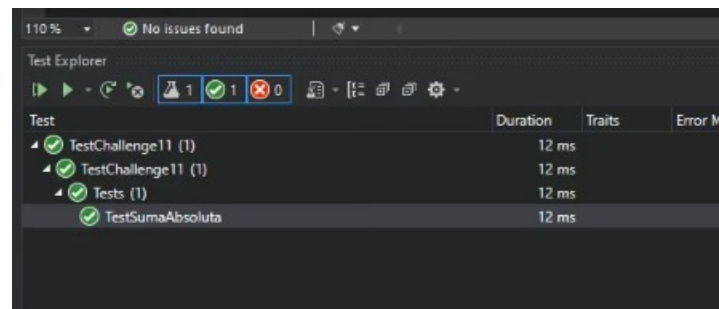
**Tomar un array de enteros (positivos o negativos o ambos) y devuelva la suma de los valores absolutos de cada elemento.**

## Código de la solución

```
1 reference | 1/1 passing
public static int SumaAbsoluta(int[] enviado)
{
    int resultado = 0;
    foreach (var c in enviado)
    {
        resultado = (c < 0) ? (resultado += (c * -1)) : (resultado += c);
    }
    return resultado;
}
```

## Casos de prueba

```
[Test]
0 references
public void TestSumaAbsoluta()
{
    int[] enviado = new int[] { 2, -1, 4, 8, 10 };
    Assert.AreEqual(25, Challenge11.Program.SumaAbsoluta(enviado));
}
```



The screenshot shows the Test Explorer window with a toolbar at the top indicating 1 passed test (green checkmark), 1 failed test (red X), and 0 skipped tests (red circle with slash). The test list below shows a hierarchy of tests, all of which passed successfully.

Test	Duration	Traits	Error Message
TestChallenge11 (1)	12 ms		
TestChallenge11 (1)	12 ms		
Tests (1)	12 ms		
TestSumaAbsoluta	12 ms		

Estimación del desafío : 15 min

Duración real: 10 min

Instrucciones

**A la potencia de \_\_\_\_**

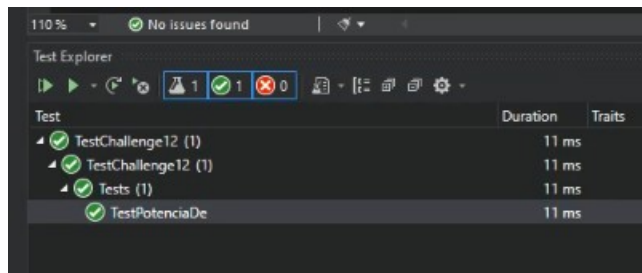
**Cree una función que tome un número base y un número de exponente y devuelva el cálculo.**

Código de la solución

```
1 reference | 1/1 passing
public static double PotenciaDe(double valor)
{
    return Math.Pow(valor, valor);
}
```

Casos de prueba

```
[Test]
0 references
public void TestPotenciaDe()
{
    Assert.AreEqual(3125, Challenge12.Program.PotenciaDe(5));
}
```



Test	Duration	Traits
TestChallenge12 (1)	11 ms	
TestChallenge12 (1)	11 ms	
Tests (1)	11 ms	
TestPotenciaDe	11 ms	

Estimación del desafío : 10 min

Duración real: 5 min

Instrucciones

## Multiplicar por longitud

Crear una función para multiplicar todos los valores de un arreglo por la cantidad de valores en el array.

Código de la solución

```
1 reference | 1/1 passing
public static int[] MultiplicarPorLongitudDeArray(int[] enviado)
{
    int[] retorno = new int[enviado.Length];

    for(int i=0; i< enviado.Length; i++)
    {
        retorno[i] = enviado[i] * enviado.Length;
    }
    return retorno;
}
```

Casos de prueba

```
[Test]
0 references
public void TestMultiplicarPorLongitudDeArray()
{
    int[] enviado = new int[] { 2, 3, 1, 0 };
    int[] esperado = new int[] { 8, 12, 4, 0 };

    Assert.AreEqual(esperado, Challenge13.Program.MultiplicarPorLongitudDeArray(enviado));
}
```

Estimación del desafío : 10 min

Duración real: 10 min

## Instrucciones

### Distancia de hamming

La distancia de hamming es el número de caracteres que difieren entre dos cadenas.

### Código de la solución

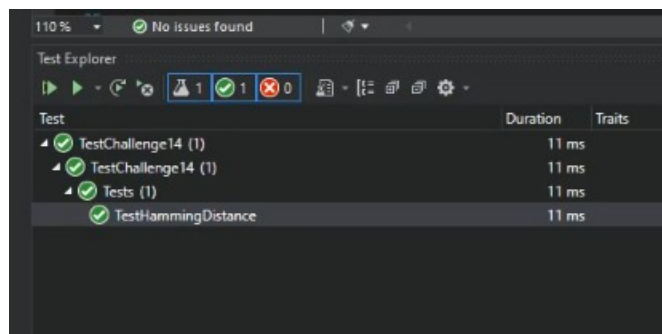
```
1 reference | 1/1 passing
public static int HammingDistance(string cadena1, string cadena2)
{
    int contador = 0;

    if (cadena1.Length == cadena2.Length)
    {
        for (int i = 0; i < cadena1.Length; i++)
        {
            contador += (!cadena1[i].Equals(cadena2[i])) ? 1 : 0;
        }
    }

    return contador;
}
```

### Casos de prueba

```
[Test]
public void TestHammingDistance()
{
    Assert.AreEqual(5, Challenge14.Program.HammingDistance("abcde", "bcdef"));
}
```



Test	Duration	Traits
TestChallenge14 (1)	11 ms	
TestChallenge14 (1)	11 ms	
Tests (1)	11 ms	
TestHammingDistance	11 ms	

Estimación del desafío : 15 min

Duración real: 15 min

## Instrucciones

### Intercambiar el nombre

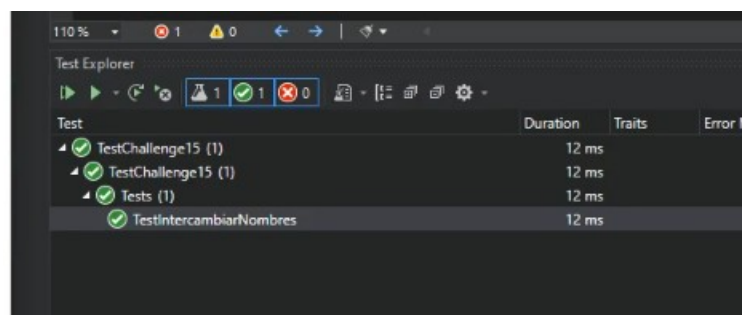
Cree una función que acepte una cadena (del nombre y apellido de una persona) y devuelva una cadena con el nombre y el apellido intercambiados.

## Código de la solución

```
1 reference | 1/1 passing
public static string IntercambiarNombres(string cadena)
{
    string[] subCadena = cadena.Split(' ');
    return String.Concat(subCadena[1], " ", subCadena[0]);
}
```

## Casos de prueba

```
[Test]
0 references
public void TestIntercambiarNombres()
{
    Assert.AreEqual("Trump Donald", challenge15.Program.IntercambiarNombres("Donald Trump"));
}
```



Estimación del desafío : 15 min

Duración real: 15 min

Instrucciones

**Número de cadena más pequeño**

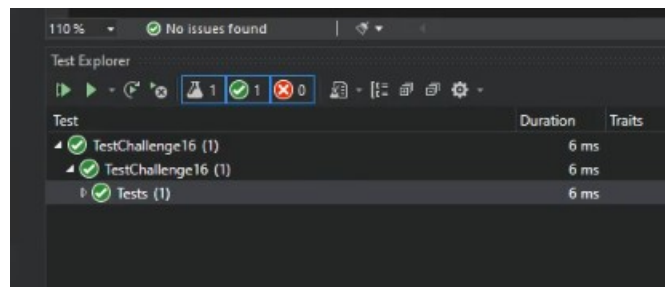
**Crea una función que devuelva el número más pequeño.**

Código de la solución

```
1 reference | 1/1 passing
public static string StringNumeroMenor(string cadena1, string cadena2)
{
    ...
    return ((string.CompareOrdinal(cadena1, cadena2) < 0) ? cadena1 : cadena2);
}
```

Casos de prueba

```
[Test]
0 references
public void TestStringNumeroMenor()
{
    Assert.AreEqual("21", Challenge16.Program.StringNumeroMenor("21", "44"));
}
```



The screenshot shows the Test Explorer window in Visual Studio. At the top, it says "110 %", "No issues found", and "1/1 passing". Below this, there are icons for test results: a green checkmark, a red X, and a green checkmark. The table below shows the test results:

Test	Duration	Traits
TestChallenge16 (1)	6 ms	
TestChallenge16 (1)	6 ms	
Tests (1)	6 ms	

Estimación del desafío : 15 min

Duración real: 10 min



## Instrucciones

### Devolver el factorial

Crear una función que tome un entero y retorne el factorial de ese entero. Esto es, el entero multiplicado por todos los números positivos menores.

## Código de la solución

```
2 references | 1/1 passing
public static int ObtenerFactorial(int v)
{
    int resultado = 1;

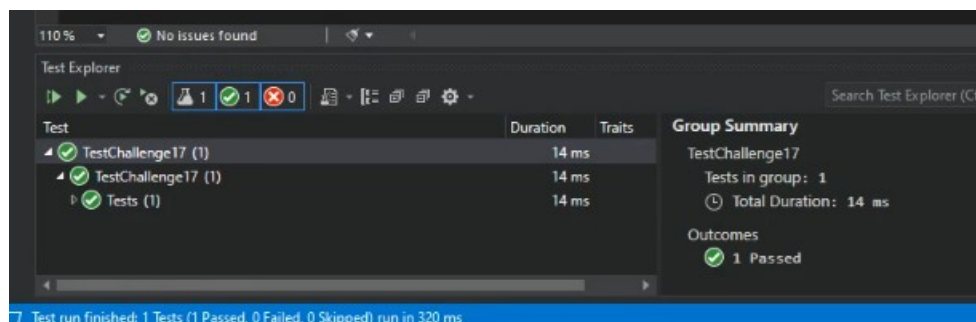
    if (v == 0) {
        return 0;
    }

    for (int i = 0; i < v; i++) {
        if (i == 0)
        {
            resultado += i * (v - i);
        }
        else
        {
            resultado += resultado * (v - i);
        }
    }

    return resultado;
}
```

## Casos de prueba

```
[Test]
0 references
public void TestObtenerFactorial()
{
    Assert.AreEqual(479001600, Challenge17.Program.ObtenerFactorial(12));
}
```



Estimación del desafío : 30 min

Duración real: 2 hs

## Instrucciones

### ¿Cuántas vocales hay?

Crear una función que tome una cadena y devuelva el número ( cantidad) de vocales contenidas en la cadena.

Código de la solución

```
1 reference | 1/1 passing
public static int FindVowels(string v)
{
    // return v.Count(c => "aeiouAEIOU".Contains(c));

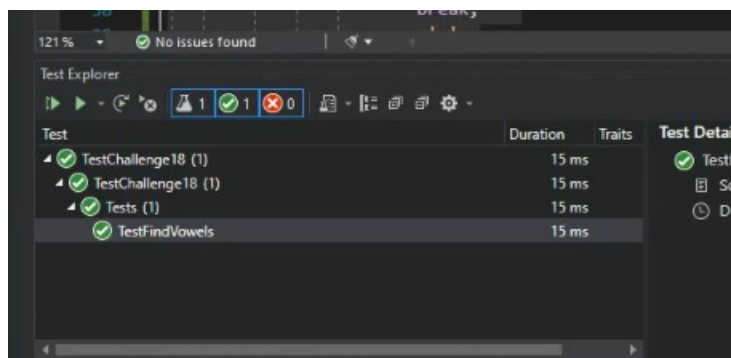
    int cantidadDeVocales = 0;

    foreach (var item in v.ToLower())
    {
        switch (item)
        {
            case 'a':
                cantidadDeVocales++;
                break;
            case 'e':
                cantidadDeVocales++;
                break;
            case 'i':
                cantidadDeVocales++;
                break;
            case 'o':
                cantidadDeVocales++;
                break;
            case 'u':
                cantidadDeVocales++;
                break;
        }
    }

    return cantidadDeVocales;
}
```

## Casos de prueba

```
[Test]
0 references
public void TestFindVowels()
{
    Assert.AreEqual(2, Challenge18.Program.FindVowels("PALma"));
}
```



Estimación del desafío : 20 min

Duración real: 30 min

## Instrucciones

### Ordenar los números en orden ascendente

Cree una función que tome una matriz de números y devuelva una nueva matriz, ordenada en orden ascendente (de menor a mayor).

- ordenar la matriz de números en orden ascendente.
- si el argumento de la función es nulo, una matriz vacía o indefinido; devuelve una matriz vacía.
- devolver una nueva matriz de números ordenados.

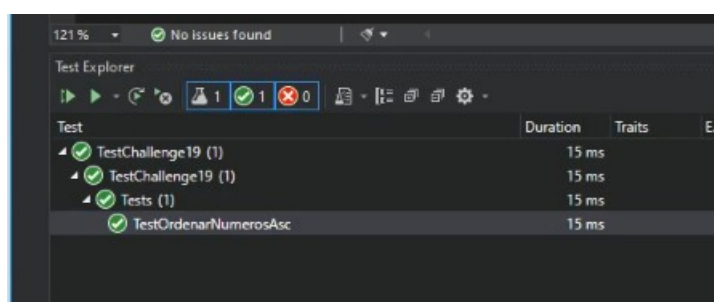
## Código de la solución

```
1 reference | 1/1 passing
public static int[] OrdenarNumerosAsc(int[] enviado)
{
    if (enviado.Length == 0)
    {
        return new int[0];
    }
    //Se dejan las líneas antes del refactor
    //int[] arrayOrdenado = new int[enviado.Length];
    //List<int> enviado2 = new List<int>();
    List<int> enviado2 = enviado.ToList();
    enviado2.Sort();
    //int [] arrayOrdenado = enviado2.ToArray();
    return enviado2.ToArray();
}
```

## Casos de prueba

```
[Test]
0 references
public void TestOrdenarNumerosAsc()
{
    int[] arrayOrdenado = new int[] { 1, 2, 5, 10, 50 };
    int[] enviado = new int[] { 1, 2, 10, 50, 5 };

    Assert.AreEqual(arrayOrdenado, Challenge19.Program.OrdenarNumerosAsc(enviado));
}
```



Test	Duration	Traits	E.
TestChallenge19 (1)	15 ms		
TestChallenge19 (1)	15 ms		
Tests (1)	15 ms		
TestOrdenarNumerosAsc	15 ms		

Estimación del desafío : 15 min

Duración real: 10 min

## Instrucciones

**Verificar si una cadena contiene solo caracteres idénticos.**

**Escribir un algoritmo que retorne true si todos los caracteres de la cadena son idénticos y sino, false.**

## Código de la solución

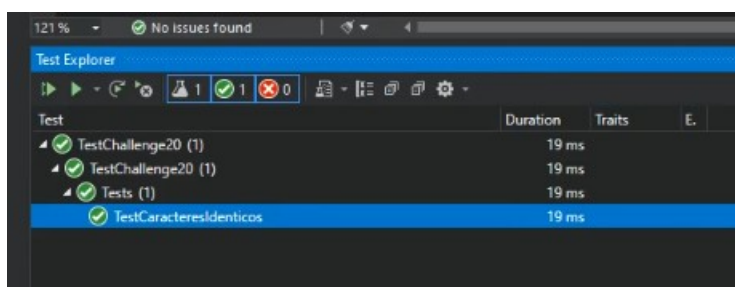
```
1 reference | 1/1 passing
public static bool CaracteresIdenticos(string v)
{
    List<char> aux = new List<char>();
    aux = v.ToList();

    return( aux.Distinct().Count() == 1 ? true : false );

    /* if (v.Length > 1)
    {
        for (int i = 0; i < v.Length - 1; i++)
        {
            if (v[i] != v[i + 1])
            {
                return false;
            }
        }
    }
    return true;*/
}
```

## Casos de prueba

```
[Test]
0 references
public void TestCaracteresIdenticos()
{
    Assert.AreEqual(true, Challenge20.Program.CaracteresIdenticos("aaaaaa"));
}
```



Test	Duration	Traits	E.
TestChallenge20 (1)	19 ms		
TestChallenge20 (1)	19 ms		
Tests (1)	19 ms		
TestCaracteresIdenticos	19 ms		

Estimación del desafío : 15 min

Duración real: 15 min

# Nivel Intermedio

## Instrucciones

### Matriz de Múltiplos

Crear una función que tome dos números como argumentos(num, lenght) y devuelva una matriz de múltiplos de num hasta que la longitud de la matriz alcance lenght.

## Código de la solución

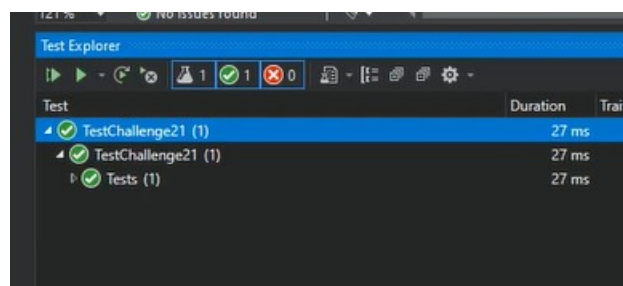
```
1 reference | 1/1 passing
public static int[] MultiplicarArray(int numero, int longitud)
{
    int[] resultado = new int[longitud];

    for (int i = 1; i <= longitud; i++)
    {
        resultado[i - 1] = numero * i;
    }

    return resultado;
}
```

## Casos de prueba

```
[Test]
0 references
public void TestMultiplicarArray()
{
    int[] arrayEsperado = new int[] { 7, 14, 21, 28, 35 };
    Assert.AreEqual(arrayEsperado, Challenge21.Program.MultiplicarArray(7,5));
}
```



Estimación del desafío : 15 min

Duración real: 10 min

## Instrucciones

### Invertir el case

Dada una cadena, crear una función para invertir el case. Todas las letras en minúsculas deben estar en mayúsculas y viceversa.

## Código de la solución

```
1 reference | 1/1 passing
public static string ReverseTheCase(string normalWord)
{
    StringBuilder inverseString = new StringBuilder("");

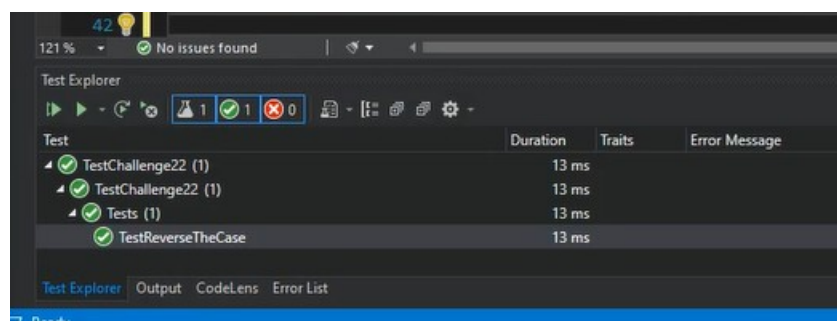
    foreach (char item in normalWord)
    {
        inverseString.Append(char.IsLower(item) ? item.ToString().ToUpper() : item.ToString().ToLower());

        /* if (Char.IsLower(item))
        {
            inverseString.Append(item.ToString().ToUpper());
        }
        else
        {
            inverseString.Append(item.ToString().ToLower());
        }
        */
    }

    return inverseString.ToString();
}
```

## Casos de prueba

```
[Test]
0 references
public void TestReverseTheCase()
{
    Assert.AreEqual("hAPPY bIRThDAY", Challenge22.Program.ReverseTheCase("Happy Birthday"));
}
```



Estimación del desafío : 30 min

Duración real: 30 min

## Instrucciones

### Encontrar los números mas grandes en un grupo de matrices

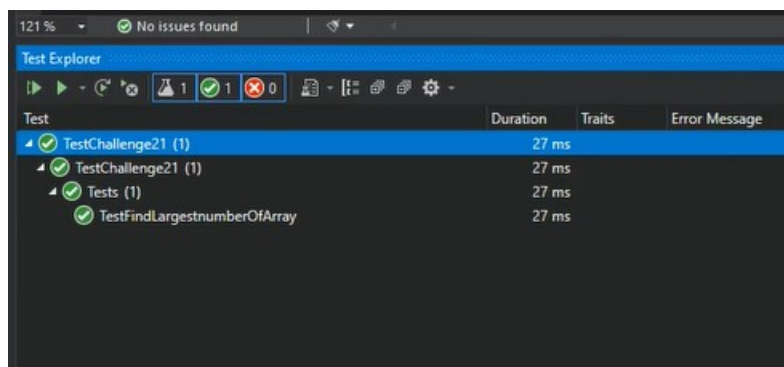
Cree una función que tome una matriz de matrices con números. Devuelve una nueva matriz (única) con los números mas grandes de cada uno.

## Código de la solución

```
1 reference | 1/1 passing
public static double[] FindLargestnumberOfArray(double[][] enviado)
{
    double[] resultado = new double[enviado.Length];
    for (int i = 0; i < enviado.Length; i++)
    {
        resultado[i] = enviado[i].Max();
    }

    return resultado;
}
```

## Casos de prueba



The screenshot shows the Test Explorer window in Visual Studio. At the top, it says '121 %' and 'No issues found'. Below the toolbar, there is a table with columns 'Test', 'Duration', 'Traits', and 'Error Message'. The table contains the following entries:

Test	Duration	Traits	Error Message
TestChallenge21 (1)	27 ms		
TestChallenge21 (1)	27 ms		
Tests (1)	27 ms		
TestFindLargestnumberOfArray	27 ms		

```
[Test]
0 references
public void TestFindLargestnumberOfArray()
{
    double[] esperado = new double[] { 7, 90, 2 };

    double[][] enviado = new double[3][] {
        new double [] { 4, 2, 7, 1 },
        new double [] { 20, 70, 40, 90 },
        new double [] { 1, 2, 0 }
    };

    Assert.AreEqual(esperado, Challenge21.Program.FindLargestnumberOfArray(enviado));
}
```

Estimación del desafío : 30 min

Duración real: 1h

# Nivel: difícil

## Instrucciones

### Seguir el robot (parte 1)

A un robot se le ha dado una lista de instrucciones de movimiento. Cada instrucción es LEFT o RIGHT, UP, DOWN, seguida de una distancia para moverse. El robot comienza en [0,0]

Desea calcular dónde terminará el robot y devolver su posición final como una matriz.

Para ilustrar, si el robot recibe las siguientes instrucciones:

*New string[] { "right 10", "up 50", "left 30", "down 10" }*

Terminará 20 a la izquierda y 40 arriba de donde comenzó, así que volvemos: int [][]{-20, 40}

## Notas

- Las únicas instrucciones dadas serán LEFT , RIGHT, UP, DOWN
- La distancia después de la instrucción es siempre un número entero positivo.

Código de la solución

```
1 reference | 1/1 passing
public static int[] TrackRobot(string[] enviado)
{
    int[] resultado = new int[2] { 0, 0};
    string movement;
    int count;

    foreach (var x in enviado) {
        string[] aux = x.Split(' ');
        movement = aux[0];
        count= Int32.Parse(aux[1]);

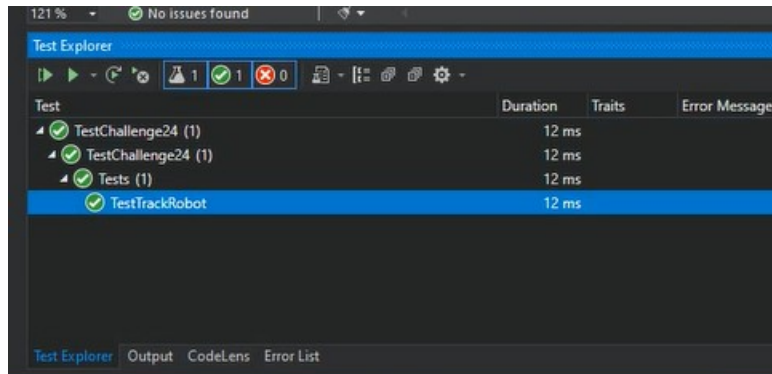
        switch (movement)
        {
            case "right":
                resultado[0] += count;
                break;
            case "left":
                resultado[0] -= count;
                break;
            case "up":
                resultado[1] += count;
                break;
            case "down":
                resultado[1] -= count;
                break;
        }
    }

    return resultado;
}
```



## Casos de prueba

```
[Test]
0 references
public void TestTrackRobot()
{
    string[] enviado = new string[] { "right 10", "up 50", "left 30", "down 10" };
    int[] resultado = new int[] { -20, 40 };
    Assert.AreEqual(resultado, Challenge24.Program.TrackRobot(enviado));
}
```



Test	Duration	Traits	Error Message
TestChallenge24 (1)	12 ms		
TestChallenge24 (1)	12 ms		
Tests (1)	12 ms		
TestTrackRobot	12 ms		

Estimación del desafío : 30 min

Duración real: 1h

## Instrucciones

### Números consecutivos

Cree una función que determine si los elementos de una matriz se pueden reorganizar para formar una lista consecutiva de números donde cada número aparece exactamente una vez.

### Código de la solución

```
2 references | 1/1 passing
public static bool ConsecutiveNumbers(int[] enviado)
{
    List<int> enviadoLista = enviado.ToList();

    enviadoLista.Sort();
    bool retornoResultado = true;

    retornoResultado = ( enviadoLista.Distinct().Count() == enviadoLista.Count() ? true: false);

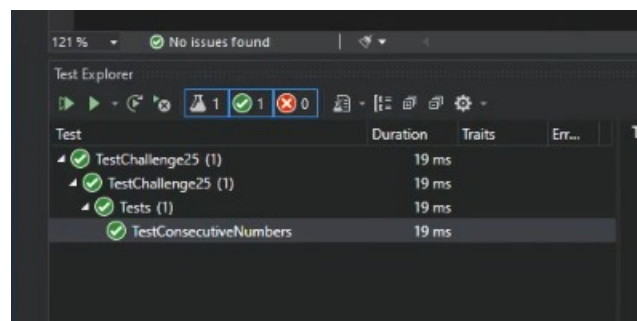
    if (retornoResultado)
    {
        for (int i = 0; i < enviadoLista.Count - 1; i++)
        {
            if (enviadoLista[i] != enviadoLista[i + 1] - 1)
            {
                retornoResultado = false;
            }
        }
    }

    return retornoResultado;
}
```

### Casos de prueba

```
[Test]
0 references
public void TestConsecutiveNumbers()
{
    int[] enviado = { 5, 4, 6, 3, 2 };

    Assert.AreEqual(true, Challenge25.Program.ConsecutiveNumbers(enviado));
}
```



Estimación del desafío : 1h

Duración real: 1h

# Nivel: muy difícil

## Instrucciones

### Orden alfabético verdadero

Cree una función que tome cada letra de cada palabra y las coloque en orden alfabético. Tenga en cuenta que la longitud de las palabras originales debe permanecer igual.

Ejemplos:

*Truealphabetic("hello world") → "dehll loorw"*

*Truealphabetic("edabit is awesome") → "aabdee ei imosstw"*

*Truealphabetic("have a nice day") → "aaac d eehi nvy"*

### notas

- Todas las oraciones estarán en minúsculas.
- No se incluirán signos de puntuación ni números en las Pruebas.

## Código de la solución

```
2 references | 1/1 passing
public static string AlphabeticalOrder(string cadenaEnviada)
{
    string[] nuevaCadena;
    StringBuilder resultado= new StringBuilder();

    nuevaCadena = cadenaEnviada.Split(' ');

    cadenaEnviada = String.Concat(cadenaEnviada.OrderBy(c => c)).Trim();

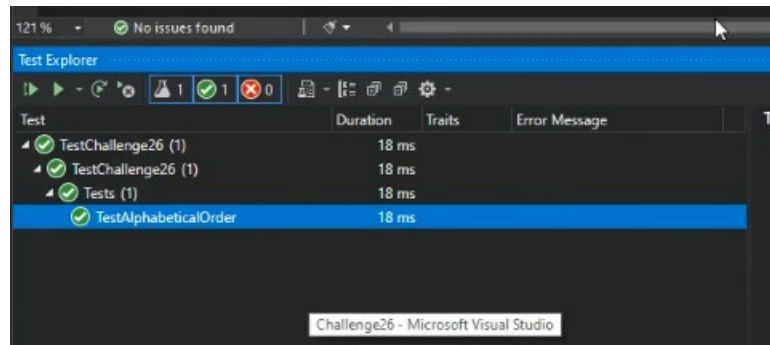
    int posicionInicio=0;

    foreach (var i in nuevaCadena)
    {
        resultado.Append(cadenaEnviada.Substring(posicionInicio, i.Length));
        resultado.Append(" ");
        posicionInicio += i.Length;
    }
    return resultado.ToString().Trim();
}
```

## Casos de prueba

```
[Test]
0 references
public void TestAlphabeticalOrder()
{
    string cadenaEnviada= "edabit is awesome";
    string retorno = "aabdee ei imosstw";

    Assert.AreEqual(retorno, Challenge26.Program.AlphabeticalOrder(cadenaEnviada));
}
```



Estimación del desafío : 1,5h

Duración real: 3 h

# Nivel: experto

## Instrucciones

### Pilish strings

En este desafío, transforma una cadena en una serie de *palabras* (o secuencias de caracteres) separadas por un solo espacio, con cada palabra teniendo la misma longitud dada por los primeros 15 dígitos de la representación decimal de Pi:

3.14159265358979

Si una cadena contiene más caracteres que la cantidad total dada por la suma de los dígitos Pi, los caracteres no utilizados se descartan y solo utilizará los necesarios para formar 15 palabras.

```
String =  
"HOWINEEDADRINKALCOHOLICINNATUREAFTERTHEHEAVYLECTURESINVOLVINGQUANTUMMECHANICSANDALLTHESECRETSO  
FTHEUNIVERSE"
```

```
Pi String = "HOW I NEED A DRINK ALCOHOLIC IN NATURE AFTER THE HEAVY LECTURES INVOLVING QUANTUM  
MECHANICS"
```

```
// Every word has the same length of the digit of Pi found at the same index ?  
// "HOW" = 3, "I" = 1, "NEED" = 4, "A" = 1, "DRINK" = 5  
// "ALCOHOLIC" = 9, "IN" = 2, "NATURE" = 6, "AFTER" = 5  
// "THE" = 3, "HEAVY" = 5, "LECTURES" = 8, "INVOLVING" = 9  
// "QUANTUM" = 7, "MECHANICS" = 9  
// 3.14159265358979
```

Además, si una cadena contiene menos caracteres que la cantidad total dada por la suma de los dígitos Pi, en cualquier caso, debe respetar la secuencia de los dígitos para obtener las palabras.

```
String = "foraloop"
```

```
Pi string = "for a loop"
```

```
// every word has the same length of the digit of pi found at the same index ?  
// "for" = 3, "a" = 1, "loop" = 4  
// 3.14
```

Si las letras contenidas en la cadena no coinciden exactamente con los dígitos, en este caso repetirá la última letra hasta que la palabra tenga la longitud correcta.

```
String = "canimakeaguessnow"
```

```
Pi string = "can i make a guess nowwwwwww"
```

```
// every word has the same length of the digit of pi found at the same index ?  
// "can" = 3, "i" = 1, "make" = 4, "a" = 1, "guess" = 5, "now" = 3  
// 3.14153 (wrong!)  
// the length of the sixth word "now" (3)...  
// ...doesn't match the sixth digit of pi (9)  
// the last letter "w" will be repeated...  
// ...until the length of the word will match the digit
```

```
// "can" = 3, "i" = 1, "make" = 4, "a" = 1, "guess" = 5, "nowwwwww" = 9
// 3.14159 (correct!)
```

Si la cadena dada está vacía, se debe devolver una cadena vacía.

Dada una cadena txt, implemente una función que devuelva la misma cadena con el formato de acuerdo con las instrucciones anteriores.

**Ejemplos:**

```
PilishString("33314444") → "333 1 4444"
// 3.14

PilishString("TOP") → "TOP"
// 3

PilishString("X") → "XXX"
// "X" has to match the same length of the first digit (3)
// The last letter of the word is repeated

pilishString("") → ""
```

**notas**

- Este desafío es un concepto simplificado inspirado en Pilish, un tipo peculiar de escritura restringida que utiliza todos los dígitos posibles conocidos de Pi. Se puede escribir un texto potencialmente infinito permitiendo la puntuación y un conjunto de reglas adicionales, que permiten evitar largas secuencias de dígitos pequeños, sustituyéndolos por palabras de más de 9 letras y haciendo que la composición se asemeje más a un poema en verso libre .
- 
- El punto que separa la parte entera de Pi de la parte decimal no tiene que ser considerado en la función: está presente en Instrucciones y Ejemplos solo para facilitar la lectura.

## Código de la solución

```
1 reference | 1/1 passing
public static string PilishString(string recibida)
{
    List<string> retorno = new List<string>();
    string result = "314159265358979";
    var arreglolongitud = result.Select(digit => int.Parse(digit.ToString()));
    int inicio = 0;

    for (int i = 0; i < arreglolongitud.Count(); i++)
    {
        if (inicio < recibida.Length)
        {
            if ((recibida.Length - inicio) >= arreglolongitud.ElementAt(i))
            {
                retorno.Add(recibida.Substring(inicio, arreglolongitud.ElementAt(i)));
                inicio += arreglolongitud.ElementAt(i);
            }
            else
            {
                StringBuilder aux = new StringBuilder();
                aux.Append(recibida.Substring(inicio));

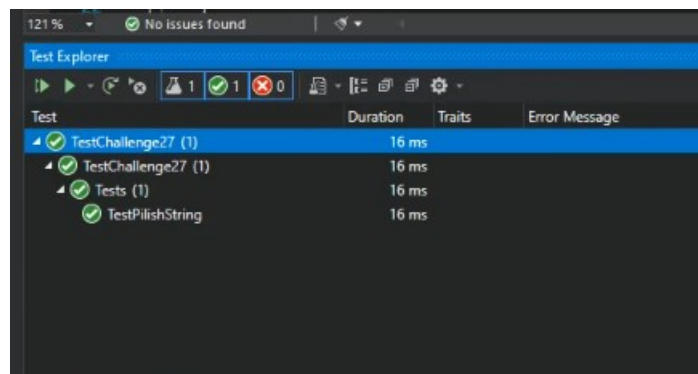
                for (int j = 0; j < (arreglolongitud.ElementAt(i) - recibida.Substring(inicio).Length); j++)
                {
                    aux.Append(recibida.Last());
                }
                retorno.Add(aux.ToString());
                break;
            }
        }
    }

    return string.Join(" ", retorno.ToArray());
}
```

## Casos de prueba

```
[Test]
public void TestPilishString()
{
    string enviado = "HOWINEEDADRINKALCOHOLICINNATUREAFTERTHEHEAVYLECTURESINVOLVINGQUANTUMMECHANICSANDALLTHESECRETSOFTHEUNIVERSE";
    string piResultado = "HOW I NEED A DRINK ALCOHOLIC IN NATURE AFTER THE HEAVY LECTURES INVOLVING QUANTUM MECHANICS";

    Assert.AreEqual(piResultado, Challenge27.Program.PilishString(enviado));
}
```



The screenshot shows the Test Explorer window with a table of test results. The table has columns for Test, Duration, Traits, and Error Message. The tests listed are TestChallenge27 (1), TestChallenge27 (1), Tests (1), and TestPilishString, all of which passed successfully with a duration of 16 ms.

Test	Duration	Traits	Error Message
TestChallenge27 (1)	16 ms		
TestChallenge27 (1)	16 ms		
Tests (1)	16 ms		
TestPilishString	16 ms		

Estimación del desafío : 3 h

Duración real: 4 h