

CICLO: DESARROLLO DE APLICACIONES WEB
MÓDULO: ENTORNOS DE DESARROLLO

CURSO: 1º

CALIFICACIÓN:

Nombre: Adrián

Apellidos: Aránegas Molina

Lee detenidamente y de forma completa cada uno de los problemas planteados y responde únicamente a lo que se pide.

Crear un directorio en el directorio [c:/Documentos](#) llamado: [NombreApellido_ExUD4-ED](#).

Una vez completada la prueba, renombrar este documento con el siguiente nombre: 'NombreApellido_Ex_UD4.odt'. Adjuntar a la entrega de la plataforma Moodle:

- 'NombreApellido_Ex_UD4.odt'
- 'NombreApellido_ExUD4-ED.zip' que incluye los directorios de trabajo de cada ejercicio.

Crear un repositorio en el usuario de GitHub. Dicho repositorio incluye un directorio para cada ejercicio del examen.

Añadir a continuación la URL de acceso al repositorio (Obligatorio, no puntua):

Enlace a ejercicios

Ejercicio 1 (5 puntos).....	2
Ejercicio 2 (5 puntos).....	5

Ejercicio 1 (5 puntos)

Requisitos previos: [SDK de .NET Core](#) Si aún no lo tienes instalado: descargar o utilizar el archivo ‘dotnet-sdk-8.0.102-win-x64’ facilitado en clase.

1. Crear el proyecto usando comandos.

Abrir la consola de ejecución de comandos de Windows y en el directorio [c:/Documentos/NombreAlumno_ExUD4-ED](#), crear el siguiente directorio: `unit-testing-using-nunit`

En este directorio, abrir un terminal Windows.

```
C:\Users\Ana González\Documents\AnaGlez_ExUD4-ED> cd unit-testing-using-nunit
C:\Users\Ana González\Documents\AnaGlez_ExUD4-ED\unit-testing-using-nunit>
```

Dentro de esa carpeta ejecutar el comando: `dotnet new sln`

Adjuntar una captura con el resultado de la ejecución del comando (fecha y hora del momento de realización de la prueba, visible en la captura).

```
Microsoft Windows [Versión 10.0.19045.5688]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\IDM2425-11>cd Documents
C:\Users\IDM2425-11\Documents>cd AdrianArangas_ExUD4-ED
C:\Users\IDM2425-11\Documents\AdrianArangas_ExUD4-ED>mkdir unit-testing-using-nunit
C:\Users\IDM2425-11\Documents\AdrianArangas_ExUD4-ED>cd unit-testing-using-nunit
C:\Users\IDM2425-11\Documents\AdrianArangas_ExUD4-ED\unit-testing-using-nunit>dotnet new sln

Esto es .NET 8.0.
-----
Versión del SDK: 8.0.403
-----
Telemetría
-----
Las herramientas de .NET recopilan datos de uso para ayudarnos a mejorar su experiencia. Microsoft los recopila y los comparte con la comunidad. Puede optar por no participar en la telemetría si establece la variable de entorno DOTNET_CLI_TELEMETRY_OPTOUT en "1" o "true" mediante su shell favorito.
Vea más sobre la telemetría de las herramientas de la CLI de .NET: https://aka.ms/dotnet-cli-telemetry
-----
Instalar un certificado de desarrollo HTTPS de ASP.NET Core.
Para confiar en el certificado, ejecute "dotnet dev-certs https --trust"
Obtenga información sobre HTTPS: https://aka.ms/dotnet-https
-----
Escribir su primera aplicación: https://aka.ms/dotnet-hello-world
Descubre las novedades: https://aka.ms/dotnet-whats-new
Explore la documentación: https://aka.ms/dotnet-docs
Notificar problemas y encontrar el código fuente en GitHub: https://github.com/dotnet/core
Use "dotnet --help" para ver los comandos disponibles o visite: https://aka.ms/dotnet-cli
La plantilla "Archivo de la solución" se creó correctamente.

C:\Users\IDM2425-11\Documents\AdrianArangas_ExUD4-ED\unit-testing-using-nunit>
```

2. Crear un directorio: *PrimeService*. Dentro de este directorio ejecutar el siguiente comando para crear el proyecto de origen: **dotnet new classlib**

Adjuntar una captura con el resultado de la ejecución del comando (fecha y hora del momento de realización de la prueba, visible en la captura).

```
C:\Users\IDM2425-11\Documents\AdrianAranegas_ExUD4-ED\unit-testing-using-nunit>cd PrimeService
C:\Users\IDM2425-11\Documents\AdrianAranegas_ExUD4-ED\unit-testing-using-nunit\PrimeService>dotnet new classlib
La plantilla "Biblioteca de clases" se creó correctamente.
Procesando acciones posteriores a la creación...
Restaurando C:\Users\IDM2425-11\Documents\AdrianAranegas_ExUD4-ED\unit-testing-using-nunit\PrimeService\PrimeService.csproj:
Determinando los proyectos que se van a restaurar...
Se ha restaurado C:\Users\IDM2425-11\Documents\AdrianAranegas_ExUD4-ED\unit-testing-using-nunit\PrimeService\PrimeService.csproj (en 63 ms).
Restauración realizada correctamente.
C:\Users\IDM2425-11\Documents\AdrianAranegas_ExUD4-ED\unit-testing-using-nunit\PrimeService>
```

3. Cambiar el nombre del archivo 'Class1.cs' a 'PrimeService.cs', editar el archivo y añadir el siguiente código:

```
using System;
namespace Prime.Services
{
    public class PrimeService
    {
        public bool IsPrime(int candidate)
        {
            throw new NotImplementedException("Please create a test first.");
        }
    }
}
```

4. Volver a la línea de comandos, situarse en el directorio 'unit-testing-using-nunit' (comando **cd..**) y ejecutar el comando:

dotnet sln add PrimeService/PrimeService.csproj

5. Crear el proyecto de prueba. Crear un directorio nuevo: 'PrimeService.Tests'

Mode	LastWriteTime		Length	Name
----	-----		-----	----
d-----	16/02/2024	11:28		PrimeService
d-----	16/02/2024	11:29		PrimeService.Tests
-a----	16/02/2024	11:29	1012	unit-testing-using-nunit.sln

Adjuntar una captura con la estructura de directorios generada (fecha y hora del momento de realización de la prueba, visible en la captura).

```
Directorio de C:\Users\IDM2425-11\Documents\AdrianAranegas_ExUD4-ED\unit-testing-using-nunit
16/03/2025 13:21 <DIR> .
16/03/2025 13:21 <DIR> ..
16/03/2025 13:18 <DIR> PrimeService
16/03/2025 13:20 <DIR> PrimeService.Tests
16/03/2025 13:19 1.012 unit-testing-using-nunit.sln
1 archivos 1.012 bytes
4 dirs 15.371.788.288 bytes libres
C:\Users\IDM2425-11\Documents\AdrianAranegas_ExUD4-ED\unit-testing-using-nunit>
```

6. Acceder al directorio 'PrimeService.Tests' y ejecutar el siguiente comando:

dotnet new nunit

“dotnet new agrega el SDK de prueba de Microsoft, el marco de pruebas de NUnit y el adaptador de prueba de NUnit.”

Adjuntar una captura con el resultado de la ejecución del comando (fecha y hora del momento de realización de la prueba, visible en la captura).

```

C:\Users\10M2425-11\Documents\AdrianAranegas_ExUD4-ED\unit-testing-using-nunit>cd PrimeService.Tests
C:\Users\10M2425-11\Documents\AdrianAranegas_ExUD4-ED\unit-testing-using-nunit\PrimeService.Tests>dotnet new nunit
Se creó correctamente la plantilla "NUnit 3 Test Project".
Procesando acciones posteriores a la creación...
Restaurando C:\Users\10M2425-11\Documents\AdrianAranegas_ExUD4-ED\unit-testing-using-nunit\PrimeService.Tests\PrimeService.Tests.csproj:
Determinando los proyectos que se van a restaurar...
Se ha restaurado C:\Users\10M2425-11\Documents\AdrianAranegas_ExUD4-ED\unit-testing-using-nunit\PrimeService.Tests\PrimeService.Tests.csproj (en 1,11 s).
Restauración realizada correctamente.
C:\Users\10M2425-11\Documents\AdrianAranegas_ExUD4-ED\unit-testing-using-nunit\PrimeService.Tests>
  
```

7. Agregar la biblioteca de clases PrimeService como otra dependencia al proyecto:

dotnet add reference ../PrimeService/PrimeService.csproj

Adjuntar una captura con el resultado de la ejecución del comando (fecha y hora del momento de realización de la prueba, visible en la captura).

```

C:\Users\10M2425-11\Documents\AdrianAranegas_ExUD4-ED\unit-testing-using-nunit\PrimeService.Tests>dotnet add reference ../PrimeService/PrimeService.csproj
Se ha agregado la referencia "..\PrimeService\PrimeService.csproj" al proyecto.
C:\Users\10M2425-11\Documents\AdrianAranegas_ExUD4-ED\unit-testing-using-nunit\PrimeService.Tests>
  
```

8. En el directorio unit-testing-using-nunit, ejecutar:

dotnet sln add ../PrimeService.Tests/PrimeService.Tests.csproj

9. En el directorio PrimeService.Tests, cambie el nombre del archivo UnitTest1.cs por PrimeService_IsPrimeShould.cs y reemplace todo su contenido por el código siguiente:

using NUnit.Framework;
using Prime.Services;

```

namespace Prime.UnitTests.Services
{
    public class PrimeService_IsPrimeShould
    {
        private PrimeService _primeService;

        public void SetUp()
        {
            _primeService = new PrimeService();
        }
    }
}
  
```

```
public void IsPrime_InputIs1_ReturnFalse()
{
    var result = _primeService.IsPrime(1);
    Assert.IsFalse(result, "1 should not be prime");
}
```

```
}
}
```

Añadir al código anterior los atributos NUnit para los siguientes requisitos de la clase prueba:

- El atributo [TestFixture] para indicar que la clase **PrimeService_IsPrimeShould** contiene pruebas unitarias.
- El atributo [Test] que indica que el método **IsPrime_InputIs1_ReturnFalse()** es un método de prueba.
- El atributo [SetUp] que indica que el método **SetUp()** es el código de inicialización.

Una vez actualizada la clase principal, ejecutar: **dotnet test**

Adjuntar una captura con el resultado de la ejecución del comando (fecha y hora del momento de realización de la prueba, visible en la captura).

Añadir una interpretación del motivo del error recibido

El error que nos sale es simplemente que la prueba echa se ha realizado correctamente que 0 han sido superadas 0 fallidas y una que hemos echo ha sido omitida porque lo hemos decidido así

10. Añadir código para que el test no devuelva error. Adjuntar captura de pantalla en caso de conseguir una prueba sin error.

11. Añadir el directorio *unit-testing-using-nunit* al repositorio Git

Ejercicio 2 (5 puntos)

1. Crear el proyecto usando Netbeans:

Categoría: Java with Maven

Tipo: Java Application

Nombre proyecto: NomApe_Ej2-ExUD4

Localización: [c:/Documentos/NombreAlumno_ExUD4-ED](#)

2. Crear una clase java: conversor.java

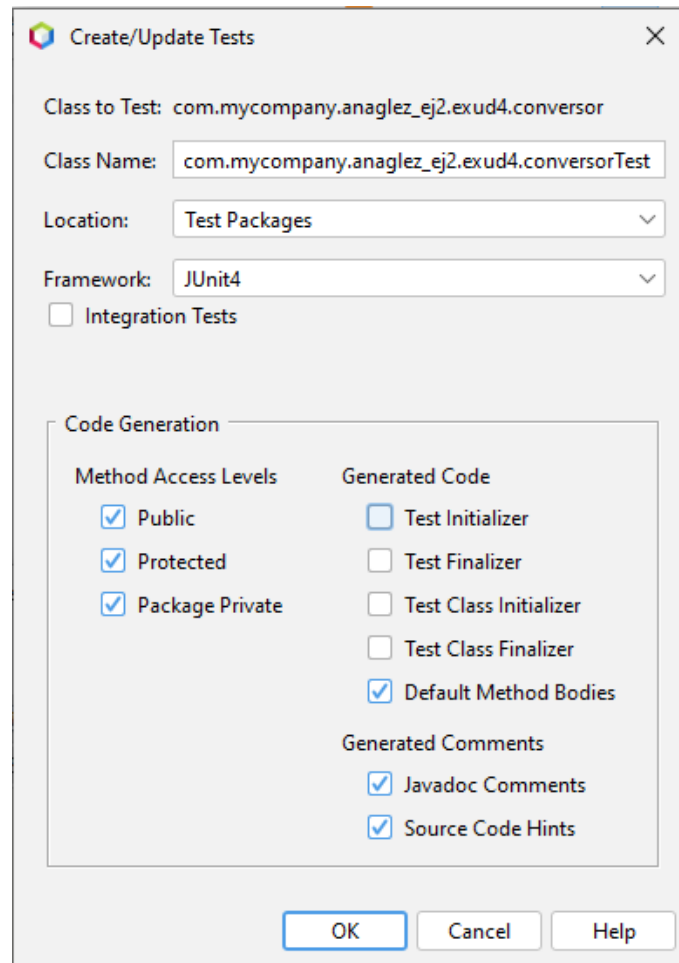
Añadir el siguiente código:

```
/**
 *
 * @author Ana González
 */
public class conversor {
    public static void main(String[] argumentos) {
        float celsius = 0f;
        // Cálculo conversión
        float fahrenheit = celsiusAFahrenheit(celsius);
        System.out.printf("%f grados celsius son %f grados fahrenheit", celsius, fahrenheit);
    }

    public static float celsiusAFahrenheit(float celsius) {
        return (celsius * 1.8f) + 32.0f;
    }

    public static float fahrenheitACelsius(float fahrenheit) {
        return (fahrenheit - 32.0f) / 1.8f;
    }
}
```

3. Crear una prueba test:



Modificar **conversorTest.java** con el siguiente código:

```
/**
 *
 * @author Ana González
 */
public class conversorTest {

    public conversorTest() {
    }

    /**
     * Test of main method, of class conversor.
     */
    @Test
    public void testMain() {
        System.out.println("main");
        String[] argumentos = null;
        conversor.main(argumentos);
        // TODO review the generated test code and remove the default call to fail.
        fail("The test case is a prototype.");
    }
}
```

```

/**
 * Test of celsiusAFahrenheit method, of class conversor.
 */
@Test
public void testCelsiusAFahrenheit() {
    System.out.println("celsiusAFahrenheit");
    float celsius = 0.0f;
    float expectedResult = 32.0f;
    float result = conversor.celsiusAFahrenheit(celsius);
    assertEquals(expectedResult, result, 0.0);
    // TODO review the generated test code and remove the default call to fail.
    fail("The test case is a prototype.");
}

/**
 * Test of fahrenheitACelsius method, of class conversor.
 */
@Test
public void testFahrenheitACelsius() {
    System.out.println("fahrenheitACelsius");
    float fahrenheit = 0.0f;
    float expectedResult = -17.777779f;
    float result = conversor.fahrenheitACelsius(fahrenheit);
    assertEquals(expectedResult, result, 0.0);
    // TODO review the generated test code and remove the default call to fail.
    fail("The test case is a prototype.");
}
}

```

5. Ejecutar las pruebas y adjuntar a continuación una captura del resultado de las pruebas

6. Modificar **`conversorTest.java`** para que las pruebas se ejecuten sin error.
7. Ejecutar las pruebas y adjuntar a continuación una captura del resultado de las pruebas.