

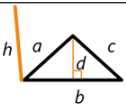
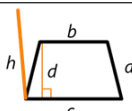
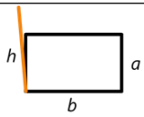
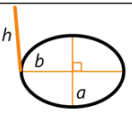
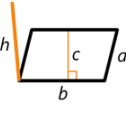
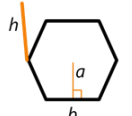
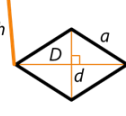
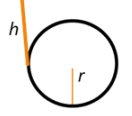
Proyecto final

Metería	Ingeniería de Software		
Integrantes (paterno, materno, nombres)	Nombres		Cuentas
	Rivas Gil Carlos Danery		16462077
Equipo	09	Figura	Traingulo
Fecha	07 de diciembre de 2024		
Git	https://github.com/Danyepro/Proyecto_IS.git		

Rubrica:

4 pts.	5 pruebas unitarias de la capa de Aplicación del perímetro.
4 pts.	5 pruebas unitarias de la capa de Aplicación del área.
4 pts.	5 pruebas unitarias de la capa de Aplicación del volumen.
4 pts.	5 pruebas de integración de la capa de Aplicación del área y volumen.
4 pts.	5 pruebas de integración de la capa Web (error 404).
5 pts.	Código y documento pdf alojado en Git.

Asignación del problema a resolver por equipo:

Equipo 01, 09 Triangulo 	$P = a + b + c$ $A = \frac{bd}{2}$ $V = Ah$	Equipo 05, 13 Trapezio 	$P = 2a + b + c$ $A = \frac{d(b+c)}{2}$ $V = Ah$
Equipo 02, 10 Rectangulo <small>n, es el numero de lados</small> 	$P = 2b + 2a$ $A = ab$ $V = Ah$	Equipo 06, 14 Elipse 	$P = \pi(a + b)$ $A = \pi ab$ $V = Ah$
Equipo 03, 11 Paralelogramo 	$P = 2(a + b)$ $A = bc$ $V = Ah$	Equipo 07, 15 Polígono Regular <small>n, es el numero de lados</small> 	$P = nb$ $A = \frac{nba}{2}$ $V = Ah$
Equipo 04, 12 Rombo 	$P = 4a$ $A = \frac{dD}{2}$ $V = Ah$	Equipo 08, 16 Circulo 	$P = 2\pi r$ $A = \pi r^2$ $V = Ah$

Comandos:

```
# Ejecutar pruebas
dotnet test .\test\Application.UnitTest
dotnet test .\test\Application.IntegrationTest
dotnet test .\test\WebApp.IntegrationTest

git log
```

Pruebas Unitarias

Xunit application triangulo

```
using System;
using Xunit;
using Application;

namespace Application.UnitTest
{
    public class UnitTestTriangulo
    {
        // Pruebas para el Perímetro
        [Theory]
        [InlineData(3, 4, 5, 12)] // Perímetro = a + b + c
        [InlineData(6, 8, 10, 24)]
        [InlineData(5, 5, 5, 15)]
        [InlineData(7, 24, 25, 56)]
        [InlineData(10, 10, 15, 35)]
        public void TestPerimetro(double a, double b, double c, double perimetroEsperado)
        {
            // Act - Realizar
            double resultado = Triangulo.Perimetro(a, b, c);

            // Assert - Verificar
            Assert.Equal(perimetroEsperado, resultado, 5); // Tolerancia de 5 decimales
        }

        // Pruebas para el Área
        [Theory]
        [InlineData(6, 4, 12)] // Área = (base * altura) / 2
        [InlineData(10, 5, 25)]
        [InlineData(8, 3, 12)]
        [InlineData(15, 10, 75)]
        [InlineData(7, 6, 21)]
        public void TestArea(double baseTriangulo, double altura, double areaEsperada)
        {
            // Act - Realizar
            double resultado = Triangulo.Area(baseTriangulo, altura);

            // Assert - Verificar
            Assert.Equal(areaEsperada, resultado, 5); // Tolerancia de 5 decimales
        }
    }
}
```

Xunit application Prisma Traingular

```
using System;
using Xunit;
using Application;

namespace Application.UnitTest
{
    public class UnitTestPrismaTriangular
    {
        // Pruebas para el Volumen
        [Theory]
        [InlineData(6, 10, 60)] // Volumen = Área * Altura del prisma
        [InlineData(12, 5, 60)]
        [InlineData(7.5, 8, 60)]
        [InlineData(20, 15, 300)]
        [InlineData(25, 12, 300)]
        public void TestVolumen(double areaBase, double alturaPrisma, double volumenEsperado)
        {
            // Act - Realizar
            double resultado = PrismaTriangular.Volumen(areaBase, alturaPrisma);

            // Assert - Verificar
            Assert.Equal(volumenEsperado, resultado, 5); // Tolerancia de 5 decimales
        }
    }
}
```

Pruebas de Integracion

```
using Xunit;
using Application;

namespace Application.IntegrationTest
{
    public class IntegrationTestApplication
    {
        [Fact]
        public void TestCalculoAreaBase()
        {
            double baseTriangulo = 5.0;
            double alturaTriangulo = 6.0;
            double areaBase = Triangulo.Area(baseTriangulo, alturaTriangulo);
            Assert.Equal(15.0, areaBase, 2);
        }

        [Fact]
        public void TestCalculoVolumen()
        {
            double baseTriangulo = 5.0;
```

```
        double alturaTriangulo = 6.0;
        double alturaPrisma = 10.0;
        double areaBase = Triangulo.Area(baseTriangulo, alturaTriangulo);
        double volumen = PrismaTriangular.Volumen(areaBase, alturaPrisma);
        Assert.Equal(150.0, volumen, 2);
    }
}
```

```
[Fact]
public void TestPerimetroBaseYAreaBase()
{
    double lado1 = 3.0;
    double lado2 = 4.0;
    double lado3 = 5.0;
    double baseTriangulo = 5.0;
    double alturaTriangulo = 6.0;

    double perimetro = Triangulo.Perimetro(lado1, lado2, lado3);
    double areaBase = Triangulo.Area(baseTriangulo, alturaTriangulo);

    Assert.True(perimetro > 0 && areaBase > 0);
}
}
```

```
[Fact]
public void TestVolumenNoNegativo()
{
    double areaBase = 15.0;
    double alturaPrisma = 10.0;
    double volumen = PrismaTriangular.Volumen(areaBase, alturaPrisma);
    Assert.True(volumen > 0);
}
}
```

```
[Fact]
public void TestAreaBaseComparacion()
{
    double basePequena = 3.0;
    double alturaPequena = 4.0;
    double baseGrande = 5.0;
    double alturaGrande = 6.0;

    double areaPequena = Triangulo.Area(basePequena, alturaPequena);
    double areaGrande = Triangulo.Area(baseGrande, alturaGrande);

    Assert.True(areaGrande > areaPequena);
}
}
}
```

Web Integration test

```
using System.Net;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Mvc.Testing;
using Xunit;

namespace WebApp.IntegrationTest
{
    public class IntegrationTestWeb : IClassFixture<WebApplicationFactory<Program>>
    {
        private readonly WebApplicationFactory<Program> _factory;

        public IntegrationTestWeb(WebApplicationFactory<Program> factory)
        {
            _factory = factory;
        }

        [Fact]
        public async Task TestHomePage()
        {
            var client = _factory.CreateClient();
            var response = await client.GetAsync("/");
            Assert.Equal(HttpStatusCode.OK, response.StatusCode);
        }

        [Fact]
        public async Task Test404ErrorPage()
        {
            var client = _factory.CreateClient();
            var response = await client.GetAsync("/nonexistent-page");
            Assert.Equal(HttpStatusCode.NotFound, response.StatusCode);
        }

        [Fact]
        public async Task TestStaticFile404()
        {
            var client = _factory.CreateClient();
            var response = await client.GetAsync("/css/invalid-file.css");
            Assert.Equal(HttpStatusCode.NotFound, response.StatusCode);
        }

        [Fact]
        public async Task TestInvalidEndpoint404()
        {
            var client = _factory.CreateClient();
            var response = await client.GetAsync("/api/invalid-endpoint");
            Assert.Equal(HttpStatusCode.NotFound, response.StatusCode);
        }

        [Fact]
```

```

    public async Task Test404ForInvalidQueryString()
    {
        var client = _factory.CreateClient();
        var response = await client.GetAsync("/?invalid=query");
        Assert.Equal(HttpStatusCode.OK, response.StatusCode); // Página existe
    }
}
}

```

Pantallas de la evidencia:

dotnet test .\test\Application.UnitTest

```

PowerShell 7 (x64)
PS C:\Users\PC FACTOR WHITE\GIT\IS10A\Proyecto_IS - final> dotnet test .\test\Application.UnitTest
Determinando los proyectos que se van a restaurar...
Todos los proyectos están actualizados para la restauración.
Application -> C:\Users\PC FACTOR WHITE\GIT\IS10A\Proyecto_IS - final\src\Application\bin\Debug\net8.0\Application.dll
Application.UnitTest -> C:\Users\PC FACTOR WHITE\GIT\IS10A\Proyecto_IS - final\test\Application.UnitTest\bin\Debug\net8.0\Application.UnitTest.dll
Serie de pruebas para C:\Users\PC FACTOR WHITE\GIT\IS10A\Proyecto_IS - final\test\Application.UnitTest\bin\Debug\net8.0\Application.UnitTest.dll (.NETCoreApp,Version=v8.0)
Versión 17.11.1 (x64) de VSTest

Iniciando la ejecución de pruebas, espere...
1 archivos de prueba en total coincidieron con el patrón especificado.

Correctas! - Con error:    0, Superado:    15, Omitido:    0, Total:    15, Duración: 20 ms - Application.UnitTest.dll (net8.0)
PS C:\Users\PC FACTOR WHITE\GIT\IS10A\Proyecto_IS - final>

```

dotnet test .\test\Application.IntegrationTest

```
PowerShell 7 (x64)
PS C:\Users\PC FACTOR WHITE\GIT\IS10A\Proyecto_IS - final> dotnet test .\test\Application.IntegrationTest
Determinando los proyectos que se van a restaurar...
Todos los proyectos están actualizados para la restauración.
Application -> C:\Users\PC FACTOR WHITE\GIT\IS10A\Proyecto_IS - final\src\Application\bin\Debug\net8.0\Application.dll
Application.IntegrationTest -> C:\Users\PC FACTOR WHITE\GIT\IS10A\Proyecto_IS - final\test\Application.IntegrationTest\bin\Debug\net8.0\Application.IntegrationTest.dll
Serie de pruebas para C:\Users\PC FACTOR WHITE\GIT\IS10A\Proyecto_IS - final\test\Application.IntegrationTest\bin\Debug\net8.0\Application.IntegrationTest.dll (.NETCoreApp,Version=v8.0)
Versión 17.11.1 (x64) de VSTest

Iniciando la ejecución de pruebas, espere...
1 archivos de prueba en total coincidieron con el patrón especificado.

Correctas! - Con error:    0, Superado:    5, Omitido:    0, Total:    5, Duración: 3 ms - Application.IntegrationTest.dll (net8.0)
PS C:\Users\PC FACTOR WHITE\GIT\IS10A\Proyecto_IS - final>
```

dotnet test .\test\WebUI.IntegrationTest

```
PowerShell 7 (x64)
PS C:\Users\PC FACTOR WHITE\GIT\IS10A\Proyecto_IS - final> dotnet test .\test\WebUI.IntegrationTest
Determinando los proyectos que se van a restaurar...
Todos los proyectos están actualizados para la restauración.
Application -> C:\Users\PC FACTOR WHITE\GIT\IS10A\Proyecto_IS - final\src\Application\bin\Debug\net8.0\Application.dll
WebUI -> C:\Users\PC FACTOR WHITE\GIT\IS10A\Proyecto_IS - final\src\WebUI\bin\Debug\net8.0\WebUI.dll
WebUI.IntegrationTest -> C:\Users\PC FACTOR WHITE\GIT\IS10A\Proyecto_IS - final\test\WebUI.IntegrationTest\bin\Debug\net8.0\WebUI.IntegrationTest.dll
Serie de pruebas para C:\Users\PC FACTOR WHITE\GIT\IS10A\Proyecto_IS - final\test\WebUI.IntegrationTest\bin\Debug\net8.0\WebUI.IntegrationTest.dll (.NETCoreApp,Version=v8.0)
Versión 17.11.1 (x64) de VSTest

Iniciando la ejecución de pruebas, espere...
1 archivos de prueba en total coincidieron con el patrón especificado.

Correctas! - Con error:    0, Superado:    5, Omitido:    0, Total:    5, Duración: 38 ms - WebUI.IntegrationTest.dll (net8.0)
PS C:\Users\PC FACTOR WHITE\GIT\IS10A\Proyecto_IS - final> |
```