NUnit Examples

1. Basic Equality Check

```
[Test]
public void That_Equality_Example()
{
   Assert.That(10 + 5, Is.EqualTo(15));
}
```

2. Not Equal

```
[Test]
public void That_NotEqual_Example()
{
   Assert.That(20, Is.Not.EqualTo(10));
}
```

3. Greater / Less

```
[Test]
public void That_Greater_Example()
{
   Assert.That(15, Is.GreaterThan(10));
}

[Test]
public void That_Less_Example()
{
   Assert.That(5, Is.LessThan(10));
}
```

4. Check Null / Not Null

```
[Test]
public void That_Null_Example()
{
   string name = null;
   Assert.That(name, Is.Null);
}


[Test]
public void That_NotNull_Example()
{
   string name = "NUnit";
   Assert.That(name, Is.Not.Null);
}
```

---

5. Check True / False

```
[Test]
public void That_True_Example()
{
   Assert.That(5 < 10, Is.True);
}


[Test]
public void That_False_Example()
{
   Assert.That(10 < 5, Is.False);
}
```

---

6. String Contains

```
[Test]
public void That_String_Contains()
{
   Assert.That("NUnit Testing", Does.Contain("NUnit"));
}
```

---

7. String StartsWith / EndsWith

```
[Test]
public void That_String_StartsWith()
{
   Assert.That("Hello World", Does.StartWith("Hello"));
}

[Test]
public void That_String_EndsWith()
{
   Assert.That("Hello World", Does.EndWith("World"));
}
```

---

8. Match Regex

```
[Test]
public void That_String_Regex()
{
   Assert.That("abc123", Does.Match(@"[a-z]+[0-9]+"));
}
```

---

9. Range Check

```
[Test]
public void That_Range_Example()
{
   Assert.That(7, Is.InRange(1, 10));
}
```

---

10. Type Check

```
[Test]
public void That_Type_Check()
{
   Assert.That("abc", Is.TypeOf<string>());
}
```

---

11. Collection Contains

```
[Test]
public void That_Collection_Contains()
{
   var list = new[] { 1, 2, 3, 4 };
   Assert.That(list, Does.Contain(3));
}
```

---

12. Collection Ordered

```
[Test]
public void That_Collection_Ordered()
{
   var list = new[] { 1, 2, 3, 4 };
   Assert.That(list, Is.Ordered);
}
```

---

13. Collection Count

```
[Test]
public void That_Collection_Has_Count()
{
   var list = new List<string> { "A", "B", "C" };
   Assert.That(list, Has.Count.EqualTo(3));
}
```

---

14. Multiple Conditions (And / Or)

```
[Test]
public void That_Multiple_Conditions()
{
   Assert.That(15, Is.GreaterThan(10).And.LessThan(20));
}
```

---

15. Using Tolerance (For doubles)

```
[Test]
public void That_Tolerance_Example()
{
   double value = 3.1415;
   Assert.That(value, Is.EqualTo(3.14).Within(0.01));
}
```

---

16. Exception with Assert.That

```
[Test]
public void That_Exception_Example()
{
   Assert.That(() => int.Parse("XYZ"), Throws.Exception);
}
```

---

17. Specific Exception Type

```
[Test]

public void That_Specific_Exception()

{

   Assert.That(() => int.Parse("XYZ"), Throws.TypeOf<FormatException>());

}
```

---

18. No Exception Thrown

```
[Test]

public void That_NoException()

{

   Assert.That(() => int.Parse("123"), Throws.Nothing);

}
```

---

19. Checking Property

```
[Test]

public void That_Property_Example()

{

   var person = new { Name = "Rama", Age = 30 };

   Assert.That(person.Age, Is.EqualTo(30));

}
```

---

20. Using AnyOf (Match multiple values)

```
[Test]

public void That_AnyOf_Example()

{

   Assert.That(5, Is.AnyOf(3, 4, 5, 6));

}
```