Contents

[Bước 1: Tạo một project C# thực hiện các phép toán đơn giản cộng, trừ, nhân, chia các số Nguyên 1](#_Toc191369399)

[Bước 2: Thiết kế giao diện tính toán như bên dưới và việc xử lý các phép tính đều thông qua lớp Caculation. 1](#_Toc191369400)

[Bước 3: Tạo một tập tin Calculation1.cs chứa lớp public Calculation1 dùng có phương thức Execute để thực hiện phép tính đơn giản với hai số nguyên. 3](#_Toc191369401)

[Bước 4: Vào winform viết code để thực hiện các phép tính: 4](#_Toc191369402)

[Bước 5: Tạo project kiểm thử để kiểm thử các phép toán trong chương trình trên 7](#_Toc191369403)

[Bước 6: Tại project Unit Test, thực hiện Add Reference để tham chiếu đến project cần thực hiện Unit Test 8](#_Toc191369404)

[Bước 7: Viết unit test 9](#_Toc191369405)

[Bước 8: Chạy unit test đã viết 12](#_Toc191369406)

MS UNIT và NUNIT (C#)

# Bước 1: Tạo một project C# thực hiện các phép toán đơn giản cộng, trừ, nhân, chia các số Nguyên

A screenshot of a computer

AI-generated content may be incorrect.

# Bước 2: Thiết kế giao diện tính toán như bên dưới và việc xử lý các phép tính đều thông qua lớp Caculation.

A screenshot of a computer

AI-generated content may be incorrect.

# Bước 3: Tạo một tập tin Calculation1\_32\_Nhi.cs chứa lớp public Calculation1\_32\_Nhi dùng có phương thức Execute để thực hiện phép tính đơn giản với hai số nguyên.

A screenshot of a computer

AI-generated content may be incorrect.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Caculation\_32\_Nhi

{

public class Caculation1\_32\_Nhi

{

private int a\_32\_Nhi, b\_32\_Nhi;

public Caculation1\_32\_Nhi(int a\_32\_Nhi,int b\_32\_Nhi)

{

this.a\_32\_Nhi = a\_32\_Nhi;

this.b\_32\_Nhi = b\_32\_Nhi;

}

public int Execute\_32\_Nhi(string CalSymbol\_32\_Nhi)

{

int result\_32\_Nhi = 0;

switch (CalSymbol\_32\_Nhi)

{

case "+":

result\_32\_Nhi = this.a\_32\_Nhi + this.b\_32\_Nhi;

break;

case "-":

result\_32\_Nhi = this.a\_32\_Nhi - this.b\_32\_Nhi;

break;

case "\*":

result\_32\_Nhi = this.a\_32\_Nhi \* this.b\_32\_Nhi;

break;

case "/":

result\_32\_Nhi = this.a\_32\_Nhi / this.b\_32\_Nhi;

break;

}

return result\_32\_Nhi;

}

}

}

# **Bước 4: Vào winform viết code để thực hiện các phép tính:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Caculation\_32\_Nhi

{

public partial class Form1\_32\_Nhi : Form

{

public Form1\_32\_Nhi()

{

InitializeComponent();

}

private void btnCong\_32\_Nhi\_Click(object sender, EventArgs e)

{

int a\_32\_Nhi, b\_32\_Nhi, ketqua\_32\_Nhi;

a\_32\_Nhi = int.Parse(txtSo1\_32\_Nhi.Text);

b\_32\_Nhi = int.Parse(txtSo2\_32\_Nhi.Text);

Caculation\_32\_Nhi c\_32\_Nhi = new Caculation\_32\_Nhi(a\_32\_Nhi, b\_32\_Nhi);

ketqua\_32\_Nhi = c\_32\_Nhi.Execute\_32\_Nhi("+");

txtKQ\_32\_Nhi.Text = ketqua\_32\_Nhi.ToString();

}

private void btnTru\_32\_Nhi\_Click(object sender, EventArgs e)

{

int a\_32\_Nhi, b\_32\_Nhi, ketqua\_32\_Nhi;

a\_32\_Nhi = int.Parse(txtSo1\_32\_Nhi.Text);

b\_32\_Nhi = int.Parse(txtSo2\_32\_Nhi.Text);

Caculation\_32\_Nhi c\_32\_Nhi = new Caculation\_32\_Nhi(a\_32\_Nhi, b\_32\_Nhi);

ketqua\_32\_Nhi = c\_32\_Nhi.Execute\_32\_Nhi("-");

txtKQ\_32\_Nhi.Text = ketqua\_32\_Nhi.ToString();

}

private void btnNhan\_32\_Nhi\_Click(object sender, EventArgs e)

{

int a\_32\_Nhi, b\_32\_Nhi, ketqua\_32\_Nhi;

a\_32\_Nhi = int.Parse(txtSo1\_32\_Nhi.Text);

b\_32\_Nhi = int.Parse(txtSo2\_32\_Nhi.Text);

Caculation\_32\_Nhi c\_32\_Nhi = new Caculation\_32\_Nhi(a\_32\_Nhi, b\_32\_Nhi);

ketqua\_32\_Nhi = c\_32\_Nhi.Execute\_32\_Nhi("\*");

txtKQ\_32\_Nhi.Text = ketqua\_32\_Nhi.ToString();

}

private void btnChia\_32\_Nhi\_Click(object sender, EventArgs e)

{

int a\_32\_Nhi, b\_32\_Nhi, ketqua\_32\_Nhi;

a\_32\_Nhi = int.Parse(txtSo1\_32\_Nhi.Text);

b\_32\_Nhi = int.Parse(txtSo2\_32\_Nhi.Text);

Caculation\_32\_Nhi c\_32\_Nhi = new Caculation\_32\_Nhi(a\_32\_Nhi, b\_32\_Nhi);

ketqua\_32\_Nhi = c\_32\_Nhi.Execute\_32\_Nhi("/");

txtKQ\_32\_Nhi.Text = ketqua\_32\_Nhi.ToString();

}

}

}

# Bước 5: Tạo project kiểm thử để kiểm thử các phép toán trong chương trình trên

Click chuột phải Solution > Add > New Projects... Sau đó chọn loại project là “Unit Test Project”

A screenshot of a computer

AI-generated content may be incorrect.

# Bước 6: Tại project Unit Test, thực hiện Add Reference để tham chiếu đến project cần thực hiện Unit Test

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

# Bước 7: Viết unit test

using Microsoft.VisualStudio.TestTools.UnitTesting;

using System;

using Caculation\_32\_Nhi;

namespace CalculatorTester\_32\_Nhi

{

[TestClass]

public class UnitTest1\_32\_Nhi

{

private Caculation1\_32\_Nhi c\_32\_Nhi;

[TestInitialize] // thiet lap du lieu dung chung cho TC

public void SetUp\_32\_Nhi()

{

c\_32\_Nhi = new Caculation1\_32\_Nhi(6,3);

}

[TestMethod]//TC1: a\_32\_Nhi =6, b\_32\_Nhi = 3, kq\_32\_Nhi= 9

public void Test\_Cong\_32\_Nhi()

{

int expected\_32\_Nhi, actual\_32\_Nhi;

expected\_32\_Nhi = 9;//kết quả kì vọng

actual\_32\_Nhi = c\_32\_Nhi.Execute\_32\_Nhi("+");//kết quả thực tế

Assert.AreEqual(expected\_32\_Nhi, actual\_32\_Nhi);// vì expected\_32\_Nhi == actual\_32\_Nhi(9==9) nên kết quả khi chạy unit test sẽ là pass

}

[TestMethod]//TC2: a\_32\_Nhi =6, b\_32\_Nhi = 3, kq\_32\_Nhi= 3

public void Test\_Tru\_32\_Nhi()

{

int expected\_32\_Nhi, actual\_32\_Nhi;

expected\_32\_Nhi = 3;//kết quả kì vọng

actual\_32\_Nhi = c\_32\_Nhi.Execute\_32\_Nhi("-");//kết quả thực tế

Assert.AreEqual(expected\_32\_Nhi, actual\_32\_Nhi);// vì expected\_32\_Nhi == actual\_32\_Nhi(3==3) nên kết quả khi chạy unit test sẽ là pass

}

[TestMethod]//TC3: a\_32\_Nhi =6, b\_32\_Nhi = 3, kq\_32\_Nhi= 18

public void Test\_Nhan\_32\_Nhi()

{

int expected\_32\_Nhi, actual\_32\_Nhi;

expected\_32\_Nhi = 18;//kết quả kì vọng

actual\_32\_Nhi = c\_32\_Nhi.Execute\_32\_Nhi("\*");//kết quả thực tế

Assert.AreEqual(expected\_32\_Nhi, actual\_32\_Nhi);// vì expected\_32\_Nhi == actual\_32\_Nhi(18==18) nên kết quả khi chạy unit test sẽ là pass

}

[TestMethod]//TC4: a\_32\_Nhi =6, b\_32\_Nhi = 3, kq\_32\_Nhi= 2

public void Test\_Chia\_32\_Nhi()

{

int expected\_32\_Nhi, actual\_32\_Nhi;

expected\_32\_Nhi = 2;//kết quả kì vọng

actual\_32\_Nhi = c\_32\_Nhi.Execute\_32\_Nhi("/");//kết quả thực tế

Assert.AreEqual(expected\_32\_Nhi, actual\_32\_Nhi);// vì expected\_32\_Nhi == actual\_32\_Nhi(2==2) nên kết quả khi chạy unit test sẽ là pass

}

//TC5:a\_32\_Nhi=10,b\_32\_Nhi=0,kq\_32\_Nhi=DivideByZeroException

[ExpectedException(typeof(DivideByZeroException))]//pass khi xuất hiện ngoại lệ DivideByZeroException khi thực hiện phép chia cho 0

[TestMethod]

public void Test\_ChiaZero\_32\_Nhi()

{

c\_32\_Nhi = new Caculation1\_32\_Nhi(10, 0);

c\_32\_Nhi.Execute\_32\_Nhi("/");

}

[TestMethod]//TC6: a\_32\_Nhi =6, b\_32\_Nhi = 3, kq\_32\_Nhi= 10

public void Test\_CongF\_32\_Nhi()

{

int expected\_32\_Nhi, actual\_32\_Nhi;

expected\_32\_Nhi = 10;//kết quả kì vọng

actual\_32\_Nhi = c\_32\_Nhi.Execute\_32\_Nhi("+");//kết quả thực tế

Assert.AreEqual(expected\_32\_Nhi, actual\_32\_Nhi);// vì expected\_32\_Nhi == actual\_32\_Nhi(10==9) nên kết quả khi chạy unit test sẽ là fail

}

[TestMethod]//TC7: a\_32\_Nhi =6, b\_32\_Nhi = 3, kq\_32\_Nhi= 2

public void Test\_TruF\_32\_Nhi()

{

int expected\_32\_Nhi, actual\_32\_Nhi;

expected\_32\_Nhi = 2;//kết quả kì vọng

actual\_32\_Nhi = c\_32\_Nhi.Execute\_32\_Nhi("-");//kết quả thực tế

Assert.AreEqual(expected\_32\_Nhi, actual\_32\_Nhi);// vì expected\_32\_Nhi == actual\_32\_Nhi(2==3) nên kết quả khi chạy unit test sẽ là fail

}

[TestMethod]//TC8: a\_32\_Nhi =6, b\_32\_Nhi = 3, kq\_32\_Nhi= 1

public void Test\_NhanF\_32\_Nhi()

{

int expected\_32\_Nhi, actual\_32\_Nhi;

expected\_32\_Nhi = 1;//kết quả kì vọng

actual\_32\_Nhi = c\_32\_Nhi.Execute\_32\_Nhi("\*");//kết quả thực tế

Assert.AreEqual(expected\_32\_Nhi, actual\_32\_Nhi);// vì expected\_32\_Nhi == actual\_32\_Nhi(1==18) nên kết quả khi chạy unit test sẽ là fail

}

[TestMethod]//TC9: a\_32\_Nhi =6, b\_32\_Nhi = 3, kq\_32\_Nhi= 1

public void Test\_ChiaF\_32\_Nhi()

{

int expected\_32\_Nhi, actual\_32\_Nhi;

expected\_32\_Nhi = 1;//kết quả kì vọng

actual\_32\_Nhi = c\_32\_Nhi.Execute\_32\_Nhi("/");//kết quả thực tế

Assert.AreEqual(expected\_32\_Nhi, actual\_32\_Nhi);// vì expected\_32\_Nhi == actual\_32\_Nhi(1==2) nên kết quả khi chạy unit test sẽ là fail

}

//TC10:a\_32\_Nhi=10,b\_32\_Nhi=0,kq\_32\_Nhi=DivideByZeroException

[ExpectedException(typeof(DivideByZeroException))]//fail khi không xuất hiện ngoại lệ DivideByZeroException khi thực hiện phép chia cho 0

[TestMethod]

public void Test\_ChiaZeroF\_32\_Nhi()

{

c\_32\_Nhi = new Caculation1\_32\_Nhi(10, 1);

c\_32\_Nhi.Execute\_32\_Nhi("/");

}

}

}

# Bước 8: Chạy unit test đã viết

A screenshot of a computer program

AI-generated content may be incorrect.

=>Kết quả:

A screenshot of a computer

AI-generated content may be incorrect.

(đã giải thích lý do pass fail trên phần viết unit test)