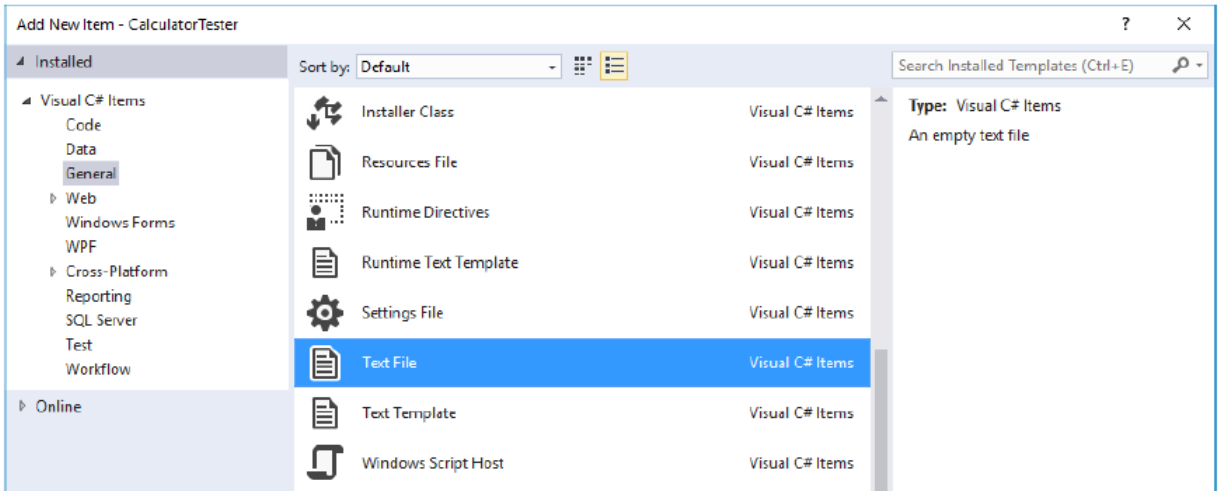
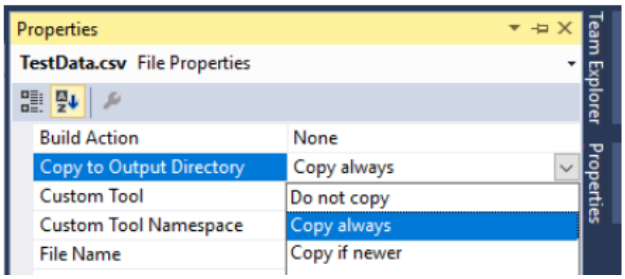
MS UNIT và NUNIT (C#) **DATA-DRIVEN UNIT TEST**

Giả sử tạo một project C# thực hiện các phép toán đơn giản cộng, trừ, nhân, chia các số Nguyên -> xem lại buổi 3.

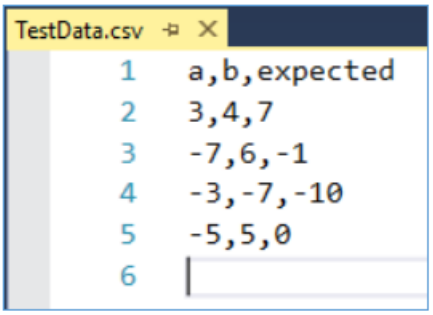
Thực thi test case với các dữ liệu test có sẵn: tạo thư mục `Data` trong project test, click phải chuột vào thư mục `Data > New Item`... tạo tập tin `TestData.csv`.



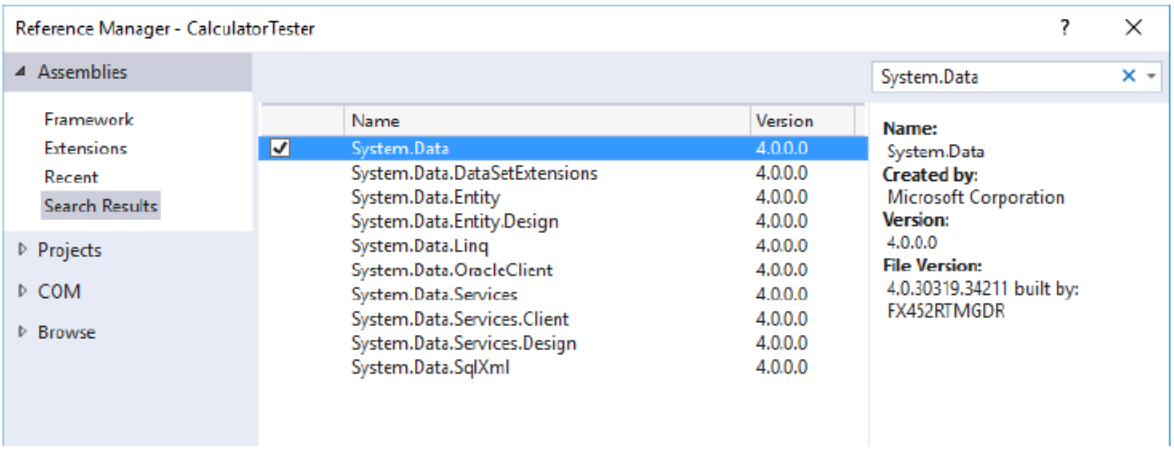
Chuột phải tập tin `TestData.csv` chọn Properties, thiết lập thuộc tính "Copy to Output Directory" thành "Copy always" để tập tin này sẽ được sao chép vào thư mục `bin` khi build project.



Nhập dữ liệu vào tập tin `TestData.csv` như sau:



Thêm reference `System.Data` vào project test



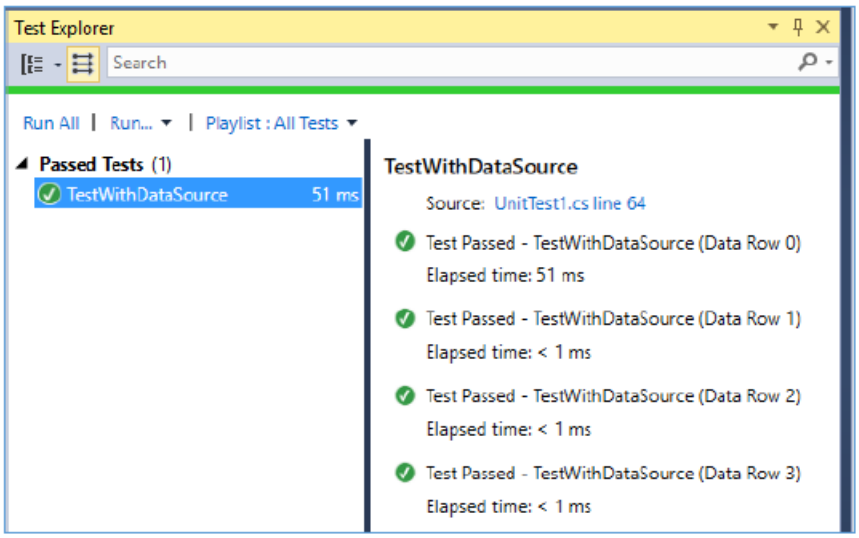
Tạo đối tượng `TestContext` trong lớp unit test như sau:

```
public TestContext TestContext { get; set; }
```

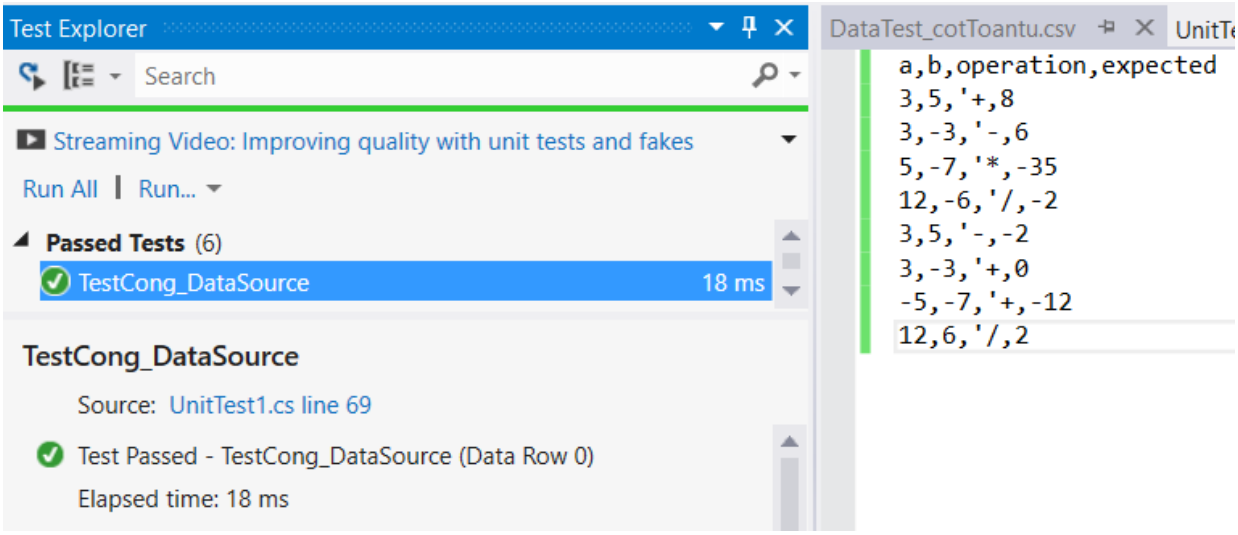
Viết test case sử dụng dữ liệu này chạy các test case như sau:

```
[DataSource("Microsoft.VisualStudio.TestTools.DataSource.CSV",
            @".\Data\TestData.csv", "TestData#csv", DataAccessMethod.Sequential)]
[TestMethod]
public void TestWithDataSource()
{
    int a = int.Parse(TestContext.DataRow[0].ToString());
    int b = int.Parse(TestContext.DataRow[1].ToString());
    int expected = int.Parse(TestContext.DataRow[2].ToString());

    Calculation c = new Calculation(a, b);
    int actual = c.Execute("+");
    Assert.AreEqual(expected, actual);
}
```

Thực thi test case trên sẽ có kết quả như sau:



Bài 1:



Bài 2:

Cho hàm tính x^n bằng đệ quy, trong đó x là số thực và n là số nguyên bất kỳ, biết x^n được tính như sau:

$$x^n = \begin{cases} 1 & nếu\ n = 0 \\ x^{n-1} \times x & nếu\ n > 0 \\ \dfrac{x^{n+1}}{x} & nếu\ n < 0 \end{cases}$$

```
static double Power(double x, int n)
{
    if (n == 0)
        return 1.0;
    else if (n > 0)
        return n * Power(x, n - 1);
    else
        return Power(x, n + 1) / x;
}
```

Viết unit test kiểm thử hàm Power với đặc tả yêu cầu như trên.

Bài 3:

Một chương trình tính giá trị đa thức tại một giá trị x nào đó. Chương trình nhập vào số nguyên n là bậc đa thức và $n + 1$ số nguyên a_i $(0 \le i \le n)$, với a_i là hệ số của x^i và giá trị biến nguyên x. Nếu người dùng nhập không đủ n + 1 hệ số cho đa thức hoặc nhập n âm thì ném ra ngoại lệ ArgumentException, với nội dung lỗi là "Invalid Data".

Viết Unit Test kiểm tra đoạn chương trình hiện thực hóa yêu cầu trên.

```
class Polynomial
{
    private int n;
    private List<int> a;

    public Polynomial(int n, List<int> a)
    {
        if (a.Count() != n + 1)
            throw new ArgumentException("Invalid Data");

        this.n = n;
        this.a = a;
    }

    public int Cal(double x)
    {
        int result = 0;
        for (int i = 0; i <= this.n; i++)
        {
            result += (int)(a[i] * Math.Pow(x, i));
        }

        return result;
    }
}
```

Bài 4:

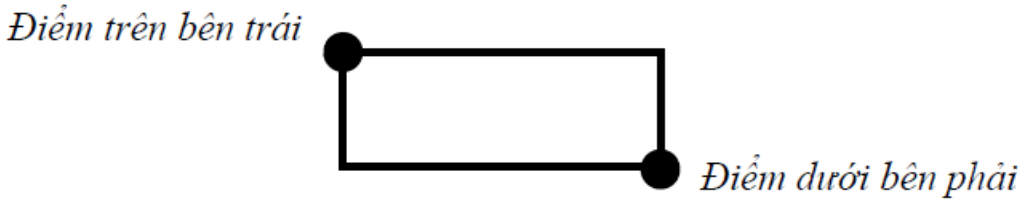
Cho chương trình chuyển đổi số nguyên dương cơ số 10 sang cơ số nguyên k bất kỳ (với $2 \leq k \leq 16$). Viết các Unit Test kiểm thử đoạn chương trình này.

```
public class Radix
{
    private int number;

    public Radix(int number)
    {
        if (number < 0)
            throw new ArgumentException("Incorrect Value");

        this.number = number;
    }

    public string ConvertDecimalToAnother(int radix = 2)
    {
        int n = this.number;

        if (radix < 2 || radix > 16)
            throw new ArgumentException("Invalid Radix");

        List<string> result = new List<string>();
        while (n > 0)
        {
            int value = n % radix;

           if (value < 10)
               result.Add(value.ToString());
           else
           {
               switch (value)
               {
                   case 10: result.Add("A"); break;
                   case 11: result.Add("B"); break;
                   case 12: result.Add("C"); break;
                   case 13: result.Add("D"); break;
                   case 14: result.Add("E"); break;
                   case 15: result.Add("F"); break;
               }
           }
           n /= radix;
        }

        result.Reverse();
        return String.Join("", result.ToArray());
    }
}
```

Bài 5:

Viết lớp `Diem` để thao tác với điểm trong không gian hai chiều bao gồm hai thuộc tính hoành độ và tung độ. Lớp `HinhChuNhat` chứa thông tin của một hình chữ

nhật, biết một hình chữ nhật được xác định bởi 2 điểm là tọa độ điểm trên bên trái và tọa độ điểm dưới bên phải:



Lớp `HinhChuNhat` có hai phương thức thực hiện các chức năng sau:

- Tính diện tích hình chữ nhật
- Kiểm tra hai hình chữ nhật có giao nhau hay không?

Viết Unit Test để kiểm thử cac chức năng của chương trinh tren.

Bài 6:

Một trung tâm gia sư cần quản l. thông tin học viên, một học viên bao gồm thông tin: m. số học viên, họ tên, quê quán, điểm của ba môn học chính. Vào cuối khoá học, trung tâm muốn tìm ra một số học viên có thành tích học tập tốt để trao học bổng khuyến khích. Một học viên được đánh giá là tốt nếu điểm trung bình ba môn học chính từ 8.0 trở lên và không có môn nào trong ba môn chính điểm dưới 5.

a) Viết chương trinh cho phép nhập danh sách học viên và xác định danh sách học viên có thể nhận học bổng.

b) Viết cac Unit Test để kiểm thử các chức năng của chương trình trên.