**Formatting Floating-Point Values**

By default, C# always displays floating-point numbers in the most concise way it can while maintaining the correct value. For example, if you declare a variable and display it as in the following statements,

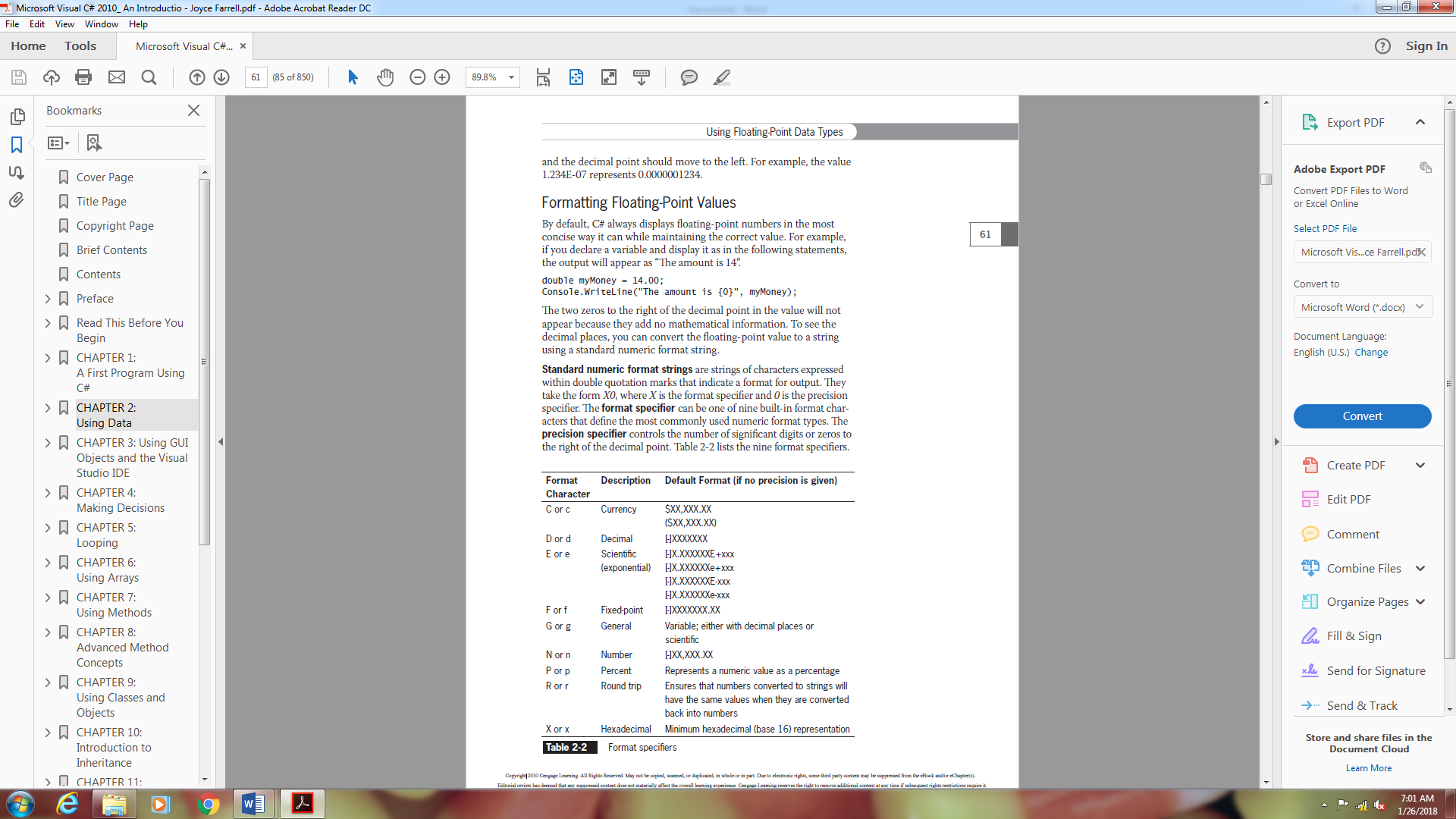
the output will appear as “The amount is 14”.

double myMoney = 14.00;

Console.WriteLine("The amount is {0}", myMoney);

The two zeros to the right of the decimal point in the value will not appear because they add no mathematical information. To see the decimal places, you can convert the floating-point value to a string using a standard numeric format string.

Standard numeric format strings are strings of characters expressed within double quotation marks that indicate a format for output. They take the form X0, where X is the format specifier and 0 is the precision specifier. The format specifier can be one of nine built-in format characters that define the most commonly used numeric format types. The precision specifier controls the number of significant digits or zeros to the right of the decimal point. The following table lists the nine format specifiers.



You can use a format specifier with the ToString() method to convert a number into a string that has the desired format. For example, you can use the F format specifier to insert a decimal point to the right of a number that does not contain digits to the right of the decimal point, followed by the number of zeros

indicated by the precision specifier. (If no precision specifier is supplied, two zeros are inserted.) For example, the first WriteLine() statement in the following code produces 123.00, and the second produces 123.000:

double someMoney = 123;

string moneyString;

moneyString = someMoney.ToString("F");

Console.WriteLine(moneyString);

moneyString = someMoney.ToString("F3");

Console.WriteLine(moneyString);

You use C as the format specifier when you want to represent a number as a currency value. Currency values appear with a dollar sign and appropriate commas as well as the desired number of decimal places, and negative values appear within parentheses. The integer you use following the C indicates the number of decimal places. If you do not provide a value for the number of decimal places, then two digits are shown after the decimal separator by default. For example, both of the following WriteLine() statements produce $456,789.00:

double moneyValue = 456789;

string conversion;

conversion = moneyValue.ToString("C");

Console.WriteLine(conversion);

conversion = moneyValue.ToString("C2");

Console.WriteLine(conversion);

Currency appears with a dollar sign and commas in the English culture. A culture is a set of rules that determines how culturally dependent values such as money and dates are formatted. You can change a program’s culture by using the CultureInfoClass. The .NET framework supports more than 200 culture settings, such as Japanese, French, Urdu, and Sanskrit.

To display a numeric value as a formatted string, you do not have to create a separate string object. You also can make the conversion in a single statement; for example, the following code displays $12,345.00:

double payAmount = 12345;

Console.WriteLine(payAmount.ToString("C2"));