ITE 365 Lab02: Reversing a number, with and without strings

Adam Lewis

May 17, 2018

Contents

1 Objectives

1. Work through the process of designing and building an app in C#.
2. Look at different ways of solving the same problem.
3. Continue to gain experience with VS and C#.

2 Background

It's helpful to work through the entire process of building an application from problem statement through complete code. In this lab, we will look at a simple math puzzle: given an integer, display the reverse of that integer. We will examine two ways of solving the problem: a solution that does not use strings and a (rather quick and dirty) solution using strings.

2.1 Problem statement

Write a program that accepts an integer number and returns the number with its digits reversed. Do this without using string variables.

3 Instructions

One never just starts writing code (anyone catch *The Princess Bride* reference?). You should first think about the problem statement and see if you can explain a solution to the problem using English and just a bit of pseudo-code.

3.1 Solution design

What makes this problem more difficult than the norm is that your Glorious Instructor (think of him as being much like a third-world dictator with delusions of evil villainy) has told you to not use strings. So we, being the canny computer scientists, know we need to apply the power of MATHEMATICS!

So, let's think about the problem... I've got to find some mathematical method of going from right to left in the number and adding the digit I find at that spot into a new number. This is where we can use the integer division and remainder operators:

```
WHILE( my number is greater than 0) DO
    placeValue <- number MOD 10;
    number <- number / 10;
    reverseNumber <- reverseNumber * 10 + placeValue;
END
```

So, we can create an app in C# that reads a number from the console, convert it to an integer, and then calls a method that implements our algorithm.

3.2 Implementing the solution

Now let's implement our solution using C#.

First thing that we need to do is to start Visual Studio.

Once you have Visual Studio started:

1. Select "New Project" or select *File>New>Project*
2. Select "Visual C#" in the templates menu
3. Select "Console Application" from the list of Project types
4. Set the project name to "ReverseApp"
5. Adjust the project location to where you want to store this project on your machine
6. Select "OK"

The IDE will grind for a few moments and then present you with a newly created project. Visual Studio works with "Solutions" and "Projects" and you can see the contents of the Solution in the "Solution Explorer" windows. By default, VS creates a C# *namespace* with a name that matches your solution name and a class named "Program" with a **main()** function. I like to rename this to something more appropriate for the project:

1. Select the file "Program.cs" in the Solution Explorer.
2. Right-click and select "Rename" from the context menu. Rename the file to something more appropriate such as "ReverseClass.cs". Note that VS will also change the name of the class in the source file.

3.3 So, let's do things in a top down fashion

That means we should start with our **main()** method. Add the following to your new class in Visual Studio:

```
public static void Main( string[] args )
{
   Console.Write( "Enter an integer (-1 to exit): " );
   int number = Convert.ToInt32( Console.ReadLine() );

   while ( number != -1 )
   {
      Console.WriteLine( "{0} reversed is {1}",
         number, ReverseDigits( number ) );

      Console.Write( "Enter an integer (-1 to exit): " );
      number = Convert.ToInt32( Console.ReadLine() );
   } // end while
} // end Main
```

Note the following:

1. The function **ReverseDigits()** is where we'll implement our algorithm. Given how it is being called, we will treat it as a class method.
2. Note that we have to read the number using the **Console.ReadLine()** method and convert what has been entered into an Integer using **Convert.ToInt32()**

3.4 Now to do the heaving lifting

So, we need to pass the inputted integer to our function and then implement our algorithm as the body of the function. We'll return back the resulting integer:

```
// display parameter number with digits reversed
public static int ReverseDigits( int number )
{
   int reverseNumber = 0; // the number in reverse order
   int placeValue; // the value at the current place

   while ( number > 0 )
   {
      placeValue = number % 10;
      number = number / 10;
      reverseNumber = reverseNumber * 10 + placeValue;
   } // end while

   return reverseNumber;
} // end method ReverseDigits
```

3.5 Build and test your application

Build your application in Visual Studio. Run your program and enter a few test cases. Set a breakpoint in **ReverseDigits()** and watch what happens to the values in the local variables as your program runs.

3.6 Programming challenge

Suppose your Glorious Instructor decides to show mercy upon you and allow you to use the string data type to solve this problem? Write a second class method named **ReverseDigitsUsingStrings()** that accepts an integer as a parameter and uses the **Convert** class and the string class methods to reverse the integer without having to do any computation. Implement your method and include it in your submission.

4 Submission instructions

Combine a copy of your source code and a screen-shot of your program into a PDF file (you can use Microsoft Word to do this) and attach it to the assignment on Blackboard.