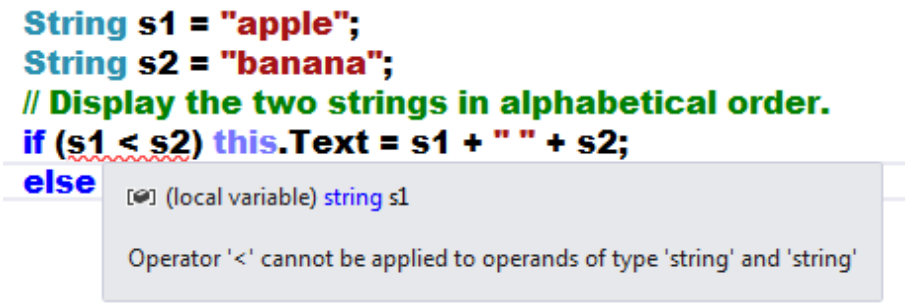
String Compare in C#

If we would like to display two string in alphabetical order, we might try this:

```
String s1 = "apple";
String s2 = "banana";
// Display the two strings in alphabetical order.
if (s1 < s2) this.Text = s1 + " " + s2;
else this.Text = s2 + " " + s1;
```

However, we get an error using the < operator as shown below:



Instead, we must use the CompareTo method:

```
if (s1.CompareTo(s2)<0 ) this.Text = s1 + " " + s2;
else this.Text = s2 + " " + s1;
```

The CompareTo method returns -1 is s1 comes before s2 (s1<s2). It returns 0 if the two strings are the same, and +1 if s1 comes after s2 (s1>s2).

We can also use String.Compare in the same way:

```
this.Text = " " +String.Compare("apple","banana"); // displays -1

this.Text = " " +String.Compare("banana","banana"); // displays 0

this.Text = " " +String.Compare("banana","apple"); // displays 1
```

The comparison is case sensitive:

```
this.Text = " " +String.Compare("APPLE","apple"); // displays 1
```

The word “Ann” comes before “Anna” alphabetically:

```
this.Text = " " +String.Compare("Ann","Anna"); // Displays -1

this.Text = " " +String.Compare("Bobby","Bob"); // Displays 1
```

If we want to compare two strings regardless of case we can, of course, use the ToUpper method in addition to CompareTo:

```
String s1 = "CAR";
String s2 = "car";
int n = (s1.ToUpper()).CompareTo(s2.ToUpper());
this.Text = " " +n; // Displays 0 meaning that the two strings that are compared are equal.
```

Strings vs. Numbers.

Strings of digits, such as id numbers, phone numbers and Social Security numbers should be declared as strings, NOT integer or double. Numeric formats should only be used for values that you would perform arithmetic operations on. While 123>35, “123” < “35”.