2.4.1 Static Formatting

Now that we've run the analysis, let's make it easier for Steve to read by adding some formatting to our table. This is the same type of formatting we did in the last module—changing font styles, adding borders, setting number formats, and so on—but we can automate formatting with VBA.

The first step to begin formatting your table is to create a new macro for the task. Following the same steps as earlier, create a new macro named `formatAllStocksAnalysisTable()`. Next, make sure the right worksheet is active in VBA. This might seem redundant because the worksheet should be active after the loop finishes, but we should always write code expecting that it will be changed in the future. Maybe we'll come back in six months and add another loop that activates a different worksheet. If we haven't specifically stated that the "All Stocks Analysis" worksheet should be active, we'll start formatting the wrong sheet. It's a good practice to explicitly activate the sheet we need before we start formatting.

Use the following code:

```
    'Formatting
    Worksheets("All Stocks Analysis").Activate
```

Visual Style Formatting

Select the header range and make the text bold by setting the `Bold` property of the `Font` property to `True`.

```
    'Formatting
    Worksheets("All Stocks Analysis").Activate
    Range("A3:C3").Font.Bold = True
```

Select the same range and add a border to the bottom edge with the following code:

`.Borders(xlEdgeBottom).LineStyle = xlContinuous`

Here's how it looks within the context of the surrounding code:

```
    'Formatting
    Worksheets("All Stocks Analysis").Activate
    Range("A3:C3").Font.Bold = True
    Range("A3:C3").Borders(xlEdgeBottom).LineStyle = xlContinuous
```

This is a small example of how to format cells (go ahead and run your macro to test it!). There are many sources of information about Excel formatting. Think about where you could find them. If you need some help, here is a hint.

SHOW HINT

Let's try to put a few more styles into place.

SKILL DRILL

Using the properties listed for the `Font` object in the **official VBA documentation (https://docs.microsoft.com/en-us/office/vba/api/excel.font%28object%29)**, change three of the font properties of the header.

Numeric Formatting

Range objects have a `NumberFormat` property that we can change by setting it equal to a special string of characters. The `NumberFormat` string for separating digits with commas is `"#,##0"`, which we'll use for the volume. To make a single-digit percentage for the return, we'll use the format `"0.0%"`.

NOTE

> In VBA number formats, the number sign (`#`) is used if you want to display only a significant digit (i.e., any digit that isn't a trailing zero). If you want to display a trailing zero, use the numerical symbol zero (`0`) instead. Adding a percent sign will move the decimal point two digits to the right.

Now let's write some code that will format the numeric cells in your sheet. Try it out on your own first, then check your solution against the one in the hint below.

SHOW HINT

Here we've used one **digit of precision** in the return amount. This implies that the percentage change is accurate up to a tenth of a percent. Excel will round the value when displaying it, but it won't affect the actual value stored in the cell.

NOTE

> Are 1.0 and 1.00 the same? Yes and no. They both store the same exact value: 1. However, these numbers imply different things.
>
> "1.0" implies accuracy up to the tenths place; the true value could be anywhere between 0.95 and 1.05. "1.00" implies that the value is accurate up to the hundredths place, so its implied true value lives in a much narrower range: 0.995 to 1.005.
>
> In scientific settings, the number of **digits of precision** is the total number of significant digits, but in financial and engineering applications, the digits of precision refer to how many digits are written after the decimal point.

SKILL DRILL

1. Add one more digit of precision to the return percentage.
2. Format the price column to use a dollar sign and two significant digits after the decimal point.

Automatically Fit Column Widths

We can change the width of a column to auto-fit the data using the `AutoFit` property, which is convenient. However, we'll need to select the column, not just the range. Luckily, that's pretty convenient too: it's just the

function `Columns`. Select column B with `Columns("B")`, then run the macro and check its result.

```
    'Formatting
    Worksheets("All Stocks Analysis").Activate
    Range("A3:C3").Font.FontStyle = "Bold"
    Range("A3:C3").Borders(xlEdgeBottom).LineStyle = xlContinuous
    Range("B4:B15").NumberFormat = "#,##0"
    Range("C4:C15").NumberFormat = "0.0%"
    Columns("B").AutoFit
```

Now that we've made static formatting updates, let's use this tool with the programming tools we already know to create conditional formatting.