Eric Brown
ISYS 520 Final Project

Executive Summary:

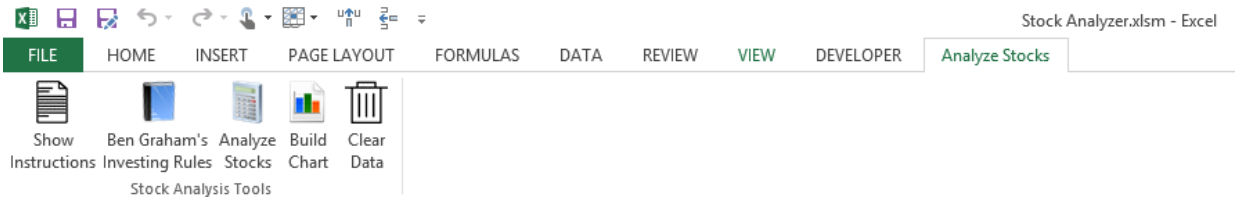
Kulakom LLC is a private investment firm formed in 2013. It has two members, and together they decide how to invest the LLC funds. This LLC was set up as an investment vehicle, but is not the sole business of either of its members. Consequently, neither member has a lot of time to spend analyzing and evaluating stocks in which to invest. The members want to actively invest in individual stocks (as opposed to investing in managed funds). So far, they have relied on intuition and gut feelings. They have made a profit so far, but only a meager one. They are looking to make smarter stock picks for their investments. In summary, Kulakom is facing a business problem of not having a good way to analyze and evaluate stocks. This leads to suboptimal investment decisions as most decisions are made off of gut feelings currently.

I have created a tool to help Kulakom evaluate stocks. Stocks are evaluated using eight tests based on Benjamin Graham's stock investing rules. The user enters a stock ticker symbol, and the program retrieves the relevant financial information from the internet, performs tests, scores the stock, and makes an investment recommendation. The user can also compare the stocks with an automated chart tool. The tool facilitates more informed decision-making for Kulakom as it makes its stock investing decisions.

Implementation:

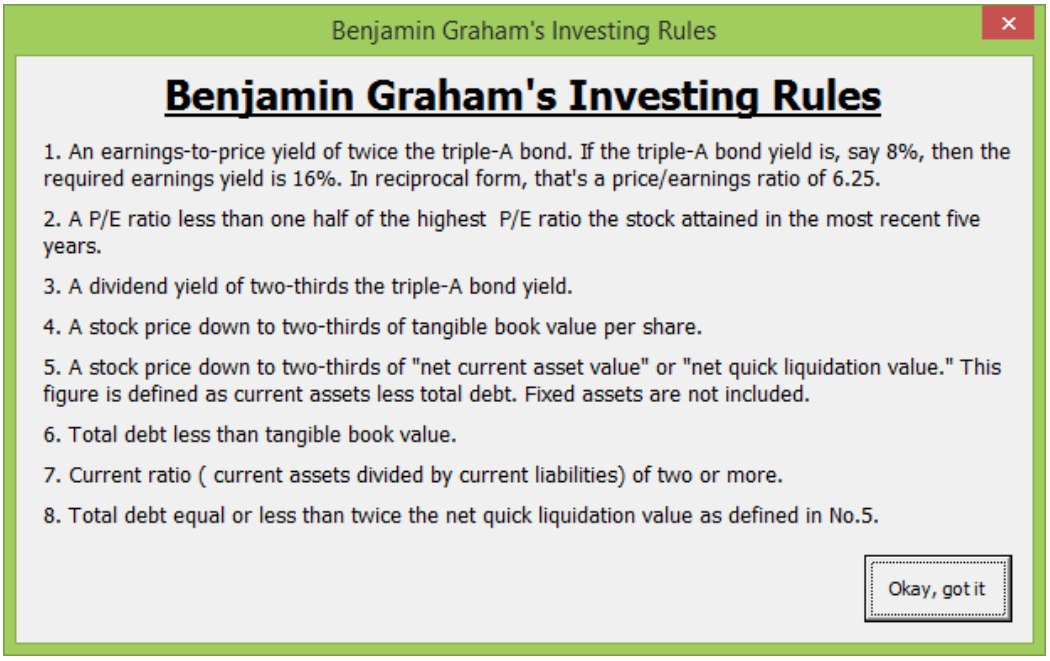
When the workbook is opened, the user is welcomed by a form that explains the instructions for using this stock analysis tool:

This instructions form explains how to use the tools. All of the tools are located on a ribbon I added to the Workbook called “Analyze Stocks.” The ribbon includes a group called “Stock Analysis Tools” with 5 buttons the user may click to perform various functions. The following discussion explains the purpose and function of each button.

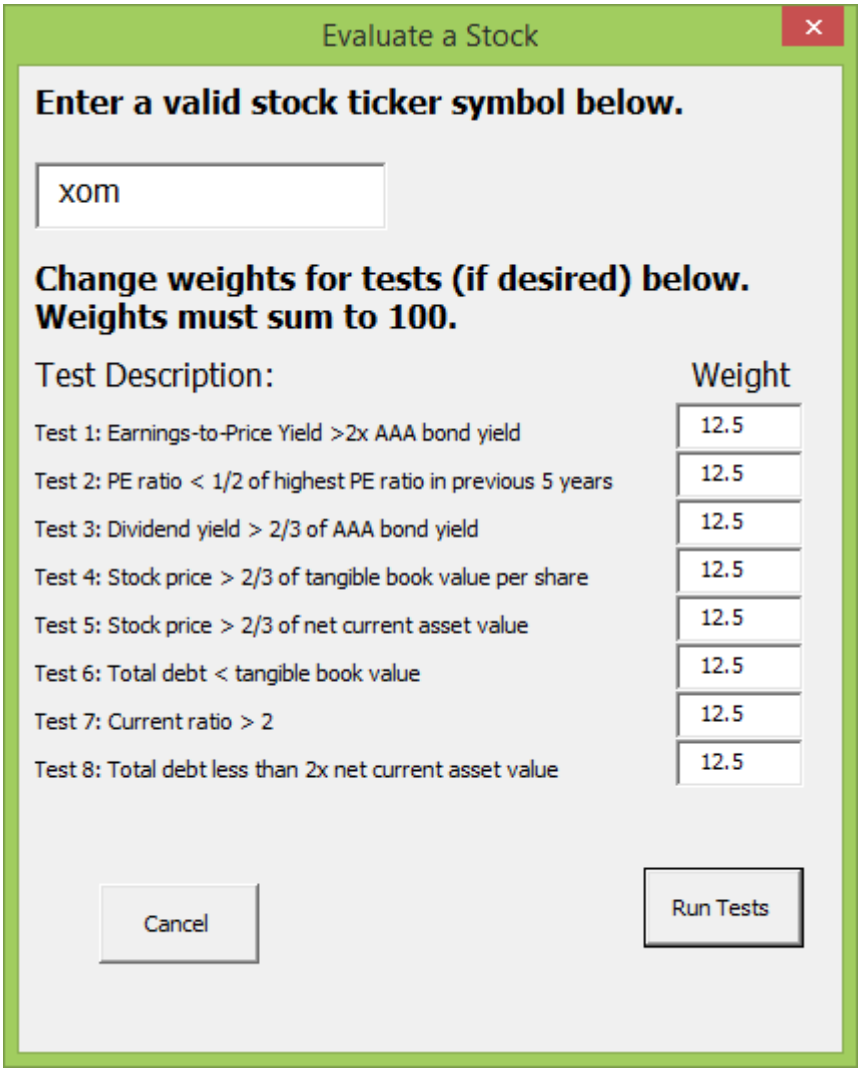


Show Instructions **–** When the user clicks this button, the instructions form is shown so the user can read the instructions and explanations of the tools if desired.

Ben Graham’s Investing Rules **–** When the user clicks this button a form is displayed that explains in detail the investment rules used in this analysis. These rules are based on Benjamin Graham’s investment rules that many investors, like Warren Buffett, have used.



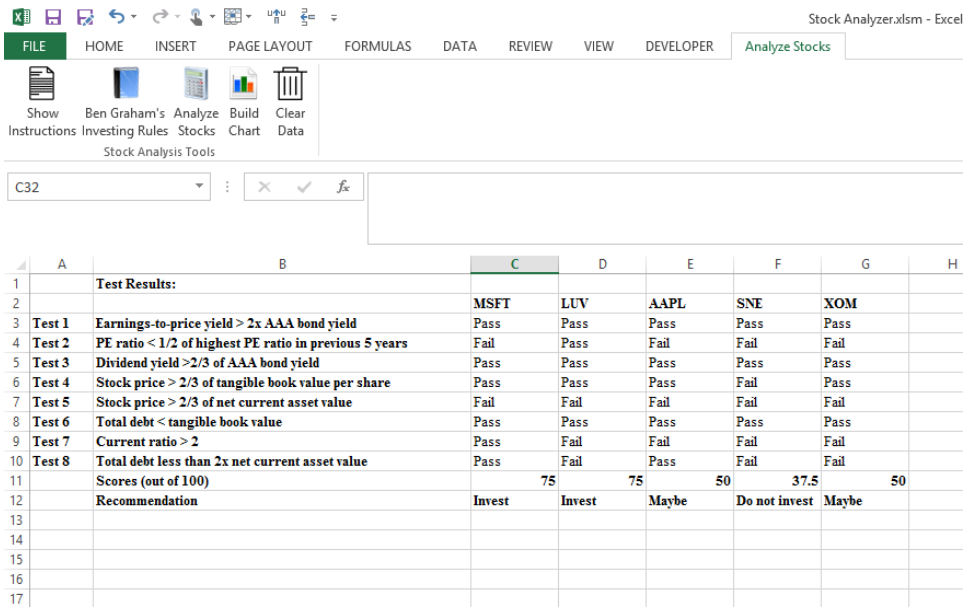
Analyze Stocks – When the user clicks this button, a form displays in order to collect information from the user (see screenshot below). The user enters a stock ticker symbol that he or she wishes to analyze. Additionally the user can specify weights for each of the eight tests. (The form defaults to equal weights for the 8 tests). This is useful if certain tests are not important to the analysis the user is performing. For example, many stock investors do not care about dividend yields (Test 3) because more modern companies choose not to distribute dividends, opting to re-invest income in the company. This was less common when Benjamin Graham initially formed his investment rules.



When the user clicks the "Cancel" button, the form is hidden and no program executes. When the user hits the "Run Tests" button several things happen:

- The program performs several web queries to obtain all the necessary financial information about bonds and the company's financial results used in the various tests.
- The program performs various calculations to get to the ratios and figures used in the tests.
- The program calculates all the test results to determine if the stock passes or fails for each test.
- The program calculates a score for the stock based on which tests it passes and fails and the respective weights specified by the user.
- The program determines an investment recommendation for the stock as "Invest," "Maybe", or "Don't invest."
- The program prints the results on the "Test Results" tab in the workbook.

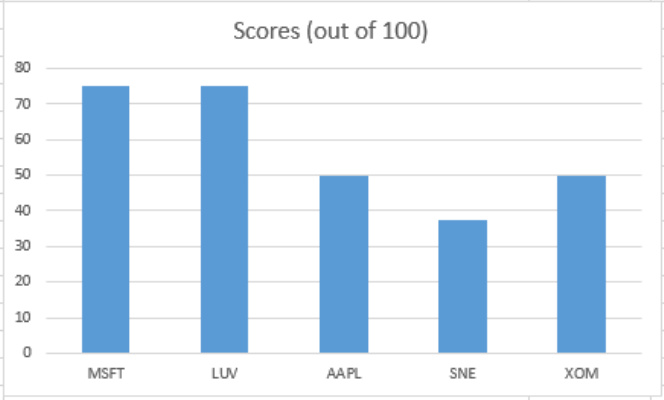
The workbook is set up with short descriptions of each test and spots for the test results to be printed. The test results for each stock are printed in the furthest left column, and older results are shifted right. Below is a screenshot of what the test results might look like after a user had tested several stocks:



	A	B	C	D	E	F	G	H
1		**Test Results:**						
2			**MSFT**	**LUV**	**AAPL**	**SNE**	**XOM**	
3	**Test 1**	**Earnings-to-price yield > 2x AAA bond yield**	Pass	Pass	Pass	Pass	Pass	
4	**Test 2**	**PE ratio < 1/2 of highest PE ratio in previous 5 years**	Fail	Pass	Fail	Fail	Fail	
5	**Test 3**	**Dividend yield >2/3 of AAA bond yield**	Pass	Pass	Pass	Pass	Pass	
6	**Test 4**	**Stock price > 2/3 of tangible book value per share**	Pass	Pass	Pass	Fail	Pass	
7	**Test 5**	**Stock price > 2/3 of net current asset value**	Fail	Fail	Fail	Fail	Fail	
8	**Test 6**	**Total debt < tangible book value**	Pass	Pass	Pass	Pass	Pass	
9	**Test 7**	**Current ratio > 2**	Pass	Fail	Fail	Fail	Fail	
10	**Test 8**	**Total debt less than 2x net current asset value**	Pass	Fail	Pass	Fail	Fail	
11		**Scores (out of 100)**	**75**	**75**	**50**	**37.5**	**50**	
12		**Recommendation**	**Invest**	**Invest**	**Maybe**	**Do not invest**	**Maybe**	
13								
14								
15								
16								
17								

Build Chart **–** If the user wants to see a visual comparison of the stocks, he or she can build a chart by hitting the "Build Chart" button. When the user clicks this button, the program will automatically build a chart of the scores for each of the stocks that are on the test results page. The chart is displayed below the test results. If there is an old chart there, the program will delete the old chart before inserting the new one. This screenshot shows the chart that is created when the user clicks this button.

	A	B	C	D	E	F	G
1		**Test Results:**					
2			**MSFT**	**LUV**	**AAPL**	**SNE**	**XOM**
3	**Test 1**	**Earnings-to-price yield > 2x AAA bond yield**	Pass	Pass	Pass	Pass	Pass
4	**Test 2**	**PE ratio < 1/2 of highest PE ratio in previous 5 years**	Fail	Pass	Fail	Fail	Fail
5	**Test 3**	**Dividend yield >2/3 of AAA bond yield**	Pass	Pass	Pass	Pass	Pass
6	**Test 4**	**Stock price > 2/3 of tangible book value per share**	Pass	Pass	Pass	Fail	Pass
7	**Test 5**	**Stock price > 2/3 of net current asset value**	Fail	Fail	Fail	Fail	Fail
8	**Test 6**	**Total debt < tangible book value**	Pass	Pass	Pass	Pass	Pass
9	**Test 7**	**Current ratio > 2**	Pass	Fail	Fail	Fail	Fail
10	**Test 8**	**Total debt less than 2x net current asset value**	Pass	Fail	Pass	Fail	Fail
11		**Scores (out of 100)**	**75**	**75**	**50**	**37.5**	**50**
12		**Recommendation**	**Invest**	**Invest**	**Maybe**	**Do not invest**	**Maybe**
13							



***Clear Data* –** When the user hits the "Clear Data" button, all test results and charts on the "Test Results" tab of the workbook are deleted. The test labels will remain for any future tests.

Learning and Conceptual Difficulties:

In this project a learned a lot about how to parse through data to make values comparable across different companies. For example I had to parse out superscripts or subscripts that were next to financial numbers for certain stocks, but not next to related numbers on other stocks. Or when my program gets the shares of stock outstanding for a company, the value it returns may be 1.5 M or 2.2 B or 307 K. I learned how to use loops to parse through a string to get the relevant numbers and also determine an appropriate multiplier based on the letter (i.e. – a "k" translated to a multiplier of 1,000).

I also learned a lot about user forms and using information input by a user to run a program. I learned how to give the user flexibility in determining inputs, but still constraining them so my program worked and the results would make sense. I learned this when I gave the user the option to assign weights to the different tests based on which tests were most relevant to him or her. By checking the values entered by the user before the program runs, I make sure that the program works and that the results returned will still be meaningful.

I encountered a few difficulties in writing this project. I struggled with validating the stock ticker symbol input by the user. I obtained a list of stock ticker symbols that had been traded in the past five years. I tried loading this data in an array that the program would check against the value entered by the user before executing. However this list was over 12,000 ticker symbols long, and loading the values into an array took a long time, checking the user's input against the array took a long time, and it was just really inefficient in general. I also tried pasting the data into another sheet and automating using the vlookup function of Excel – but this approach had the same problems with speed and efficiency.

I also had some difficulty with obtaining good data for each company. My biggest problem is that sometimes the necessary data is not available on the internet. My program uses web queries to obtain financial data from Yahoo! Finance and Reuters. I noticed that sometimes the balance sheet data for a company is not available for certain companies. For example, I successfully used my program with Google's stock ticker, but the very next week when I tested it again, my program failed. I found out that the balance sheet data that was on Yahoo Finance! was no longer there. I assume this has something to do the timing of a company releasing annual and quarterly financial statements. Eventually I decided to create a catch-all error handler that resets the program in the event that something goes wrong. The most common error would be a user entering an incorrect stock ticker symbol. If this happens, the web query will not find the information it would normally find, and the program will display an error message and the program will reset so the user can try again. The instructions sheet explains the most common types of errors. I decided building a catch-all error handler was easier and more efficient that trying to check each stock ticker for validity or explain each possible error that could happen throughout the entire program.

If I were to develop the program further I would try to build better error handling. Perhaps I could automate an auto-correct for common tickers, or figure out a better way to check the validity of the user's input. Additionally, I could possibly automate the web query to try other financial websites if certain financial information is unavailable on Yahoo Finance! or Reutres.

Assistance

I received no assistance in writing the code for this program. A classmate helped me obtain a list of ticker symbols of stocks, but I ended up not including it in the project. I used internet searches to familiarize myself with certain programming concepts I implemented in my project.