Cody Marsh

Dr. Allen ISys 520

Final Project Write-Up

Amazon.com Competitive Scraping Macro Documentation

Introduction

This documentation will describe the coding for the macro that scrapes Amazon.com for pricing and shipping information. The purpose of this macro is to allow a user to import into the spreadsheet a list of Amazon Stock Identification Numbers (ASINs), and retrieve Amazon's current pricing and shipping costs, while also telling the user the name of the item. I have learned a lot while developing this macro and much of that learning was by trial and error. The format of this documentation will therefore follow the order in which the final macro is written. I will describe the purpose of each section of the macro line by line, going from top to bottom. I realize that this is not the way that the macro is spliced by the computer, but I believe it to be the most user friendly method for me to provide documentation. After a description of the functionality of the macro, I will comment on the things that I have learned while doing this project, specifically the problems that I had to overcome.

In the workbook available for testing are 50 ASINs that I grabbed from our database and used to test the macro after I deemed it ready to turn in. This is the version that I gave to my boss at Overstock, and they were thrilled with the results. Apparently our current scraping software is about 60% dependable, and takes a long time to run, even if only for a small number of ASINs. The benefits of this macro are that it is correct more often, works especially well for small amounts of ASINs (say less than 50), and can be deployed at any time of the day without doing causing problem to the company bandwidth.

Of this list of 50 ASINs, there are only 4 where the macro fails to find out all the information a user would find if he were looking at the site itself. These 4 are all due to the flash technology Amazon uses for a limited number of products. The other 46 either bring back an Amazon total, or inform us the item is out of stock, the webpage is down, or that we need to estimate the shipping revenue. It is a 92% accurate for these particular ASINs. My boss was very well pleased!

The Macro

Set-up work

- The first thing I do is declare some public string variables. These variables are used in a couple different sub procedures and I needed them to retain their values throughout the module.
- The worksheet allows the user a few different options, each option tied to a button. The first sub procedure allows the user to scrape the Amazon information starting from whatever point in the list he chooses, and continuing to the end of the list of ASINs.
- The next sub procedure specifies some of the details of the main button I plan on being used, which is the button that allows for the scraping of the entire list of ASINs. It specifies a uniform

row height for each row, and column widths for each of the different columns used to store information. This is purely to make the page look a little nicer. It also formats the positioning of the text in each cell. This was important because many items have long names, and I wanted the first part of the name to be visible in each cell, regardless of modifications the user might have made to the page. It then activates the cell with the first ASIN and loops through the main macro until the end of the list.

The main sub procedure

- I declare various string and integer variables, which I use throughout the sub procedure for loops and temporary holders of the strings that I need to use to display the requested price, shipping, and name information. I then turn off the screen updating so that the process is easier on the eyes of the user.
- I grab the current ASIN that I want to gain the information for, which I use to bring in the main webpage for that ASIN.
- This brings in the first possibility of an error. Occasionally the webpage related to that ASIN does not exist, either because it was incorrectly recorded in the database or Amazon has taken the webpage down. I call a sub procedure whose purpose is to see if the webpage exists. If the webpage does not exist, the macro returns to the main page and writes onto the sheet that the webpage does not exist and readies itself to grab the information for the next ASIN. I choose to display the message rather than a text box because the macro is designed to run for many ASINs at once and I do not want anything halting to process in the middle.
- Once the page has been successfully imported, I call a sub procedure that gets the name of the item, and that sub procedure writes the item's name in the appropriate column on the Amazon Data sheet.
- I then call a sub procedure that identifies if the item is currently available. If the item is not in stock through Amazon or any other seller, then Amazon does not display any pricing or shipping information for that ASIN. I then display a message that informs the user that the item is unavailable and exit the sub procedure, leaving the code ready for the next ASIN.
- Once the page is imported and the item is identified as being available, the code then identifies if the item is not sold by Amazon. Although Amazon has huge inventory and sells a lot of its own merchandise, it is still a popular avenue for other sellers to list their products. When Amazon is only acting as an exchange between sellers and buyers, the pricing and shipping information does not consistently show up on the main product page. If this is the case, the macro then goes back to the web and imports the product details page, which specifies who the people are that are selling the product. The page by default sorts itself from lowest price plus shipping amount, so the macro brings that page in and finds that price and shipping amount and writes it on the Amazon Data sheet.
- The next part of the sub procedure deals with a different style of page on Amazon where they announce the product as being available at external websites, but not through Amazon. The macro identifies these types of pages and finds their pricing information and writes it on the Amazon Data sheet

- Now that the macro has dealt with all the non-standard pages, and whether the webpage exists and the item is available, then the macro finds the current Amazon price on the imported page. The hiccup that happens here is that sometimes it is necessary to add the item to the cart to see a special price. If that is the case, then a note will be written to the user letting him know that that is the case.
- The next step was to get the shipping information for an item on a standard imported webpage. This information is generally in one of two places, depending on if the item ships for free or has an actual shipping charge. The macro first checks to see if the item ships for free. If it does it writes a $0 shipping charge and moves on.
- The next possibility that it checks for is if the item is eligible for free shipping on orders over $25. For these kinds of items, which by themselves cost less than $25, the shipping information is unavailable until the customer puts in the location that he wants it shipped to. At Overstock we have a fairly crude method for estimating a shipping cost, but it requires some information from the Overstock database. I have currently instructed the macro to write "estimate shipping" into the shipping column so that the user will know that that item needs to have its shipping charge estimated in order to accurately find its total price.
- The next part of the macro is important, as Overstock sells only new items, and sometimes Amazon will sell used items. While they always advertise the new items first, if they only have inventory in used items, then the macro will grab that price. This part of the macro identifies the item as being a used item, and while still displaying a price, notes in the shipping column that the item is used.
- Occasionally the item's main webpage does not display shipping information. If that is the case then the macro imports the item's product details page and searches for shipping information on that page. If it finds it, it grabs it and writes it onto the Amazon Data sheet, and if there is still no shipping information it writes the string "not sure" into the shipping column.
- It then turns on screen updating so the user can see the work that it has done, and ends the sub procedure

Explaining the private subs that follow the main sub procedure

- The first private sub totals up pricing and shipping to populate the Amazon total column. If both pricing and shipping are numbers, it adds them together and displays the total price. If the shipping column has some sort of text, then it displays the price and adds the word "plus" and then displays the text in the shipping column.
- The next private sub is the one that gets the item's name. It searches in the originally imported page to find the item name and write it onto the Amazon Data sheet.
- The next private sub is the one that checks if the webpage exists. Again this is not the chronological order used, but the order in which the code is written. If the webpage query brings back an error, it writes in a note that the item is unavailable and exits the main sub procedure.
- The next private sub is the one used to check for item availability. If the item is unavailable it writes that information in on the price column and exits the sub procedure.

Problems I ran into

- During the documentation step, I briefly mentioned some of the problems that I ran into while writing this macro. The first thing that I had to learn about was error handling. This macro does a lot of checking to see whether certain conditions exist, and the only way I know of to move on when conditions don't exist is through manipulation of the error handler. I googled VBA error handling and looked at different sites' examples using it, and then used a lot of trial and error to get the code to do what I thought I was telling it to do. I had to be very careful that the 'on error resume next' statement was only in the places where I knew an error would be unavoidable, and was ready to handle those errors that might come. As soon as I was outside of that error area, I turned the default handler back on so that I would be messaged if areas of the code were having errors when I did not expect them too. The reason this final project has been carefully checked for error handling problems is because I had a fairly big issue where I could not get a certain type of webpage to bring me accurate shipping information and fault was in an error that was occurring that I did not know was happening because I had not yet reset the error handler. It was through line by line running of the code that I was able to realize the problem.
- The biggest headache was the non-uniformity of how Amazon presents its products. This led to many exceptions, and is the reason why I have to check so many conditions before even trying to find a price and shipping. Even then, the shipping was often not even imported on the main page, or not even advertised at all. This is why there are a few different text messages in the shipping column when a large column of ASINs is analyzed.
- This led me to a creative solution, as I was trying to figure out how to calculate a total price when I had some text values in the shipping column. I found that even if price and shipping were declared as strings, that if they were numerical values I could still add them together by adding the cell value once I had written the price into the cell. Also, since the variable type was string, I could easily append them together as strings, so that the total price would look like "$8.88 plus Estimate Shipping" which lets the user know what he would need to do to find an accurate total price.
- On a few items, Amazon has its pricing done by option, rather than the item itself, and uses Adobe Flash or something similar to interact with the user. This caused a big problem because the price would not display unless one of the options had been chosen, and also Overstock only compares pricing on items, not on options. In those cases (I only ran into it once, and I have run this macro on around 300 ASINs or so up to this point) I simply message that the price cannot be found.
- More commonly I ran into troubles with Adobe Flash when it came to finding shipping values. In these instances the shipping would not import with the page (as it was not displayed until an option was clicked on) and in those cases I messaged a "not sure" in the shipping column.
- Occasionally the name comes back as something different. Amazon's webpage brings in the name in a few different styles, and as I have gone through, I have corrected for the most common things that confuse the macro, but that section is written in such a way that there are almost limitless ways to trick it. It does not happen much, and on a random sample of 50 ASINs I

would expect this error to occur somewhere either once or even zero times, but I acknowledge that it is a possibility.

Those are the biggest problems that I remember running into as I wrote this macro. It was a great learning experience. I did a lot of searching online, and also incorporated many of the statements that we learned in class. It took patience and creativity to figure out how to solve these problems, but it was great feeling to see it work in the end.