VBA Final Project: User's Manual

Sean Baenziger

April 2016

Introduction:

BYU'S MBA program is designed to help aspiring young professionals of capability and faith to accomplish their career goals. In this vein, the graduate school developed a mentoring program in which second year students mentor first year students to search for an internship and otherwise navigate MBA life. This spreadsheet is a new spreadsheet designed to track statistics and progress for this mentoring program.

Executive Summary:

There were several tasks that needed to be accomplished in order to complete this spreadsheet:

1. Weekly data created in a CSV file though learning suite needed to be imported and placed in the appropriate "Weekly" and "Cumulative" sections. This import process needed to be robust enough to handle common glitches such as students who do not report every week, and students who report twice. An algorithm was developed to accomplish these tasks

2. A simple, ready to use, entry method to input data needed development for first year mentees, "climbers" and second year mentors, "sherpas". Several userforms were developed in order to input this data easily.

3. A reporting system needed to be developed in which sherpas were notified of their climbers' progress and could meet to discuss areas of emphasis. An email message notification system was developed in order to meet this need, and a userform was created in order to select which MBA group would receive the email messages.

4. A clear contents option was needed in order to reset the sheet after every year. A sub-procedure as developed in order delete the entered data and resent other items to their null values.

5. A custom ribbon needed to be added so that users could easily access the functionality of the .xlsm spreadsheet. The ribbon was titled "macros" and contained buttons for the five key functionality areas.

Implementation:

My solution was designed first and foremost to make managing the Sherpa program easy and simple. The five component solutions are as follows:

First, to take learning suite CSV data and input it into the appropriate places in the spreadsheet there I developed several steps. Originally I had designed to allow Excel to connect to learning suite, download the file automatically, and then

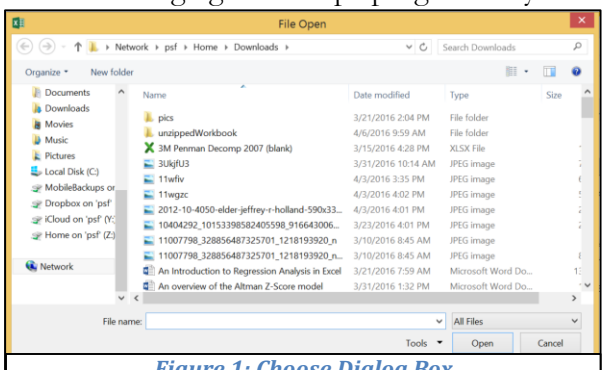

Figure 1: Choose Dialog Box

upload the data, but upon conferring with those for whom this spreadsheet was designed, they stated that they preferred a semi-automated process in which they could choose the files to upload and when to upload them. As such, my first step in the process was to create a "choose file" file dialog box, in which the user could identify the specified .CSV file to open and upload the data. The user selects the file that he or she would like to open, and the VBA coder places the data in both the weekly sections and additively in the cumulative sections of the tabs for "All Climbers" as well as the track specific climber reports. This task was accomplished using several algorithms. First, the program uses a searching algorithm to identify which of the climbers in the "All Climbers" tab is first, and then searches the climber data from the newly opened spreadsheet for that climber's name. Next, the program then identifies the climber, and pulls information from the appropriate row next to the climber's name and transfers it to the appropriate tabs in the "SHERPA TRACKING SHEET" spreadsheet, individually for the weekly reports and additively for the cumulative reports. The algorithm then deletes the student's name in the KPI data report if there is a duplicate so that if the student added two week's data in one week, the appropriate data would only be added once. If the student does not report that week, then the program skips the student and moves to the next student using a basic loop function. Thus, all the data is placed in the appropriate places.

FINANCE TRACK
WEEK 1

	TEAM #	TEAM LEAD	INT'L	HRS WORKED	# JOB APPS	# INFO INTERVIEWS	# MOCK INTERVIEWS	# REAL INTERVIEWS	# OFFERS	GRADE
Student 1		Sean Baenziger	YES	6	3	4	1	0	0	
Student 2				7	2	1	0	0	0	
Student 3				2	4	0	0	0	0	
Student 4				5	0	0	1	0	0	
Student 5			YES	2	0	1	0	0	0	
Student 6										
Student 7				15	0	3	0	0	0	
Student 8				1	1	0	0	0	0	
Student 9				1	0	1	0	0	0	
Student 10				0	0	0	0	0	0	
Student 11				1.83	0	0	0	0	1	

Figure 2: Track Specific Climber Report Output

Common problems that could result in required debugging are:

- If the format of the CSV file changes (BYU's career services offices has discussed such a possibility, but to my knowledge nothing has occurred yet);
- If someone changed the structure of the Sherpa Tracking Spreadsheet by adjusting rows or columns

Second, to enter in the appropriate data on the appropriate tabs at the beginning of the semester, several userforms were created that would assign climbers to their appropriate tracks (majors), teams, and Sherpas. Another userform was created in order to input Sherpa data as well. The student userform has 6 inputs on it, two text and four combo boxes. The text inputs are the name and the interest, while the combo boxes contain information for the track, the Sherpa (loaded using an array from the list of Sherpas), the option to select whether or not the student is an international student, and the team number. This data then allows the climber to be placed on three spreadsheets, the "All Climbers", the "[Track] Report", and the "[Track] Team Leads" with the appropriate data. If a climber is input onto a team that is already full, then a message box will appear alerting the user that another team is needed.

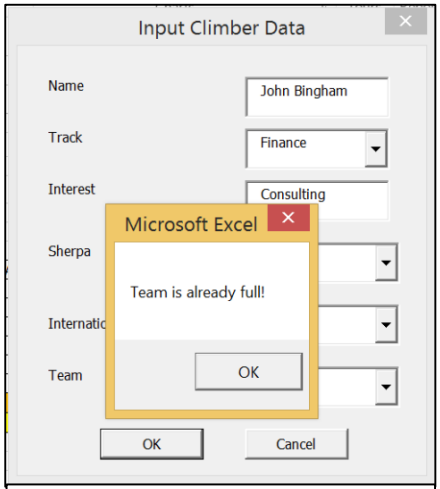

Figure 3: Userform and "Team is Already Full" Message

The Sherpa userform allows the user to enter in the appropriate data using three combo boxes and three text boxes. The combo boxes are all preloaded with the appropriate data from track, cell carrier, and team number, and the text input allows the user to enter the requisite information to later identify the Sherpa and their assignment. The algorithm then auto-populates the data in the appropriate tabs for later usage using loop functions and end(xldown) functions.

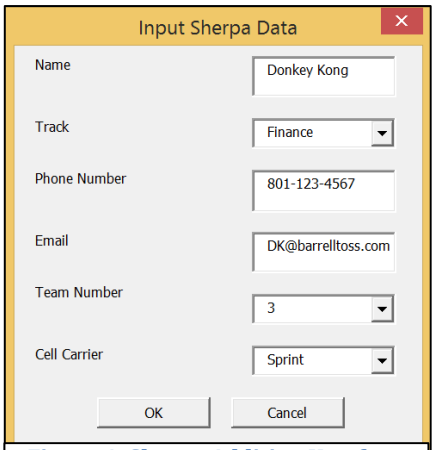

Figure 4: Sherpa Addition Userform

Common problems that could result in required debugging are:

- If the format of the Sherpa Tracking Sheet is altered. While several precautions were taken such as using end(xldown) commands and others that adjust to changing columns and rows, if the underlying format of the spreadsheet is adjusted, the algorithms will need to be appropriately adjusted to achieve functionality.

Third, an email message notification system was developed that allows the spreadsheet to automatically communicate updates to Sherpas about the progression of their climbers. The notification system is built off of the programming learned in VBA class, and creates a send email function that is designed to run within a sub procedure. The sub procedure creates a template, adds the sherpa's name, the climber's name, and the relevant statistics imported from the CSV file to the template, analyzes the areas in which the climber is performing below the average relative to his or her peers, and then uses this analysis to compile a list of areas of possible focus for upcoming Sherpa meetings as a final addition to the template message. The sub then calls the function which adds this template to an email message, links up to an already created gmail address, and sends an email with the report to the appropriate Sherpa.

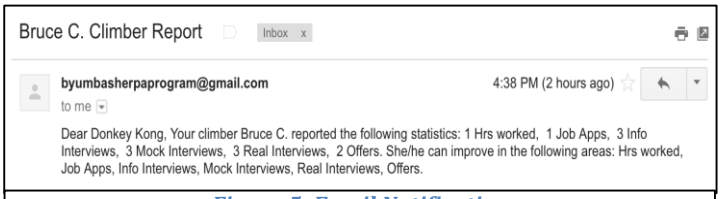

Figure 5: Email Notification

Common problems that could result in required debugging are:

- If gmail adjusts its algorithms or formatting the function will need to be appropriately adjusted
- Again, if the format of the spreadsheet is altered
- If the password is changed on the gmail account

Fourth, the clear contents function simply used arrays and nested loops to cycle through the sheets with the data, delete the appropriate data, and re-input the null values. Common debugging issues include the same aforementioned issue of whether or not the columns and rows of the spreadsheet had been altered.

Finally, the spreadsheet ribbon was altered using the Custom UI tool and the instruction provided

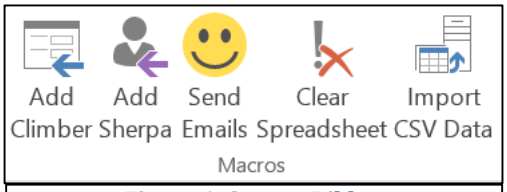

Figure 6: Custom Ribbon

in class. The techniques used were the essential code modifications as well as the button seeking. There are very few debugging needs with this facet of the project, unless someone were to access the VBA code and change the actual macros (and their names) that were already written.

Learning Moments:

This project encountered several difficulties that provided learning experiences for me. First of all, while I had already completed several data adjustment and input assignments for this class in the past, this spreadsheet had several idiosyncratic wrinkles in both the team assignment format, and the data adjustments from the CSV file into multiple tabs. Specific challenges included:

- The aforementioned challenges of identifying the climbers who had entered in data twice or not at all.
- The input of association with the appropriate teams
- Identification of glitches within the data and the appropriate corrections

While several iterations were necessary in order to place the data in the proper format and locations, several techniques were developed in order to adjust the algorithm to return appropriate results. First, with the duplicate names in the data for climbers who entered in their information twice, a portion of the algorithm was created that would delete the name on the data input spreadsheet and skip the row variable addition function so that the finding algorithm would find the first iteration of the data the first time, then move to the second iteration afterwards. By this process, both sets of data would be entered into the cumulative sections, while only the most recent would appear in the weekly reports section. The concerns with the appropriate team entry given the design of the spreadsheet was a relatively simple adjustment using an a multiplier on a row variable and then end(xldown) function in order to ascertain which was the next open spot on the team was. A further layer of code was used to identify whether or not the team was full using the text of the bottom line of the team box if the end(xldown) function returned the end of a full data box (the text was "TRACK AVE", and was used as the baseline to determine whether or not the team was full). Finally, in order to ascertain which data to put in and which to leave out, simple isnumeric functions were utilized in order to identify appropriate data and place them in the appropriate locations on the tracking sheets. While these elements were all elements that I had used successfully in other projects the application of these concepts in this specific project broadened my understanding of their usage and how I could best apply them in the future.

Additionally, one further learning moment came in the skipping function in which if a student's name is not found on the KPI spreadsheet using an "On Error GoTo" function. The function worked the first time through the code, but on the second pass through using a loop it kept giving me an error. I tried adjusting the location of the "On Error GoTo:" statement, but the same result kept occurring. Finally, I turned to the all-powerful google for help. I discovered that when using an "On Error GoTo" function, the code somehow or another places the program into an error handling mode and a "Resume" statement

needed to be inserted into the code to take the program out of the mode for the next pass through of a loop. It was an interesting learning I had not anticipated.

The email message notification system also had several learning moments that broadened my understanding of VBA code. First of all, there were several errors that I needed to work through in order to enable functionality. First, during the course of the project I set up a gmail account that would enable the sending of email messages, but shortly thereafter ran into an error in which the gmail account had the security features enabled which did not permit VBA access. I then conducted a google search about how to address the problem and identified toggles on my account settings that enabled me to send email messages regarding climber updates. This was an educational moments in terms of cyber security and systems understanding.

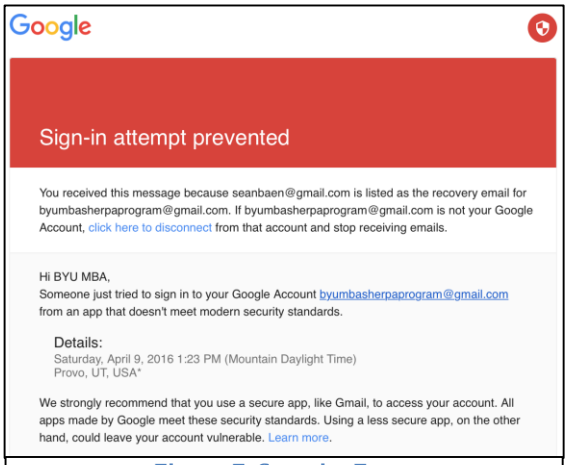

Figure 7: Security Error

Finally, one shortcoming of my project was that I wanted to include a feedback system in which emails would be sent out using google sheets in which Sherpas could enter in a brief 2 sentence summary of their climbers' progress over the course of the past week. The summary would then be stored to a google doc and downloaded and input into an appropriate location on the spreadsheet. The efforts I made to complete this was primarily to talk to Gove to determine the viability of such an approach. He told me it would be difficult and time consuming, but possible. I then conducted a little bit of my own research on the feasibility of the project. After creating an educated guess about how much time the project would already consumer, as well as the available resources I had, I decided against going further with this section of the project. I did clear it with Gove as well to make sure my project was still within the appropriate scope.

Significant Assistance:

I had significant assistance from Gove, both TA's Cameron and Nathan, Google and the Holy Ghost (seriously, sometimes the only thing that'll get you through an annoying VBA bug is prayer… ☺). Gove was a saint and let me bug him in his office about the email glitch, both TA's frequently had me in their office hours as I worked through some of the loops and userform functions, and google returned to me several of the find functions and so on.

Conclusion:

I did not incorporate a separate code write up as part of my report as I tried to identify specific concepts and methodologies used to formulate this code in the various parts of my write up. Overall, this has been an educational project that has allowed me to give back to the University that has given me so much. Thank you for your time.