Executive Summary

As an officer in the National Guard I am responsible for the physical readiness of the soldiers under my charge. Soldiers undergo a semi-annual physical fitness exam to determine their readiness. There two areas tested: Strength, which is tested on push-ups and sit-ups, and cardiovascular endurance, which is tested on a 2-mile run. Because Guardsmen are "citizen soldiers" there is little way to ensure that soldiers are exercising consistently. As a result, too many soldiers perform poorly on physical fitness exams.

My project is a fitness tracker that will track activity in these two areas, strength and cardiovascular endurance. The program sends out an email with a link to a google form every week to all of my soldiers. It links to the responses on google sheets and pulls in data every time the excel project is opened or anytime the user chooses to refresh the dashboard. Finally, there is a dashboard that displays the data for the user (me) allowing me to see workout and physical fitness testing scores among the groups of soldiers that I am responsible for. Ultimately, I will be able to have data to support discipline among my soldiers in their personal physical readiness.

How it Works

My program is broken down into three main functions:

1. A resetting of the program. If I need to make changes to the roster of soldiers, where they land in the organization, it will mean a change of how the data is consolidated. This clears the sheets and information then rebuilds the information based on the roster I have input. I don't expect this to happen very often as I rarely need to change the organizational structure at this level in my leadership.

2. An automatic emailer. This sends out a simple email to the soldiers with a link to the google form each Sunday. The form is pre-filled to make the task easier for the soldiers. Rather than put in a zero on each event they did not do, all they have to do is select who they are, what group they are in, and the time spent on activities that they completed during the week. I have a prefilled date that is populated by a current time cell function so that I get the correct date (the date it was sent out) for each submission regardless of how soon or late the respondent finally submits the form.

3. The main function. This function connects to the google sheet of responses and imports those responses to the excel program. Then it goes through and ensures that all responses have been logged with the correct soldier and aggregates the data. The dashboard is a set of graphs that display the main information I want from the main function. The function is composed of many

other functions. The main function simply scrolls through the list of responses after it has been updated and then executes other functions as needed. Once the list of responses becomes empty the main function terminates. The following is a description of what this main function does chronologically.

- **Functions to set public variables and an array**. There are two functions executed before anything else happens. One establishes a set of variables so that I have less to have to write into the other functions used. The second sets an array of the worksheets I use so that I can use for loops to perform a series of activities on each worksheet.

- **Determine if it is a new week.** Each week new numbers are aggregated and displayed. Therefore, the next step is to determine if it is a new week or not. If it is a new week then there is a general shifting of data and resetting of daily report data. This sets the program up to import new data from the most recent week. The new week is determined by matching the date now against the date that the program should update to a new week. If it is not a new week then all updating activities will just change data that have changed since last update without any major changes in data location or reporting dates.

- **Update data per soldier.** This calls a series of functions.

    i. The first two functions loop through worksheets to find a match of group and soldier and define where data should be imported for that soldier.

    ii. The next function imports data for the week

    iii. A next series of functions import and aggregate the weekly, monthly, and yearly totals. For all soldiers in a group.

    iv. The last aggregates physical fitness testing data per group

        1. In here there are functions to match fitness performance against the official scoring matrix to determine soldier physical fitness scores.

        2. Another set of functions list who the high performers are, who needs improvement, and who have failed their physical fitness testing. Doing this will help me identify and take actions to improve physical readiness among my soldiers.

- The dashboard charts automatically update when information is input and do not need a macro to do so.

Discussion of Difficulties

Importing data

I found it easy to establish a simple refresh function, but I was having trouble getting the macro to pause and wait for the import to complete before running the rest of the main function. As a result my program was running functions and calculating data before it was completely updated. I spent time looking online and only found information about sleep and wait commands which either would make the import wait or wouldn't do anything at all. It was in class that Professor Allen mentioned the backgroundquery property. After that I was able to find multiple examples of how to set up the import and was able to get it to run smoothly.

Aggregating and designing data

The process of moving data from one sheet to another is relatively easy. The process of aggregating that data was also relatively easy. The actual process was somewhat time consuming because I have little experience coding. It was the design and layout of that information that was the most difficult to tackle. I had to determine how I would display the difference between current week information, last week, current month, last month, and cumulative yearly data.

Ultimately, I planned for this display to be background information I could access as I needed, but the most important part of the information was how it would ultimately result in charts and displays in my dashboard. In order to code I had to determine where I thought data would end up. If I was going to compile twelve months, fifty-two weeks, or three hundred sixty-five days of data per soldier then how would I lay out the spreadsheet so that I had room for each soldier's information?

I still don't believe that I have the most optimal layout. I'm still not certain what other data I would want to display or how I might want to interact with the spreadsheet. But the foundation I have developed would make it easy to add or adjust layout to create a more robust program.

Looping and passing data

In order to create more efficiency in my program I knew that the best method would be to do only two loops. One to loop through the imported data and one to loop through the various spreadsheets in order to determine where to put that information.

In order to accomplish that efficiency it would be necessary to pass through worksheet names, row, and column information. I was able to figure this out for most programs, but got a little lost over the cumulative number of functions I created. I made iterations on functions to make things run more smoothly which resulted in cluttered functions that were no longer necessary. I also created copies of functions that performed the same function, but with different results. I know that they could be optimized into one function, but chose to complete the project and optimize later.

As a result I feel that my code is clunky. I certainly need to work on naming conventions, on design and flow, and on optimizing functions and public variables so that I can reduce the overall code of programs. This would also reduce the amount of memory my code would use to run.

Google

I found it relatively easy to find solutions to my programming needs by simply typing in a description of what I was trying to do with the word “vba” next to it. Along with seeing examples of how to fix my problem I would also find better ways to optimize other parts of my program.