Dallon Jarman
Assignment 3
CYB136

For this assignment, I created a mock dynamic storage program. The purpose of this is to dynamically allocate memory to the user's name and to use constructors to store the names. With the program, the user is allowed to add multiple people to the program and add characteristics about the user. From there the user may add another person, delete a person, create another characteristic, remove a characteristic, or find a person. I ran into some very weird issues while writing this code. The first being is almost every 5 minutes, cin and cout will refuse to work and every instance of cin and cout will throw an error that wasn't very helpful. I was able to fix this by removing and adding the '>' in #include <stdio.h> and it would automatically fix the issue but only until I recompiled. Another issue I kept running into was when I used getline. The first couple of times it's used in the program it will work flawlessly. Then after it will refuse to work no matter what I tried. It refused to grab user input which makes the program not function correctly when adding space to your input. This can become a huge issue when the user inputs a two-word characteristic but in the delete characteristic, getline doesn't work making it impossible for the user to delete a two-word characteristic. To go more in-depth into what the program does. The program will start by asking the user how many people they would like to add. Then it will prompt the user to add all the people. From there it will ask the user how many characteristics they will like to add such as hair color, gender, age, etc. Once the initial setup is finished, then it will provide the user with 7 inputs. The first is, "add a name". This will allow you to add another user to add another person to the program. The second input is, "delete a name". This is useful if you need to remove a person or change the name of a person and all their values at once. The next input is, "add a category". This is useful when you need to collect another form of data on another person such as collecting their address or social security number. The next input is "delete a category". This is handy when the government may come after you for collecting sensitive data or when data is no longer needed. The next option is, "print the list". This will print a fancy list of all the users and their assigned data within a category. Finally is, the "search name". if you have thousands of users stored. It may be hard to find a certain user. This will make it much easier to find a user and all their relevant data.

The broken code may not seem to have any significant issues. It may even be hard to figure out if it is broken. But in the function, "deleteName" and "addName". They are missing a key component. They are missing the delete[] names. I believe this can cause a lot of issues in the future. Without deleting the old array, the computer will continue to store the memory that was allocated causing it to eat up all your memory preventing you to store anything else. This can be bad when you are trying to add anything to the program. If you have "deleted" a bunch of users the computer will continue to store and this can cause a memory leak. This can cause drastic consequences such as too much of the available memory may become allocated and all or part of the system or device stopping working correctly, the application failing, or the system slowing down vastly due to thrashing.

The language feature this misuses is pointers. When you create a pointer it pointers to a data value. When you forget to clear that pointer, it will continue to point to the data value indefinitely. This won't cause any red flags in the program when you try to run it but it will slowly eat away at your memory till you have none left whatsoever and it will be incredibly tricky for an end user to figure out what is causing their device to slow down.

This program will work properly under normal use and has enough error checking to prevent the user from entering an incorrect value. To fix the data leak it is a very very simple one-line code that is deleted []. That is all you need. It is so small that developers may overlook it and not even think twice about it.

Many companies need ways to store user data such as birthdates, gender, race, military status, and much more. Every business needs a way to store information about users. The issue with every business needing software like this bugs will be found. This will cause issues for thousands of users and many users to panic if your code isn't perfected. Even with perfect code. Someone will input something that will cause all the code to break.
All bugs are hard to find and detect. The majority of bugs are found from rigorous testing and are usually never found by just looking over the code. When you create your code it's even harder to find bugs that exist within it. You know to look for certain things but there are so many things you would never think to look for. Having people test your code before releasing it to the public is the best way to debug code. Having some of your friends just go through the code and try their best to break it is a great way to debug.
Finally, flaws in code are tricky to find but with enough trial and error and plenty of testing they can come to the light, and then as a developer it is their job to remove the bug and not create other bugs in the process.