## Monday, 9/5, 8pm-10pm

* Began project
* Worked on research about making GUIs in Java, what libraries I needed, and Java Class Structure

## 

## Tuesday, 9/6, 8pm-9pm

* Started officially coding
* Ran into problems very quickly concerning how the GUI would allow a program window to shutdown the entire software
* Solved this by looking at java API and what exactly System.exit(0) does to a program

## 

## Wednesday, 9/7, 8pm-10pm

* Worked on setting up Buttons for all of the Essential Programs
* Ran into problems again with ActionListeners and ActionPreformed Methods
* Problem not solved before bed

## 

## Thursday, 9/8, 8pm-10pm

* Solved the problem, took way too long, may or may not cause problems down the road, but that's for another day

# Week 2, 9/5 - 9/11

**Summary**: My plan for week two was to teach myself how to program a GUI in Java, which is a lot harder than I thought. Initially there are many libraries to choose from. The choice is very important though, as many libraries do not have the methods that are pretty much required for this project. The thought of creating my own methods did occur, however I do cherish my own sanity, and I have a schedule to keep. This week was a very big learning experience for me, just like how I think most weeks for me will be. Coding in java for me this far has really only been solving problems using loops and recursion and scanners. This is way outside of my usual wheelhouse, which in a way is very exciting and terrifiying. Thankfully, my goal was accomplished, and I am pretty proud of how the project is going so far.

## Monday, 9/12, 9pm-11pm

* Added all of the buttons needed for the text encryption and decryption program, and got all of the ActionListeners and the ActionPreformed methods to work with the buttons.
* Started work on the actual Caesar Cypher algorithm method that is called with the ActionPreformed method.
* Ran in to a little problem with the algorithm, mostly just the logistics

## Wednesday, 9/14, 8pm-11pm

* Soley worked on getting the Text Encryption part of the first program down. The concept of a Caesar Cypher is very easy to understand. You use the key and shift the alphabet that many units and spit back out the inputted phrases with the shifted alphabet. However, coding this is a fairly complex problem to understand. That being said, I spent almost all of my time today creating something that didn't actually work. I had a big problem with being able to go through an alphabet at separate indexes so you can move the alphabet according to the key given.

## Friday, 9/16, 9pm-11pm

* Finally got the Encryption algorithm to work, Thank Fucking God. I had to scrap my previous algorithm because it was almost completely wrong. I figured out a major problem, where my program would freak out if you typed in different cases. I employed a quick fix where the code automatically puts the input text into lowercase to avoid this problem. The one problem with this is it can't output the correct casing when giving the outputted text.

# Week 3, 9/12 - 9/18

**Summary**: This week was WAAAAYYYY more efficient than the first week of my project. There was less looking at GUI libraries and Java API and more actual coding, the stuff that I'm good at. The GUI is all complete concerning the first program, and it looks more beautiful now that the GUI is a little more fleshed out. I spent probably more time than I should have working on the algorithm of the Caesar Cypher Encryption, but I'm super proud of how everything is going. There's just something about clicking the “Encrypt'' button and having it take the text in the “Plaintext'' box and seeing it come back as random letters is so satisfying. The one problem with all of this is: I still haven't done the Decrypting part. This is a problem for next week.

## Monday, 9/19, 8pm-10pm

* Started work on the decrypting part of the Caesar Cypher program, was relatively easy as it uses the same concepts as the encrypting part of the program. I have yet to format the GUI with all the corresponding buttons that are needed to make the program actually function, but that will be for tomorrow

## Wednesday, 9/21, 9pm-10pm

* Was only able to work for a little bit tonight, but was able to implement all of the text boxes and label boxes for the decrypt portion to look as good as the encrypt portion. The next thing to work on is the hashing algorithm, which I wasn't able to get to today, but is a priority for the next day that I work on the project.

## Thursday, 9/22, 8pm-10pm

* Today was very unproductive. In the middle of the day I got inspiration to try to remake the GUI of my program to look more sleek and well designed. I was going to try to implement a tab system for the GUI, where there's one pop-up window and you click on one of the tabs in the upper left hand corner to change to a different screen that shows a different program. However, this did not work at all. I flopped around looking for fixes for problems that I made for myself by trying to switch my GUI. I ended up scrapping the idea for the Tabbed GUI and resorting back to my previous program so I could make some progress on the project. The next time I work on the project is going to be towards the hashing algorithm, 100 percent.

# Week 4, 9/19 - 9/25

**Summary**: My plan for week 3 was to get my first program down in the GUI and coded so it would function, and I have successfully done that. The second part of my plan for this week was to get a decent start on research and brainstorming for my second program so that I can start adding GUI elements onto my program in the beginning of week 4. This would lead into me programming a good chunk of the second program the rest of the week, so only a little bit of cleanup would be left for week 5. I am a little behind thanks to my tangent I took on Thursday, but I should be able to catch some small catch up to get back on track.

## Monday, 9/26, 8pm-10pm

* First order of business today was brainstorming ideas for how I wanted the GUI to look for my second program. I got some inspiration from my first program on how it should be laid out: plain and simple, very minimal places to get stuck. Once that was done I went ahead and coded the GUI for the second program, which wasn't too painstaking of a process. The last thing I did today was start researching how to even create a hashing algorithm, turns out it's kind of hard. Codewise it might not be, but there's a lot of mental gymnastics and thinking you have to do to make it actually function. Obviously you have to make sure you can't reverse the hash (that would defeat the whole purpose of a hash), and you have to minimize collisions (still don't know how i'm going to do that). I'm going to leave the brainstorming for the code tomorrow.

## Tuesday, 9/27, 8pm-9pm

* Took the first few tries at making a hashing algorithm, and it honestly wasn't that hard. The one problem is that I don't really know how many collisions it has. When I looked back at making a hashing algorithm, I didn't notice that java has specific methods that make it super easy to make your own hashing algorithm. Like I said before, I don't know how many collisions my algorithm has, I pretty much just smashed a bunch of methods together and got something that looks good. I'm not really satisfied with the outcome, it's not very impressive or cool, actually pretty skill barren.

## Thursday, 9/29, 8pm-9:30pm

* I decided I'm going to change the hashing algorithm to something different. Instead, I'm going to make a program that can accurately calculate a file's hash value for different hashing algorithms (Specifically MD5, SHA-256, SHA-512, and crc32). You will then be able to check the file's hash vs the hash of another file, or even be able to check if the sum of a file represents the sum of a file you've downloaded from the internet. I find this way more interesting than what I was doing before, and also way more useful. This way ill be able to grasp on how actually useful hashing algorithms work and how to implement them usefully.

# Week 5, 9/26 - 10/2

**Summary**: So things took a little detour this week, but that's fine, things happened. I ended up finishing the second program, and then hating it and making plans to redo it. But the good news is that I like the new idea way more than the old idea, and it will be more useful to actual real world applications, thank god. That being said, I think I understand hashing algorithms way more than I did before thanks to my little failed program. Before I thought it was just a bunch of random string generators, but obviously that doesn't make any sense. Now I know that making a real hashing algorithm is crazy fucking hard, ill actually link a very interesting article that goes through the [SHA-2](https://blog.boot.dev/cryptography/how-sha-2-works-step-by-step-sha-256/) algorithm. Safe to say that my algorithm sucked compared to a real one, I'm glad I'm doing it differently now. Lots of work to do next week to catch up.

## Monday, 10/3, 8pm-9pm

* Got to work remaking the GUI for the second program. This time it's more time intensive than with the previous version. I need to be able to allow the java program to access a person's computer files, and allow them to open up their File Explorer. Now, the caveat to this is that only Windows users can use this program, as this might cause some bugs when using another OS’s version of a File Explorer. I've been watching a youtube channel that is an amazing help when learning how to code GUIs in java, and he gives some really good tips on how to form class structure, it's called [Bro Code](https://www.youtube.com/channel/UC4SVo0Ue36XCfOyb5Lh1viQ). I think this guy has to be a genius. Anyways, I was able to get a really good start on the GUI part and I'm frankly excited to work on the project tomorrow.

## Wednesday, 10/5, 8pm-10pm

* HOLY SHIT THIS IS SO COOL. Ok, so I ended up getting the File Explorer to work thanks to a little bit of research. Pretty much half of the time I spend on my project is looking at the Java APIs of random ass libraries that do some pretty incredible things. SO, now when you click a button, it opens up your File Explorer and lets you choose a file from your computer to select, and once you do that it gives you the path to that file in your computer. Pretty much the entire program relies on that file path to compute the hash of your file. It works off the principle of opening that file by passing the file path to a MessageDigester that will take in the entire file and compute the hash using whatever is read in from the file, PRETTY FUCKING COOL! Seriously hyped to keep working on this project.

## Thursday, 10/6, 8pm-10pm

* Ok, got to work on the real meat of the new program. It's actually really easy to compute the hashes of a lot of the major hashing algorithms out there, because there is thankfully a method for all of them. If I had to actually code them I wouldn't be doing this program, that would be way too hard. I figured out through some research that my program would be super memory heavy, meaning: depending on how big the file you're trying to read into the digester, you could be using quite a bit of memory from the computer, and that's a big problem. Thankfully what i've figured out in my haze of strep throat, is that I can feed the digester a single line of the file at a time, and once it's done with that line, hashes it and moves onto the next line, which is hashed and the concatenated with the line before and calculated again to make a new hash. Using this method, only a small bit of memory will need to be used at a time because storing a hash and a line of a file is way more efficient than storing the whole file at a time. I actually feel like a genius right now. I'm pretty tired and sick, so i'm definitely not going to class, but ill work on my project when I wake up, just so I can say I finished the second program.

## Friday, 10/7, 12pm-12:30pm

* I feel like shit right now. The strep throat meds are not helping right now. I hooked up all the GUI buttons and shit to the second program, and it works, so there. I haven't tested it against bigger files yet, but I'll try that later, not really trying to crash my computer right now. I'm really trying to be excited about this, but I really don't feel good. Good night.

# Week 6, 10/5 - 10/9

**Summary**: I'm writing this Friday night, so I still don't feel good. In short, I finished the second program, but I'm decently off schedule. I need to make some changes to my plans or else I might not get what I want done. Short but sweet.

## Monday, 10/10, 5pm-6:30pm, 7pm-9pm

* So I still don't feel good, but I'm a little better. I have made an incredible revelation. I figured I might as well start the keylogger now because that's the thing that I wanted to do the most out of the entire program. Getting started was a little hard. I eventually figured out how to hide the program in the background while the keylogger is doing its proverbial magic. Now the cool part, JNativeHook. This is an entire API surrounding typing, it might as well be a keylogger's dream. I'll admit, this might be a little cheaty, but it pretty much gives you everything you need to make a fairly competent keylogger. I have to catch up with other classes this week because of my strep throat, so this is really the only day I can work on the project, but I'm fairly confident in everything so far. The GUI is pretty simple, and I've set the program to dump all of the text into a file once the esc key is pressed. I'll try to work on the program this weekend if I can.

# Week 7 Summary, 10/10 - 10/16

**Summary**: Ok, so I had a little bought with pink eye for a little while, and man did it kick my ass. I couldn't wear my contacts so I was pretty much walking around blind for the entire time. I couldn't really see what was happening on my computer screen, so I scrapped working on my project at all on the weekend. Pretty fun times actually. I'm behind, but surprisingly not too behind. I made up some ground by working pretty hard on my project on Monday. The plan for next week is to finish the keylogger, but that's a pretty vague goal. I'm going to need to do some more research on keyloggers, and what's reasonable for me to complete in my project. I don't want to go completely overboard on the keylogger and drop the ball on everything else. Until monday, see you later.

## Monday, 10/17, 9pm-11pm

* So first things first, I did some research and some testing. I needed to make sure that my program would be hidden in the background while it was doing its logging business. Keyloggers shouldnt be noticeable, obviously. I wanted to make sure that my program would recognize my key inputs even when I had “vanished” the window. Turns out, it does. This is going a little easier than expected, but I still have to set up actions for EVERY KEY ON A KEYBOARD, except for the esc key, which is the safe key. Obviously this won't be very hard, but man I'm really not looking forward to it, it'll be extremely tedious.

## Wednesday, 10/19, 8pm-11pm

* Ok, today I found something out. I was going on about this all wrong from the start. Instead of adding loggers for every single key on a keyboard (like ActionListeners for Buttons), I CAN JUST ADD A LOGGER TO A SCREEN. JNativeHook has an object called Logger. With JNativeHook, you can set the whole computer screen as the global screen, then you can put a NativeKeyListener to that screen, and then you can slap on the Logger to that Listener, which will allow the collection of everything that is typed. This shit is absolutely crazy. I'm starting to realize what kind of insane things you can do with coding, you can literally do anything you want. There are still some logistics that I need to get around, like how the hell I'm going to deal with the space bar. It sucks, because I can't really just tell the computer to skip a line and then keep writing on that same line, store the info and then slap it into a text document. If it was that easy I would be done by now. I definitely feel like I'm dealing with things that I don't really fully understand, and with the next part that's coming up I really need to understand how this whole thing is going to work. Looks like some research is in order when I work on this next.

Week 8 Summary, 10/17 - 10/23

**Summary**: This week was a big week for me since I now have the means to create the famous keylogger that I have really wanted to make. I'm hoping that I don't run into too many problems with the keylogger, as I'm using JNI (Java Native Interface) to make this keylogger work. JNI provides an interface for java to “communicate” with other coding languages, so java can do some crazy things that wouldn't normally be possible in strictly java. Since java only has control over what happens in the Java Virtual Machine, utilizing JNI allows java to call and to be called by other native languages, like C, C#, C++ and Assembly. I found a few articles that helped me understand JNI a little conceptually and in actuality, so I'll post them [here](https://www.ibm.com/docs/en/sdk-java-technology/8?topic=jni-overview) and [here](https://www3.ntu.edu.sg/home/ehchua/programming/java/JavaNativeInterface.html). The important thing to note is I don't know the ins and outs of how all of this stuff works which is a scary thing as solving problems could be quite hard to deal with, but I'm hoping that I can navigate my way through this keylogger with minimal issues. Here's to making a ton of progress next week!

## Monday, 10/24, 8pm-9pm

* OK, today there was a problem that I probably should have assumed would happen. Since I have never used anything outside of the native java library, ive never had to import a library before, and turns out im either really dumb, or importing a library is really hard. I had to follow a tutorial because it wasn't plainly simple how to import the library, but it wasn't working like how the tutorial said it would. I spent a good hour trying to import the library to no avail. The problem that I'm seeing is that I'm downloading the .jav file for the library I want, and then I'm adding it to my classpath, but it says that it has no idea what I'm trying to import in my Keylogger class. I think I'm going to take a break on this until tomorrow to see if maybe I'll think of something that I was doing wrong. I'm just too confused by the whole thing, I mean I haven't even been able to see how I'm going to do the keylogger. Hasta Manana.

## 

## Tuesday, 10/25, 8pm-10pm

* ACTUAL PROGRESS WAS MADE TODAY! So to start things off I spent a good hour fixing my import problem with the JNativeHook library. Turns out my problem wasn’t with the .jav file not being in the right classpath, it was because I was using the wrong import statements. Since I was importing a library from a github page, the whole file structure was different, so my import statement in my class was wrong. After changing that, IT ACTUALLY WORKED. So since I was able to import the right library, I got to work on actually programming the keylogger, and damn is it confusing. I sifted through the github page, and was able to find a decent amount of documentation that helped me at least begin to understand how the different methods and components worked together. I'll post it [here](https://github.com/kwhat/jnativehook). JNativeHook is actually incredibly impressive, as it allows you to track ALL human inputs on a computer. That means JNativeHook can track your mouse coordinates, your mouse wheel coordinates, your mouse clicks and what you click on and where you clicked, and your keyboard strokes. And the thing is, that's not even all it does, it's honestly kind of crazy what you can do with this library. Back to the issue at hand, after looking at some documentation it became somewhat clear to me what I had to do, and it seemed quite simple. Since I was only working on logging keystrokes, I didn't even have to focus on the other parts of the library, and there were only a few required methods to make a keylogger function, pretty slick. Now that I know what I'm doing a little more, I feel pretty comfortable leaving the actual coding for tomorrow. I've got other homework to do.

## Thursday, 10/27, 9pm-10pm

* I'm quite annoyed currently, and don't worry I will explain why. Today was about me just wetting my feet a little with JNativeHook, and seeing what I could and couldn't do. I realized somewhat early on, that I needed to make some workarounds to how to implement my keylogger. The problem that i have run into, is calling multiple methods that make multiple windows appear on my screen. To explain a little, when using the GlobalScreen method in JNH (JNativeHook), you have to pass it a parameter, that parameter being an instance of a Keylogger. The way the Keylogger works is you have to create the keylogger through the GlobalScreen, as that's how JHN logs human interaction with devices. The way that my program works is that clicking the “KasperLogger” button on the home page activates an ActionListener that then automatically calls the main method of my keylogWindow, which then calls the keylogWindow constructor to make the GUI window and all of the GUI attributes with it. You can kind of start to see a problem forming. I cannot use both my keylogWindow constructor and GlobalScreen.addNativeKeyListener(new KeyLogger()) at the same time, as it just makes two “KasperLogger” windows pop up, and I can't just pass that problem off as a feature. I was hoping I could get some decent coding progress in, but I guess not. I have a plan on how to fix this, but it's late and i have other shit that's due tomorrow, so i'll work on this some time this weekend.

## Saturday, 10/29, 10pm-11pm

* That's right, look who's here on a Saturday! So I enacted my plan, and it's damn wonky but it's also damn good too. What I did was I took all of my keylogger code, and I shoved it into a new method, and I put this KEY() method in a new class called Keylogger. I then added a START button to my keylogWindow class that when pressed, would activate an ActionListener that would call the KEY constructor, which would then create a new instance of the Keylogger class, which would then act as my keylogger, pretty cool right? It looks a little janky, but that's ok with me because it works. That's all for today, I'm tired after the game, so good night. More to come on Monday, SO STAY TUNED!

Week 9 Summary, 10/24 - 10/30

**Summary**: Oh man this week was a rollercoaster of emotions. This week was pretty much all about me absorbing info about how JNI and JNH work. I can honestly say that I'm a smarter man now that I've done this research. So far, my keylogger is not in working order like i wanted it to be, but that's fine, i still have time to make it work. The one thing I am worried about is running into too many problems. I've already had a run in with some annoying problems, but I worked past them, so hopefully there are no more problems along the way, and I can stride across the finish line with grace (this is a joke btw, i know this won't happen). All in all, a lot of quality work was done this week, and I'm super proud of it all. If I really think about it, of course the keylogger is going to be the hardest and take the longest to implement, because it's the coolest. A little down the line I also need to start thinking of what would be the next best program to implement into my program because there are some doozies left to complete. Next week, I'm hoping to actually have a working keylogger so I can absolutely blow people's minds when I show off my program. Until next week, Bye!

## Monday, 10/31, 9pm-11pm

* New week, new progress baby! So the first thing I wanted to do was actually make the keylogger function. So far the only thing my keylogger does is open up a logged window that is waiting for keystrokes to happen. What I ended up doing was just making a few print statements that would print things out to the console, fairly simple. I realized pretty quick that “NativeKeyEvent.getKeyText(e.getKeyCode()” quite literally got the key that was pressed, meaning if you pressed the enter key, you would get “Enter” printed out to the console, which isn't exactly what i want. I implemented quite a few “if” and “if else” statements to help print out the actual character representation for some a decent chunk of the keys, so when typing the console would actually somewhat represent actual lines that someone would type. Once I found out that worked pretty well, I went on to figuring out a way to make text files using java and writing to those text files. After I got that down and made a few test files and typed text into them from java, I went straight into implementing it into my keylogger. This is where I discovered some issues that I can't really explain. My thought process when coming into the keylogger was to implement a safe key which would stop the keylogger so it didn't have to go on forever until I terminated the program. I picked the “Escape” key, as people don't usually hit that key very often and it would be safe from accidental clicking by an unsuspicious person. Currently I'm thinking that somehow the “Escape” key is messing up my code. I've tried a few troubleshooting things and so far I cant get my program to write anything into my newly created text file, it's always blank. I'm a little tired, and i dont have the patience to work on this right now, so I'm calling it an early bed time tonight. Maybe i'll dream of a way to fix this, and my Capstone will be perfect and I'll get an A in this class, maybe. Until then, Bye.

## Wednesday, 10/2, 8pm-9pm

* Ok wow this is actually an issue. So today I looked back at creating files and writing to them in java, and I actually made a new project and practiced just like Dan always tells me to do, and it works perfectly. The concept and the reality of creating and writing to files is actually really fucking easy, its like two or three lines of code. However, when I go back to my keylogger, IT DOESN'T WORK LIKE IT'S SUPPOSED TO. I'm starting to think java is failing me at this point. I really don't know where the problem is exactly, but it must be with the way that I wrote the method. I mean, I even tested this method with different types of Writer methods to see if that would change something, and none of them worked. My last ditch effort was to see if maybe creating the file was the issue, so I made a .txt file on my computer, and then had my program try to write to it. Everything worked up until I opened the .txt file, and it was blank. There are no errors, no exceptions, and nothing out of the ordinary, my program just doesn't work, and it's frustrating as hell. I kind of want to take a break for a bit, this shit is breaking me a little. Yeah I think I'm done until next week, and i don't even care if I regret this break later on, I need it.

Week 10 Summary, 10/31 - 11/6

**Summary**: There are truly only a few words to describe this week, and they would be: no progress. This week was tough, a lot of homework and absolutely nothing to look forward to. Currently there is something mysterious going on with my capstone, where whatever i do it does not function as intended. Maybe I can pass it off as a feature, maybe not. I have almost gone through all of the stages of grief with this project because of this problem. I genuinely don't know how to get around this. It actually might be time to ask for some help, because this issue will probably not get solved on its own. I'm behind a decent b it right now, but i've decided that the keylogger is going to be my second to last program i do for my capstone. The project ideas only get harder from here, and since I'm having this much trouble with the keylogger, I don't even want to know what the other programs will bring. Next week's plan is a little tough, as I have two big ass papers and an outline to complete before Sunday at 11:59, so there probably won't be a lot of capstone work if any at all. There's a possibility that I may have to come up with a new plan next week, like maybe switching gears to a different program and scrapping the keylogger altogether. Well, until next week, Bye.

Week 11 Summary, 11/7 - 11/13

**Summary**: Like prophesied in last week's summary, I didn't work on my project at all, I barely got everything done actually, as I'm writing this on a sunday night at 11:47 after I just turned in my last 8 page paper. I have made up my mind, I'm going to fight it out with this keylogger, if it's this hard to fix, then it's worth it to keep trying. I haven't looked at my capstone in a while, but in reality there has to be a way to solve my problem, I'm just not looking at it the right way. THERE HAS TO BE A WAY. Java has a fix for everything, so next week it's back to the grindstone. The plan is still the same, get the keylogger done by any means possible. Until tomorrow, Liam is out.

## Monday, 11/14, 8pm-10pm

* OK, so today there was not a lot of progress made. I spent a decent bit of time just looking at my code and seeing if there were some issues with my code. I found a few points of interest, but sadly they weren't the problem. At first I thought that maybe the problem was because I wasn't flushing or closing the PrintWriter after typing something, but that wasn't it. Next I thought maybe I needed to use the .ToString() method to make “NativeKeyEvent.getKeyText(e.getKeyCode())” work, but that also wasn't even close to true. The next thing I tried was to just write a pre-set string to the file from the main method, and guess what, IT WORKED. So I know I can get it to work, it's just something that I'm doing wrong that I'm not seeing. Maybe it's a weird thing with Java? Maybe it's because of the way that I'm calling the KEY() method? I don't really know. This Friday I need to ask for some help, because not making any progress is really hurting my capstone at this point. I'll be back on wednesday, i'm tired of this project. Good night.

## Wednesday, 11/16, 8pm-9pm

* Today was a little bit of a waste, and a pretty bad mistake on my part. I have decided to wait until Friday so I can ask for help on the keylogger, so until then I might as well work on another program in the project. This was a huge mistake. So I went snooping around for some ideas on how to implement a packet sniffer, an anti malware service, and an image encryption service into my project. For starters all of these projects are way out of my scope. The inherent nature of these programs requires a lot of customization to be available to the end user. This means an insane amount of variation and coding to make a decent looking GUI. Fuck that. I'll link one Github project I looked at when trying to find ideas for a packet sniffer [here](https://github.com/orbisgis/java-network-analyzer). Before you look at this project, I understand that what I would have to do is not this insane. The amount of time put into the JNA is actually crazy, and the amount of work that was needed to make that was even more insane. If I were to implement something like a network analyzer, it would be a lot more simple, but the thing is even a simple version of that program would involve more headaches than I know what to do with. There would be no realistic way for me to finish any of these programs before the due date, so I'm scrapping this idea. The plan is to keep trying to find a solution to the keylogger problem. Until Friday, See ya.

Week 12 Summary, 11/14 - 11/20

**Summary**: OK, so this week wasn't much better than last week, and progress was also non existent. I found out that I am currently stuck with the keylogger no matter what I do. I either finish it and be done with the project, or I scrap it and be done with the project, those are the only two options at this point. With that being said, I talked to Ian and he gave me some ideas to think about. He took a small look at my Keylogger class and said he couldn't find anything that popped out at him, but that i should review the API for all of the Writers in java and see if i can find a reason why my program isn't working like it should. He then told me to send him a discord chat if I ever needed some help or assistance. That was pretty cool. Next week the plan is to implement Ian’s ideas and see if something sticks out to me. If not, then it is going to be me and Ians job to fix this mess. This was a short summary, but honestly what else am I supposed to write about? There was no progress made and no revolutionary findings, so yeah. Until tomorrow, hasta manana.

Week 13 Summary, 11/21 - 11/27

**Summary**: Ok i know what you're going to say, but i have no motivation to work on this project. Practice this week fucking sucked, im super tired, and its thanksgiving weekend. I'm enjoying myself and not worrying about this project. It can wait until tomorrow. The plan is to actually do some serious work to catch back up, because I'm really not making this easy on myself. I'm thinking I might just have to settle for a lackluster project, and have the text be outputted to the console, and no more .jav file that runs the program either, that was hella ambitious of me. Tomorrow a lot of work is going to be done tho, I promise.

## Monday, 11/28, 8pm-10pm

* Well well well, look who's back. I FIGURED IT OUT. But that doesnt mean I fixed it. So, the problem the WHOLE TIME was the fact that i initialized my file and my PrintWriter in the nativeKeyPressed() method. For a little explanation, this method is called whenever a key is pressed, meaning this method gets called quite a lot. Since I initialized both the File and the PrintWriter in a method that gets called over and over again, I am effectively erasing all of my progress when I type. The file gets overwritten by the next file that gets initialized when I type! This took me WAY longer than it should to notice, but now that I know what to do I think I can fix the problem. With that being said, I have a huge headache, and it's like 11pm right now, so I'm hitting the hay. This past semester I have been looking at my computer screen for way too long, my eyes genuinely hurt. Good Night.

## Friday, 12/2, 12-3pm

* IAN IS MY SAVIOR, well kind of. So I waited until I could get some help from Ian personally because I hate being taught how to do things online. He told me that what i said makes sense, he also didnt realize that the nativeKeyPressed() method was called multiple times, so im not totally dumb. We did a bunch of problem solving and tests and it turns out I was right, the file is being overwritten. The reason why the files would always be blank is because the “Escape” key was my terminating key to stop the keylogger, and I made it so pressing “Escape” wouldn't type out the string “Escape” in the .txt file. Since the “Escape” key was the last one I would press every time, it overwrote the last file with a new one and would put in a blank string, and then flush and close the PrintWriter. That was my problem all along. So Ian gave me a few Ideas, and I went to my room with some serious determination. Well, turns out none of them work. First idea was to initialize the file as a global variable so it would be stuck in the nativeKeyPressed() method and overwrite itself. For some reason this still doesn't work, it still gives me a black .txt file. This might be my best option though. The next thing I tried was making both the PrintWriter and the File global variables, but that doesn't work because I need to surround the PrintWriter with try/catch, and if it's not in a method I can't do that. Another thing i tried was putting the PrintWriter and the File in another method and then sending them as parameters, however since this method is actually just receiving keystrokes, i can't send a PrintWriter or a file with each keystroke, it's just not physically possible. At this point this keylogger is my arch nemesis. This thing sucks and I don't really understand why. It's about crunch time, so I really need to finish this thing. Well, I'm off to the gym, Ta-ta for now!

Week 14 Summary, 11/28 - 12/4

**Summary**: So this week was a little more exciting than the other weeks, just because I kind of got somewhere. It really sucks because I have put a lot of effort into this project, and this one part is giving me the hardest time imaginable. The plan for next week is to focus on getting ready for my finals, and then asking Dan to sit down with me and go through why my program doesn't work. This is the only shot I have realistically. I feel like i have done all i can do to make this keylogger work, so I will be waiting until friday, maybe even earlier to ask Dan for some help. I'm really hoping he has something that can help me because this whole project kind of rides on his wizard knowledge. THE GOAL IS TO JUST FINISH THIS DAMN KEYLOGGER! I will get this project done, and I will get an A on it, even if I have to fight somebody for it. Until next Friday, i really hope i have some good ideas and maybe a fix for this problem, See ya!

## Friday, 12/2, 11-3pm

* I FUCKING DID IT, YES THATS RIGHT I FUCKING DID IT. OH YEAH, OH YEAH, OH YEAH. ITS DONE, ITS FINISHED!!!!! Ok so let me explain. I was trying to use a PrintWriter to print every single character inside of a text file. This didn't work because it kept opening the text file, replacing the character that was already there, and then closing the text file. Obviously, if the method that prints to the text file is called every time I press a key, it's going to rapidly replace the contents of the file over and over and over again, which isn't what I want. My thought process turned to strings. Since it seems that I can only make one commit to the text file, I instead thought I could use strings, and add on to them every time I pressed a key. Surprisingly, that worked. I then made a method that would take that string when the Escape key was pressed and send it to a PrintWriter that would then send it to my file that was initiated as a global variable. AND IT WORKED. It's crazy because I was stuck for so long, and it turns out all along the solution was going back to the basics! After figuring this out, I went back to my apartment and did a shit ton of bug fixes, ran my program, and she works without a hitch!!!!! So, this is as far as this capstone is going, no more time for implementation and such. Time to go work on the other parts of the final project that I just learned about today! This is really it, this is my last journal entry, SO SEE YA!