***CSC 413 Project 2 Documentation***

***Summer 2023***

***Wing Lee***

***920558688***

***CSC-0413-01***

[***https://github.com/csc413-SFSU-Souza/csc413-p1-ayanamesu***](https://github.com/csc413-SFSU-Souza/csc413-p1-ayanamesu)

* Introduction
* Project Overview

completed

* Technical Overview

completed

* Summary of Work Completed

The summary of work completed is as follows I first worked on the RunTimeClass implementation because this class is the only independent class that did not rely on other classes. Then I worked on the ByteCode classes starting with the first one ArgsCode and all the way to WriteCode. While I wrote these bytecode classes, I also need to add code to the virtual machine classes. I also can't tell if they work or not so I head onto the Program class and code resolve address and after that I can see if Goto, label, Call and FalseBranch work.

* Development Environment

The Java version used is Java 20 SDK   
The IDE used is IntelliJ IDEA 2023 1.2 Ultimate Edition

* How to Build/Import your Project

I first imported this assignment from GitHub repository and copied the HTTP link and then went to my Terminal and used the command git clone <https://github.com/csc413-SFSU-Souza/csc413-p1-ayanamesu.git>. Then I began building the project by opening IntelliJ IDEA and using the old project and select files then selecting Import Project from existing resources. Then select the root folder as the source root of the project > Create Project from existing sources > Enter in a project name then keep hitting next > then click finish and once done you should see the C symboles next to the files.

* How to Run your Project

To run this project you would first need to make sure the selected testing arguements files all have "DUMP ON" in them. The test arguements are: factorial.dump.cod/factorial.x.cod , fib.x.cod, functionArgsTest.cod. Then you would go to run/debug configurations and edit configuration, first select the "Interpreter class" and then in the program arguements you would enter in either of the 3 test arguements listed above. This is because the program arguements run off of the interpreter class.

* Assumption Made

Some assumptions to be made would be that I expected the test arguement to be true and correct and would display the correct results. I also assumed that the project given to me would work properly.

* Implementation Discussion

//do this on monday

* Class Diagram  
  Please see the documentation folder there is a Diagram png with all the folders that has classes inside of them such as ByteCodes, loaders, virtual machine and the Interpreter.
* Project Reflection

In my opinion this project was harder than the previous project, at first glance and while working on it as well. The difficulty was stem from the pdf because the pdf leaves many important tasks that we should do and too many instructions and some of them are very miniscule with no example outputs or examples at all. Now diving deep into the project what I found most difficult was the dump() part because if I got that wrong then the bytecode classes are all wrong. In the project there are still some small errors such as the Halt dump because its dumping when it should not be dumping. I found that the bytecode classes themselves were pretty self explanatory I got to retouch on switches with ternary conditions.

* Project Conclusion/Results

Overall I found this project very difficult. I also went through many phases of emotions through out this project. At some points I had a mental breakdown because I couldn't get the project running, This is caused because of the (For loop) that was made during class. Once the professor helped me resolve that issue I finally had my hopes up and felt better as the project ran as I wanted initially. Before I got help though I spent days debugging and finding out why the project wouldn't run and then I also fixed many bytecodes along the way for factorial.x.cod file then after I got help I finally went and dived into the other test arguements.