CSC 413 Project 2 Documentation

Summer 2024

Student name: Guiran liu

Student ID: 923620812

Class : CSC413-02

GitHub Repository Link:

https://github.com/csc413-SFSU-SU2024/interpreter-GuiG2023

Table of Contents

1 Introduction 3
1.1 Project Overview 3
1.2 Technical Overview 3
1.3 Summary of Work Completed 3
2 Development Environment 4
3 How to Build/Import your Project 4
4 How to Run your Project 4
5 Assumption Made 4
6 Implementation Discussion 5
6.1 Class Diagram 5
7 Project Reflection 5
8 Project Conclusion/Results 6

1 Introduction

1.1 Project Overview

This project involves developing an interpreter using a structured approach with various classes and folders. The goal is to read command lines from a template provided by our professor, prompt the user for input, and generate the correct output based on the input.

1.2 Technical Overview

Our purpose is to mock language X as a simplified version of Java. Basically, we have three different folders: the first one is byte codes, the second one is loaders, and the third one is a virtual machine. All of these courses are served for the interpreter class. Also, we are not allowed to change any code about the interpreter class.

The interpreter is responsible for processing byte codes created from source code files with the extension x. The interpreter and the Virtual Machine will work together to run a program written in Language X.

The two sample programs are recursive versions of computing the nth Fibonacci number and finding the factorial of a number. These files have the extension x.cod. Also, there is a verbose mode, which shows detailed information of the stacks inside where we are running our program.

1.3 Summary of Work Completed

It took a lot of time for me to make the whole project run correctly. First of all, I spent a lot of time reading the professor interpreter PDF file to try to understand and get the big picture of the whole project and photos with the project lecture introduction of the assignment. I started with the runtime stack class and tried to figure out what happened underneath the interpreter's hood.

The second step is to create the bytecode interface, create different bytecode classes, implement the interface, and write down the execute, initialize, and tostring methods. I also followed the professor's second code table and loader instructions. Also, without the professor's help, I cannot finish this part by myself because it's difficult for me. I learned a lot from that, including reflection on Java.

The third step is the test and debugging step. This part took me the most time. When I run the program, I always find exceptions, errors, or logical errors in my product, so I debug from different classes one by one, going over the interpreter PDF file, again and again, to see if I lost something. Finally, I got the correct output from the professors.

2 Development Environment

OpenJDK 21.0.2 2024-01-16

OpenJDK Runtime Environment (build 21.0.2+13-58)

OpenJDK 64-Bit Server VM (build 21.0.2+13-58, mixed mode, sharing)

IDE: IntelliJ IDEA 2023.3.3 (Ultimate Edition)

3 How to Build/Import your Project

I cloned the GitHub repo from the professor to my own local repo. Then I used intellij open to find the repo, trust the project, and add it to my working environment, which is my intelligent idea.

4 How to Run your Project

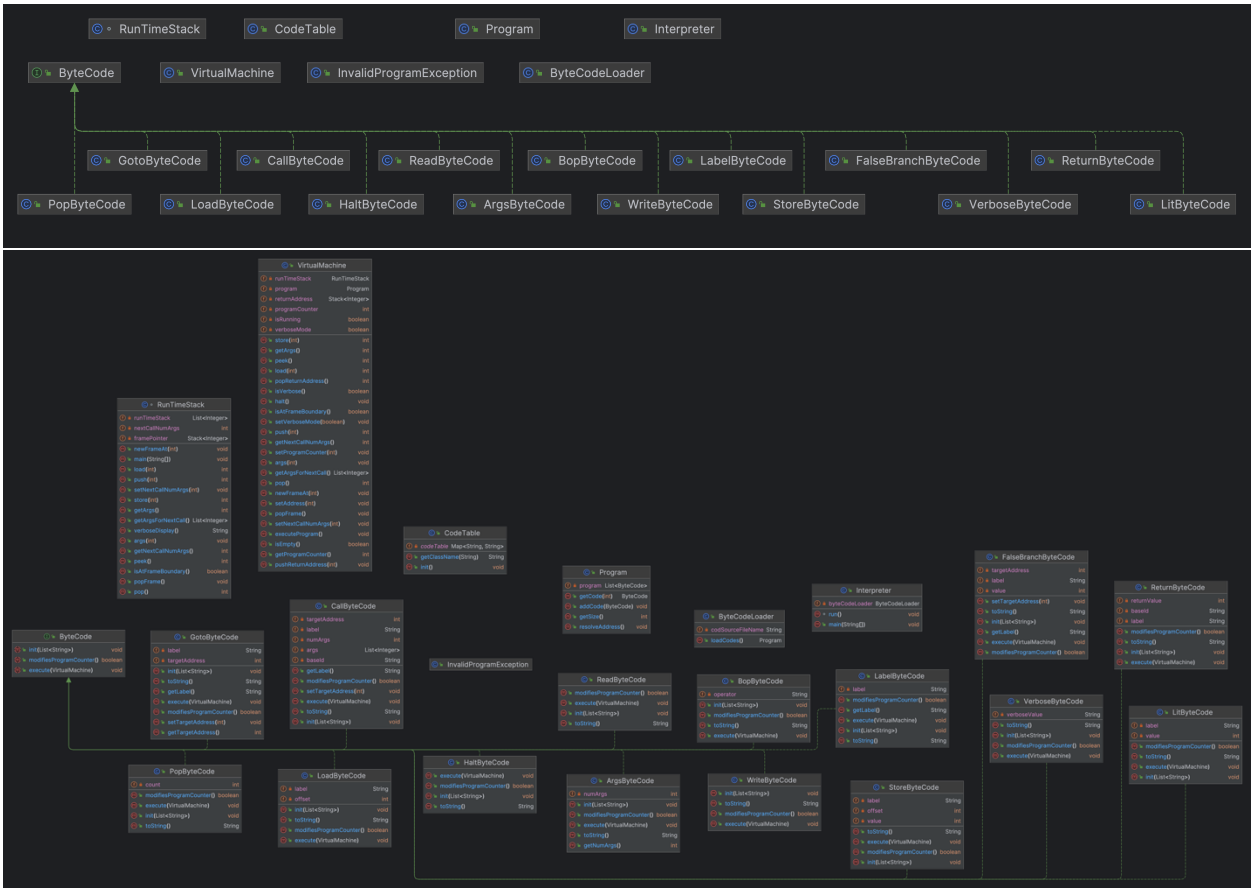
Running this project is different from running project one. First, I set up my configuration, which means I have to enter the code that the professor gave us into the configuration settings. Then, I could run the project from the interpreter class, the main method.

5 Assumption Made

The assumption is these will assume that we have to run the project to get the correct output and compare it with his professor's sample output.

6 Implementation Discussion

6.1 Class Diagram



7 Project Reflection

Thankfully, I've finished the project, and I believe I got the correct answers. However, during the process, I was not confident about completing the entire project because I encountered many issues.

First of all, when I started testing with my runtime stack, I faced numerous exceptions. It took some time to fully understand what was happening inside the runtime stack and the relationship between the frame pointer and the stack. After that, I was unsure about how to build a code table until the professor provided guidance. Following his instructions, I successfully completed the code table class.

Next, when I tried to finish all the byte codes, I needed to understand the big picture of the entire project and the purpose of the bad code. I revisited the PDF file multiple times to figure out the format of the byte code and what happens between bad calls. I also had to understand how to connect the bad code class with the bad code loader, the program class, and the virtual machine class.

After I thought I had finished the project, I clicked the icon to run it, but it crashed due to an empty stack exception. Following the exception message, I went back to my runtime stack, double-checked everything, and solved the problem. When I reran the project, it crashed again due to a number format exception. I repeated this process of debugging and fixing issues multiple times.

Finally, when I managed to run the project successfully, I encountered a logic error: the output was completely different from the professor's sample output, and I couldn't use my verbose mode. I went back to the PDF file again to double-check the requirements and see if I had missed anything. I re-implemented parts of the project several times until my output closely matched the professor's output. After thorough testing with all the codes provided by the professor, it finally worked correctly in the end.

8 Project Conclusion/Results

This is the most 'complicated' project I have ever had in a school class. It's not just about the number of classes but also about learning how to explore the relationships between classes and figure them out and manage time. I had to understand the connections between different parts of the project. The 40-page PDF file the professor provided was challenging to read at first, but it was the most useful resource.

Through this project, I learned to grasp the big picture first and dive step by step into the details. I also learned how to explore and quickly understand new concepts. This was a great project because of the technical challenges and because it taught us how to learn independently rather than just following the professor's instructions.