CSC 413 Project Documentation

Fall 2018

Mesoma Esonwune

915852059

413.02

https://github.com/csc413-02-spring2019/csc413-p1-mesonwune

Table of Contents

1 Introduction

1.1 Project Overview

I worked on Project two which was creating an Interpreter in Java

1.2 Technical Overview

I worked on Project two which was creating an Interpreter in Java

1.3 Summary of Work Completed

I got some of the bytecodes working, started the RunTimeStack, and have completed ByteCodeLoader and Program.

2 Development Environment

Java 11.0.2 in IntelliJ

3 How to Build/Import your Project

a) Install IntelliJ if you don't have it.

b) Go to repository and get the HTTPS file.

c) Go on terminal and install file.

d) Open IntelliJ and click Import Project

e) Choose file

f) Click Next when it says create project from existing sources

g) Click Next when it shows project name

4 How to Run your Project

a) Right-click on Interpreter.java

b) Click Run.

5 Assumption Made

I assumed that I wasn't supposed to add too many methods to the files. I keep on forgetting that adding new methods to the skeleton is not a bad thing. I also thought that reading it several times would be enough. I missed out on a lot of information because of how dense the pdf was and I ended up missing key information.

6 Implementation Discussion

6.1 Class Diagram

Attached on next page

I had misread the suggested order of tackling the assignment and had stressed myself out trying to figure out what needed to be in each specific ByteCode class. Once I reread and saw what I was supposed to do, I started looking on the ByteCodeLoader. This was pretty easy to tackle since it was mapped out for us on the pdf. Unfortunately, I got stuck with the Program.java.

I had to create an addByteCodes so that program can store the ArrayList of ByteCodes. Then I moved on to resolveAddrs. I asked Professor Souza on how to start this and he had told me that the simplest way was to parse through the Arraylist program to track and store where LabelCode was index-wise. Then, find each branch ByteCode and see where its Label was and replace the symbolic address with the actual index. I had a hard time figuring out how to replace the address until I was given the idea of creating methods that can change it for me rather than trying to do it manually. I realized that I needed to create getters and setters for the branches so I can change the address value.

7 Project Reflection

This was insanely difficult and it made me feel awful about myself. I put as much time as I possibly could but I kept on running into walls. This project made me want to quit this major honestly because I felt uncomfortable asking questions when so many people were far ahead of me. Even if this project was extended, I couldn't put effort in it because I have a midterm on Friday and I'm lost in that class as well. It just reminded me that I am still playing catch up.

8 Project Conclusion/Results

For some reason, it compiles but I keep getting errors in two separate areas.

The first one is in ByteCodeLoader.java. I keep on getting a ClassNotFoundException for interpreter.bytecode.null even though I have a catch for it.

The second one was in Program.java. I keep getting IndexOutOfBoundsException for this line "return this.program.get(pc);" I think it's because I haven't properly implemented it yet.

Please zoom in!!!!!
It's readable