CSC 413 Project 2 Documentation

Semester YEAR

Yakoub Alkabsh

917108002

413 Summer 2023 R4

https://github.com/csc413-SFSU-Souza/csc413-p2-Yakkubs

Table of Contents

1 Introduction

1.1 Project Overview

- This project is a program that acts an interpreter of our X files which is a coding language that we create. The program converts the contents into byte codes that our algorithm treats as computer instructions and runs them like any other coding language.

1.2 Technical Overview

- This program acts as an interpreter that reads X files(fake coding language that we defined) and tokenizes all the codes within it. We make use of a runtime stack class that has a runtime stack and a frame pointer stack to store and run the codes properly. The codes being byte codes that we define that are similar to machine code you find in MIPS, with codes such as Dump and Halt. With all this we have a mock interpreter that runs on java.

1.3 Summary of Work Completed

- From the given skeleton code, I fully implemented the runtime stack class with all the functions that it needs to work properly with the vm class. This includes functions like peek, pop, newFrame, popFrame…etc. With the runtime stack being completed I implemented all the byteCodes and properly implemented their methods, for instance Goto bytecode sets the first argument as the value of its label variable then sets its target address to the program counter. I completed the implementation of the program class and added a new data structure that holds label codes and their positions so that my implementation of the resolveAddress function works properly. Finally finished the implementation of the VirtualMachine class which executes our program while updating the program counter and can Dump the state of the frame if prompted. My implementation of the Vm class contains many functions that our byteCodes use to access data, that helps keep our code incapsulated.

2 Development Environment

- IntelliJ IDEA 2023.1.2
- Java 18

3 How to Build/Import your Project

- Click on the repo link provided above and click download zip, alternatively assuming you have git installed, you can go to your terminal and clone the repo with git clone https://github.com/csc413-SFSU-Souza/csc413-p2-Yakkubs.

4 How to Run your Project

- Once downloaded, open folder in intellji and head to the interpreter file from this path, csc413-p2-Yakkubs > interpreter > Interpreter.java. once you have this file open run the file by clicking the run/play button on the top right and enter the proper x file.

5 Assumption Made

- I assumed that there are no floats/doubles for arguments, only Ints.
- I assumed that X file does not contain non existing bytecodes

- I assumed there were no syntactical errors within the X files
- I assumed all the frame boundaries are correct

6 Implementation Discussion

- Some design choices I made while coding the project was to use a hashmap for label code names and positions. Using a hashmap made getting the locations that my jumpCodes would use a lot simpler because I would just need to use the hashmaps built in get function to get the position without having to loop through the entire map. Another thing was creating a jump code interface class solely for all the byte codes the jump, that made coding for jump conditions simpler and cut down on repeated code. Finally another notable design choice I made was that instead of having 2 loops within the resolveAddress function that checked for label codes and jump codes, I just had one that checked for jump codes but added an if statement in my addByteCode function that checked for label codes there instead because I felt like that's more intuitive and maybe more efficient then doing it the other way.

6.1 Class Diagram

- Check documentation folder

7 Project Reflection

- This project was a lot simpler then what I initially thought it would be. In the beginning I was overwhelmed because I didn't understand what the project is supposed to do and what it was but once I finished the runtime stack class everything started to click. A big take away that I got was that I forgot that the main thing when it comes to coding is that I need to always make sure that I understand what the code/project is doing before I try and tackle it. In assignment 1, I was confused with what a loop was doing which caused me to make an in efficient/intuitive algorithm for processing operators. Since I understood the project everything became easier and it felt like basic 210/340 coding after that.

8 Project Conclusion/Results

- I feel like I did great for this assignment, I started at a good time and made sure to stay on pace with the whole project. I do have to admit that in the beginning since I was confused about runtime stack class I googled what it is and how people would implement it but once I realized it was simple I was fine after that. Lastly, from experience I now know that I need to fully read the PDF multiple times instead of reading it partially/bit by bit because I ended up asking a lot of clarifying questions in the discord when they were already answered in the PDF.