CSC 345 Lab – Import PseudoAssemblyObf and Run Program

Overview

In this lab you will install the PseudoAssemblyObf library in the local Maven repository on your computer. You will then import the PseudoAssemblyObf library into a Java project. Finally, you will run an assembly language program to make sure it works.

Part 1

Install the PseudoAssemblyObf library in the local Maven repository of the computer.

- Download the PseudoAssemblyObf-1.0.jar file to the local machine. Make sure to unzip it.
- Open a command prompt (type in cmd in Windows search bar to open a command prompt).
- Here is the format of the Maven command that you will need to run:
  mvn install:install-file -Dfile=<path-to-your-jar-file> -DgroupId=<group-id> -DartifactId=<artifact-id> -Dversion=<version> -Dpackaging=jar
  - -Dfile: The full filename of the .jar file you are installing (this includes the whole path of where the file is currently located).
  - -Dgroupid: The group id of the dependency (org.example for PseudoAssemblyObf).
  - -DartifactId: The artifact id of the dependency (PseudoAssemblyObf).
  - -Dversion: The version of the dependency (1.0 for PseudoAssemblyObf).
  - -Dpackaging: The packaging of the dependency (jar for PseudoAssemblyObf).
- Sample Maven install command (assumes the .jar file is in the C:\temp directory):
  mvn install:install-file -Dfile=c:\temp\PseudoAssemblyObf-1.0.jar -DgroupId=org.example -DartifactId=PseudoAssemblyObf -Dversion=1.0 -Dpackaging=jar
  Note: The mvn command requires the JAVA_HOME environment variable to be set to the location of the JDK.

Part 2

Setup the IntelliJ project.

- Create an IntelliJ console application. Make sure to use Maven as the build system when creating the project.
- Add the following dependency to the project's pom.xml file:

```
<dependencies>
  <dependency>
    <groupId>org.example</groupId>
    <artifactId>PseudoAssemblyObf</artifactId>
    <version>1.0</version>
  </dependency>
</dependencies>
```

- After adding the dependency to the pom.xml file do the following:
    - Open the Maven tab in IntelliJ (on right side).
    - Press the Download Sources button (in Maven toolbar).
    - Press Generate Sources and Update Folders for All Projects (in Maven toolbar).
    - Press Reload All Maven Projects (in Maven toolbar).
    - Note: Skipping the above will cause IntelliJ to not give prompts for import statements.

Part 3

Paste the following code into the main method and run it.

```
String code = "";
code += ".data\n";
code += "var int x\n";
code += ".code\n";
code += "loadintliteral ri1, 88\n";
code += "storeintvar ri1, x\n";
code += "printi x\n";
code += "printi ri1\n";

int numVirtualRegistersInt = 32;
int numVirtualRegistersString = 32;
String outputClassName = "MyLabProgram";
String outputPackageNameDot = "mypackage";
String classRootDir = System.getProperty("user.dir") + "/" + "target/classes";

PseudoAssemblyWithStringProgram pseudoAssemblyWithStringProgram = new
PseudoAssemblyWithStringProgram(
      code,
      outputClassName,
      outputPackageNameDot,
      classRootDir,
      numVirtualRegistersInt,
      numVirtualRegistersString
);

boolean parseSuccessful;
parseSuccessful = pseudoAssemblyWithStringProgram.parse();

if (parseSuccessful == true) {
   // Creates a Java bytecode class file
   pseudoAssemblyWithStringProgram.generateBytecode();

   // Run the Java bytecode class file and show output on the console
   PrintStream outstream = new PrintStream(System.out);
   pseudoAssemblyWithStringProgram.run(outstream);
}
```