CS 335 Semester 2022–2023-II: Project Description

12th Mar 2023

Due Your submission is due by Mar 29th 2023 11:59 PM IST.

General Policies

- Do not plagiarize or turn in solutions from other sources. We WILL check your submission(s) with plagiarism checkers. You will be PENALIZED if caught.
- Each group will get a total of FOUR FREE DAYS to help with your project submission deadlines. You can use them as and when you want throughout the duration of the project. For example, you can submit Milestone 1 two days late without penalty, and you will have two free days remaining to potentially utilize for the rest of the semester.

Description

The goal of this project is to implement a compilation toolchain, where the input is in Java language and the output is x86_64 code.

Milestone 3

Extend your 3AC IR from Milestone 2 to add runtime support for procedure calls. Your implementation of activation records MUST INCLUDE the following fields:

- space for actual parameters,
- space for return value,
- space for old stack pointers to pop an activation,
- space for saved registers,
- space for locals.

You can optionally add other fields required by your implementation. Remember that you may need to define 3AC instructions corresponding to various features allowed in the input program (e.g., `call`, `pushparam`, and `return` where the semantics is obvious from the name).

The exact order of the fields depends on you and the calling conventions of the target x86 architecture [x86 calling conventions, x86 Disassembly/Calling Conventions].

Output.
For correct input programs, your implementation should output a text dump of the 3AC of the input Java program with runtime support for activations.

Looking ahead. The plan for the next and final Milestone is to generate correct x86_64 assembly from 3AC which can be run (via NASM or GAS on Linux). So it is important that you do a good job with runtime support for procedures that is tailored to the target assembly syntax.

We will discuss code generation shortly in class, but you might want to refer to Chapter 8 from the Dragon book (2nd edition) to get a feel of the requirements for code generation.

Submission.

- Submission will be through Canvas.
- Create a zip file named "`cs335_project_ <roll>.zip`". The zipped file should contain a folder `milestone3` with the following contents:
  - All your source files must be in `milestone3/src` directory.
  - Use `Makefile` (or equivalent build tools like `ant`) for your implementation. You are free to use wrapper scripts to automate building *and* executing your compiler.
  - Use the directory `milestone3/tests` and your previously-designed test cases to showcase your implementation.
  - Your submission must include a `milestone3/doc` directory and a `PDF` file. The `PDF` file should describe any tools that you used, and should include compilation and execution instructions. Document all command line options.

    You SHOULD USE LaTeX typesetting system for generating the PDF file.

Evaluation.

- Your implementation should follow the prescribed steps so that the expected output format is respected.
- We will evaluate your implementations on a recent Debian-based distribution.
- We will evaluate the implementations with our OWN inputs and test cases, so remember to test thoroughly.
- Our evaluation will only focus on correctness, the compilation time is not important.
- The TAs will meet with each group for evaluation. Make sure that your implementation builds and runs correctly. A typical evaluation session could be as follows.

```
$ cd <milestone3>/src
$ make
$ ./milestone3 --input=../tests/test5.java --output=test5.3ac
$ cat test5.3ac
```