CIS*4650 (Winter 2024) Compilers

Implementation Project - Checkpoint Three

Due time: Apr. 3, 2024, by 11:59 pm.

Functionality:

After the first two checkpoints, your compiler should be able to detect syntactic and semantic errors for a C- program and produce an annotated syntax tree if the given program is valid. With Checkpoint Three, your compiler implementation should be complete, generating assembly code for valid *C*- programs that can be executed on the TM simulator (downloadable from CourseLink).

Execution and Output

You are now responsible for all command-line options specified for the project (-a, -s, -c). In particular, the -c option should now produce assembly code for the TM simulator. Please refer to the lecture notes on "11-TMSimulator" for some suggestions about the implementation of the runtime environments.

Documentation

Your submission should include a project report describing the entire compiler construction process. In addition to what has been done for this checkpoint, the overall design, lessons gained in the implementation process, assumptions and limitations, possible improvements, and the contributions of each member if relevant, you can now outline any major changes in your design that took place over time (i.e. any significant modifications from the previous two checkpoints). Some of the things you might want to discuss may include issues you encountered in translating syntax trees into assembly code, error handling, parsing and semantic analysis, source language issues, attempted repair of detected errors if implemented, and so on. The document should be reasonably detailed, about six double-spaced pages (12pt font) or equivalent and organized with a suitable title and headings. Some marks will be given to the organization and content of this document in the marking scheme.

Test Environment and Programs

You should verify your implementation on the Linux server at linux.socs.uoguelph.ca, since that will be the environment for evaluating your demos. For incremental development, we recommend that you follow these steps: (a) download and get familiar with the TM Simulator; (b) focus on the expressions and assignments in the "main" function for the initial implementation; (c) expand the implementation to cover control structures, function calls, nested blocks, and arrays; (d) handle the runtime error for "out of bound" of indexed variables; and (e) revisit the previous two checkpoints for possible improvements.

In addition, each submission should include ten *C*- programs, which can be named `[1234567890].cm`. In particular, programs 1.cm through 3.cm should compile without errors (producing valid assembly code that can be executed on the TM simulator). Note that these programs should represent different levels of complexity, not all easy or brutal. 4.cm through 8.cm should exhibit various syntactic and semantic errors (but no more than 3 per program). For 9.cm and 0.cm, anything goes and there is no limit to the number and types of errors in them. All test files should have a comment header clearly describing the key aspects of the compiler being tested, including the nature of any errors that may be present.

Makefile

You are responsible for providing a Makefile to compile your program. Typing "make" should result in the compilation of all required components to produce an executable program: *CM*. Typing “make clean” should remove all generated files so that you can type “make” again to rebuild your program. You should ensure that all required dependencies are specified correctly.

Deliverables

(1) Submission to the related drop box on CourseLink by the due time:
- All source code required to compile and execute your “*CM*” compiler
- Makefile and all the required test files
- The README file for build-and-test instructions
- A project report as described in the “Documentation” section above
- Tar and gzip all the files above and submit it on CourseLink by the due time.

(2) Demo: You should schedule a brief meeting of about 15 minutes with one of the TA’s so that we can evaluate your demo on April 4, 2024. A link to the shared spreadsheet will be emailed to you several days ahead so that you can sign up on an available time slot for your demo. Later when the marking is done, your mark along with the feedback will be posted in the related drop box of your CourseLink account. Wish everyone the best in completing the compiler project as fully as possible!