Design Report

Architecture

We invoke the Antlr4 runtime, adding error listeners for better formatting/behavior when the lexer fails.

Implementation Strategy

First, we implemented the grammar exactly as was specified but in Antlr4's syntax. Since Antlr4 works with non-LL(1) grammars, we were able to make sure we got this baseline grammar correct on the public test cases first.

Then we transformed it into an LL(1) grammar by hand. We removed ambiguity and made it LL(1) concurrently. This is because we have direct way to make the grammar LL(1), and no LL(1) grammar can be ambiguous. To do this, we wrote a script that quickly tested the grammar against the public test cases. We removed left recursion and did left factoring in small steps, ensuring we didn't start failing correct test cases or passing incorrect ones.

Engineering Challenges

By default, Antlr4 seems to lazily run the lexer when generating the AST for a Tiger file. We needed to circumvent this, so we could preven the parser from running when the lexer fails, which required a bit of digging into the Antlr4 runtime.

Known Bugs

No known bugs.

Build & Usage Instructions

We have a python3 frontend so you will just need to `pip3 install antlr4-python3-runtime`.

Then run the frontend with

```
./run.sh [filename.tiger]
```