Compiler Design Assignment 2 (Parser)

Group Details

- Project Group Number: 35
- Members (order is alphabetical and does **not** reflects contribution):
  - 160308 Jatin Jindal
  - 160479 Parv Mor
  - 160534 Raghav Garg

Instructions to run lexer

- `python lexer.py --cfg=../tests/cfg1/some-cfg ../tests/input1/some-input --output=../output/some.html`
- Or you can run `./run_lexer.sh` to generate all combinations of HTML files.
- To generate a config file: `cd tests && python color_cfg_gen.py <name>`

Instructions to run parser

- `python parser.py ../tests/input2/some-input --output=../output/some.html`
- Or you can run `./run_parser.sh` to generate all combinations of HTML files.

Instructions to run IR and symbol table generator

- `cd src && python semantic_parser.py ../tests/input3/some-input --ir=../output/irs/some.ir --st=../output/sts/some.st`
- Or you can run `./run_ir.sh` to generate all IRs for `input3` tests.

Instructions to generate assembly

- `./compile.sh` to generate assembly for all tests.
- Use spim to simulate them.

Language

Our SIT triplet is:

- Source language: `golang`
- Implementation language: `python`
- Target machine code/language: `mips`

Following features distinguish our source language from `C:`

- Auto assignment, i.e. type inference
- Multiple return values
- a, b, c = 1, 2.0, "Hello"
- func(a, b int) along with func(a int, b int)

Reference for the grammar of `golang:` https://golang.org/ref/spec