

Python Compiler - Milestone 1

Akshat Gupta *

Devansh Kumar Jha †

Shashwat Gupta ‡

March 3, 2024

1 Compilation Instructions

To build this repository follow the following steps:

```
cd src          # Go the the source code directory
make all        # Run the Makefile to compile the toolchain
./cs335 --help  # Get information about the functioning of the compiler
./cs335 --input <path to input file>
make clean      # Clean all the compiled binaries
```

For more specific output you can also run the following commands:

```
# Give input directly from the terminal. Use --verbose or --full to display debug output.
./cs335

# Redirect debug outputs to the output file.
./cs335 --input <path to input file> --output <path to output file> --verbose
./cs335 --input <path to input file> --output <path to output file> --full

# Generate AST pdf and DOT script for the input program
./cs335 --input <path to input file> --dot <path to store dot script>
./cs335 --input <path to input file> --ast <path to store pdf file for AST>
./cs335 --input <path to input file> --dot <path to store dot script> --ast
<path to store pdf file for AST>
```

NOTE:

- The testcases are present in the *tests* subdirectory. Therefore, to test the *./cs335* binary on a few testcases, use the command such as:

```
./cs335 --input ../tests/test1.py
```

- The following condition should be met by every testcase: **Input test cases will use two spaces for indentation and will have a final newline.**
- While using **–ast** and **–dot** flags, the path to store this files should at the very least include filename. For example, if you want to generate the file **ast.pdf** in the same directory as the binary, use the command:

*Fourth Year Undergraduate, IIT Kanput, akshatg20@iitk.ac.in

†Fourth Year Undergraduate, IIT Kanput, dkjha20@iitk.ac.in

‡Fourth Year Undergraduate, IIT Kanput, shashwatg20@iitk.ac.in

```
./cs335 --input <path to input file> --ast ./ast.pdf
```

- We recommend that you store the dot scripts in the `/milestone1/output/dot` subdirectory and ast PDFs in `/milestone1/output/ast` subdirectory.

2 Different flag functionalities

2.1 help

```
./cs335 --help
```

Display of information about the compiler flags.

2.2 input

```
./cs335 --input <path to input file>
```

Adds the path of the input file. By default it is the standard input. The next argument after `-input` flag should correspond to the location where the input file is present.

2.3 output

```
./cs335 --input <path to input file> --output <path to output file>
```

Adds the path of the output file. By default compiler output is displayed at the standard output. The next argument after `-output` flag should correspond to the location where the output is to be stored.

2.4 verbose

```
./cs335 --input <path to input file> --verbose
```

Prints detailed compilation log and debug outputs.

2.5 full

```
./cs335 --input <path to input file> --full
```

Prints the complete debug output.

2.6 error

```
./cs335 --input <path to input file> --error <path to error file>
```

Adds the path of the error file. By default it is the standard error. The next argument after `-error` flag should correspond to the location where the error is redirected.

2.7 ast

```
./cs335 --input <path to input file> --ast <path to store pdf file for AST>
```

Configures the compiler to output Abstract Syntax Tree of the input program in a PDF. The next argument after `-ast` flag should correspond to the location where the ast file is to be stored.

2.8 dot

```
./cs335 --input <path to input file> --dot <path to store dot script>
```

Configures the compiler to output the DOT script corresponding to input program in a .dot file. The next argument after `--dot` flag should correspond to the location where the dot file is to be stored.