CS335: Milestone 5

Group 14

Members:

Name	Roll No.	Email	Contribution
Supreeth Baliga	180801	supbal@iitk.ac.in	20%
Chinmay Goyal	180206	chinmayg@iitk.ac.in	20%
Aaryan Srivastava	180007	aaryans@iitk.ac.in	20%
Nikhil Agarwal	180475	nikhilag@iitk.ac.in	20%
Sanchit Agrawal	180664	sanagrwl@iitk.ac.in	20%

Source Language (S): *C*
Implementation Language (I): *Python3*
Target Language (T): *X86 Assembly - 32 bit: AT&T Syntax*

Repo Link: https://github.com/SupreethBaliga/Compilers/

Directory Structure:

```
TOP
├── src/
│   ├── helpers/
│   │   └── libstring.a (Archive file for string functions)
│   ├── lexerClass.py (Lexical analyser of the compiler)
│   ├── parserClass.py (Creates the AST, does semantic analysis and emits TAC)
│   ├── SymbolTable.py (Handles all functionality related to symbol table)
│   ├── TypeTable.py (Handles all functionality related to type table)
│   ├── TAC.py (Handles all functionality related to three address code)
│   └── codeGen.py (Converts the TAC to Assembly code)
├── tests/
│   └── (Contains all tests for respective milestones)
├── PDFs
│   └── (Contains the corresponding PDF files for milestones)
├── requirements.txt
├── install.sh (Script file to setup dependencies)
├── lex.sh (Script file to run just the lexer)
├── c-- (This is the main script file with which you run the compiler)
├── README.md
└── .gitignore
```

Instructions to run:

- Install all the python modules and set up virtual environment by executing *$ ./install.sh*
- Activate the virtual environment by running: *$ source venvcompiler/bin/activate*
- To use the compiler execute *$ bash c–* and provide the test file(s) as command line arguments

Main Features supported by our compiler:

- Handles data types like char and int (bool can be treated similar to a char)
- Supports usage of derived data types arrays (multidimensional), structs and pointers
- Handles type checking and precedence of operators
- Support for short circuit evaluation
- Handles nested if-else, iterators, loops and conditionals
- Supports any number of parameters passed to the function and handles recursion
- Scales well to size of the expressions
- Supports usage of basic functions like printf, scanf, malloc, sizeof and free

Basic Features NOT supported by our compiler:

- Function declarations are not allowed. Only function definitions are allowed
- Array initializations are not allowed (Workarounds exist for char arrays)
- Only static integers are supported and they should be initialised with a non-negative integer. Global variables cannot be initialised while declaring.
- Pointer arithmetic is not supported (Workarounds exist)
- File I/O is not handled

Advanced Features supported by our compiler:

- Mixing of declarations and statements, unlike ANSI C
- Support for multiscoping and multi-level pointers
- Support for static integer variables and global variables
- Handles floating point integers and all it's operations
- Avoids generating redundant labels by the use of backpatching
- Dynamic memory allocation/management
- The stack pointer is set in an inline manner according to the current scope (This reduces number of instructions)
- Added support for using functions from the math library and string functions

Output of *'$ git show final'*:

```
commit edd87223dbb678e38fb78f4cfe6152c700be975c (HEAD -> main, tag: final)
Author: Supreeth Baliga <supreethbaliga@gmail.com>
Date:   Wed May 12 16:06:32 2021 +0530

    PPT of presentation added

diff --git a/PDFs/CS335-Group14.pdf b/PDFs/CS335-Group14.pdf
index 32db776..f8130b1 100644
Binary files a/PDFs/CS335-Group14.pdf and b/PDFs/CS335-Group14.pdf differ
```