CS335 Milestone 3

Group 17
Apoorva Gupta
Krish Sharma
Varun Tokas

April 21, 2024

1 Compilation Instructions

The following steps should be followed to create the parser

1. Go to directory `milestone3/src`
2. Build the parser using `make create`
3. Run the generated executable named `gmc` using command line argument `-h` to generate the help message

2 Execution Instructions

The following lines display a sample help message

- Usage: `./gmc -i <input\_file> -o <output\_file> -v`

```
Options:
-i, --input: Input file name [Default - stdin]
-o, --output: Output file name (for x86) [Default - asm_code.s]
-t, --tac: Output file name (for tac) [Default - tac.t]
-s, --symbol_table: Output file name (for symbol table) [Default - symbol_table.csv]
-v, --verbose: Debug mode
-h, --help: Display this help message
```

3 Tools used

The following tools were used and are required to be installed and configured to run the code:

1. **Flex :** Version 2.6.4

2. **Bison :** Version 3.8.2
3. **Graphviz (dot) :** `dot` version 2.43.0
4. **GNU Make :** Version 4.2.1
5. **GCC :** Version 9.4.0 (or later)

Please note that the system was built and tested solely on a Unix-like system and is designed to be run on input files with **LF** format newlines. Support for older Macintosh **CR** or Windows-style **CRLF** newline formatting is not present. As per discussion in class, the input is required to end in a newline without any indentation to be read properly by the lexical analyzer.
It is also required that the compiler have the `unistd.h` library to facilitate **exec()** calls in the parser code. This is utilized to directly compile the abstract syntax tree into pdf form from within the executable itself.

Features Implemented

- support for integer, boolean, and string data types. While we conduct type checking for floats, their execution is not currently supported.
- Control flow mechanisms such as if-elif-else blocks, while loops, and for loops with explicit ranges are implemented.
- Class functionalities encompass constructors, methods, and objects, with provision for multiple inheritance.
- Functions and classes operate within separate scopes to maintain encapsulation.
- Function calls can be made with or without return values, accommodating both primitive and composite data types.
- The system facilitates the creation of 1-dimensional lists comprising integers, booleans, strings.
- All fundamental operators are functional for integers and booleans.
- Relational operators are applicable to string comparisons.
- Essential functions such as print, range, and len are supported.
- Core programming constructs like control flow statements, recursion, and class definitions are incorporated.

Changes from milestone2

- A new 3AC instruction has been implemented to support printing strings, extending the functionality of the print operation.
- An additional 3AC instruction, return none, has been introduced to facilitate the clearing of registers in the 3AC process.
- The range function now supports ranges generated from the length of arrays, enabling more versatile iteration.

Not Support

- Accessing objects with objects.
- Floating point in X86
- In bool instead of true /false we are printing 1/0.
- While all specified language features have been embraced, various aspects of the language itself have been overlooked.

Effort Sheet

- All members have contributed equally