Compiling the compiler-Milestone3

Vaishnavi(211140) Mokshagna(210761) Kruthi(210088)

17 April 2024

1 Installing the required files:

Implementation of Lexer and Parser is in Flex and Bison. So install the following if not previously installed-(in Ubuntu)

```
sudo apt-get update
sudo apt-get install flex
sudo apt-get install bison
```

2 Instructions for execution:

The command line options include-
1)-i or -input : give the file name from which input is to be read
2)-o or -output : File name to which compiler outputs the 3AC
3)-v or -verbose : Prints the derivation (parse tree) on the command line
-h or -help : Manual page

Above commands are all optional, By default, input is read from test.py
For compilation-

```
$make
$./prob -i test.py -o output.txt > new1.s
$gcc -c new1.s -o new1.o -no-pie; gcc new1.o -o new1 -no-pie ;
$./new1 > result.txt
```

-On running the above commands, where input file is test.py, 3AC goes into output.txt and assembly file goes into new1.s.
-And the result obtained from assembly goes into result.txt
We have also included a header file classes.hpp for some class declarations
We also used bits/stdc++.h library for our implementation.
We have submitted 5 test cases, please make sure they are in LF and not in CRLF before running.
No manual changes required to the generated assembly file or to run it successfully.

3 Language features supported:

- We implemented a statically typed python language.
- We assumed that all uses of variables for the first time will include the type and will be initialized hence if it's followed, error will be shown.
- In assembly, we supported for int, 1-D list, bool, strings, classes(single and multi level inheritance and constructors)
- All basic operators given below are supported for int and bool(wherever applied):
  Arithmetic operators: $+, -, *, /, //, \%, **$

  Relational operators: $==, !=, >, <, >=, <=$

  Logical operators: and, or, not

  Bitwise operators: $\&, |, \wedge, \sim, <<, >>$

  Assignment operators: $=, += , -=, *=, /=, //=, \%=, **=, \&=, |=, \wedge=, <<=, >>=$
- Control flow via if-elif-else, for, while, break and continue.
- Supported the function len() for lists and strings.
- Supported Range() function with 1 or 2 arguments.
- Supported library function print() for int, bool, list access, basic operators on them and strings.
- Supported Recursion.
- Function arguments can be list, objects, list access, int, bool, string.
- Methods and method calls, including both static and non-static methods
- We assumed that main() function will always be defined.

4 Modifications of 3AC from Milestone2 to Milestone3:

1)Changed the way list-base appears during list initialization and list access(say for a list A, before it was t1=list_base, now it changed to t1=list_base_a)
2)Changed the format of function names in objects.
3) Changed the format for storing return values of functions if any.

5 Required features- unsupported

1)Static polymorphism via method overloading isn't supported.
2)Relational operators on strings.

6 Effort Sheet:

Vaishnavi (211140) - 33.33%
Mokshagna (210761) - 33.33%
Kruthi (210088) - 33.33%