CS335 (Compiler Design) : Project Milestone 4

Deepak Sangle
200860
Department of Computer Science
sangleds20@iitk.ac.in

Shreyasi Mandal
200956
Department of Computer Science
shreyansi20@iitk.ac.in

Vartika
201089
Department of Computer Science
vartikag20@iitk.ac.in

JAVA17 Compiler

The goal of this project is to implement a compilation toolchain, where the input is in Java language and the output is x86_64 code. In this milestone, we have implemented a translator to generate x86_64 assembly instructions from the three-address code generated in the previous milestone.

Language Features Supported

Basic Features

- Basic Operators: $+$, $-$, $*$, $/$, $++$, $--$, $\hat{}$, !, $\sim$
- Assignment Operators: $+=$, $-=$, $*=$, $/=$, $\&=$, $|=$, $\hat{}=$, $\%=$
- Conditional Operators: $==$, $!=$, $>=$, $<=$, $>$, $<$, $||$, $\&\&$
- Shift Operators: $<<$, $>>$
- Control Flow: if-then, if-then-else, for and while
- Primitive data types: int, byte, short, long
- 1D arrays with primitive data types
- Methods and method calls
- Recursion
- Library function println()
- Classes and objects

Basic Features - Not Supported

- Multi-Dimensional Arrays

Optional Features Supported

- Do-While Loop

Example

```
class arrays {
    public void main() {
        int arr[] = new int[10];
        for(int i=0;i<10;i++){
            arr[i] = i * 5;
            System.out.println(arr[i]);
        }
        return;
    }
}
```

The above code will generate the following assembly x86_64 code -

```
.LC0:
  .text
  .string "%lld\n"
  .globl main
main:
  pushq %rbp
  movq  %rsp, %rbp
  subq  $96, %rsp
  pushq %r12
  pushq %rbx
  pushq %r10
  movq  $0,%r12
  movq  %r12, -88(%rbp)
.L1:
  movq  -88(%rbp),%r12
  movq  $10,%rbx
  cmpq  %rbx, %r12
  setl  %r12b
  movzbq  %r12b, %r12
  cmpq  $0, %r12
  je  .L2
  movq  -88(%rbp),%r12
  movq  -88(%rbp),%rbx
  movq  $5,%r10
  imulq %r10, %rbx
  leaq  -80(%rbp), %r10
  movq  %rbx, (%r10, %r12, 8)
  movq  -88(%rbp),%r12
  leaq  -80(%rbp), %rbx
  movq  (%rbx, %r12, 8), %r12
  movq  %r12, %rsi
  leaq  .LC0(%rip), %rdi
  movq $0, %rax
  call  printf
.L3:
  movq  -88(%rbp),%r12
  movq  %r12, %rbx
  addq  $1, %r12
  movq  %r12, -88(%rbp)
  jmp .L1
.L2:
  popq  %r10
  popq  %rbx
  popq  %r12
  addq  $96, %rsp
  leave
  ret
```

Execution Instructions

To execute the code, run the following commands in the given sequence -

```
cd src
make
./java-assembler --input=../tests/test_1.java --output=./test_1.s
make asm test_1.s
./test
```

Error messages

- **Conditional expression** :- Give error for type mismatch of operands.
  For example: int k = (1)?5:9 ; It will give error for first operand that must be boolean and error will be like this - "int cannot be converted to boolean."
- **And(&&) & OR(||) expressions** :- Give error if either or both operands are not boolean.
  For example: boolean b = true||7; It will give error for second operand that must be boolean and error will be like this - "bad operand types for binary operator '||' "
- **Bitwise expression** :- Give error if either or both operands are not integer data types
  For example: long v = 125 & 5.2; It will give error for second operand that must be integer data types and error will be like this - "bad operand types for binary operator '&' "
- **Equality and Inequality expression** :- Give error if both operands are incomparable
  For example: boolean v = true==1 ; It will give error for both operand that must be of same type and error will be like this - "Incomparable types: boolean and int"
- **Relational expression** :- Give error if both operands are incomparable(operands can be of short,int,float or double type)
  For example: boolean v = 5.62 < true; It will give error for second operand that must be of int or double or long and error will be like this - " bad operand types for binary operator '<' "
- **Shift expression** :- Give error if either of operands are not of type char or short or int or long
  For example: long t=22«8.2; It will give error for second operand that must be of char or byte or short or int or long and error will be like this - " bad operand types for binary operator '«' "
- **Arithmetic expression** :- Give error if either of operands are of not of type char or short or int or long or float or double
  For example: int a = 4*6 + false ; It will give error for second operand that must be of char or byte or short or int or long or float or double and error will be like this - " bad operand types for binary operator '+' "
- **Unary plus/ minus expression** :- Give error if operand is boolean or array or string or void
  For example: boolean a = +true ; It will give error for second operand that must not be of boolean or array or string or void type and error will be like this - " bad operand type boolean for unary operator '+' "
- **Increment and decrement expression** :- Give error if operand is boolean or array or string or void
  For example: boolean a = –false; It will give error for operand that must not be of boolean or array or string or void type and error will be like this - " Bad operand types for decrement operator"
- **Tilde operator** :- Give error if operand is not char, byte, short, int, long
  For example: double d = 5.2; It will give error for operand that must not be of boolean or array or string or void type and error will be like this - " bad operand type double for unary operator ' ' "
- **Assignment operator** :- Give error if operands are of invalid types for assignment.
  For example: int d = 5.2+4*3; It will give error for operand that cannot be assigned and error will be like this - " incompatible types: possible lossy conversion from double to int "

Tools Used

The following tools are used for creating **AST Generator**.

- Flex - This is used for the lexer implementation of the java program
- Bison - This is used to generate the parser

Contributions

SL. No.	Member Name	Roll Number	Member Email	Contribution (%)
1	Deepak Sangle	200860	sangleds20@iitk.ac.in	40
2	Shreyasi Mandal	200956	shreyansi20@iitk.ac.in	30
3	Vartika	201089	vartikag20@iitk.ac.in	30