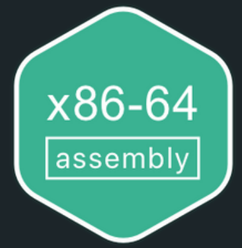
April 26, 2023

WCTC
CS335 – COURSE PROJECT
JAVA TO X86_64



WHO COMPILES THE COMPILER?

KNOWLEDGEMENT ACKNOW

KNOWLEDGEMENT ACKNOW
KNOWLEDGEMENT ACKNOW

We would like to thank our professor, Mr. Swarnendu Biswas for teaching this course and pushing us to do something which seemed absolutely impossible at first. Bringing this project to fruition was stressful, but fun at the same time. The satisfaction on seeing the code compile correctly was exhilarating.

We would like to thank our TA, Mr. Deepak Raj for their feedback.

We feel honored to be one of the last few batches to do this course as a department compulsory.

Milestone 1

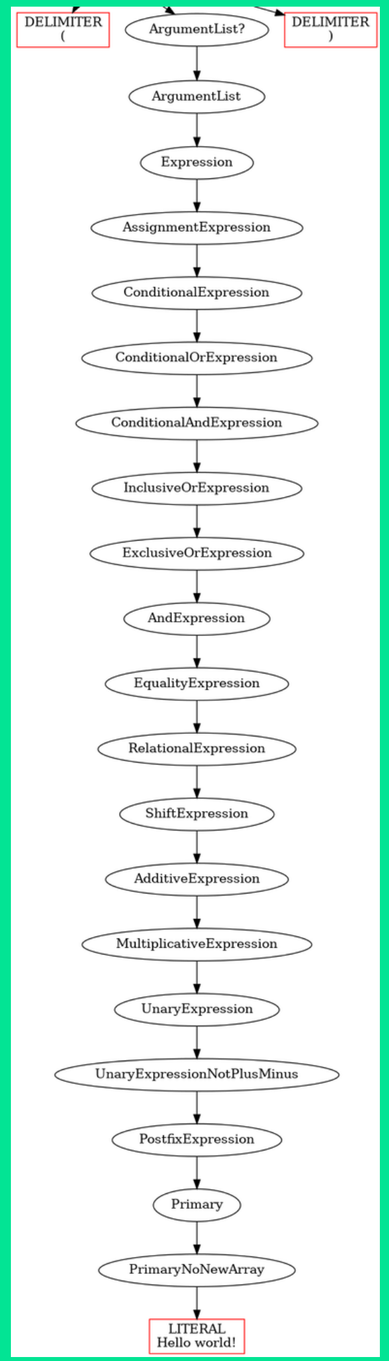
- Used Flex for lexical analysis and Bison for parsing
- Removed conflicts by -
  - Removing epsilon transitions
  - Introducing dummy non-terminals
  - Introducing operator precedence and associativity
  - Delegating some checks for future phases
- Used another lexer to construct the parse tree

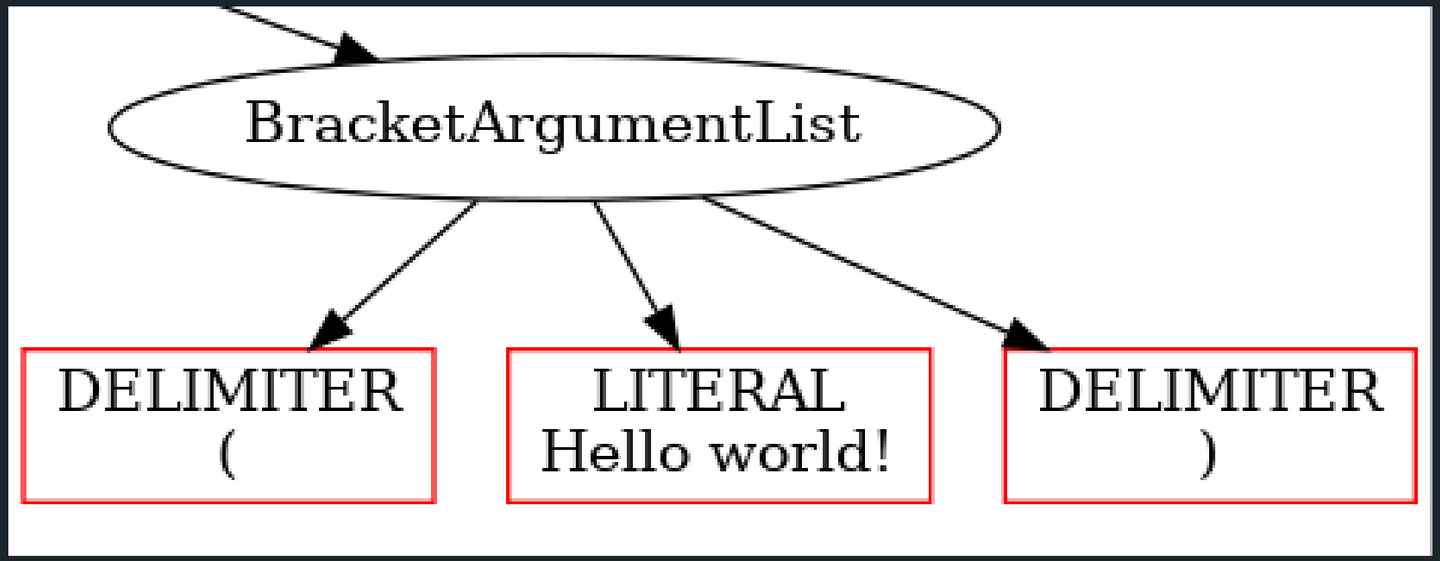
Milestone 1

- Constructed the AST from Parse Tree by -
  - Removing nodes with a single child
  - Removing leaves like ';' and other delimiters
  - Promoting operators
- Increased readability by -
  - Printing line numbers of nodes
  - Introducing color coding for operators, keywords, IDs, etc.

Delimiter Keyword Identifier Operator

LESTONE 1 MILESTONE 1 MILESTONE 1 MILESTONE 1 MILESTONE 1 MILESTON


DELIMITER
(
ArgumentList?
DELIMITER
)
ArgumentList
Expression
AssignmentExpression
ConditionalExpression
ConditionalOrExpression
ConditionalAndExpression
InclusiveOrExpression
ExclusiveOrExpression
AndExpression
EqualityExpression
RelationalExpression
ShiftExpression
AdditiveExpression
MultiplicativeExpression
UnaryExpression
UnaryExpressionNotPlusMinus
PostfixExpression
Primary
PrimaryNoNewArray
LITERAL
Hello world!


BracketArgumentList
DELIMITER
(
LITERAL
Hello world!
DELIMITER
)

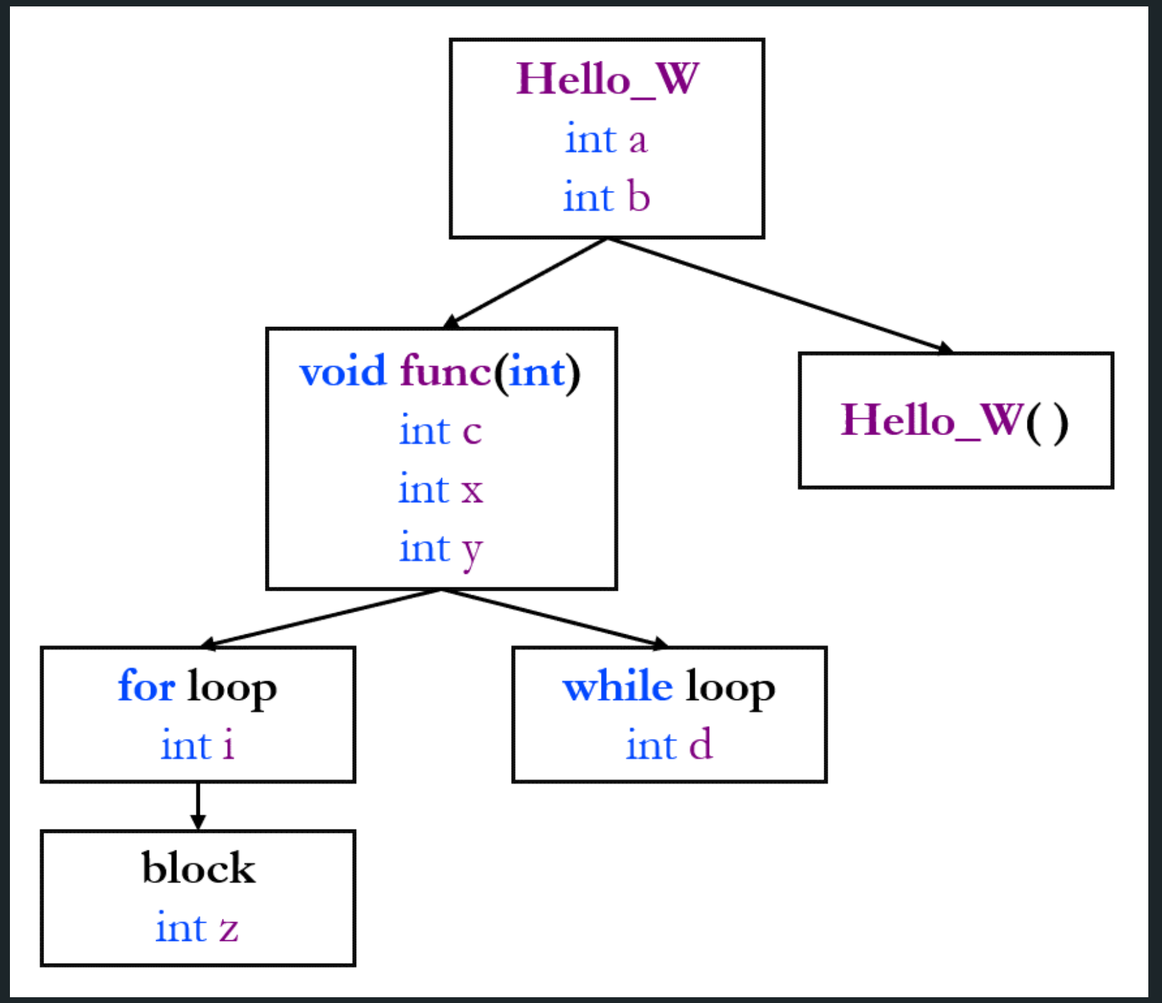
Milestone 2

- Walked the AST multiple times
  - Walk 1 - Establishes scope hierarchy, initializes class and method symbol tables
  - Walk 2 - Populates symbol tables, ensures declaration and initialization before use
  - Walk 3 - Type checking
  - Walk 4 - Modifier checking
- Between walks, other checks were done -
  - Default constructors added to classes
  - Class sizes, and field member offsets calculated
  - Function local variable offsets calculated

```
class Hello_W {
    int a, b;
    public void func(int c){
        int x = 10;
        int y = 20;

        for(int i=0; i<20; i++){
            x += 1;
            y = 2*i;
            {
                int z = x + y;
                z *= c;
            }
        }

        while(y > 0){
            int d = 20;
            c += d;
            y--;
        }
    }
}
```



Milestone 2

- 3AC Generation
  - Use temporaries corresponding to nodes in AST
    - Guarantees uniqueness
    - Easy to retrieve
  - Position numbers
  - Populate jump statements with relative values
  - Replace with absolute values at the end

goto J+33
↓
302: goto J+33
↓
302: goto 335

LESTONE 2 MILESTONE 2 MILESTONE 2 MILESTONE 2 MILESTONE 2 MILESTO

Milestone 3

- 3AC Optimization
  - Conservative - used
  - Removed redundant temporaries using copy propagation - introduced due to template code generation
  - Renamed temporaries and line numbers
- Name mangling using function parameters
- Stack pointer manipulation for function calls
  - Space for parameters, return address and function locals
  - Register space ignored to keep 3AC architecture independent

Milestone 4

- More 3AC optimizations!
  - Constant Folding + Propagation
  - Strength Reduction - Algebraic Simplifications
- Template x86 instructions for each 3AC instruction
- 16 byte Stack alignment for allocmem and print

Milestone 4

- Dynamic array support via heap allocation
  - Dimension sizes stored as metadata at base address
  - Arrays can have any number of dimensions
  - Array sizes can be determined at compile time*
- Object creation support via heap allocation
  - Field member offsets stored in symbol table
  - Any field member can be accessed using "this" variable

Milestone 4

- Activation record follows x86_64 conventions
  - Caller responsibility:
    - Push caller saved registers
    - Push parameters
    - Call function (push return address)
  - Callee responsibility:
    - Push old base pointer
    - Push callee saved registers
    - Space for locals and temporaries

Unsupported Features

- Support for smaller types
  - For better stack manipulation, we supported full fledged type-casting till 3AC only
- Access without the 'this' pointer
  - Field members can however be accessed as 'this.x'
- Static Variables
  - Support for static methods and method calls, but not for variables

BONUS BONUS BONUS BONUS

ARRAYS

Arbitrary number of dimensions and dynamic sizes

STATIC POLYMORPHISM

Function Overloading

3AC OPTIMIZATION

Algebraic Simplification, Constant Folding, and Constant Propagation

PRIMITIVE TYPE CASTING

Both explicit and implicit typecasting

DEFAULT CONSTRUCTOR

Supplied by the compiler in the absence of user defined constructor

DO WHILE LOOP

Implemented similarly to the while loop

HANK YOU THANK YOU THA
HANK YOU T HANK YOU TH A
HANK YOU T A
HANK YOU T A

ADITYA TANWAR (33%)
tanwar20@iitk.ac.in
200057

JANHVI ROCHWANI (30%)
janhvir20@iitk.ac.in
200467

SOHAM SAMADDAR (37%)
sohams20@iitk.ac.in
200990

github.com/cliche-niche/WCTC

JAZ Hands