

Lexical analyzer

token stream

Syntax analyzer

AST

Semantic analyzer

Symbol table, AST

Code generator

moon code

← { ① DFA  
②. implement the DFA to  
to tokenize the source file.

← { ①. eliminate left recursion and ambiguity.  
②. generate first sets and follow sets.  
③ recursive descent predictive parsing.

← { ① Convert previous AST to a new AST to  
to fit ast visitor.  
② generate symbol table.  
③ type checking

← { ①. update symbol table, add temp var  
for expression and function call  
②. compute member size and offset.  
③. stack based code generator.

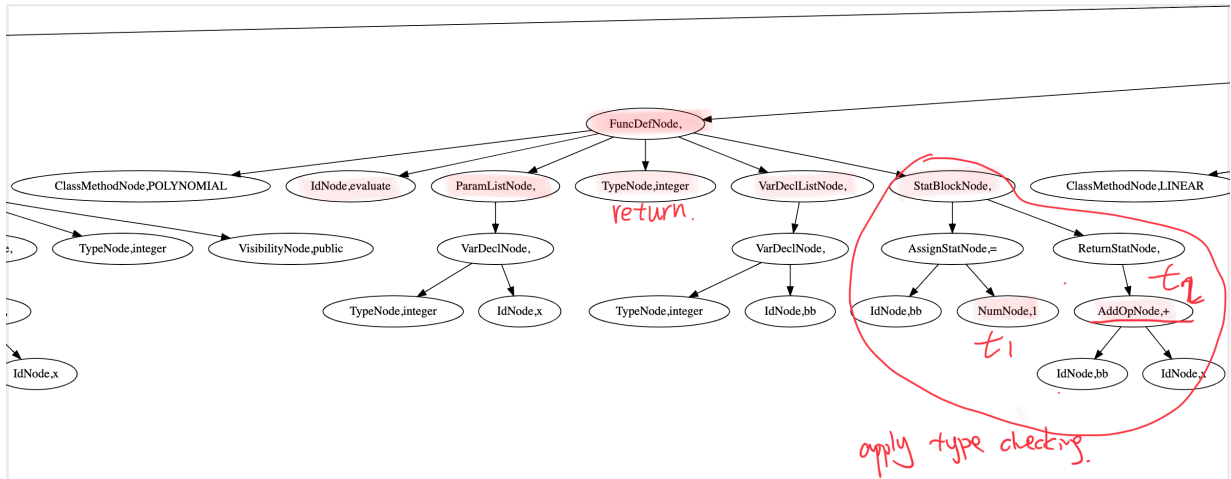
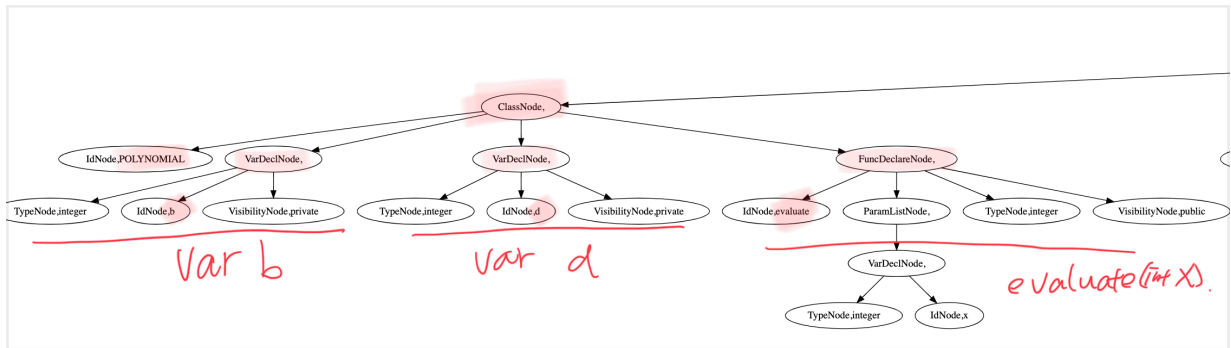
```
class POLYNOMIAL{
    private integer b;
    private integer d;
    public func evaluate(integer x) : integer;
};

class LINEAR inherits POLYNOMIAL
{
    private integer a;
    private integer b;
    POLYNOMIAL polynomial_obj;           // object data members

    public func evaluate(integer x) : integer;
};

// ===== Function Definitions ===== //

func POLYNOMIAL::evaluate(integer x) : integer
{
    var
    {
        integer bb;                       // if x = 2, then return 3;
    }
    bb = 1; t1
    return(bb+x);                          // the function has a return
} t2.
```



class   POLYNOMIAL					
table: POLYNOMIAL   scope size: 8					
			Size	offset	
var	b	integer	4	-4	
var	d	integer	4	-8	
func	evaluate	integer			
table: evaluate   scope size: 24					
param	x	integer	4	-12	
var	bb	integer	4	-16	
litval	t1	integer	4	-20	
tempvar	t2	integer	4	-24	

added in type checking stage.

added in computer member size stage.

```

evaluate0 sw -4(r14),r15 } return link.
% processing: t1 := 1
addi r1,r0,1 } save 1 to t1
sw -20(r14),r1
% processing: AssignStatNode(bb=1;)
% processing: bb := t1
lw r2,-20(r14) } t1 -> bb.
sw -16(r14),r2
% processing: t2 := bb+x
lw r1,-16(r14),bb
lw r4,-12(r14),x
add r5,r1,r4 } bb+x
sw -24(r14),r5 } r5 -> t2.
% processing: return(return(bb+x);)
lw r3,-24(r14),t2
sw 0(r14),r3 } t2 -> return value slot.
lw r15,-4(r14),t2
jr r15 } return
  
```

offset.

bb -16

t1 -20.

x -12.

t2 -24.

