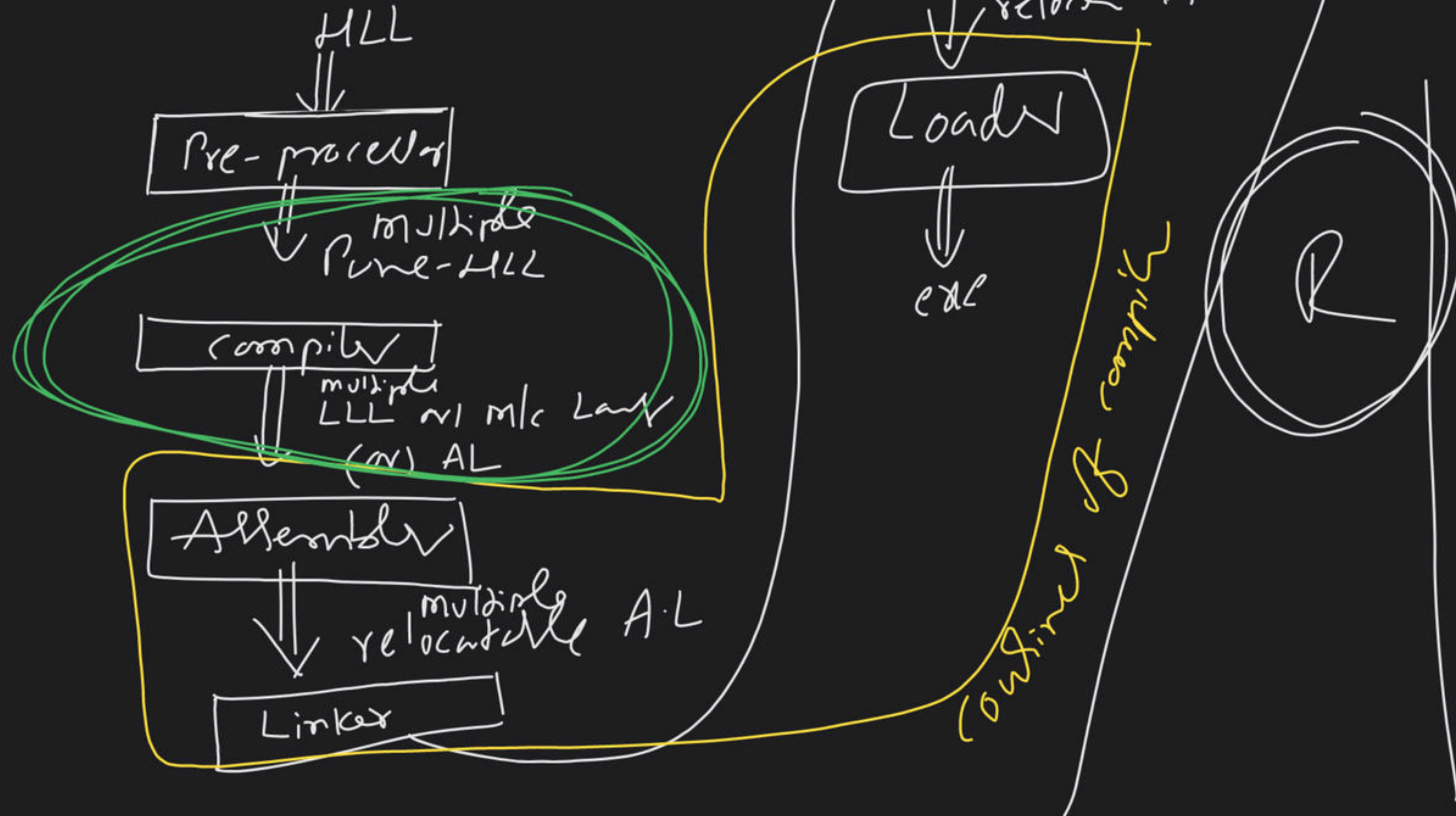# Compiler Design (6-10)
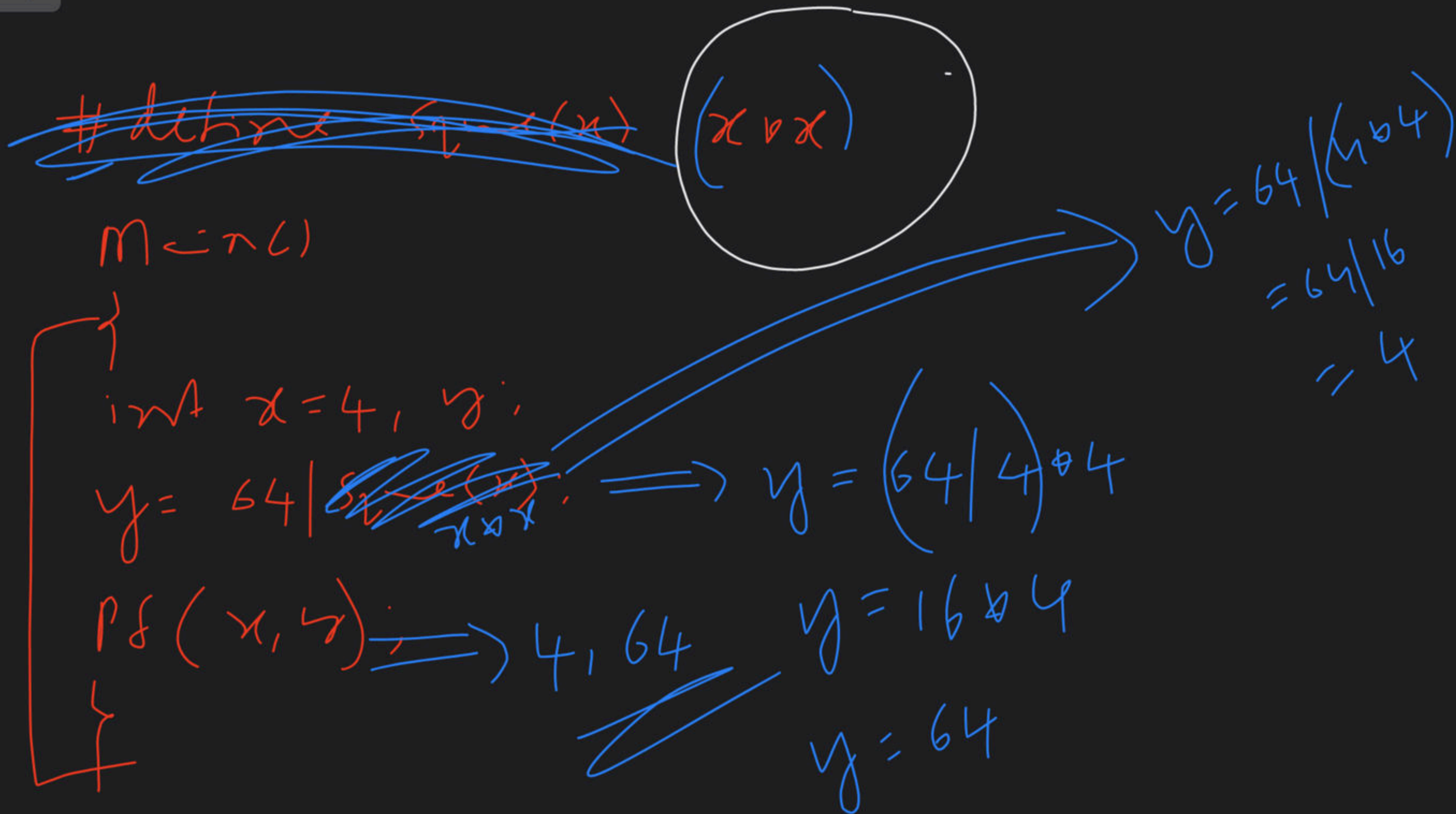
① Introduction

② Lexical Analyzer

③ Syntax " (Parser)

④ Syntax Directed Translation

⑤ Runtime Environment

⑥ Intermediate Code Generation

⑦ Code optimization
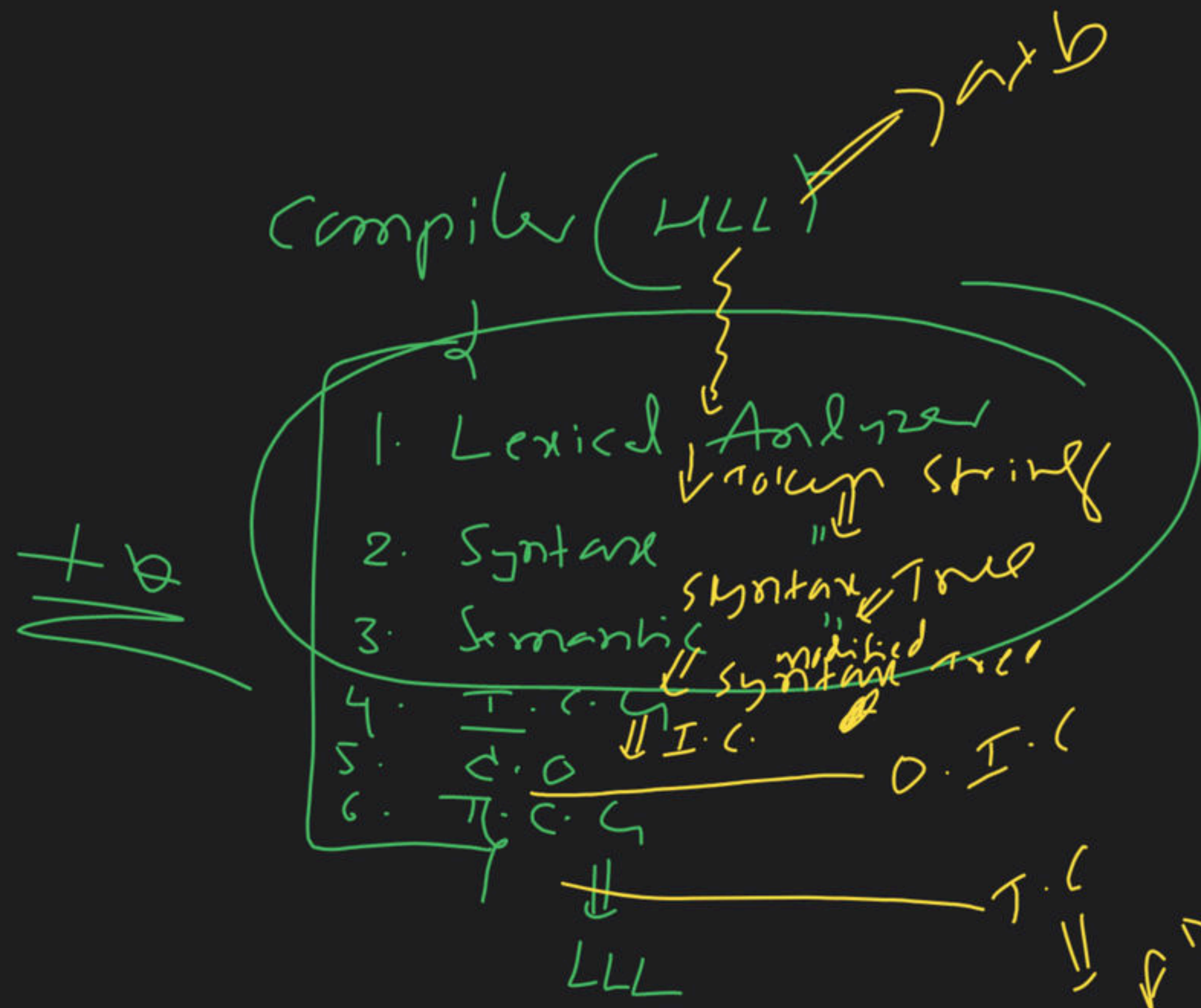
=> my Notes

Introduction

HLL

Pre-processor

multiple
Pure-HLL

compiler

multiple
LLL ~ m/c Lang
(~) AL

Assembly

multiple
relocatable A.L

Linker

Single
relock - AL

Loader

exe

Combined of compiler

R

#define ~~f,~~~~(x)~~ $(x \triangleright x)$

$M = \pi()$

int $x = 4$, y;

$y = 64 | \cancel{x \cdot x}:$ $\Rightarrow$ $y = (64 | 4) \triangleright 4$

$pf(x, y): \Rightarrow 4, 64$ $y = 16 \triangleright 4$

$y = 64$

$y = 64 | (4 \triangleright 4)$

$= 64 | 16$

$= 4$

Compiler Introduction

Compiler (HLL) $\longrightarrow$ $a \times b$

1. Lexical Analyzer
   Token String
2. Syntax
   Syntax Tree
   modified
   Syntax Tree
3. Semantic
4. T.C.C   I.C.
5. C.O   O.I.C
6. T.C.G

T.C

$\rightarrow$ P.D.P. R.P.

LLL

int a = 10
int a = 20

ex