CS & IT ENGINEERING

Compiler Design

Lexical Analysis & Syntax Analysis

Lecture No. 1

By- DEVA Sir

TOPICS TO BE COVERED

01 Introduction to Compiler Design

02 Important Topics

03 Weightage in GATE

04 How to Prepare CD?

05 Doubts

① phases of a Compiler

② Lexical Analysis

→ *** ③ Syntax Analysis

④ Syntax Directed Translations

** ⑤ Intermediate Code Generation

→ ** ⑥ Code optimization

⑦ Run Time Environments

$\underbrace{\text{5M — 7M}}$

15 days

$\leq$ 30 Hours

How to prepare CD?

$\hookrightarrow$ 1st : Regular to class

2nd : Notes $\Longrightarrow$ practice

3rd : GATE PYQs

Doubts:

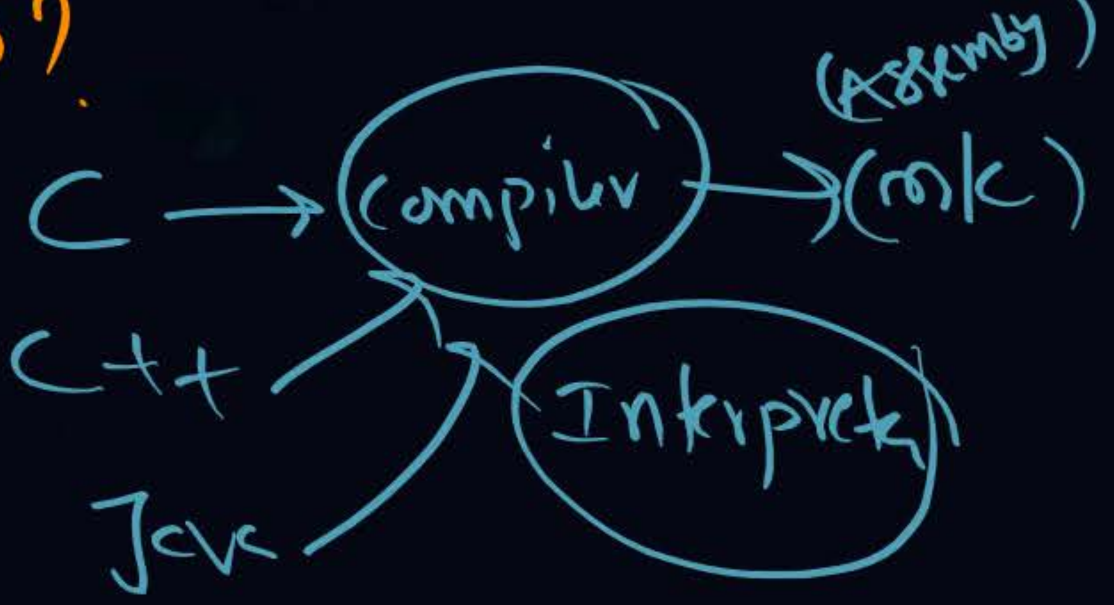
↳ 1) Why to study CD?

↳ 1st: GATE syllabus

2nd: programming knowledge

2) practice applications of CD?

↳ Compiler



3) IS TOC required?

↳ Depends on faculty

For DEVA Sir:

↳ To learn CD, TOC not required

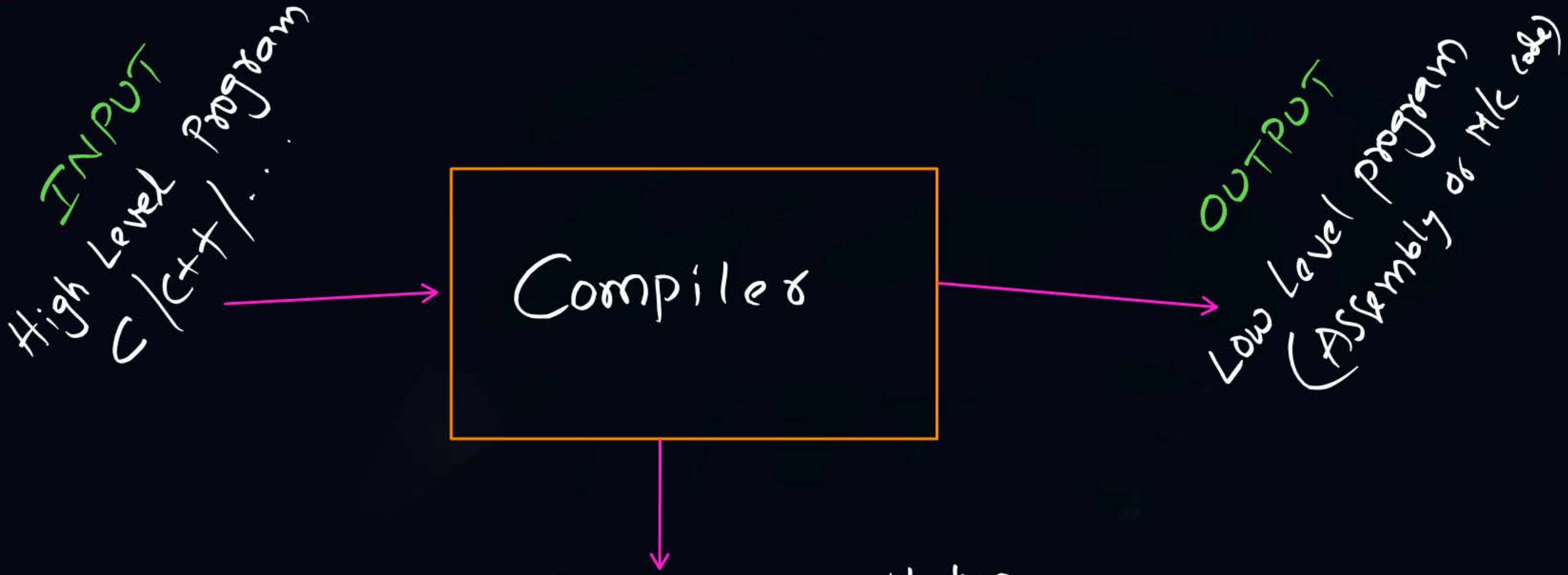
while learning CD,

↳ I will take care of TOC concepts inside CD.

Introduction:

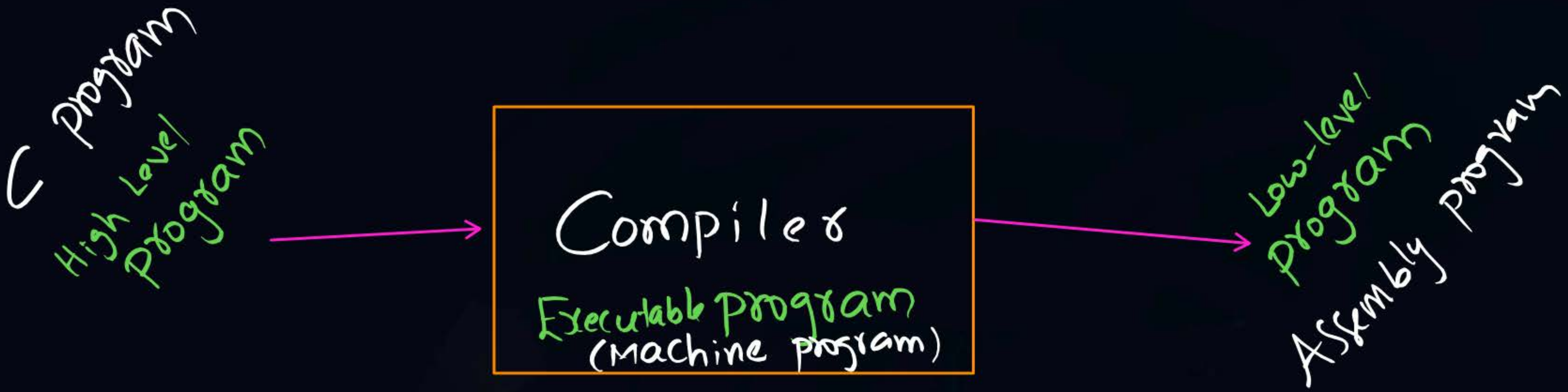
1) What is compiler? ⟶ Translator
2) Where we use Compiler? ⟶ Language Translation
3) How compiler is designed? ⟶ 7 phases
4) Types of Errors in programming?

Compiler:



Also reports compilation errors if any

Compiler:


C program
High Level Program
Compiler
Executable program
(Machine program)
Low-level Program
Assembly program

How the compiler was Implemented ?

M/C code

Assembly code

} Today, we don't use

$\boxed{\text{High level code}}$

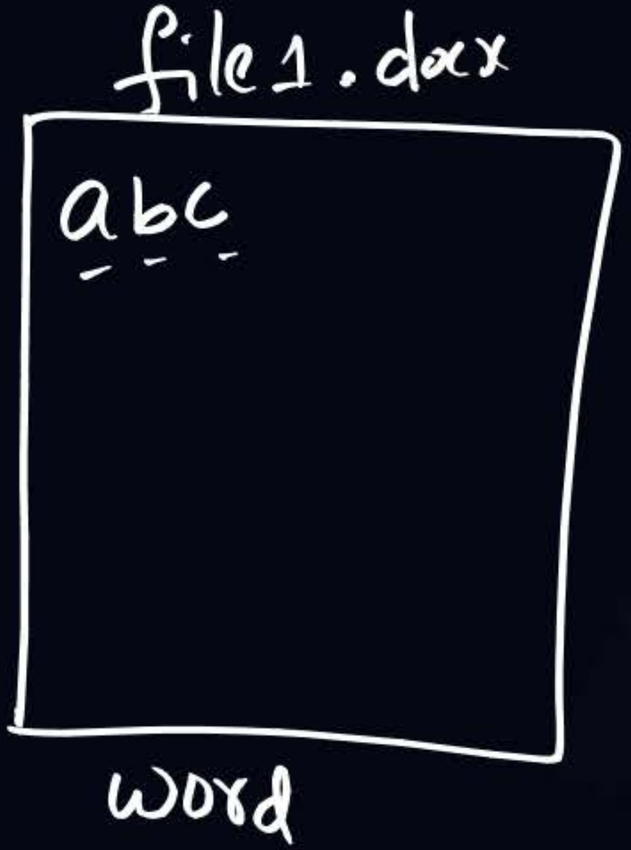
- → C
- → C++
- → Java

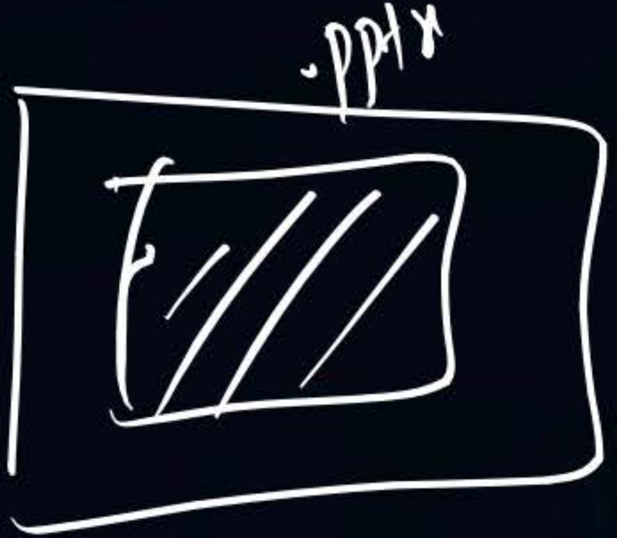
Translator :

$$\begin{tikzcd}\text{Source} \arrow[r] & \boxed{\text{Translator}} \arrow[r] & \text{Target}\end{tikzcd}$$

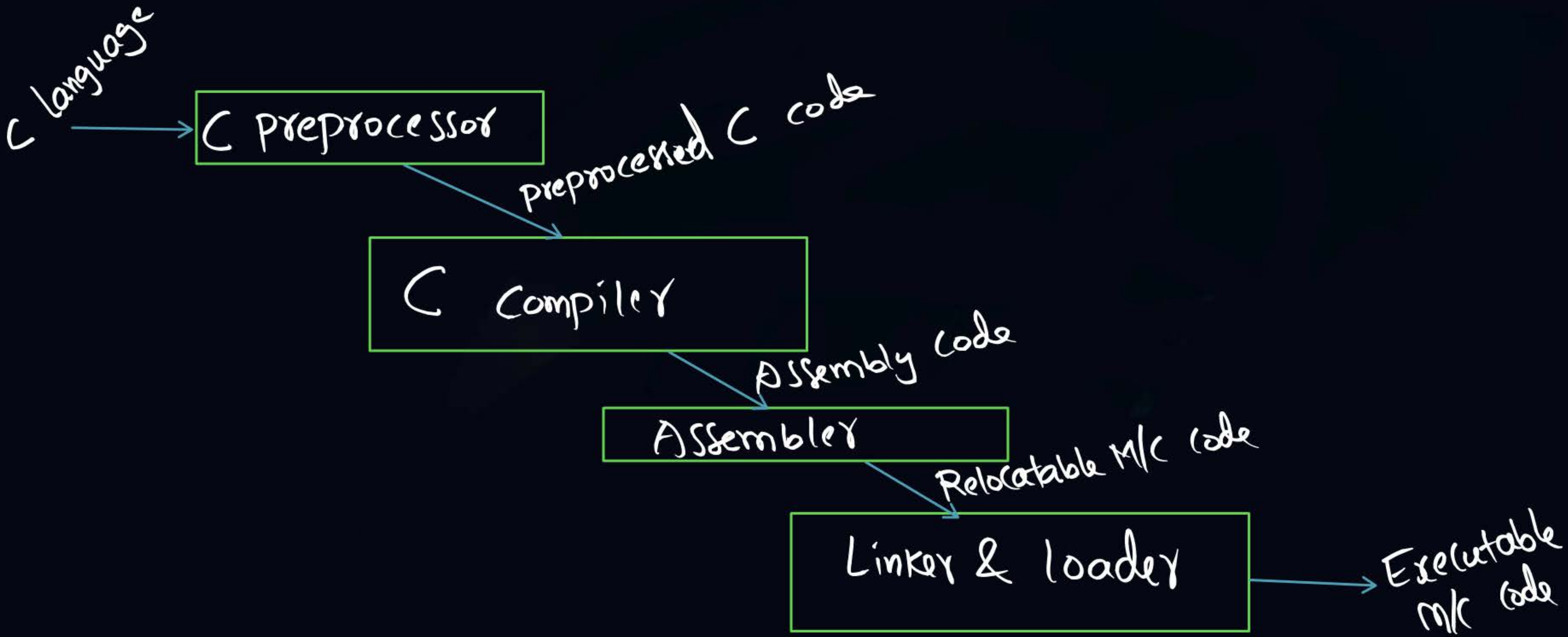
1) Compiler
2) Interpreter
3) Editor
4) preprocessor
5) Assembler
6) Linker
7) Loader
8) spell checker
9) Word | Excel | powerpoint | . . .

Is Word Translator ?

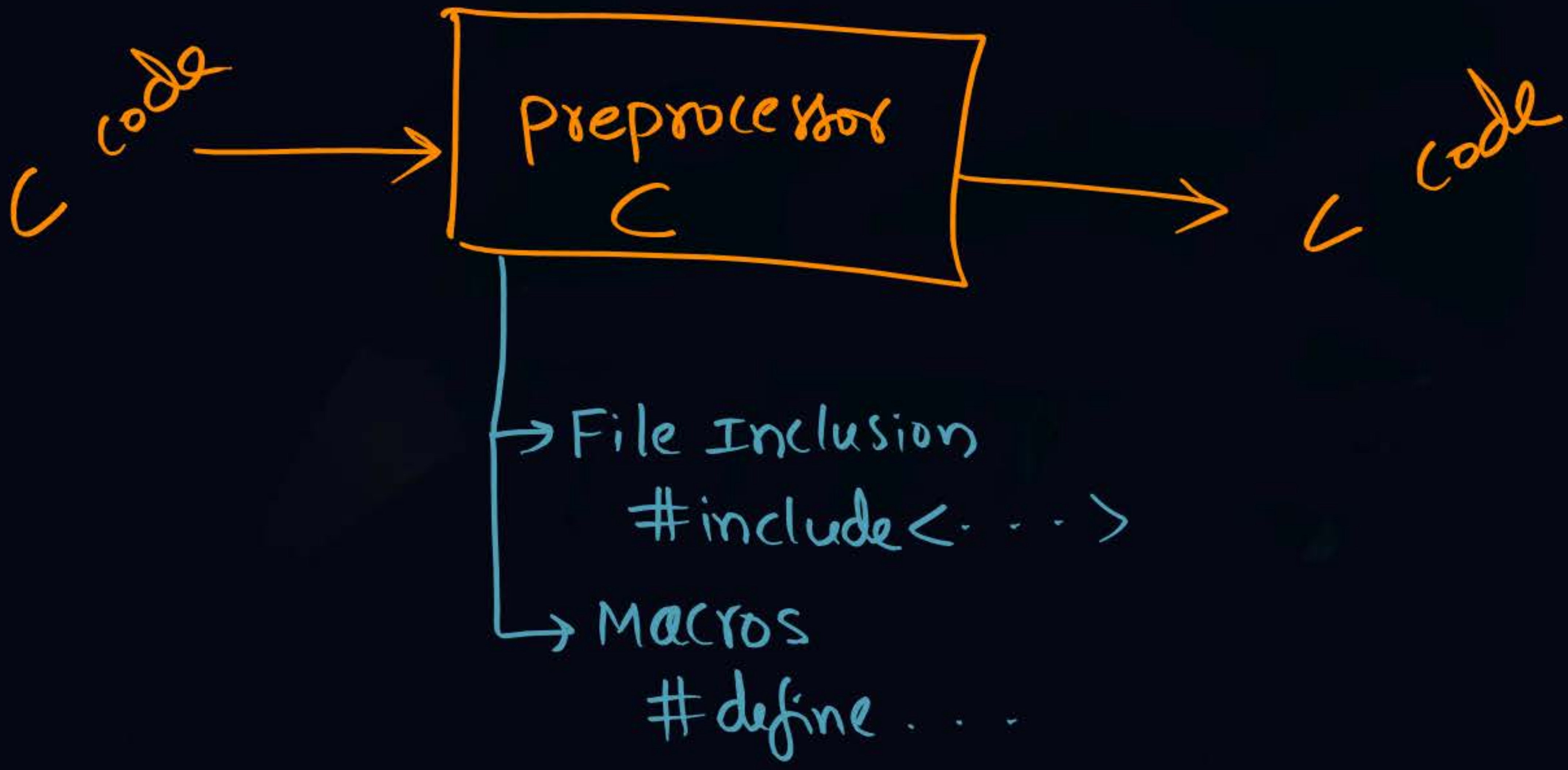


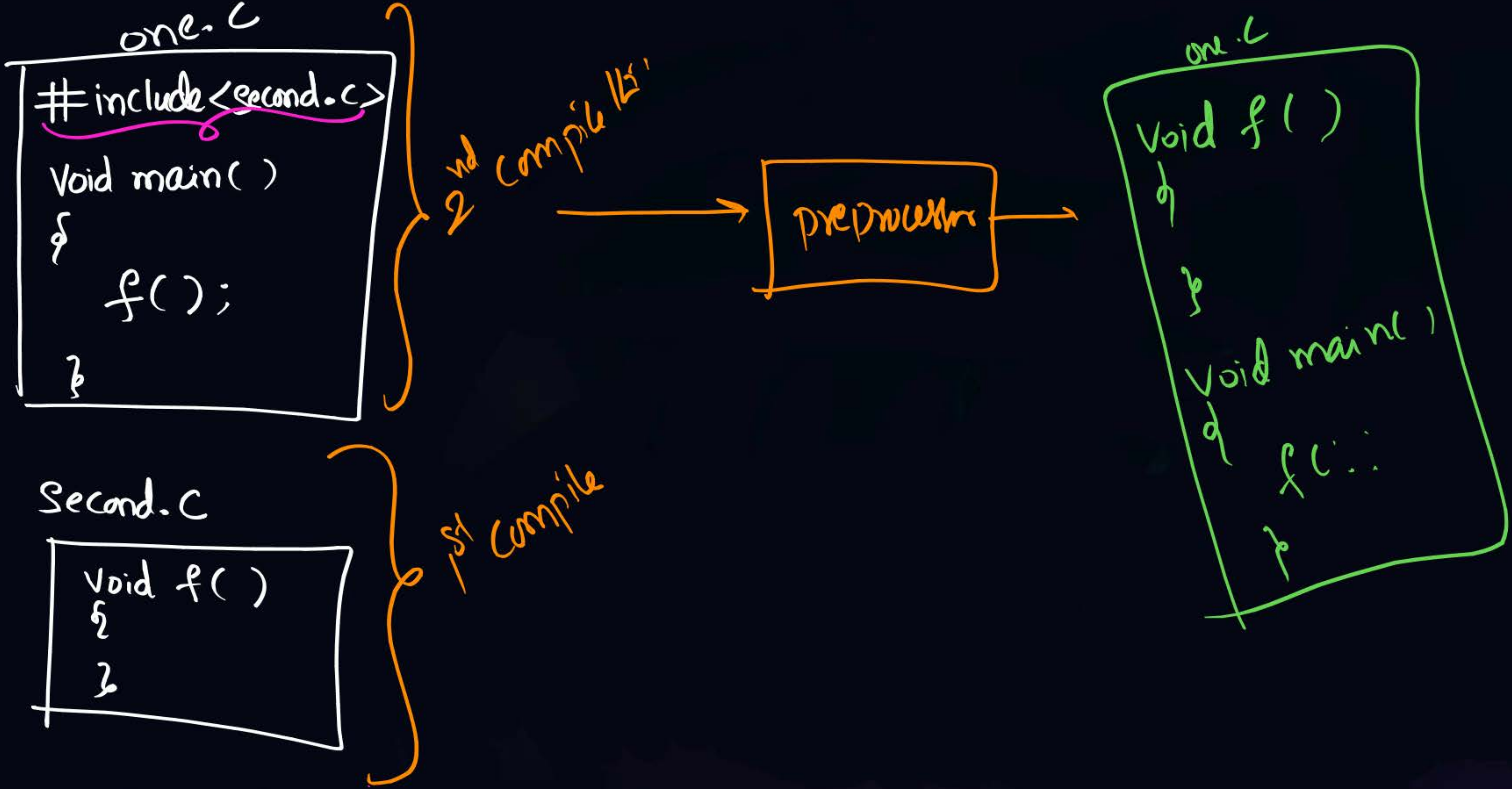


Language Translation : [C language $\Longrightarrow$ Executable code]



Preprocessor :





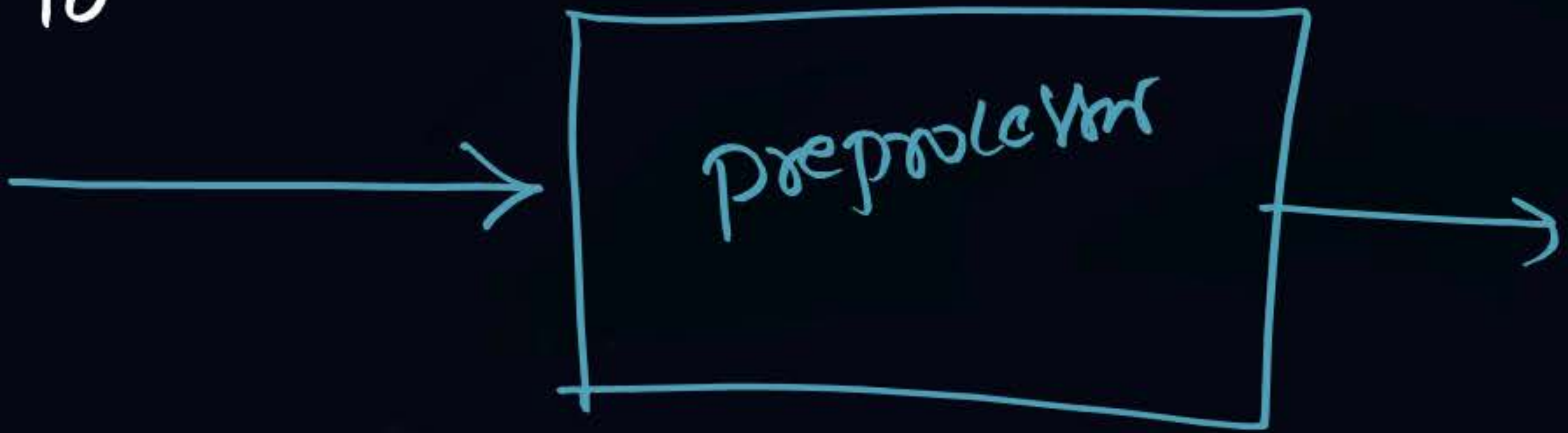
```
#include<second.c>
Void main( )
{
    f( );
}
```

```
Void f( )
{
}
```

```
Void f( )
{
}
Void main( )
{
    f( );
}
```

$\#\text{define}\ \overbrace{\text{MAX}}^{\text{constant}}\ \overbrace{10}^{\text{Expression}}$

```
int x=MAX;
Void main( )
{
    int y=MAX;
}
```



```
int x=10;
void main( )
{
    int y=10;
}
```

#define constant exp

Replace constant with exp.

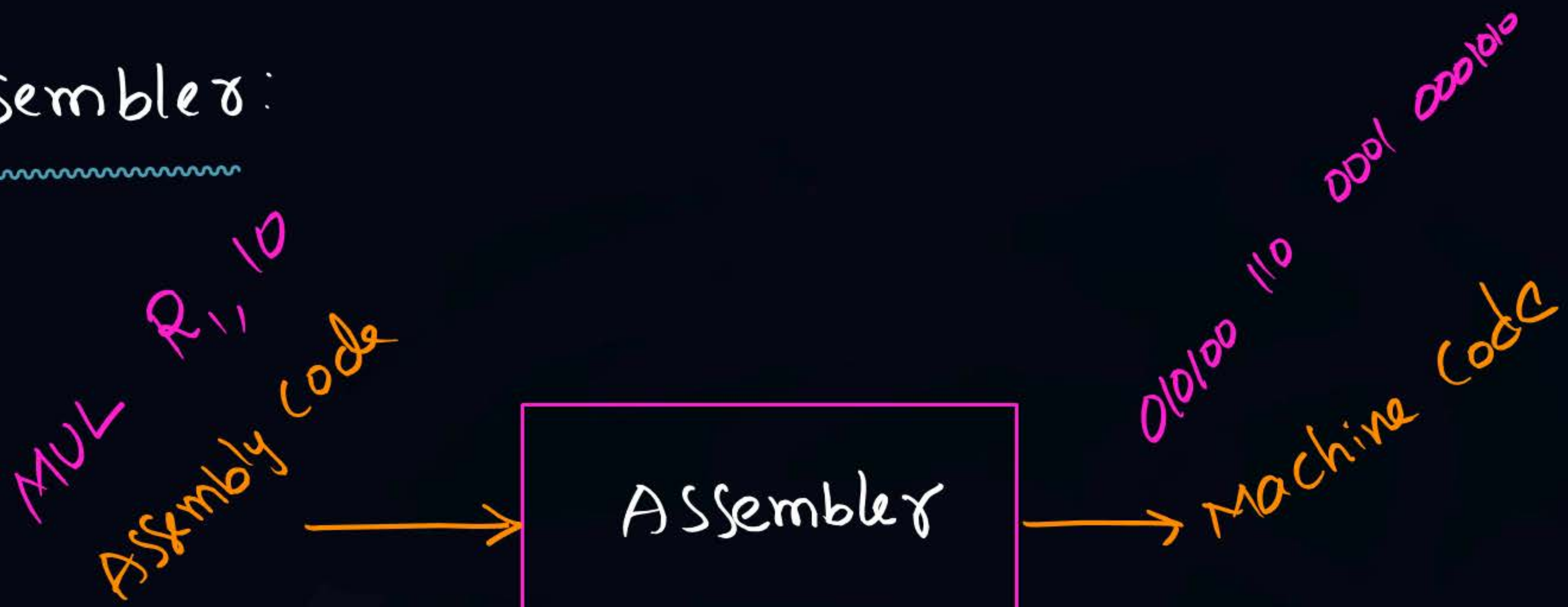
$\underbrace{\#\ \ldots\ldots}_{\text{preprocessor statement}}$ in C

#define . . -
#include . . .
#if
#else
#endif
:
,
,

$\underbrace{\#\text{incde<stdio.h>}}_{\text{preprocessor error}}$ ✗

#include<stdio.h> ✓

Assembler:


MUL R1, 10
Assembly code
Assembler
010100 110 0001 0001010
Machine code

Linker

↳ It resolves all external references

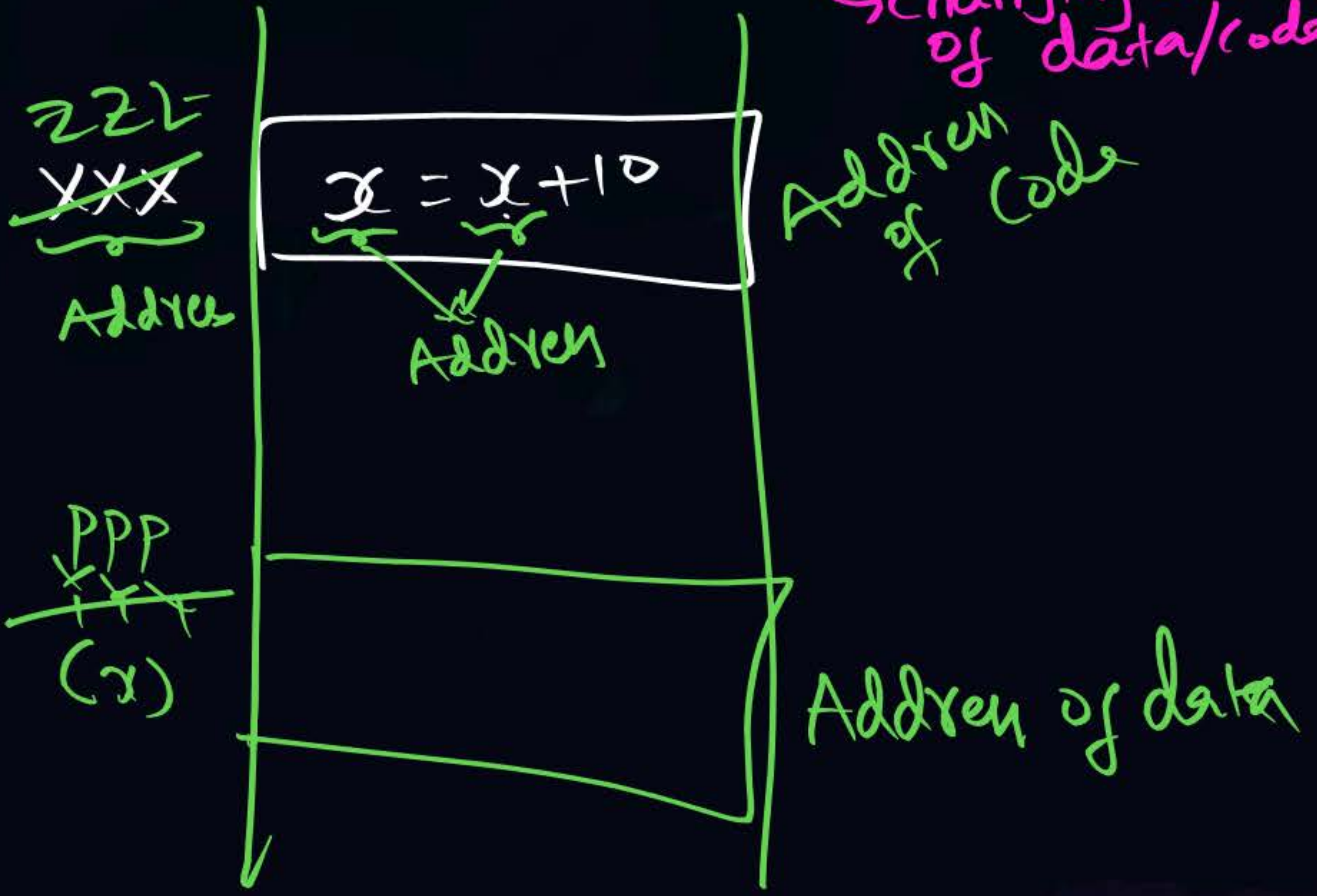
```
extern int x;

void main()
{
    printf("%d", x);
}
```

what are external references?

Loader

↳ It performs relocation
↳ changing address of data/code



High level
(m/c Independent) C code $x = x + a;$

Low level
(m/c Dependent)
$\left\{ \begin{array}{l} \text{A code} \\ \\ \text{M code} \\ \text{(B code)} \end{array} \right.$

ADD R_1, R_2 ; $R_1 \leftarrow R_1 + R_2$

$\underbrace{010100}_{\text{ADD}}$ $\underbrace{011}_{\text{type of Addressing mode}}$ $\underbrace{0001}_{R_1}$ $\underbrace{0010}_{R_2}$

Loaders

- → Boot Strap Loader
- → Swapper
- → other Loaders

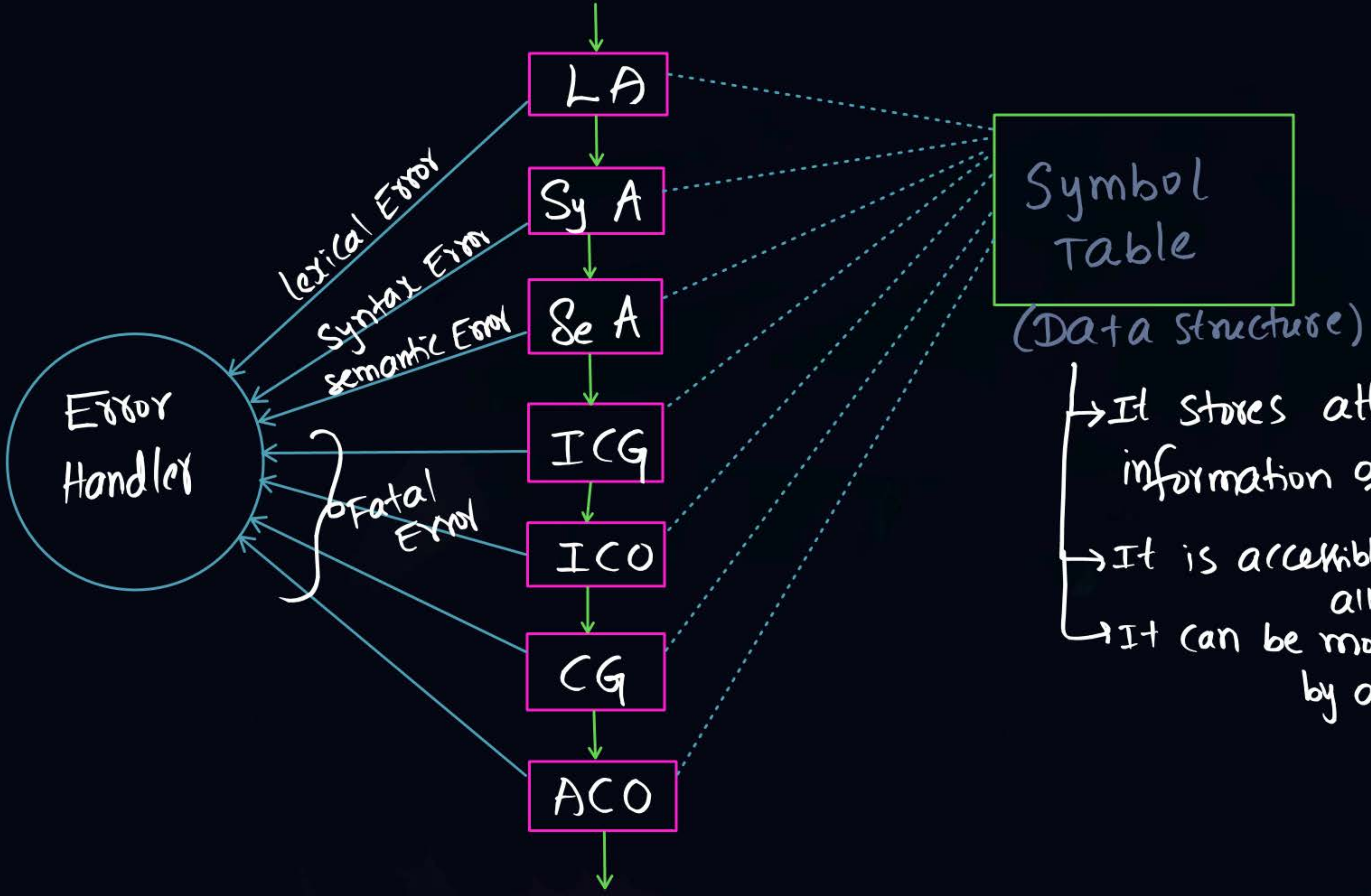
In Language Translation : Loader performs relocation

In general : Relocation, loading modules/programs/applications/. . .
Allocation/Deallocation

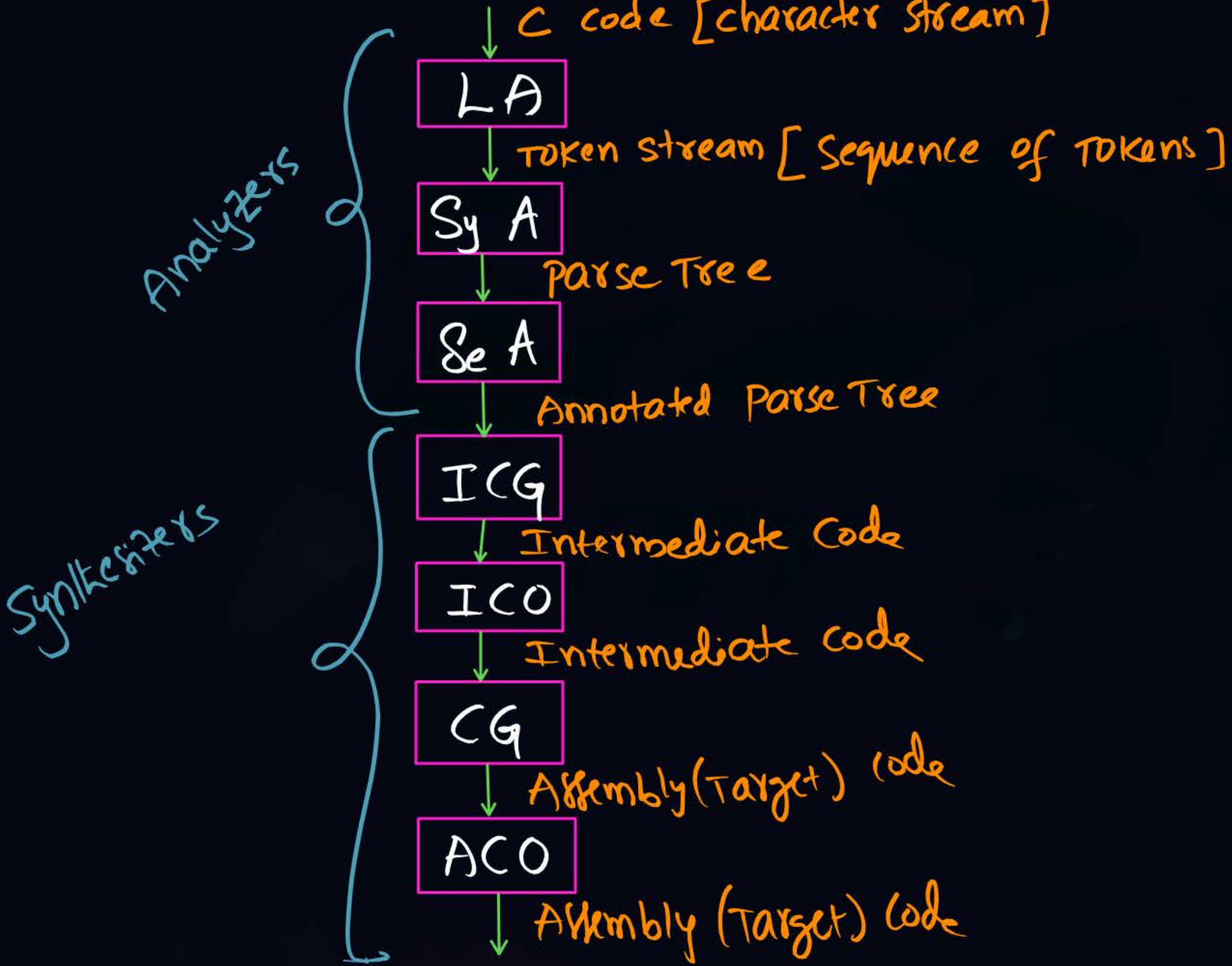
Phases of a Compiler:

7 phases

1) Lexical Analysis (LA)
2) Syntax Analysis (Sy A)
3) Semantic Analysis (Se A)
4) Intermediate code Generation (ICG)
5) Intermediate Code optimization (ICO)
6) Code Generation (CG)
7) Assembly (Target) Code optimization (ACO)



- It stores attribute information of Identifiers
- It is accessible to all phases
- It can be modified by all phases


C code [character stream]
LA
Token stream [Sequence of Tokens]
Sy A
Parse Tree
Se A
Annotated Parse Tree
Analyzers
ICG
Intermediate Code
ICO
Intermediate code
CG
Assembly (Target) code
ACO
Assembly (Target) code
Synthesizers

m/c Independent (Front end)

C code } Language Dependent

I code

Language Independent

M/c dependent (Back end)

A code

M code

E code

$$\begin{tikzcd}
{} \arrow[d] \\
\boxed{LA} \arrow[d] \\
\boxed{Sy\ A} \arrow[d] \\
\boxed{Se\ A} \arrow[d] \\
\boxed{ICG} \arrow[d] \\
\boxed{ICO} \arrow[d] \\
\boxed{CG} \arrow[d] \\
\boxed{ACO} \arrow[d] \\
{}
\end{tikzcd}$$

H.W.

Data	Code	Statement	program	Language

C Languag	C compiler

Summary

- Basics of Translation
- phases of a compiler [continue]
  - Errors
  - LA.

THANK YOU GW
SOLDIERS !