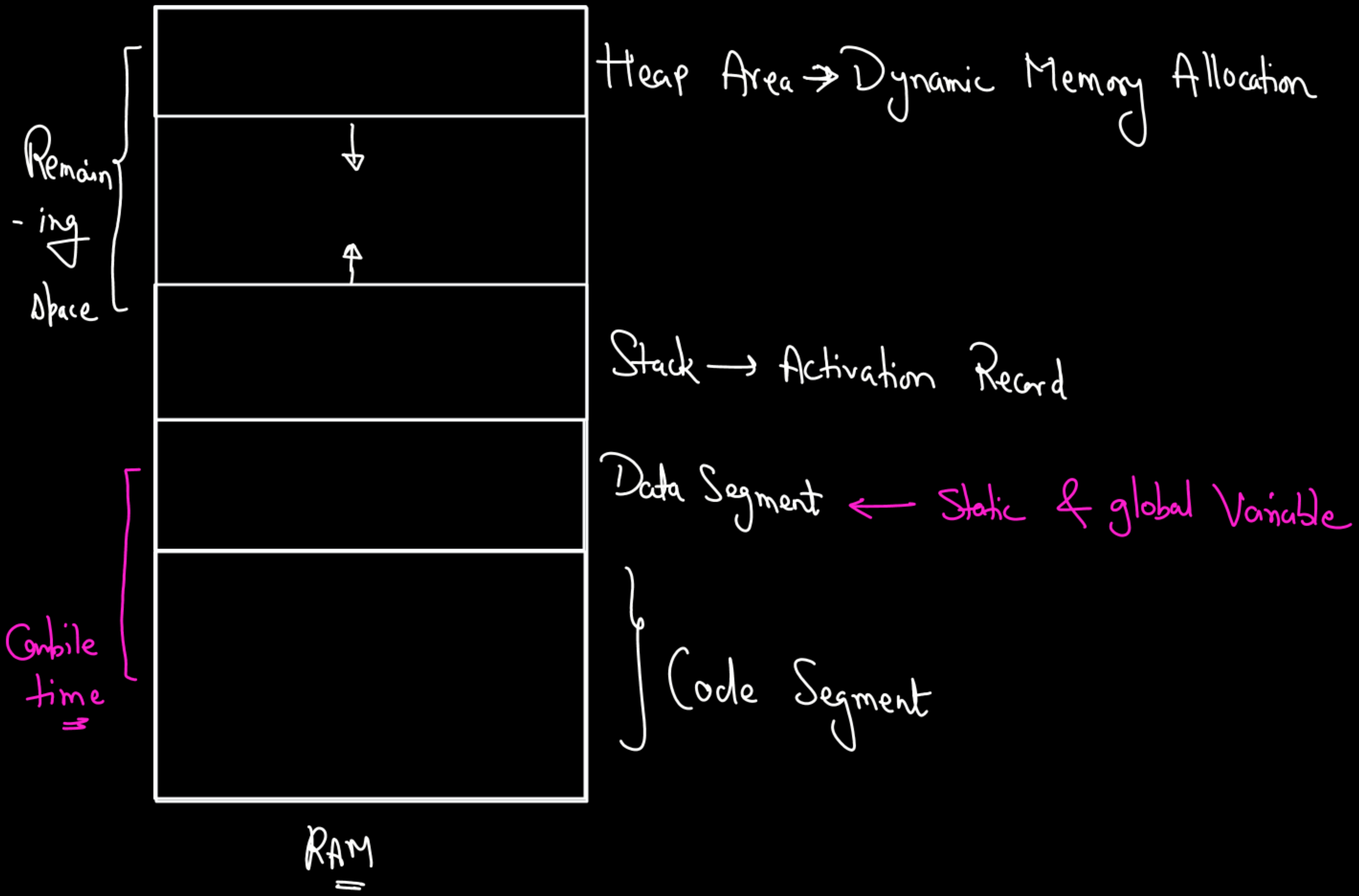
Lecture - 28

Storage Classes in C

⇒ Programming in C


Heap Area → Dynamic Memory Allocation
Remain-ing Space
Stack → Activation Record
Data Segment ← Static & global Variable
Code Segment
Compile time
RAM

Storage Classes

↳ Defines → [Scope, lifetime, Visibility & Default value] ← of Variables

1) Scope → [पहुँच] → [दायरा] → Visibility, limit

2) lifetime → A variable stands for How much time ?

3) Default Value → The value automatically set during variable declaration

4) Storage Area → location of the Variable in memory

$$\begin{tikzcd} & \text{RAM} \\ \text{memory} \arrow[ru] \arrow[r] \arrow[rd] & \text{Cache} \\ & \text{Register} \end{tikzcd}$$

Block ⇒

↳ Describe an entity and Space

↳ Set of Statement that define the scope

```
int main( ) {
    int x = 10;
    {
        int x = 20;          ← this x is the part of the block
        printf("%d", x);     ← 20
    }
    printf("%d", x);         ← 10
return 0;
}
```

$$\begin{tikzcd}
\text{main} \arrow[r] \arrow[d] & x = 10 \\
\text{Block} \arrow[r] & x = 20
\end{tikzcd}$$

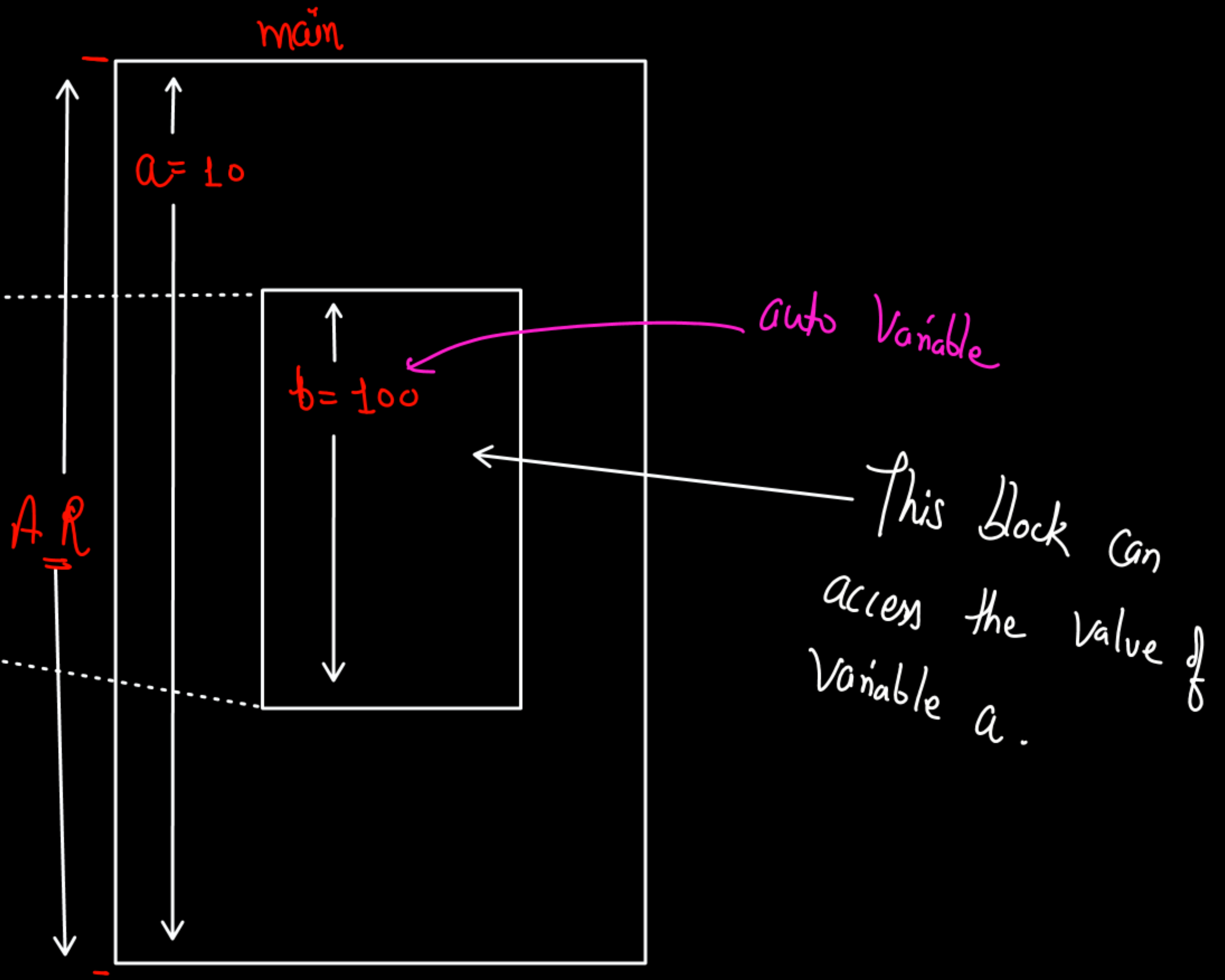
* local Scope has highest precedence

* The block automatically destroyed after executing.

  ↓

  Variables also destroy

```
int main ( ) {
    int a= 10;
    {
        int b= 100;
        printf ("%d", a+b) ;
    }
    printf ("%d", a);
    printf ("%d , b);
    return 0;
}
```



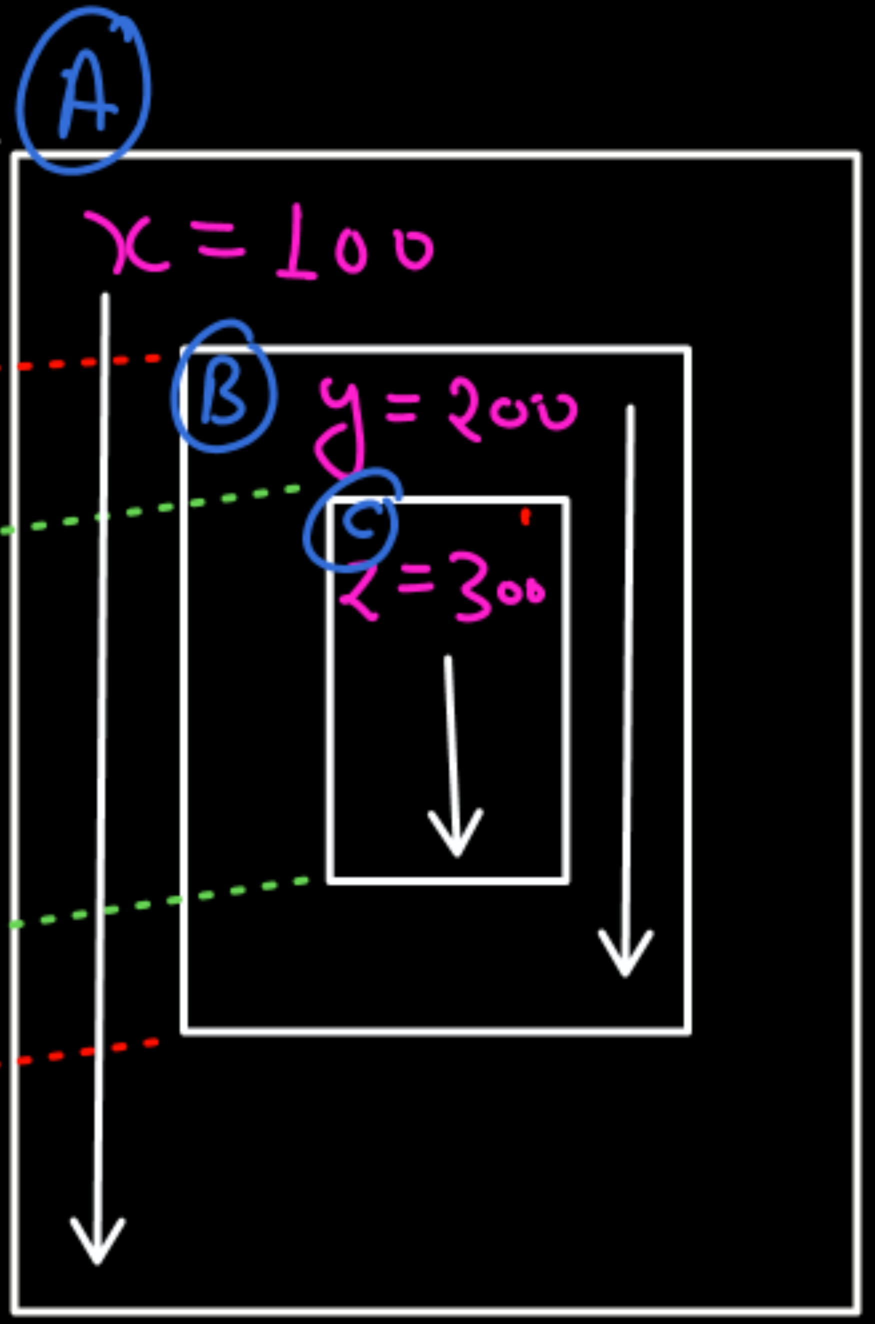
(A) Auto Variable ⟵ default

int x; or auto int x; ⟵ Same

Lifetime : Destroyed when block exited, during the block execution

Scope : Within the block, and nested blocks, Where the current block in parent block

```
{
  int x= 100;
  {
    int y = 200;
    {
      int z= 300;
      printf("%d", x+y+z);
    }
  }
}
```



$C \rightarrow B \rightarrow A$

$A \xrightarrow{X} B \xrightarrow{X} C$ (X) default Value : garbage Value

int x ; or auto int x ;

printf("%d", x) ← garbage Value

Storage Area : RAM (Stack)

↖ Memory

debug Mode | O/s (New) |

Modern Compiler | Security

⇓

$\boxed{\text{Value} = 0}$

(B) Register :

↳ Occupy CPU Registers to store the value.

```
register int x = 20;
```

↖ Requesting the o/s to get a block for int x in register memory

- → Accept → Register Memory
- ↳ Reject / Deny → RAM ← Stack

Register
CPU Memory
fastest

Lifetime : local, within the block, if block exited, the variable destroyed.

Scope : local scope, within the block

Default Value : garbage Value

Storage Area : CPU Register (if available)

otherwise Memory (Stack)

$\uparrow$ RAM

(c) Static Variable:

- Stored within data Segment of memory.
- Accessable for all if global

main → fun → y = 99
x = ~~10~~ 11 (Static)

print $\xrightarrow{x}$ 10
print $\xrightarrow{y}$ 99

fun → y = 99
x = ~~11~~ 12 (updated)
print → x = 11
y = 99

Static Variable are declared once.

```
void fun( ){
    int y = 99;
    static int x = 10;
    printf ("%d", x);
    printf ("%d", y);
    x++;
}
int main( ){        Entry
  ⟹ fun( );
  ⟹ fun( );
     fun( );
     fun( );
   return 0;
}
```

* The Static Variable remains active for the whole program.

```
void function( ){
    static int x = 0;
    printf("%d", x);
    x+= 2;
    int y = 100;
    printf("%d", y);
}

int main(){
    function();
    function();
    function();
return 0;
}
```

local static

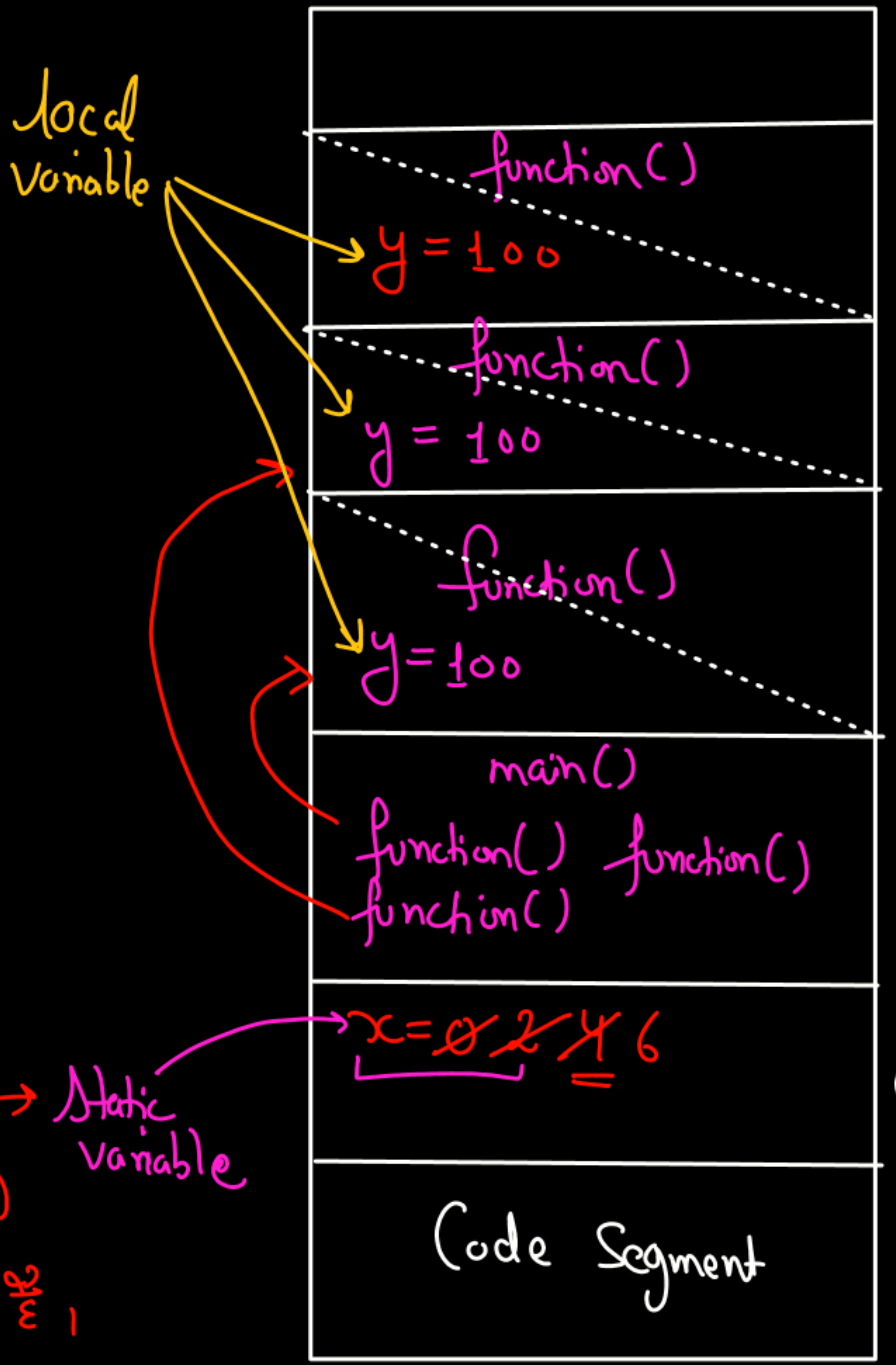
$\leftarrow$ ~~0~~ ~~2~~ 4

x = x+2

$\leftarrow$ local

$\longrightarrow$ 100

entry



data Segment $\leftarrow$ External Variables

function( )
$\downarrow$
का part है।

Output

Static $\rightarrow$	0	100
	2	100
	4	100

local (auto)

2nd call
↳ x is already defined

* The initialised at once
* Scope within whole program (if global)

  otherwise, local Scope ← Can only be accesed by the function itself who defined the static Variable.

```
A ( )
  ↳ Static int x = 10;      ← This x is only accessable by A ( )

B ( )
  ↳ Static int y = 20;      ← is Visible only for B ( )
```

Data Segment

* Lifetime : Entire program duration.

* Stored in Data Segment in RAM, NOT in Stack
* Default Value = 0

(D) External Variable :

a . c

↳ int x = 20

b . c

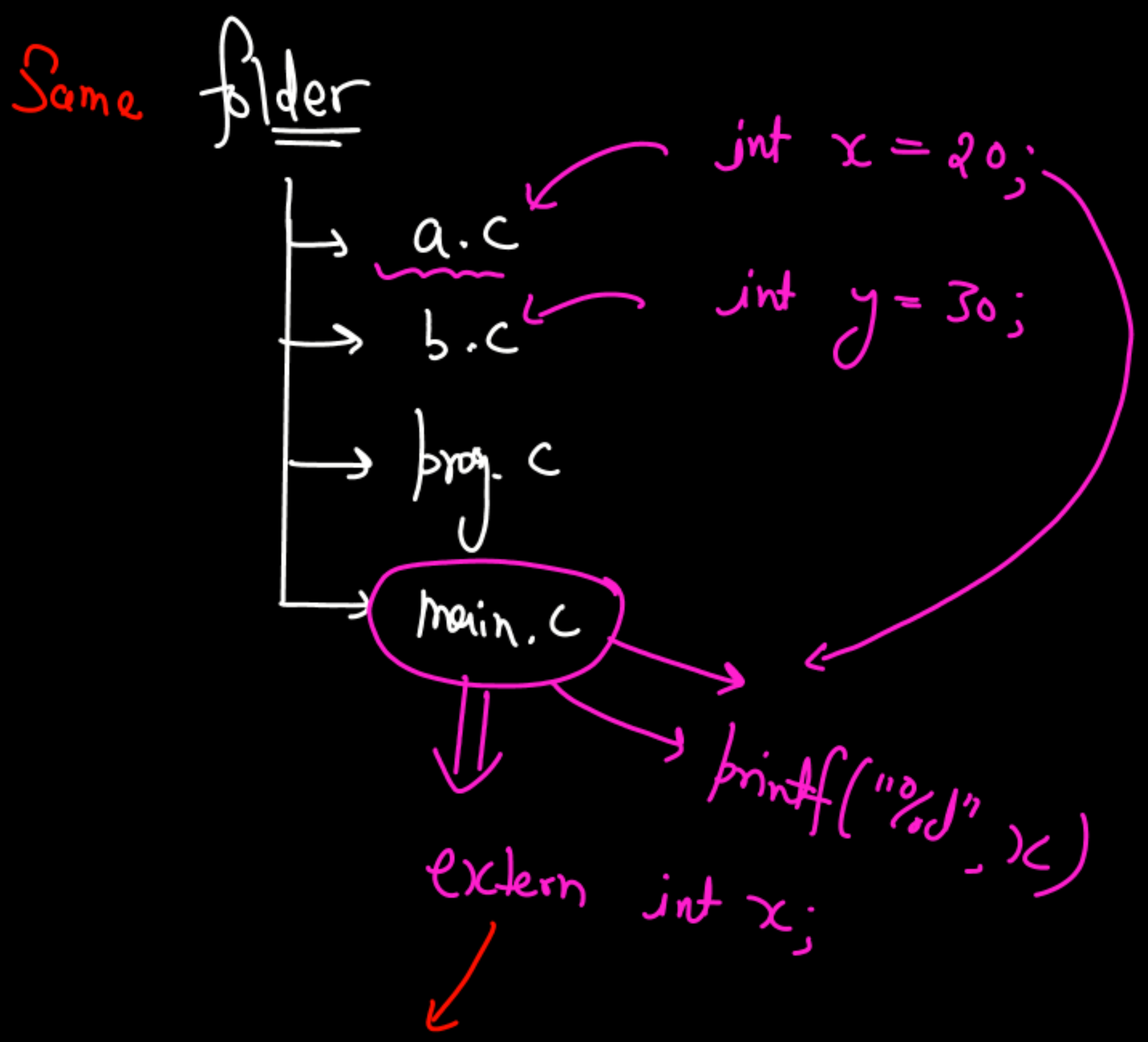
↳ int y = 30;

main . c

↳ extern int x;

extern int y;

printf ("%d", x+y);



The Variable x is defined in another c program.

Lifetime : Entire Program duration

Scope : Global Scope ← Can be accessed by any function

Storage Area : Data Segment
↳ RAM

Default Value : Zero (0)