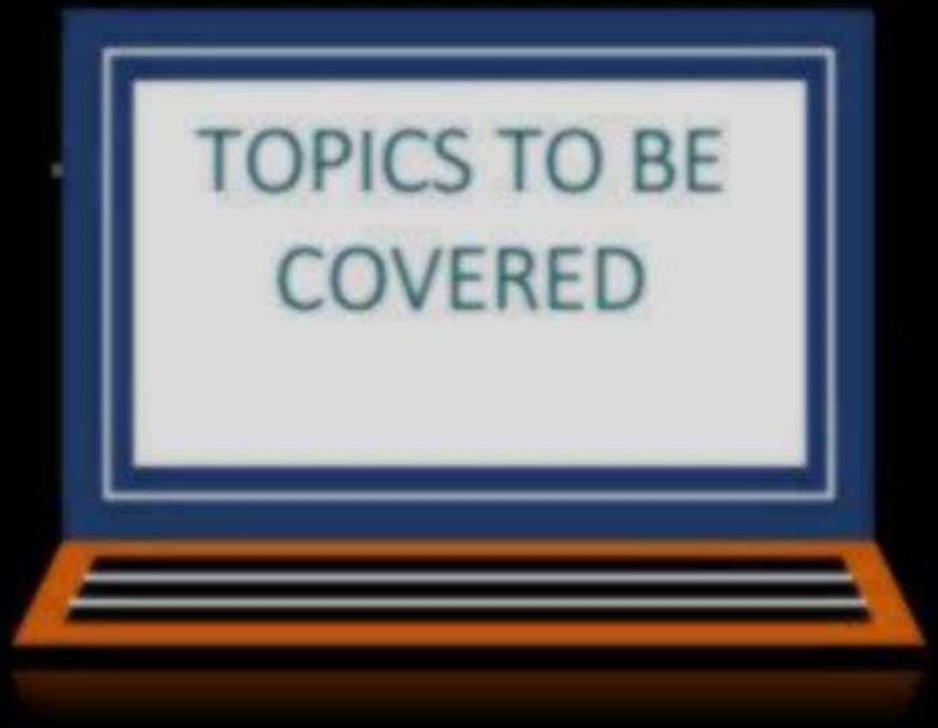
CS & IT ENGINEERING

Compiler Design

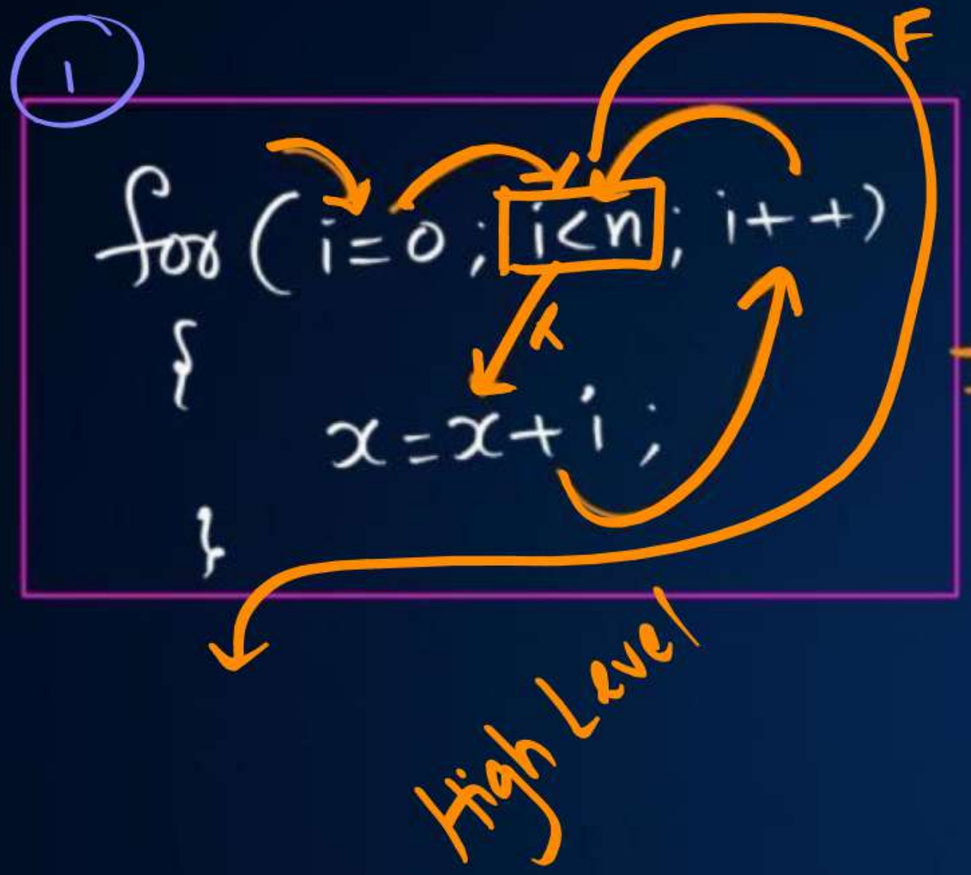
Intermediate code and code optimization

Lecture No. 2

By- DEVA Sir



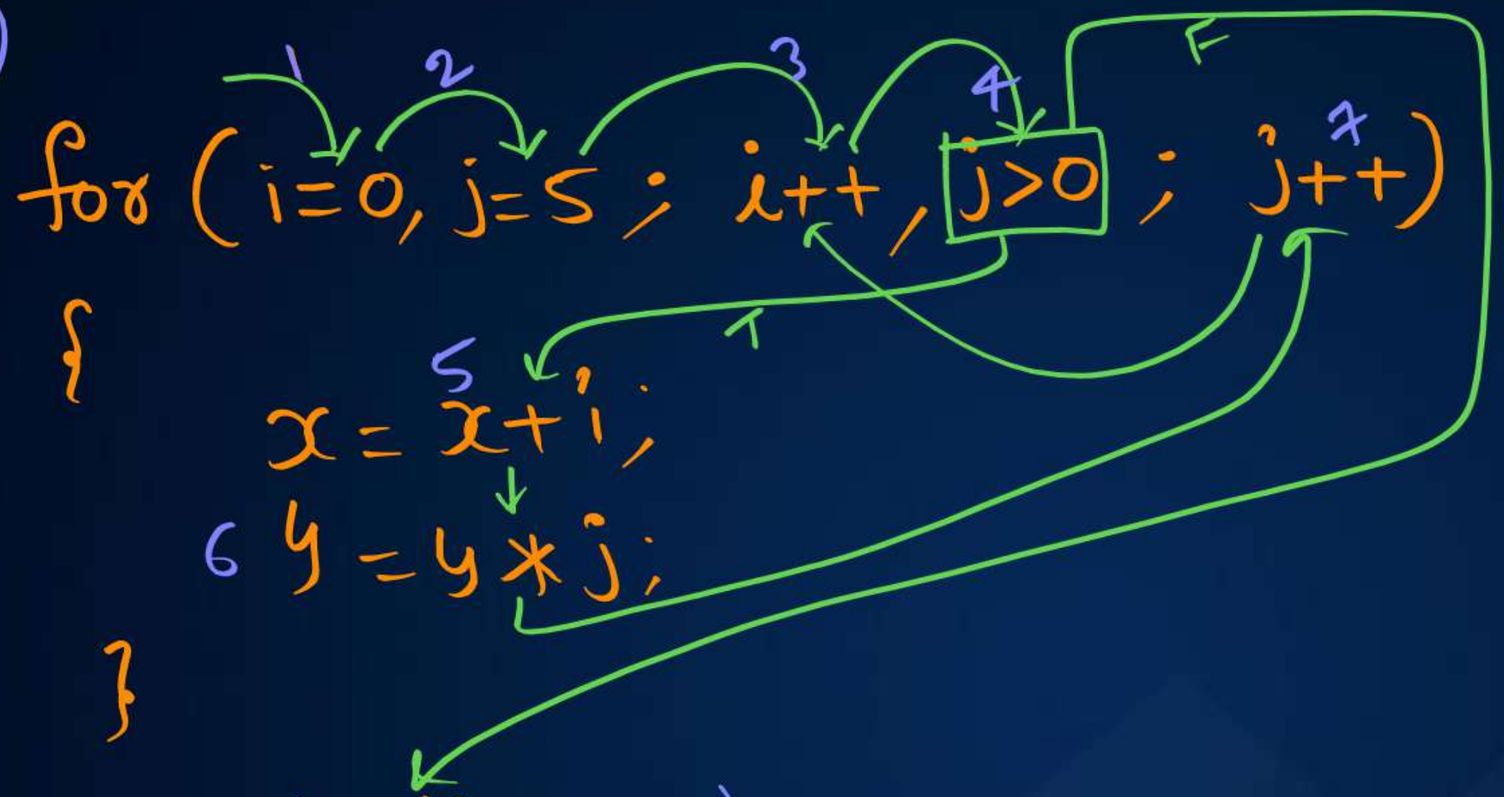
- CFG (Control Flow Graph)
- 3AC



1. i=0
2. if (i≥n) goto 5
3. x=x+i
4. i=i+1, goto 2
5. exit

3AC

(2)



High Level

1. i=0
2. j=5
3. i=i+1
4. if(j≤0) goto 8
5. x=x+i
6. y=y*j
7. j=j+1, goto 3
8. a=x+y;

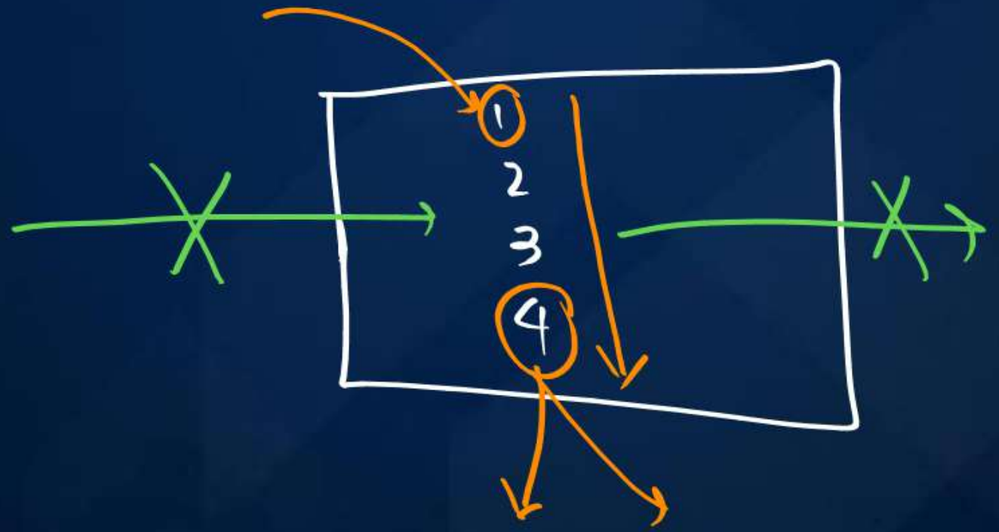
3AC

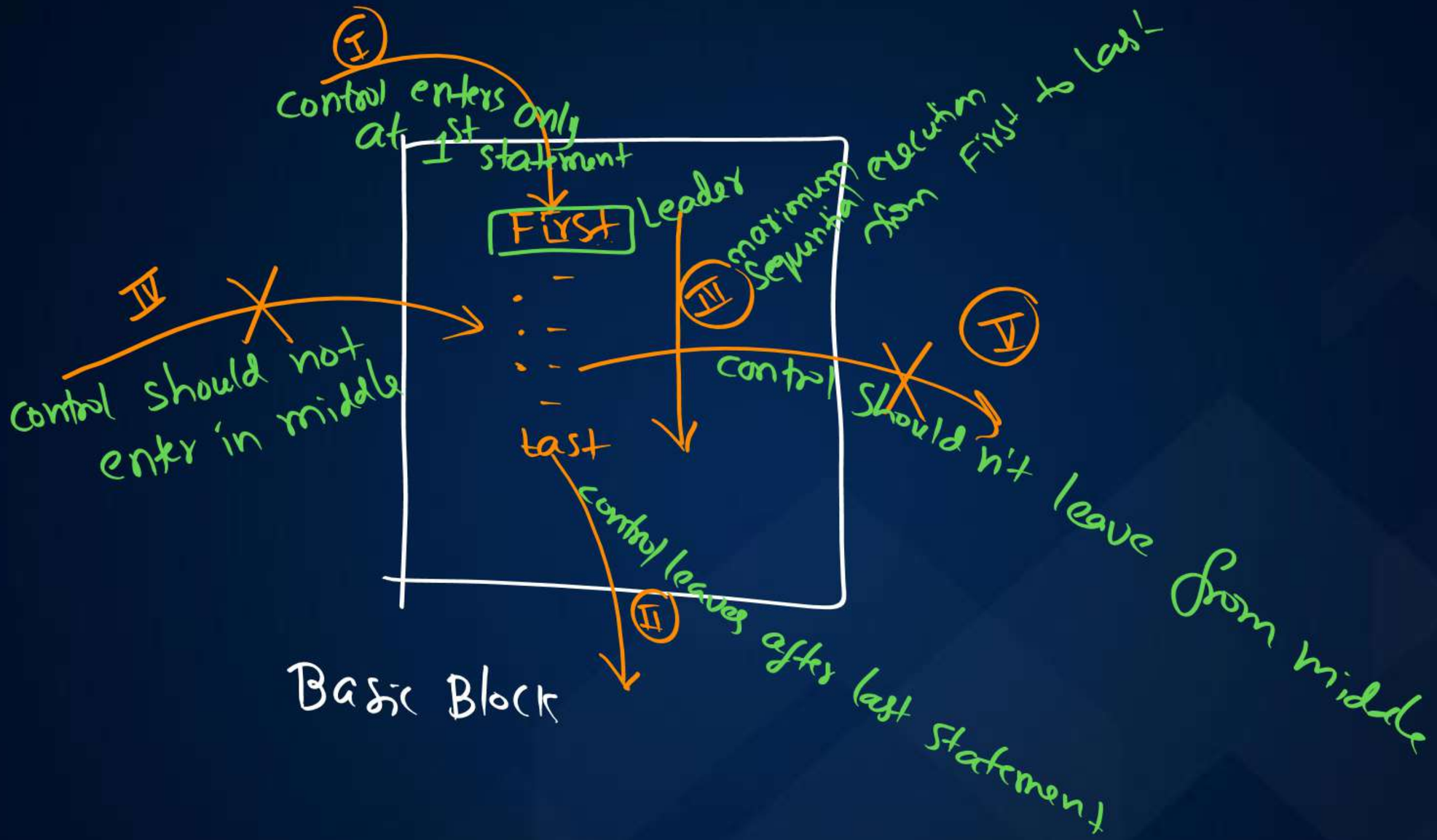
Control Flow Graph (CFG)

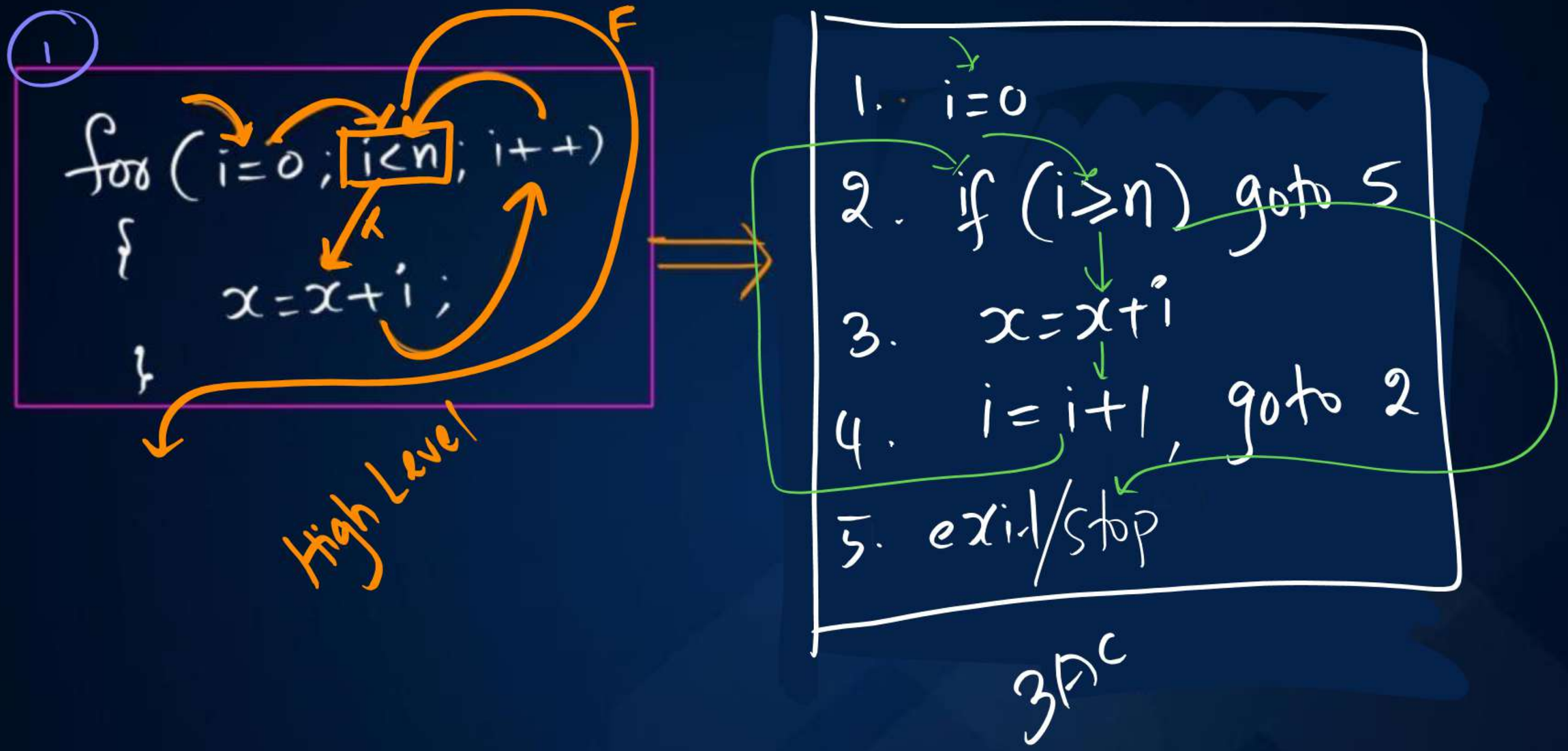
- It is "Flow Graph"
- Collection of nodes and edges
- Collection of "Basic blocks"

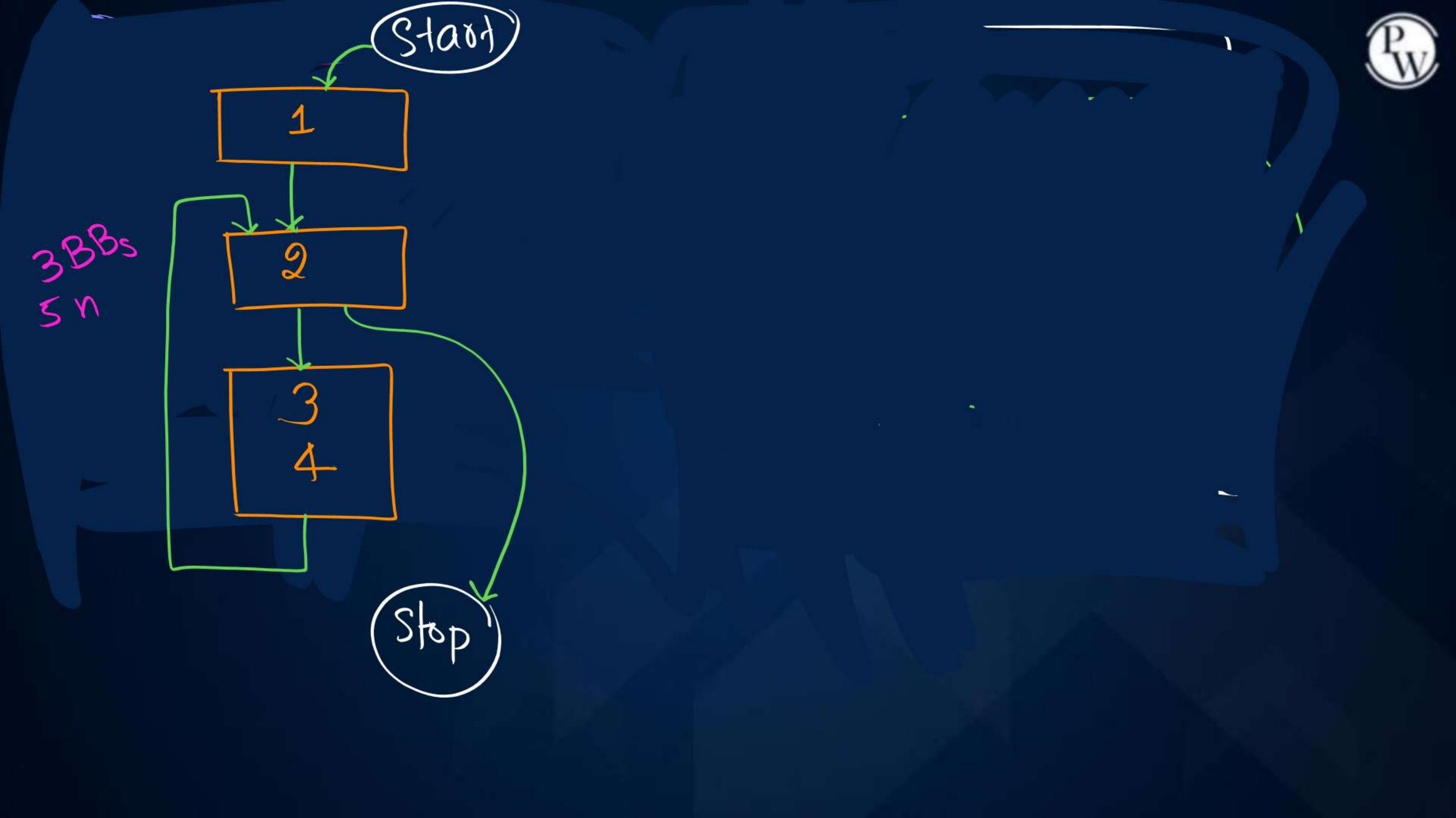
What is Basic Block (BB) ?

↳ It is maximum (sequence) set of statements where control enters only at 1st statement and only leaves after last statement.

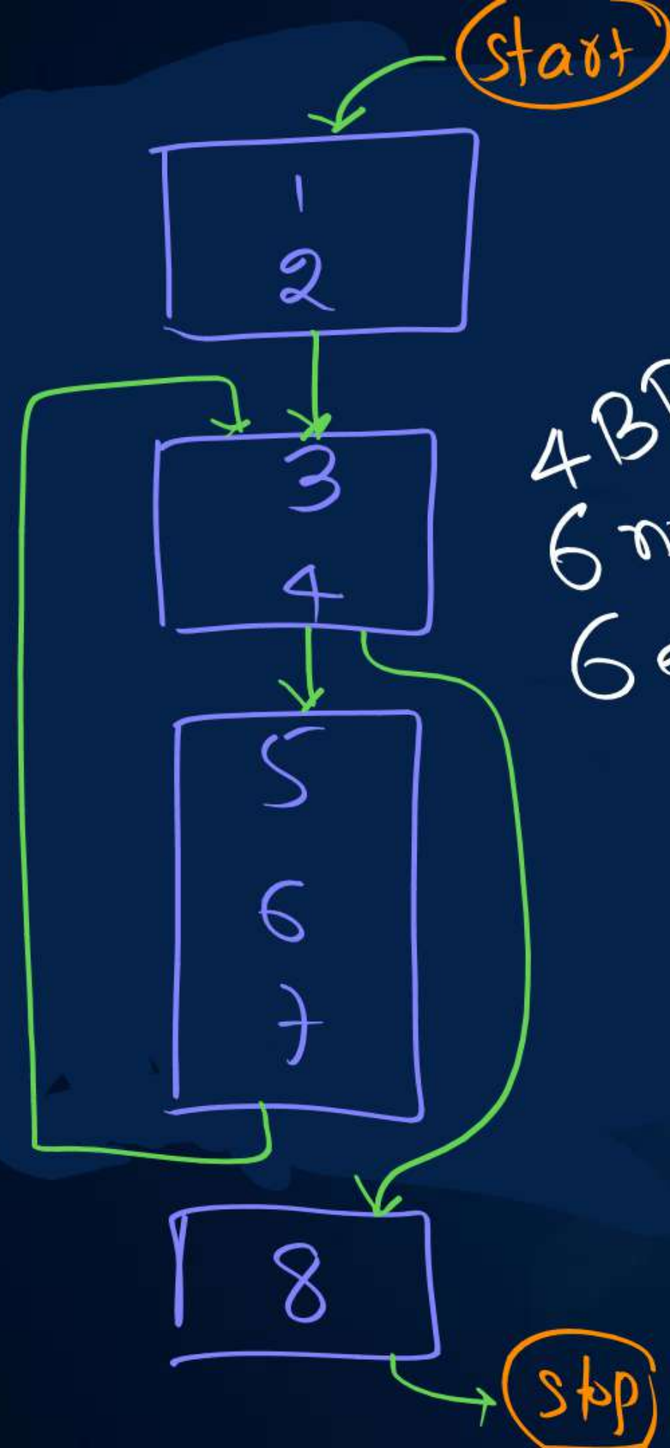



I
Control enters only at 1st statement
FIRST
Leader
III
maximum sequential execution from First to Last
IV
Control should not enter in middle
V
control should n't leave from middle
last
control leaves after last statement
II
Basic Block

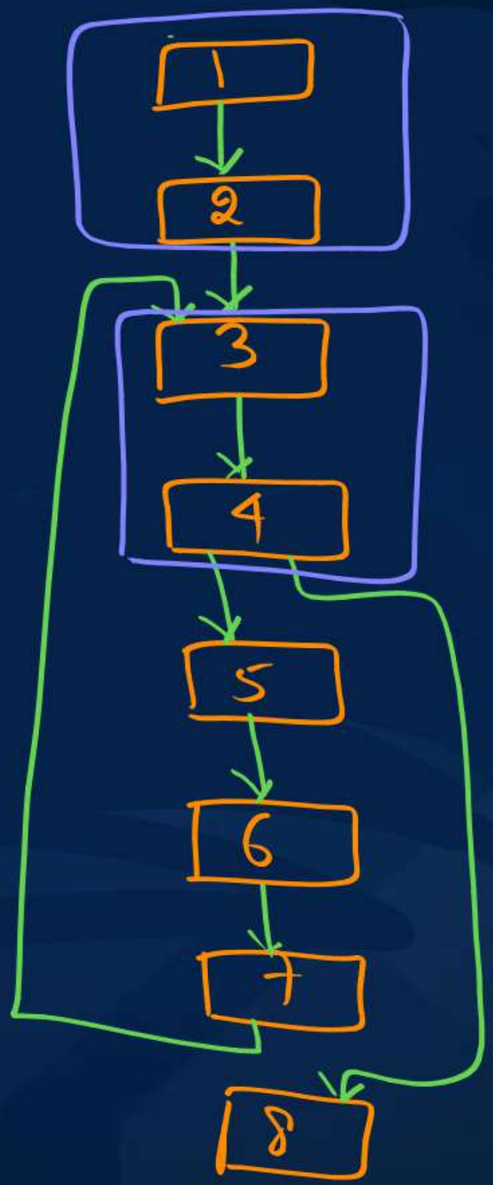

1
for (i=0; i<n; i++)
{
x=x+i;
}
F
High Level
1. i=0
2. if (i≥n) goto 5
3. x=x+i
4. i=i+1, goto 2
5. exit/stop
3AC


Start
1
2
3
4
Stop
3 BBs
5 n

(2)



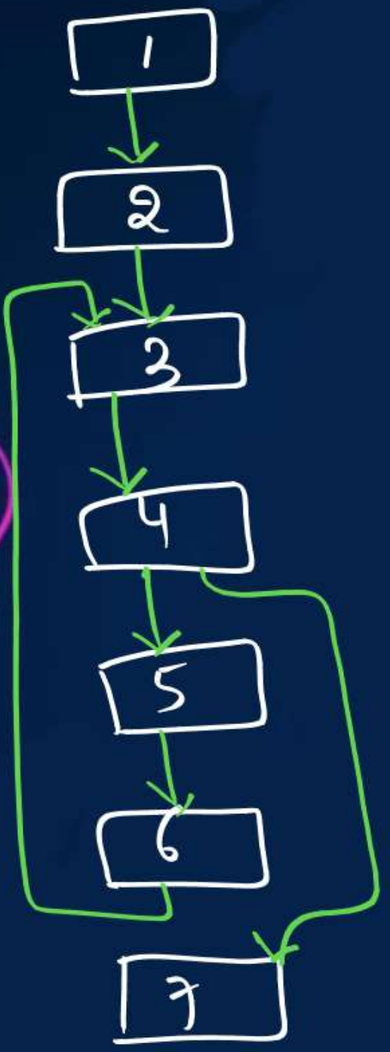
4 BBs
6 nodes
6 edges

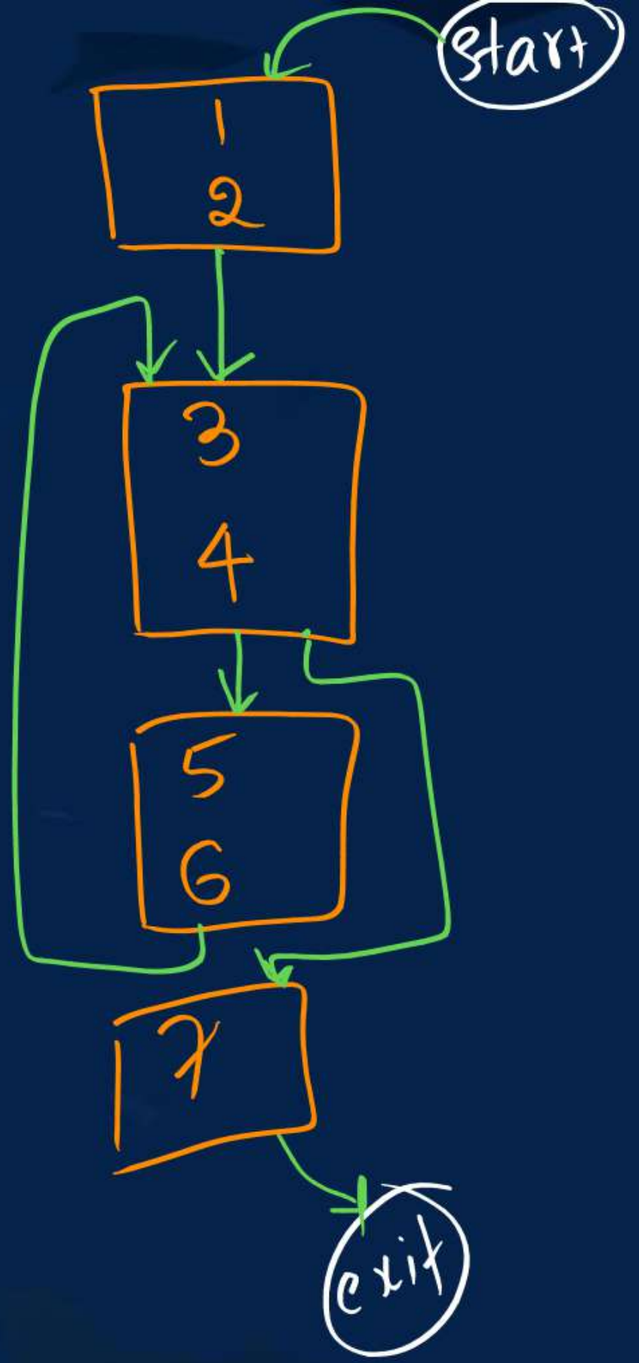


1. $i = 0$
2. $j = 5$
3. $i = i + 1$
4. if ($j \leq 0$) goto 8
5. $x = x + i$
6. $y = y * j$
7. $j = j + 1$, goto 3
8. $a = x + y$;

(3)

1. $x = a + 1$
2. $y = x * 2$
3. $z = b - y$
4. if $(y > z)$ goto 7
5. $x = x - 1$
6. $y = y - 1$, goto 3
7. $a = a + y$



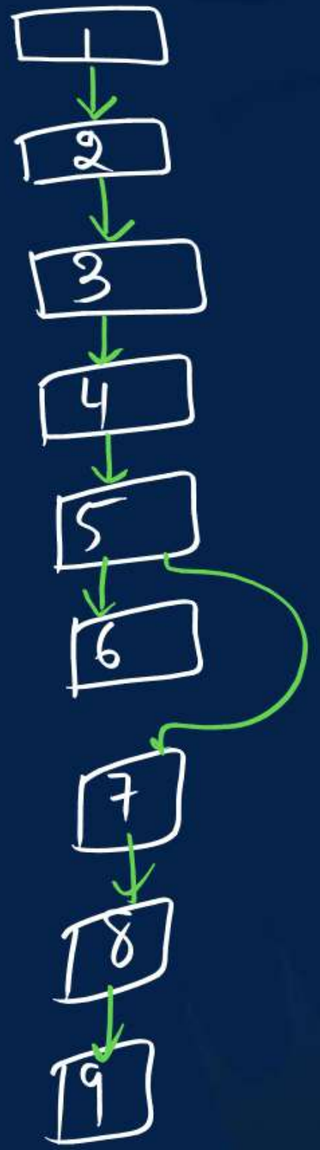


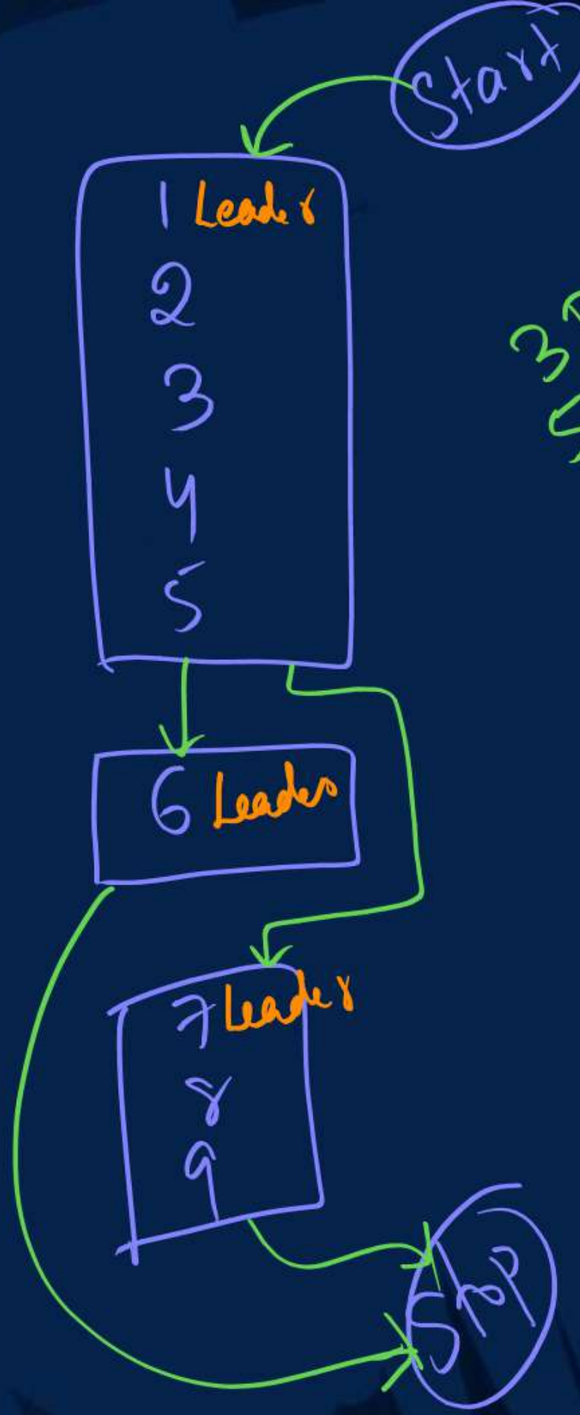
6 nodes
4 BBs
6 edges

4

Find no. of nodes & edges

```
1. c=a+b
2. d=c*a
3. e=c+a
4. x=c*c
5. if(x>a)
6.     { y=a*a }
7. else
8.   { d=d*d,
9.     e=e*l;
   }
```

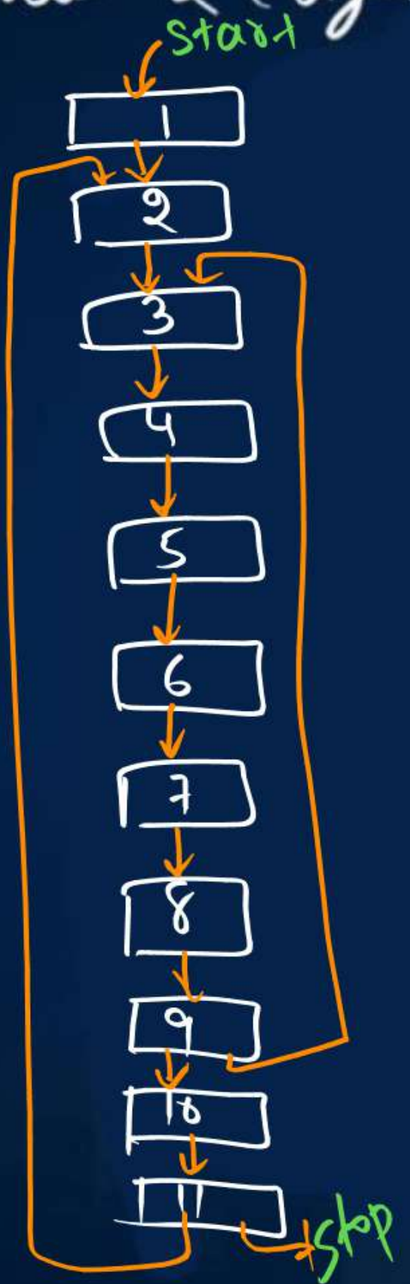


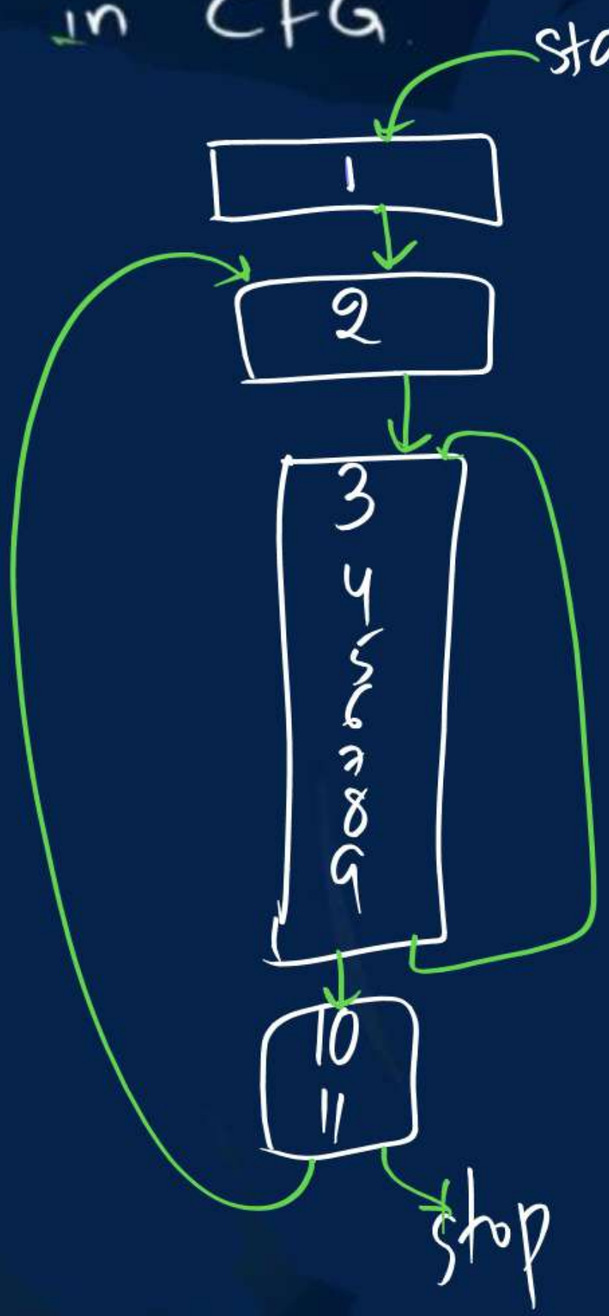


3 BBs
5 nodes
5 edges

(5) Find no. of nodes & edges in CFG

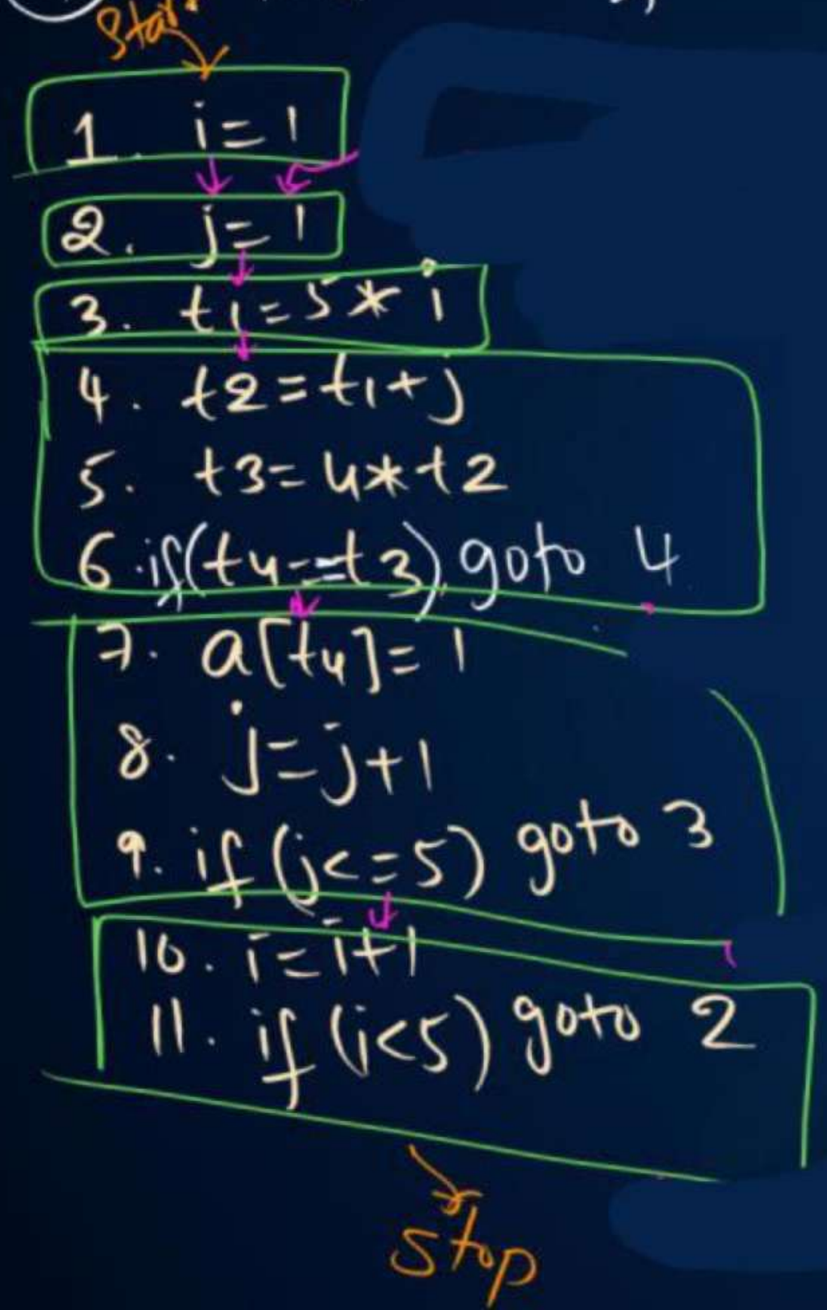
1. i=1
2. j=1
3. t1=5*i
4. t2=t1+j
5. t3=4*t2
6. t4=t3
7. a[t4]=1
8. j=j+1
9. if (j<=5) goto 3
10. i=i+1
11. if (i<5) goto 2





4 BBs
6 nodes
7 edges

(6) Find no. of nodes & edges in CFG.



H.W

Three Address Code : [We can store 3AC in following ways]

1. Triple Form [–, –, –]
2. Quadruple form [–, –, –, –]
3. Indirect triple form

3AC:

$x = y + z$

$a = x * x$

Triple Form

	operator	Left operand1	Right operand2
1000	+	y	z
1010	*	1000 (Address of 3AC code)	1000 (Address of 3AC code)

Advantage: Less Space

Disadvantage: If any lvalue used many times then it has to be calculated every time

1000 : (+, y, z)

1010 : (*, 1000, 1000)

3AC:

$x = y + z$

$a = x * x$

Quadruple:

	operator	Left operand 1	Right operand 2	Result
1000	+	y	z	x
1015	*	x	x	a

Advantage: Saves time if any variable (lvalue) is used many times

Disadvantage: More space

1000: (+, y, z, x)

1015: (*, x, x, a)

3AC:

$x = y + z$

$a = x * x$

Indirect Triple Form

	operator	Left operand1	Right operand2
1000	+	y	z
1010	*	5000 (Address of address of 3AC code)	5000

Indirect table:

5000	1000
5010	1010

Three Address Code :

↳ TO Store 3AC, we have 3 data structures.

① Triple Notation

$x = (y * z)$
$y = (x + a)$
$z = (-y)$

Address	Operator	Left operand$_1$	Right operand$_2$
1000	*	y	z
1010	+	[1000]	a
1015	−		[1010]

② Quadruple Notation

	operator	Left operand	Right operand	Result
5000	*	y	z	x
5012	+	x	a	y
5020	−		y	z

③ Indirect Triple

6000	[1000]
6005	[1010]
6010	[1015]

	Triple Notation		
1000	*	y	z
1010	+	[1000]	a
1015	−		[6005]

Triple Notation

$x = y * z$

$y = x + a$

$z = (-y)$

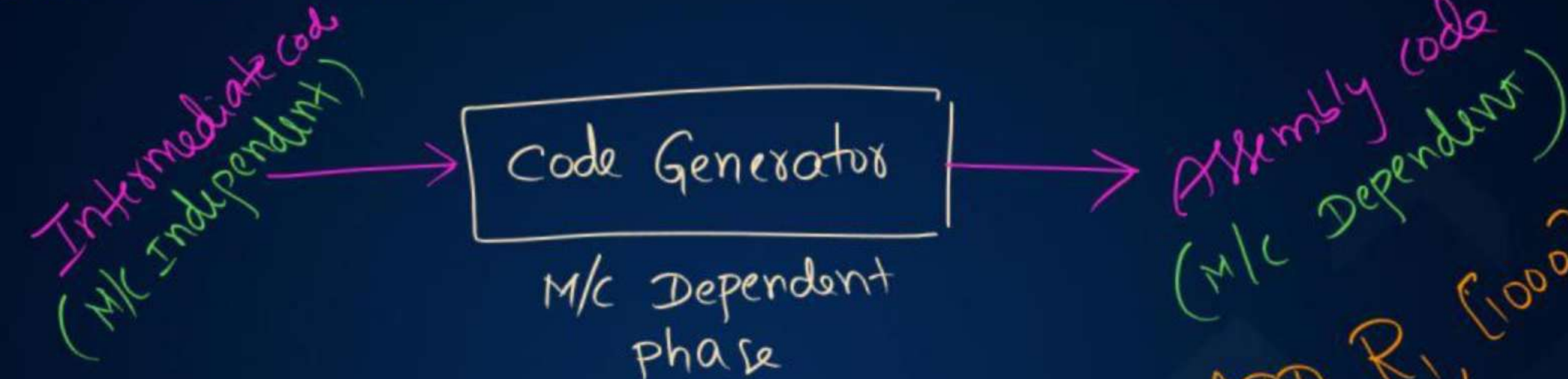
Address	Operator	Left operand$_1$	Right operand$_2$
1000	*	y	z
1010	+	[1000]	a
1015	−	.	[1010]

1000 : (*, y, z)

1010 : (+, [1000], a)

1015 : (−, , [1010])

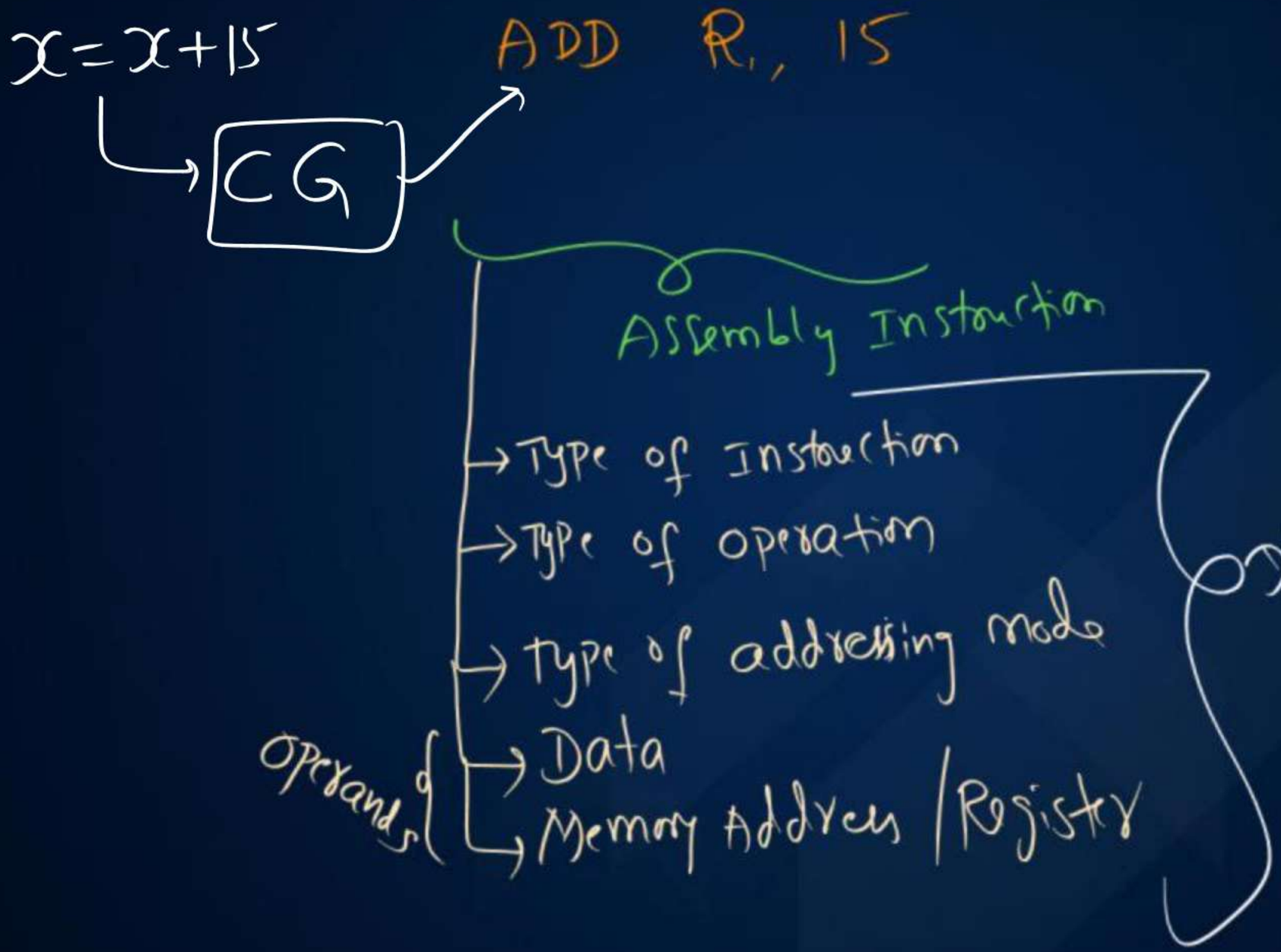
Code Generator :



$x = x + y$

x and y are variables

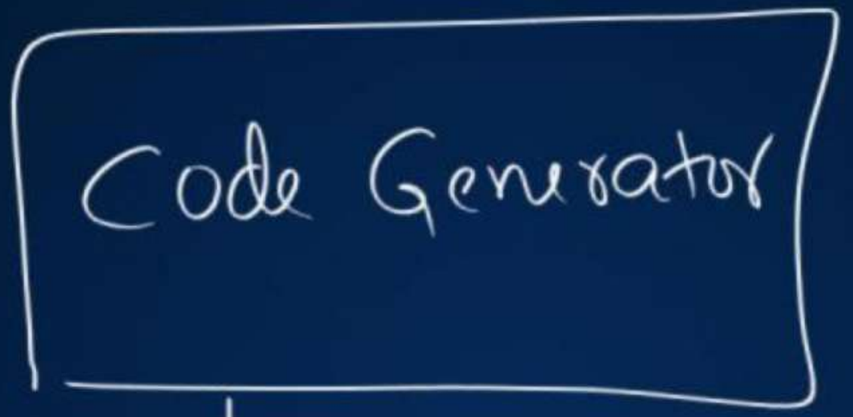
x and y are registers/memory addresses


x = x + 15
CG
ADD R, , 15
Assembly Instruction
→ Type of Instruction
→ Type of operation
→ Type of addressing mode
Operands
→ Data
→ Memory Address / Register
Depends on m/c Architecture

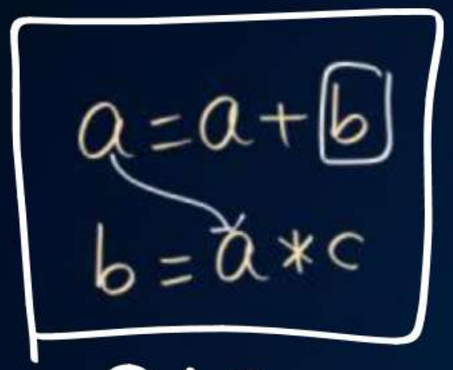
Register
↳ speed
Memory
↳ Not speed compared to Reg.

ADD R_1, R_2

$x = x + y$



↳ Register Allocation
↳ Graph coloring
↳ To allocate min no. of registers



We have only 1 register

How many min memory spills required ?

$R \leftarrow a$

$Mem_1 \leftarrow b$

$R \leftarrow R + mem_1$

$Mem_1 \leftarrow c$

$R \leftarrow R * mem_1$

⇒ one memory spill

$a = a + b$

$b = a * c$

We have only 2 register

How many min memory (Register) spills required ?

$R_1 \leftarrow a$

$R_2 \leftarrow b$

$R_1 \leftarrow R_1 + R_2$

$R_2 \leftarrow c$

$R_1 \leftarrow R_1 * R_2$

$= 0$

ICG


Analyzed C code
ICG
Intermediate Code

CG


Intermediate Code
3AC or CFG
CG
Assembly code

Algebraic laws $x = a+b-a+c$

Reordering $x = (a*b)+a+(b*c)$

c

1st

c will be from

Common sub expression elimination

$(a+b) * (a+b)$

a a

Find no. of variable in SSA

exp given ⟹ Best 3AC ⟹ Best SSA

3AC given ⟹ exp / Best 3AC ⟹ Best SSA Code

DAG given ⇄ (exp) Best 3AC ⟹ Best SSA Code

Syntax Tree ⇄ exp ⇄ Best 3AC ⇄ Best SSA Code

Any → expression


THANK YOU
GW
SOLDIERS !