

Mirroring Lecture_4

```
Sum.java LotteryGame2.java X
LotteryGame2.java > LotteryGame2 > main(String[])
3 public class LotteryGame2 {
4     public static void main(String[] args) {
26
27         } else if (n >= 1100 && n <= 1500) { ←
28             System.out.println(x: "You won a Cycle!"); ←
29
30             if (n >= 1100 && n <= 1300) { ←
31                 System.out.println(x: "Brand: Avon Cycle");
32             } else { (1301→1500) ←
33                 System.out.println(x: "Brand: Hero Cycle");
34             }
35
36         } else if (n > 50 && n <= 80) { ←
37             System.out.println(x: "You won a Bike!");
38
39             if (n > 50 && n <= 65) {
40                 System.out.println(x: "Model: Bullet");
41             } else {
42                 System.out.println(x: "Model: Rajdoot");
43             }
}
(not included)
(50 → 80)
50 → 65 (66 → 80)
else
```

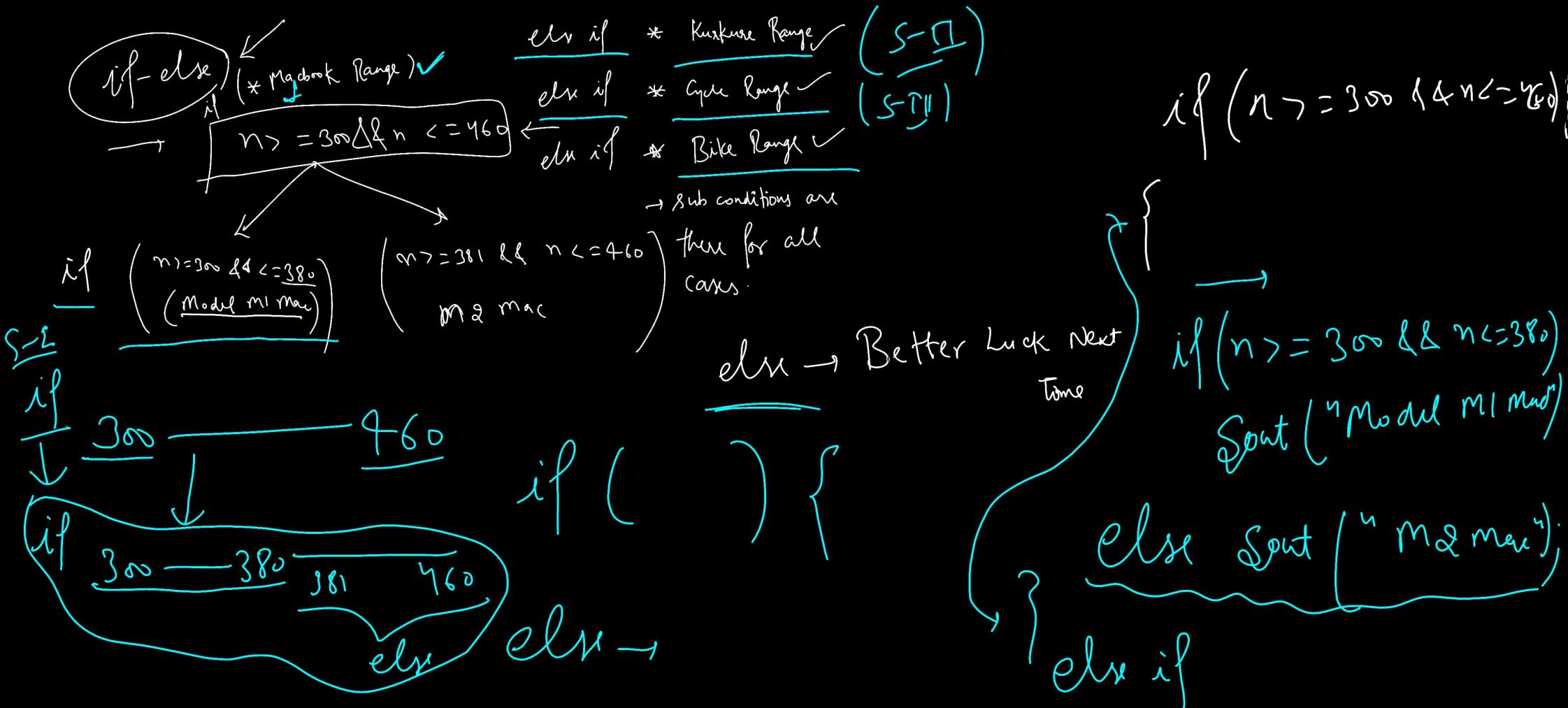
Run Testcases 0 0 Java: Ready Ln 18, Col 43 Spaces: 4 UTF-8 CRLF {} Java Chat quota reached Go Live Prettier ENG IN 10:24 AM 2/19/2026

Mirroring Lecture_4

```
Sum.java LotteryGame2.java X

LotteryGame2.java > LotteryGame2 > main(String[])
3 public class LotteryGame2 {
4     public static void main(String[] args) {
38         if (n > 50 && n <= 65) {
39             System.out.println(x: "Model: Bullet");
40         } else {
41             System.out.println(x: "Model: Rajdoot");
42         }
43     }
44     } else {
45         System.out.println(x: "Better luck next time.");
46     }
47 }
48 sc.close(); ← // we have to close the scanner class
49
50 }
51
52
53 }
```

Run Testcases 0 0 Java: Ready Ln 38, Col 1 Spaces: 4 UTF-8 CRLF {} Java Chat quota reached Go Live Prettier ENG IN 10:37 AM 2/19/2026

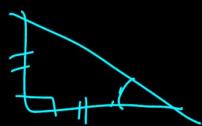




Scanned with OKEN Scanner

A photograph of a handwritten note on lined paper. The note is written in cursive ink and reads: "You need 10 white carnations". The handwriting is clear and legible.

Right Angled $\triangle \rightarrow 90^\circ$



$\rightarrow \gamma_{0w\delta} = y_1$

Acute Angle $\Delta \rightarrow$

row

3rd son

\downarrow $\star \rightarrow \text{col}$ \perp

* → CS

* * → ω }

row	col
1	1
2	2
3	3
4	4
:	:
n	n

$\gamma_{0W} = 1$

while ($xow \leq y$) {

```
int col=1;
```

while (col < = row) {
Spew out point (" * "

3 $\text{col} = \text{col} + 1;$

· `Sont()` ·

YOW++)

Pattern 2 :-

1	2	3
2	*	*
3	*	*

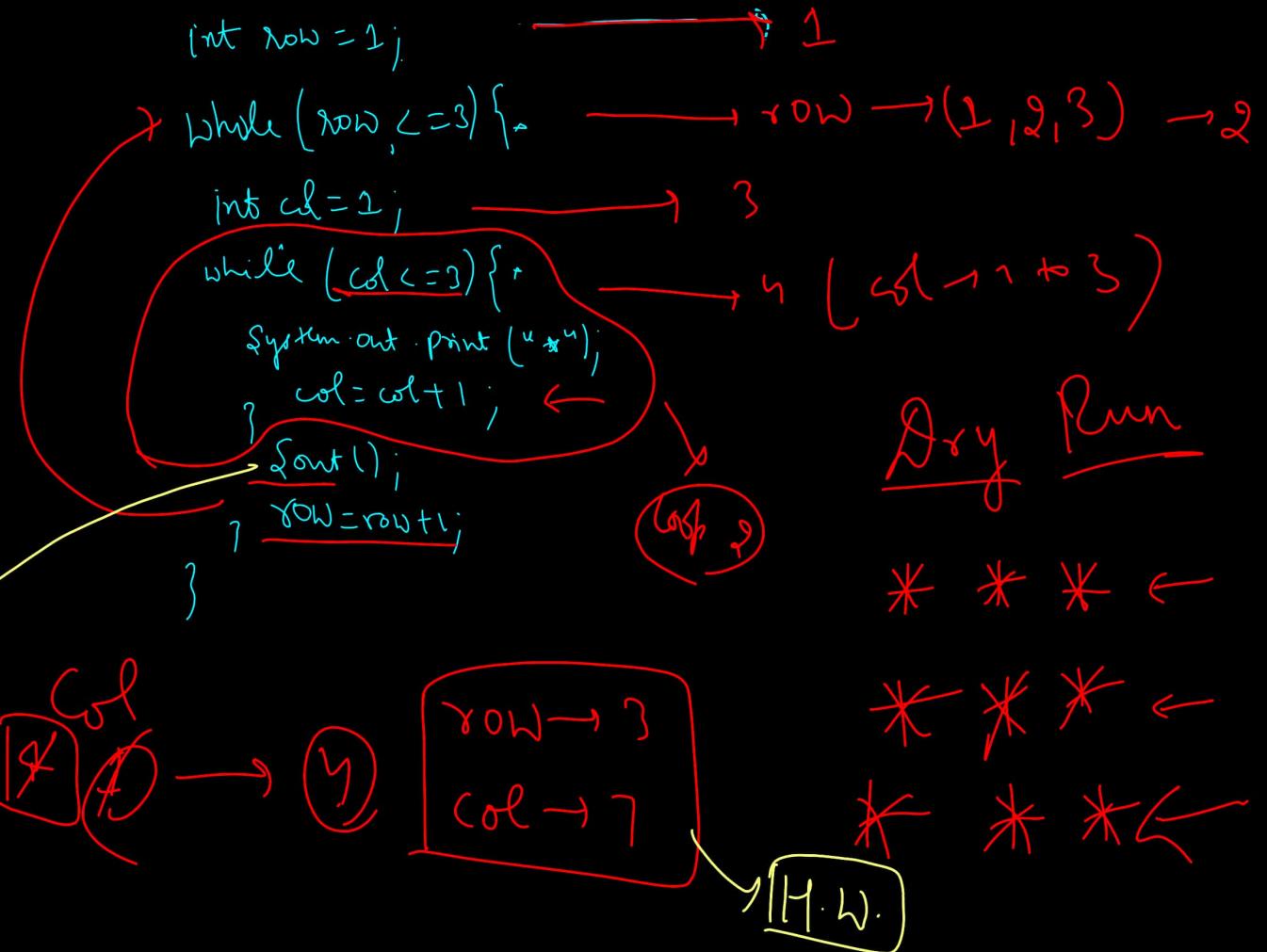
→ 3 Rows, 3 columns

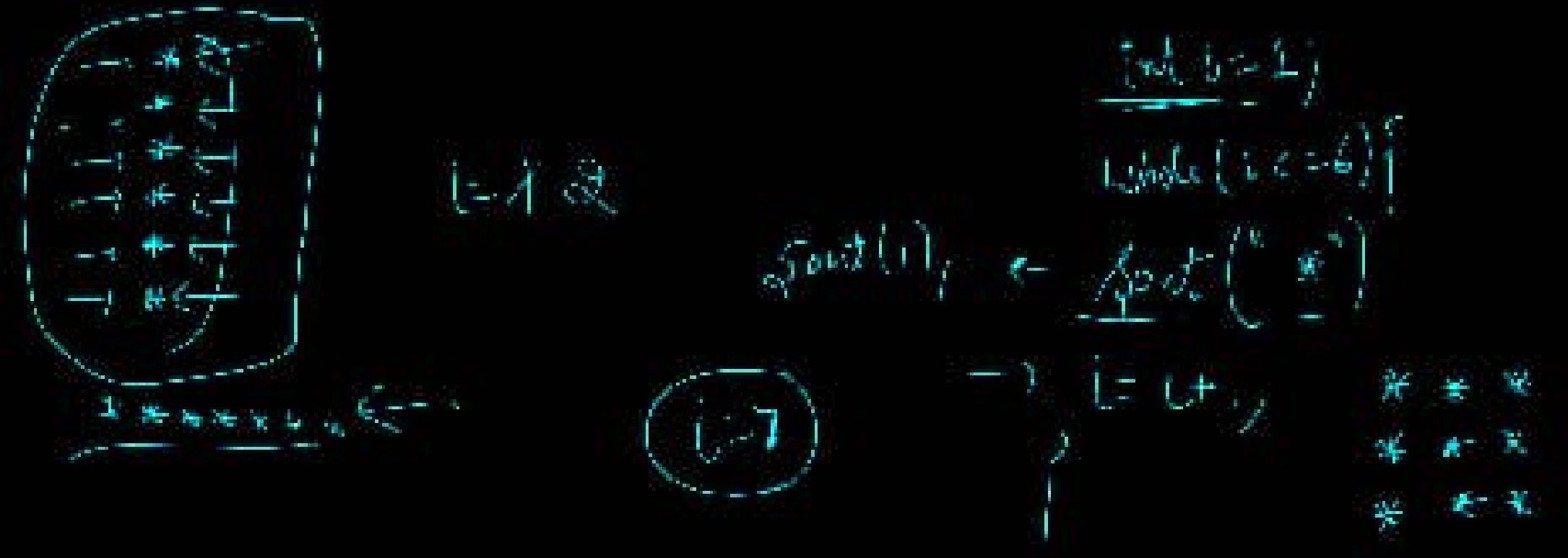
→ Constant = '*', Variables = rows, col

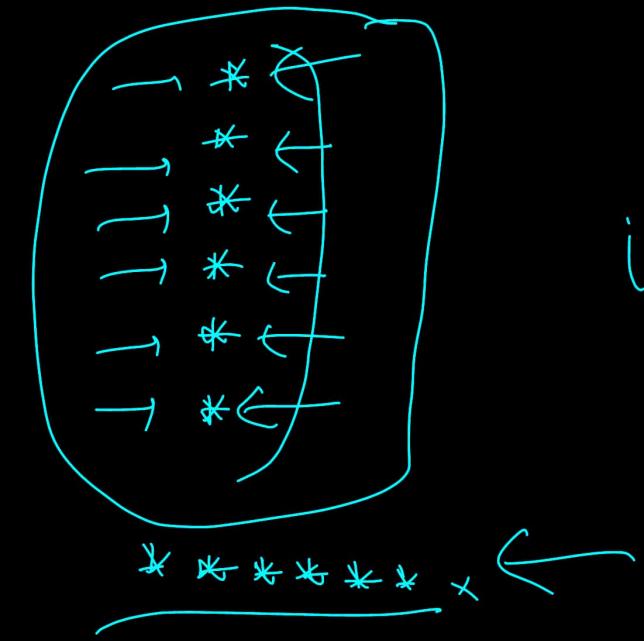
→ rows = 1 to 3

↓
(col = 1 to 3) [inside the row]

No need to write anything



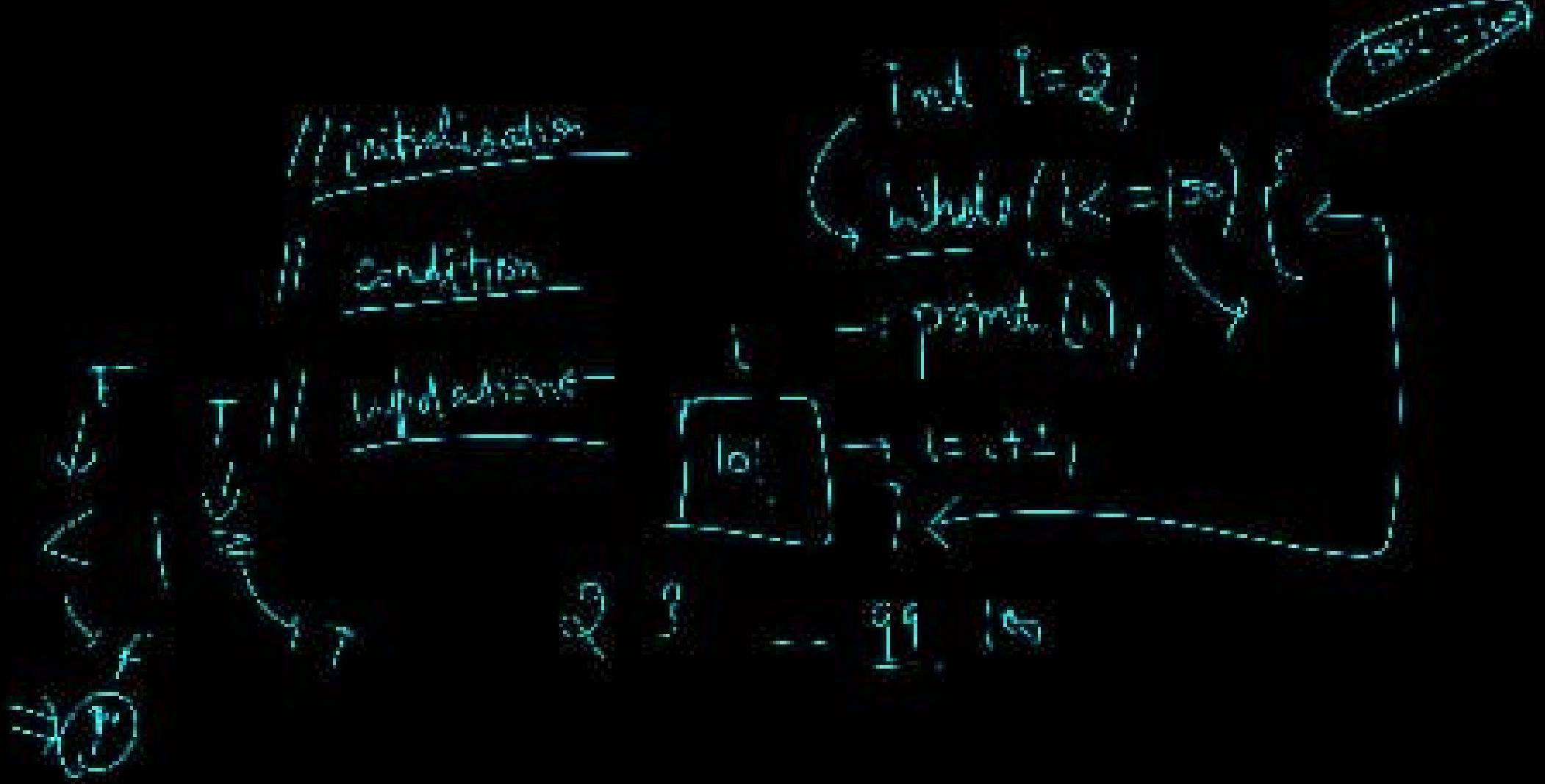




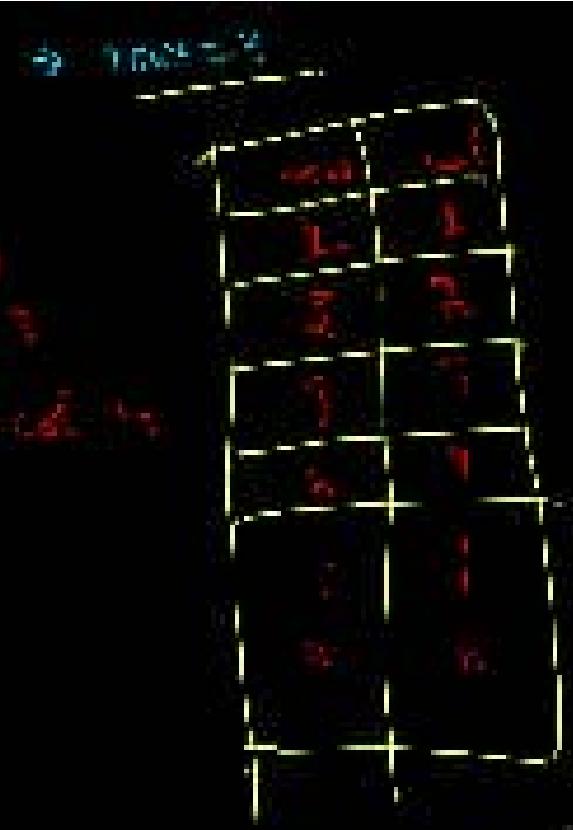
$i=7$

int i=1;
while (i <= 6){
 $\quad \quad \quad \text{fout}(i); \quad \leftarrow \quad \underline{\text{fout}(" * ")}$

$\quad \quad \quad \rightarrow \quad i = i + 1;$ * * *
 * * *
 * * *



A fluorescence microscopy image showing a C. elegans neuron. The neuron is stained with a green fluorescent antibody against synaptosomal-associated protein 25 kDa (SAP-25), which labels the presynaptic boutons. The neuron has several varicosities, which are swellings along the axon where neurotransmitter is stored. The soma (cell body) is visible as a bright green spot. A red fluorescent signal, likely from a different marker or a different channel, is visible as puncta (small spots) within the varicosities, representing synaptic vesicles. The background is dark, making the green and red signals stand out.



3. $\text{Coll} = \text{Coll}(\kappa + 1)$
 4. $\text{GCH} \rightarrow \text{Coll}(\kappa + 1) \vdash$
 $\forall \alpha < \kappa \exists f_\alpha : \omega \rightarrow \kappa$
 $\forall \beta < \kappa \forall n \in \omega \exists m \in \omega$
 $f_\beta(n) < f_\beta(m) \rightarrow f_\beta(n) = f_\beta(m)$
 $\forall \beta < \kappa \forall n \in \omega \exists m \in \omega$
 $f_\beta(n) < f_\beta(m) \rightarrow f_\beta(n) \neq f_\beta(m)$

The diagram illustrates the structure of a while loop. It features a vertical sequence of symbols: a brace on the left, followed by a condition involving ' \leq ' and ' f ', then a brace on the right, and finally a brace at the bottom. Above this sequence, three horizontal arrows point from left to right, labeled with ' F ', ' T ', and ' U ' respectively. The word ' $initialisation$ ' is written above the first arrow, ' $condition$ ' above the second, and ' $updation$ ' above the third. A circled ' T ' symbol is located at the bottom left, and the number '2' is at the bottom right.

```
int i=2;  
while(i<=100)  
    point(i);
```

→ point (

|o|

2 3 - - 99 100

