

Conditionals, Loops & Patterns

Instructor: Love Babbar

→ Discord → invalid link

A++
V++

1 min
wait

Conditionals

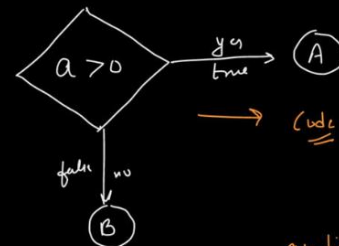
if statement
if-else

learn.thecodehelp.in harsh97310@gmail.com +919068225017

if (condⁿ)
{

execute this

}



audio → Read
issue or
Repeat

```

if ( age > 17 )
{
    you are eligible to
    vote
}
  
```

Handwritten code and logic for a grade assignment and a cricket match outcome.

```

if (marks > 95)
{
    cout << "A grade";
}
  
```

Logic for cricket match outcome:

score = 335
 India wins
 Pak wins
 score = 123

```

int main ()
{
    if (score < 300)
    {
        cout << "India wins";
    }
    else
    {
        cout << "Pak wins";
    }
}
  
```

Annotations for the cricket code:

- 123 < 300 is true.
- 335 < 300 is false.

Handwritten notes on if-else and multiple conditions.

Q 1) if { }

Q 2) if { } else { }

Q 3) if-else

if { } else { }

multiple conditions

Logic for multiple conditions:

- >= 90 → A
- >= 80 → B
- >= 60 → C
- >= 40 → D
- < 40 → F

Handwritten notes on C++ control flow statements:

if statement:

```
if ( )
{
}
```

else if statement:

```
else if ( )
{
}
```

else statement:

```
else if ( )
{
}
```

else statement (skipped):

```
else if ( )
{
}
skip else
```

if-else statement:

```
if ( )
{
}
else if ( )
{
}
```

if-else-if statement:

```
if ( )
{
}
else if ( )
{
}
```

else statement (skipped):

```
else if ( )
{
}
else
{
}
```

Loops

- for
- while
- do-while
- for-each

Repeat

name \rightarrow 5 times

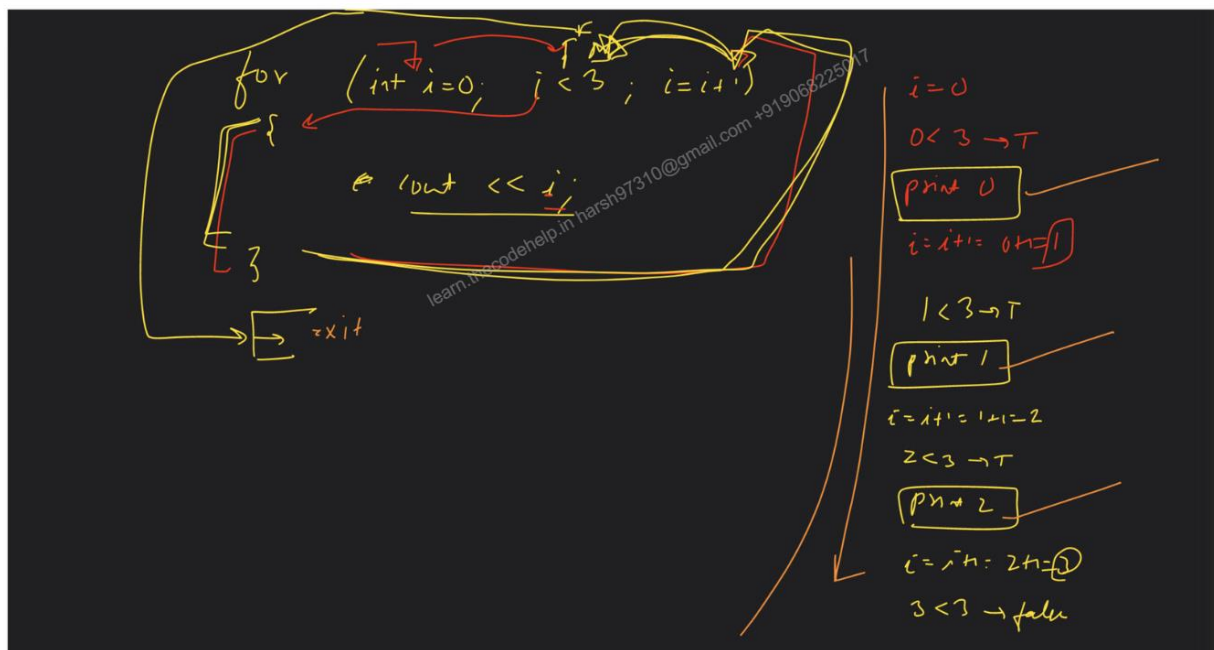
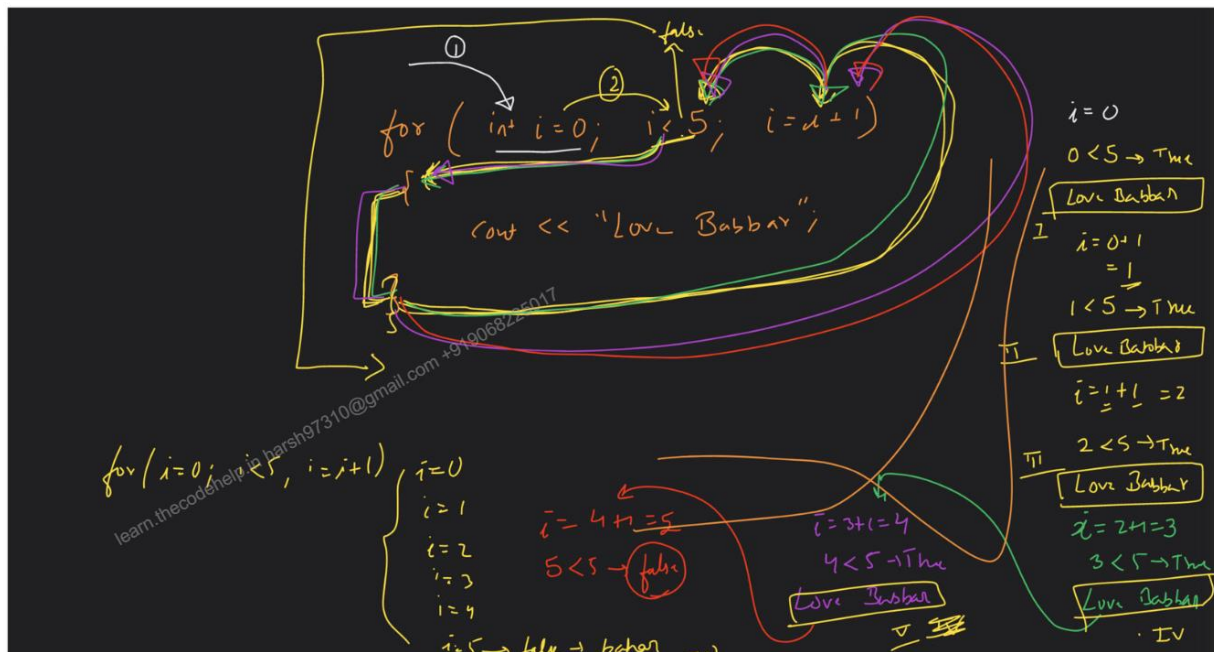
loop

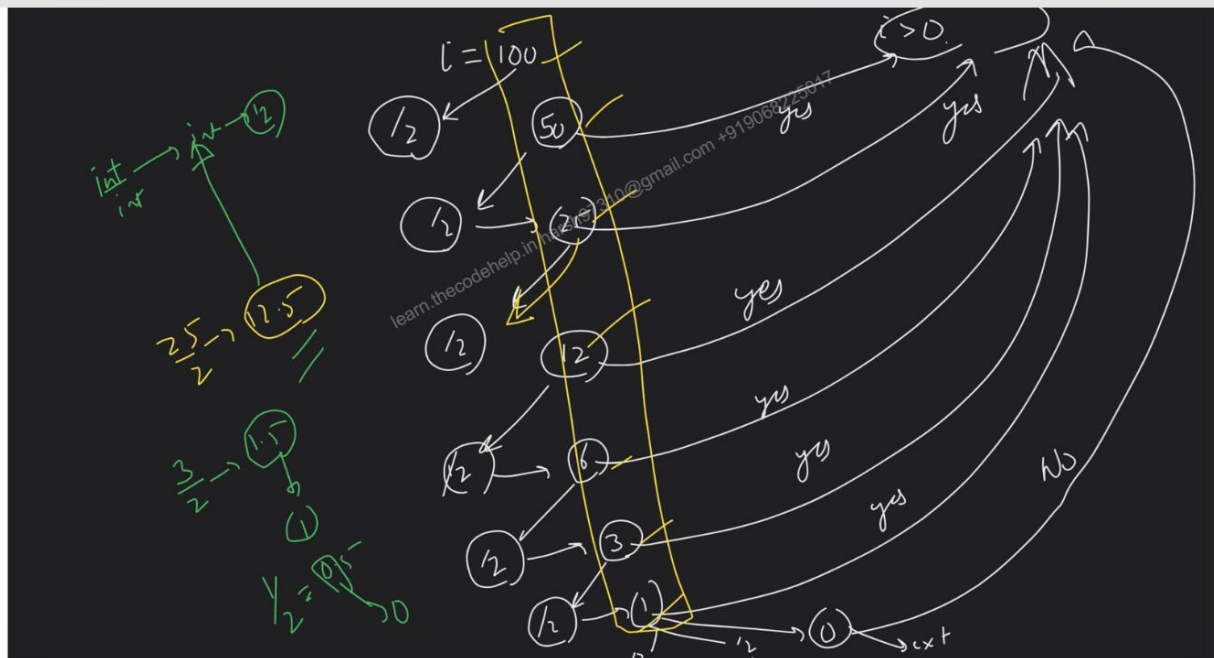
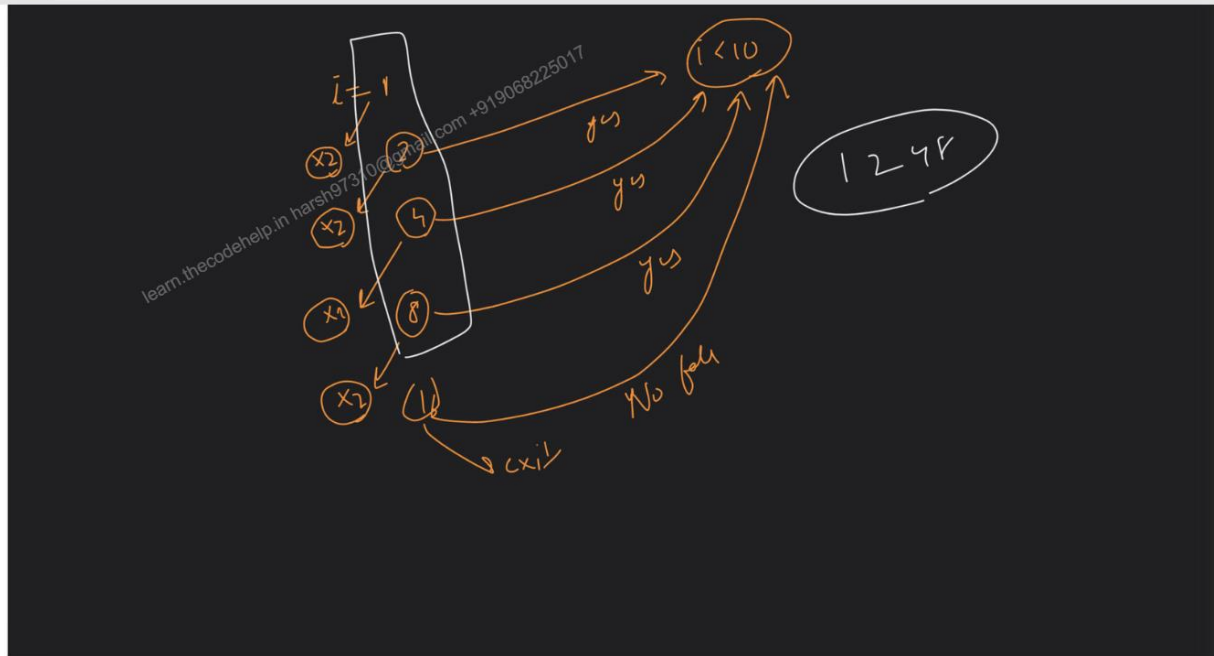
counting \rightarrow 1 \rightarrow 5

```

for (
    init i = 0;
    cond i < 5;
    incl/decl i = i + 1
) {
    // Code
}

```





Handwritten notes on a blackboard:

Left side: A box contains the numbers 0, 2, 4, 6. An arrow points from the box to the condition $i \leq 5$. Below the box, the word "false" is written.

Right side: A multiplication table is shown:

$i \rightarrow 1 \times 2$	$\rightarrow 2$
2×2	4
3×2	6
4×2	8
5×2	10
6×2	12
7×2	14
8×2	16
9×2	18
10×2	20

Watermark: learn.thecodehelp.in harsh97310@gmail.com +919068225017

Handwritten notes on a blackboard:

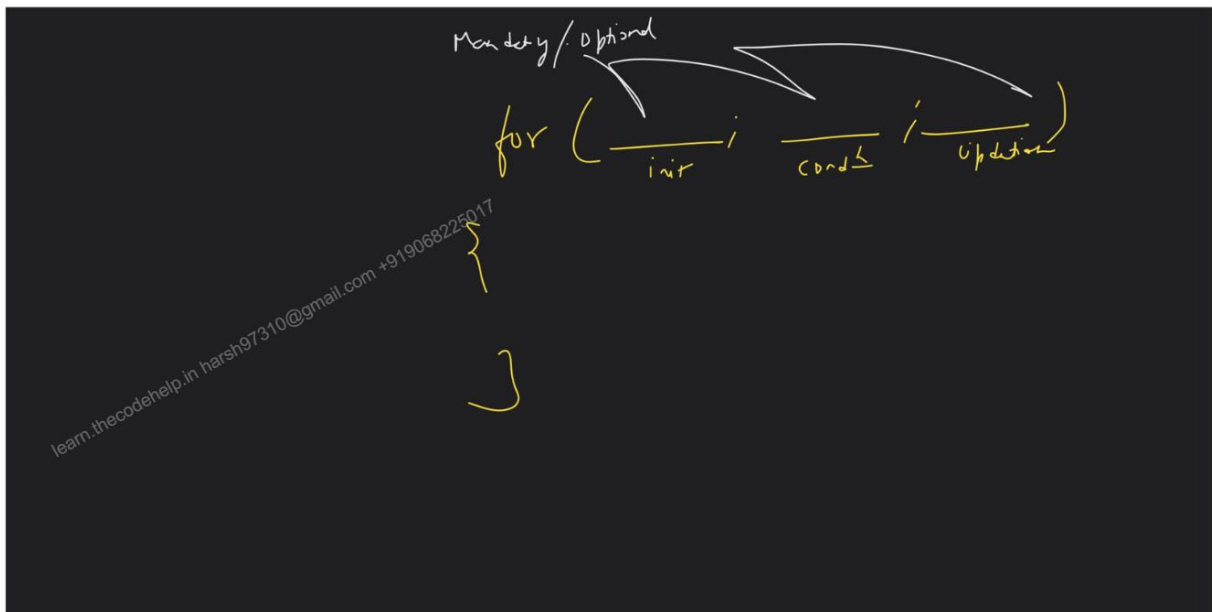
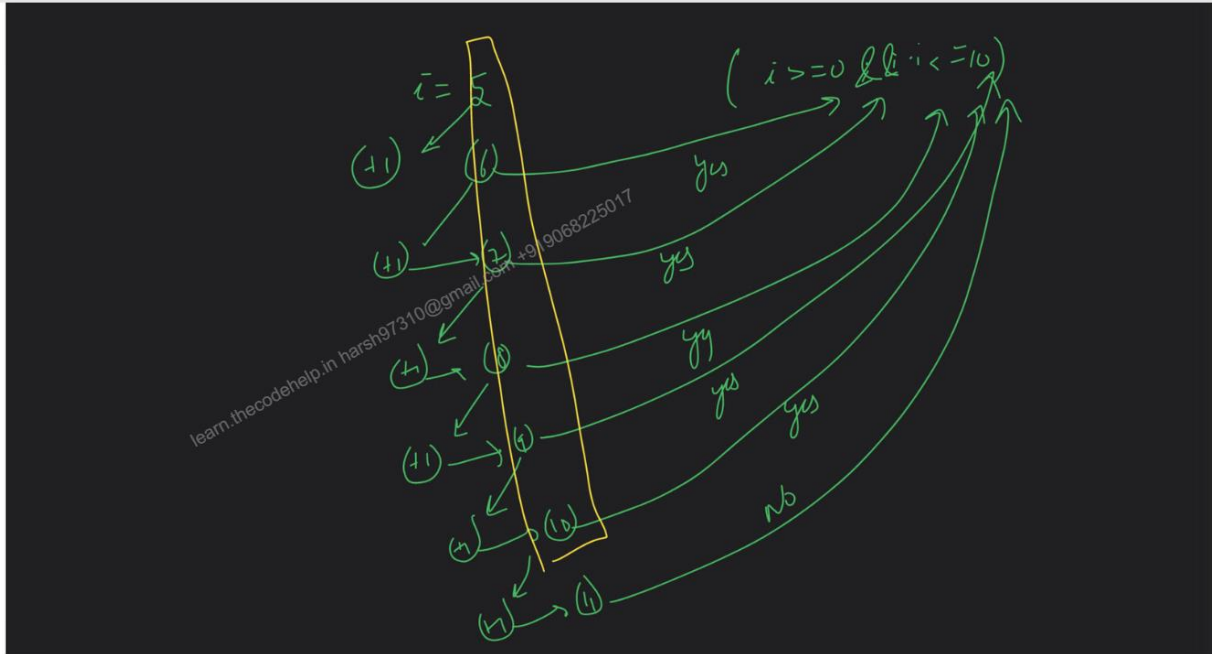
Left side: A box contains the numbers 1, 3, 5, 7. Arrows point from the box to the condition $i \leq 5$. Below the box, the word "false" is written.

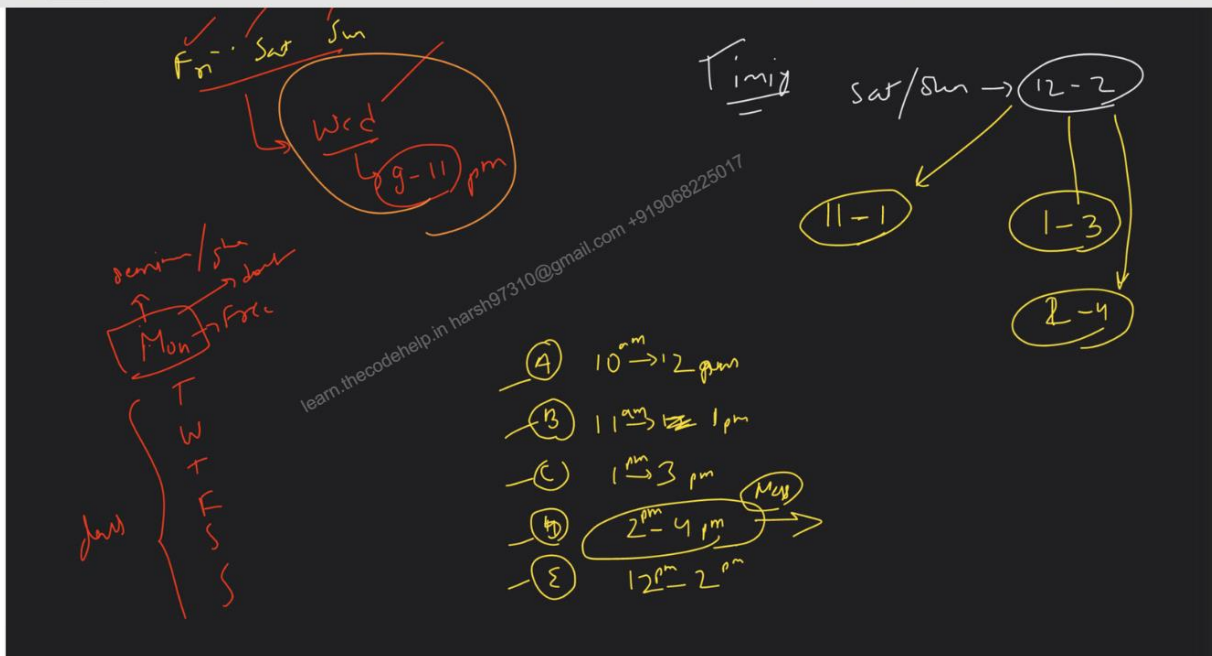
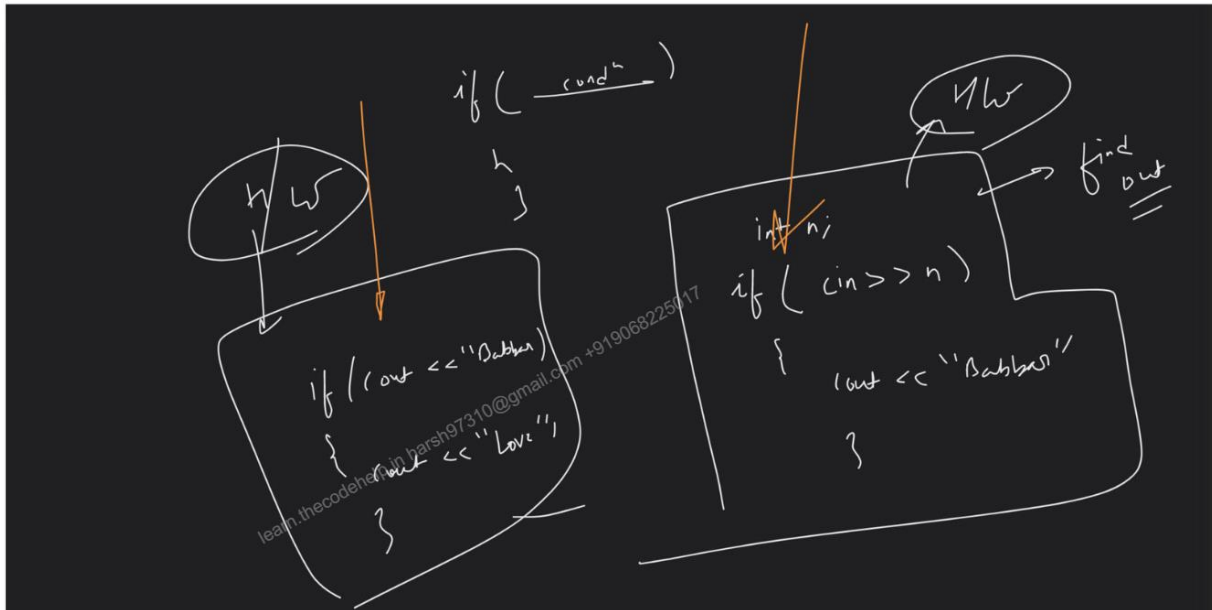
Right side: A loop body is shown:

```

i = 1
do {
  i = i + 2
} while (i <= 5)
  
```

Watermark: learn.thecodehelp.in harsh97310@gmail.com +919068225017





Handwritten code and notes for a C program:

```

int bronum;
<in>> bronum;

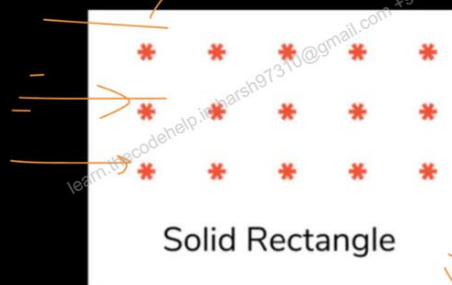
if (bronum == 0)
{
    cout << "Baat Bangagi";
}
else
{
    cout << "Baat Nahi";
}
  
```

Notes:

- 30 min h if-else
- bro
- bro num → variable create
- take input
- bro num == 0
- print "Baat bangagi"
- agar nahi h
- cout << "Baat nahi bangagi"

Solid Rectangle

single solution



Solid Rectangle

2 min
Brow

why?

5 min

Logic Build

Logic Build

Loops → strong

① Row - observation

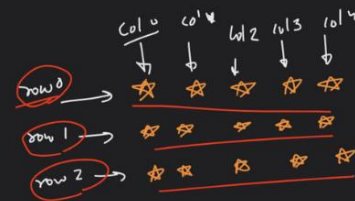
total rows $\rightarrow 3$

② Col - observation

col 0 $\rightarrow 3$ ★
col 1 $\rightarrow 3$ ★
col 2 $\rightarrow 3$ ★
col 3 $\rightarrow 3$ ★
col 4 $\rightarrow 3$ ★

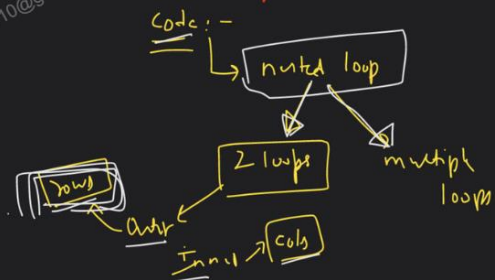
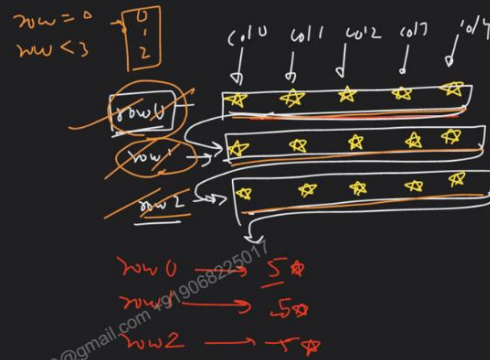
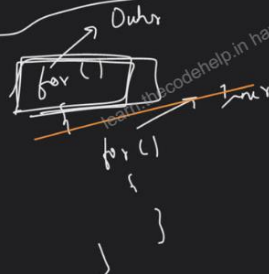
han column
me 3 ★ print karne hai
Rage

Solid Rectangle



Output \rightarrow row \rightarrow row count

```
for (int row = 0; row < 3; row = row + 1)
{
    for (int col = 0; col < 5; col = col + 1)
    {
        cout << "★";
    }
    cout << endl;
}
```



Handwritten code and execution flow for a 3x5 star pattern:

```

for (int row = 0; row < 3; row = row + 1)
{
    for (int col = 0; col < 5; col = col + 1)
    {
        cout << " * ";
    }
    cout << endl;
}

```

Execution Flow:

- row = 0**
 - 0 < 3 → T
 - col = 0
 - 0 < 5 → T
 - col = 1
 - 1 < 5 → T
 - col = 2
 - 2 < 5 → T
 - col = 3
 - 3 < 5 → T
 - col = 4
 - 4 < 5 → T
 - 5 < 5 → false
- row = 1**
 - col = 0
 - 0 < 5 → T
 - col = 1
 - 1 < 5 → T
 - col = 2
 - 2 < 5 → T
 - col = 3
 - 3 < 5 → T
 - col = 4
 - 4 < 5 → T
 - 5 < 5 → false
- row = 2**
 - col = 0
 - 0 < 5 → T
 - col = 1
 - 1 < 5 → T
 - col = 2
 - 2 < 5 → T
 - col = 3
 - 3 < 5 → T
 - col = 4
 - 4 < 5 → T
 - 5 < 5 → false

Pattern Output:

```

* * * * *
* * * * *
* * * * *

```

Handwritten code and execution flow for a 3x5 star pattern with row and column indices:

```

for (int row = 0; row < 3; row = row + 1)
{
    for (int col = 0; col < 5; col = col + 1)
    {
        cout << " * ";
    }
    cout << endl;
}

```

Execution Flow:

- row = 0**
 - col = 0
 - col = 1
 - col = 2
 - col = 3
 - col = 4
 - col = 5
- row = 1**
 - col = 0
 - col = 1
 - col = 2
 - col = 3
 - col = 4
 - col = 5
- row = 2**
 - col = 0
 - col = 1
 - col = 2
 - col = 3
 - col = 4
 - col = 5

Pattern Output:

```

* * * * *
* * * * *
* * * * *

```

Diagram:

A diagram showing the pattern output with row and column indices. The pattern is a 3x5 grid of stars. The row index is 0, 1, 2 and the column index is 0, 1, 2, 3, 4. The pattern is shown as a 3x5 grid of stars.

Flowchart:

```

graph TD
    row = 0
    col = 0
    col < 5 --> T --> col++1
    col < 5 --> F --> row++1
    row < 3 --> T --> col = 0
    row < 3 --> F --> End

```

①

```

for (int row = 0; row < 4; row = row + 1)
{
    for (int col = 0; col < 4; col = col + 1)
    {
        cout << " * ";
    }
    cout << endl;
}

```

0 → 4 1 → 5
 Square Pattern

row 0 → * * * *

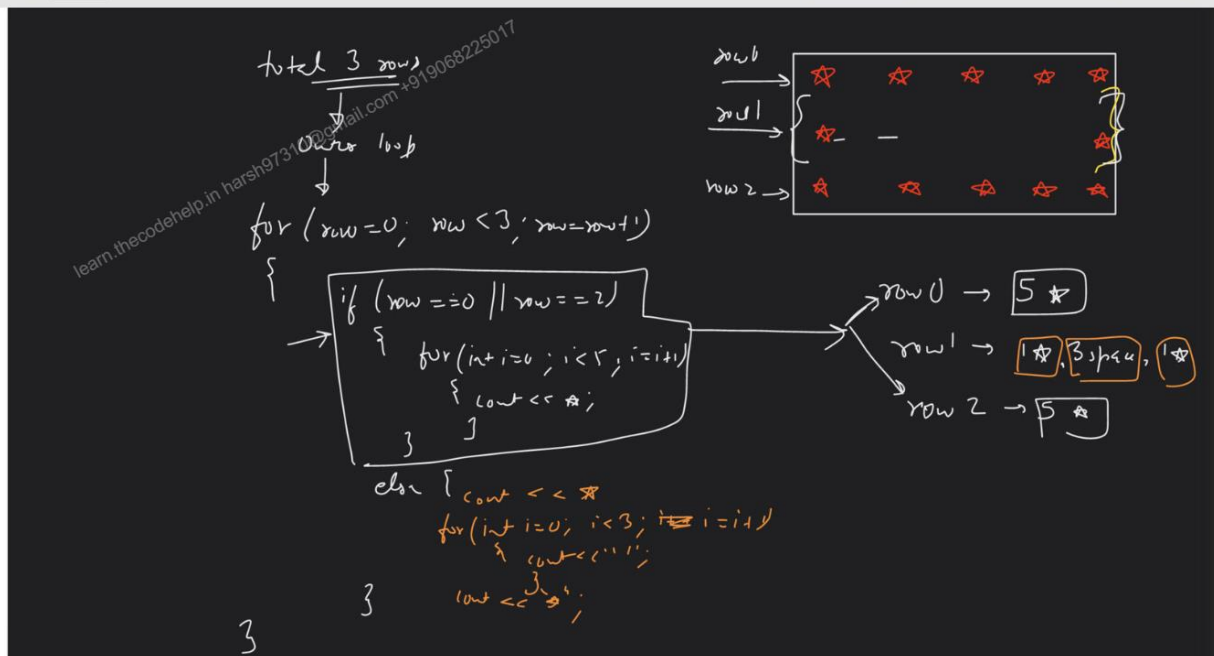
row 1 → * * * *

row 2 → * * * *

row 3 → * * * *

row 0 → 1 *
 row 1 → 4 *
 row 2 → 9 *
 row 3 → 16 *

Yes



total rows $\rightarrow 6$

Output

```

for (int row=0; row<6; row=row+1)
{
    if (row==0 || row==5)
    {
        for (int col=0; col<5; col=col+1)
        {
            cout << '*';
        }
    }
    else
    {
        cout << ' ';
        for (int col=0; col<3; col=col+1)
        {
            cout << "-";
        }
        cout << ' ';
    }
    cout << endl;
}

```

Diagram showing the output pattern of stars and spaces for rows 0 to 5.

Row 0 \rightarrow 5 stars

Row 1 \rightarrow 1 star, 3 spaces, 1 star

Row 2 \rightarrow 1 star, 3 spaces, 1 star

Row 3 \rightarrow 1 star, 3 spaces, 1 star

Row 4 \rightarrow 1 star, 3 spaces, 1 star

Row 5 \rightarrow 5 stars

for (int row=0; row<6; row=row+1)

Diagram showing the output pattern of stars and spaces for rows 0 to 5, with annotations for row counts and column counts.

Row 0 \rightarrow 5 stars

Row 1 \rightarrow 1 star, 3 spaces, 1 star

Row 2 \rightarrow 1 star, 3 spaces, 1 star

Row 3 \rightarrow 1 star, 3 spaces, 1 star

Row 4 \rightarrow 1 star, 3 spaces, 1 star

Row 5 \rightarrow 5 stars

Annotations:

- Row count: row=0, row=1, row=2, row=3, row=4, row=5
- Column count: col=0, col=1, col=2, col=3, col=4
- Row 0: 5 stars
- Row 1: 1 star, 3 spaces, 1 star
- Row 2: 1 star, 3 spaces, 1 star
- Row 3: 1 star, 3 spaces, 1 star
- Row 4: 1 star, 3 spaces, 1 star
- Row 5: 5 stars

Diagram showing the output pattern of stars and spaces for rows 0 to 5, with annotations for row counts and column counts.

Row 0 \rightarrow 5 stars

Row 1 \rightarrow 1 star, 3 spaces, 1 star

Row 2 \rightarrow 1 star, 3 spaces, 1 star

Row 3 \rightarrow 1 star, 3 spaces, 1 star

Row 4 \rightarrow 1 star, 3 spaces, 1 star

Row 5 \rightarrow 5 stars

Annotations:

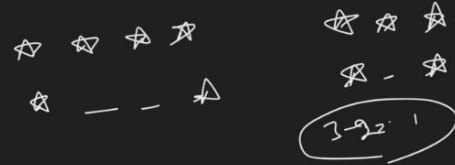
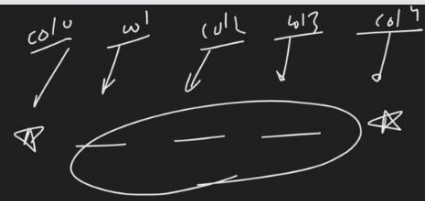
- Row count: row=0, row=1, row=2, row=3, row=4, row=5
- Column count: col=0, col=1, col=2, col=3, col=4
- Row 0: 5 stars
- Row 1: 1 star, 3 spaces, 1 star
- Row 2: 1 star, 3 spaces, 1 star
- Row 3: 1 star, 3 spaces, 1 star
- Row 4: 1 star, 3 spaces, 1 star
- Row 5: 5 stars

5 col \rightarrow 3 spec

$$8 - 2 \rightarrow 6 \text{ sp}^2$$
$$n-2 \rightarrow 2 \text{ space}$$

no. of $10^1 - 2 \rightarrow$ spaces

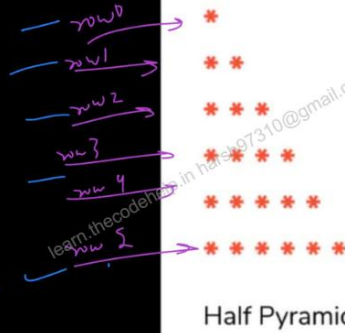
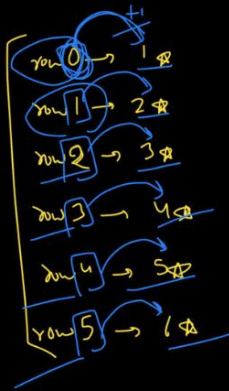
colloist - 2



```
for (int i = 0; i < s.length(); i++)
```



Half Pyramid



total rows → n

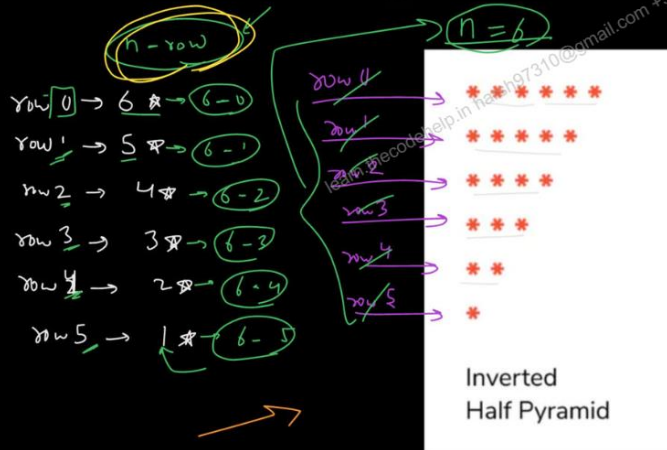
```
for (int row = 0; row < n; row++)
{
    for (int col = 0; col < row + 1; col++)
    {
        cout << "★";
    }
    cout << endl;
}
```

total rows → 5

```
for (int row = 0; row < 5; row++)
{
    for (int col = 0; col < row + 1; col++)
    {
        cout << "★";
    }
    cout << endl;
}
```



Inverted Half Pyramid

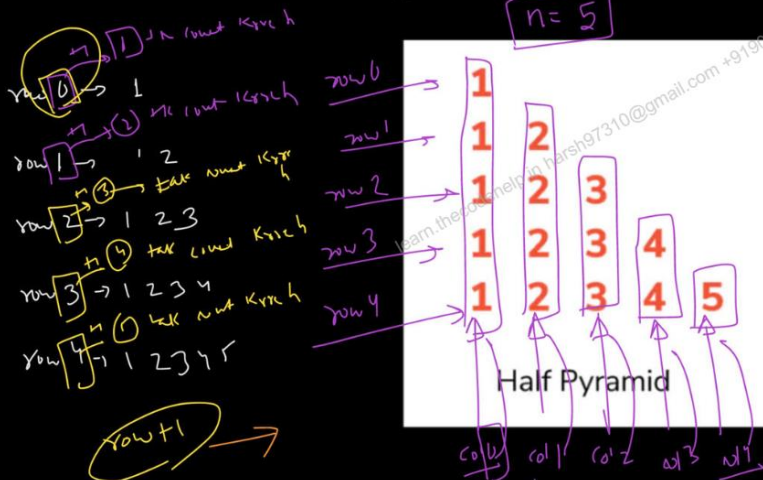


```
for (int row=0; row < n; row=row+1)
```

```
{
    for (int col=0; col < n-row; col=col+1)
    {
        cout << "x";
    }
    cout << endl;
}
```

Steps: - (1) row count → row loop
 Step 2 → row break → formula → inner loop

Numeric Half Pyramid



```
for (int row=0; row < n; row=row+1)
```

```
{
    for (int col=0; col < row+1; col=col+1)
    {
        cout << col+1;
    }
    cout << endl;
}
```

row = 0
 $0 < 3 \rightarrow T$
 Inner loop starts now
 $col = 0$
 $0 < row + 1, 0 < 1 \rightarrow T$
 $print\ col + 1 \rightarrow print\ 1$
 $col = 1$
 $1 < 1 \rightarrow F$
 break new inner loop
 $row = 1$
 $1 < 3 \rightarrow T$
 Inner loop starts
 $col = 0$
 $0 < row + 1, 0 < 2 \rightarrow T$
 $print\ col + 1 \rightarrow print\ 1$
 $col = 1$
 $1 < 2 \rightarrow T$
 $print\ col + 1 \rightarrow print\ 2$
 $col = 2$
 $2 < 2 \rightarrow F$
 Inner loop ends
 $row = 2$
 $2 < 3 \rightarrow T$
 Inner loop
 $col = 0$
 $0 < row + 1, 0 < 3 \rightarrow T$
 $print\ col + 1 = 0 + 1$
 $col = 1$
 $1 < 3 \rightarrow T$
 $print\ col + 1 = 1 + 1 = 2$
 $col = 2$
 $2 < 3 \rightarrow T$
 $print\ col + 1 = 2 + 1 = 3$
 $col = 3$
 $3 < 3 \rightarrow F$
 Inner loop ends
 $row = 3$
 $3 < 3 \rightarrow F$
 Outer loop ends

n = 3

Inverted Half Pyramid

n = row = 5 - 0 = 5

row 0: 1, 2, 3, 4, 5
 row 1: 1, 2, 3, 4 (n - row = 5 - 1 = 4)
 row 2: 1, 2, 3 (n - row = 5 - 2 = 3)
 row 3: 1, 2 (n - row = 5 - 3 = 2)
 row 4: 1 (n - row = 5 - 4 = 1)
 row 5: (n - row = 5 - 5 = 0)

n = 5

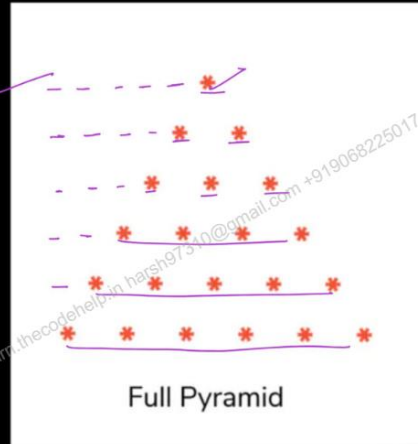
Inverted Half Pyramid

col 0 col 1 col 2 col 3 col 4

```
for (int row = 0; row < n; row = row + 1)
{
    for (int col = 0; col < n - row; col = col + 1)
    {
        print col + 1
    }
    WCD
    g - 11 gn
}
```

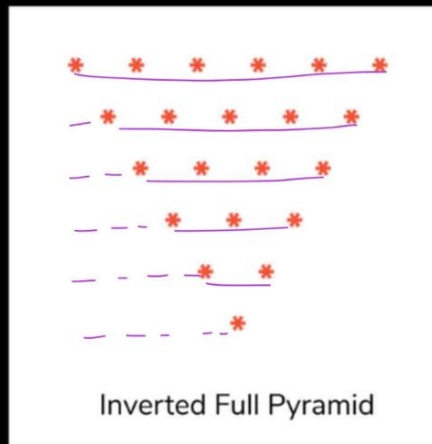
Full Pyramid:

3 class → padhai
 1 class → review
 (W.C.)
 doubt
 discussion
 motivation
 game
 feedback
 QnA
 topic



```
for ( )
{
  // space
  for ( )
  {
  }
  // star
  for ( )
  {
  }
}
```

Inverted Full Pyramid:



```
for ( )
{
  // space
  for ( )
  {
  }
  // star
  for ( )
  {
  }
}
```

Numeric Full Pyramid:

```
  1
 2 3 2
3 4 5 4 3
4 5 6 7 6 5 4
5 6 7 8 9 8 7 6 5
```

Full Pyramid

Numeric Hollow Full Pyramid

```
  1
 1 2
 1 3
 1 4
1 2 3 4 5
```

Hollow Full Pyramid