

~~→ ↓ ← Lecture - 4 → ←~~

### 13) Solid Diamond

\* \*

\* \* \* → space → n - How -

\* \* \* \* → star → How + 1

\* \* \* \* \* → space → How

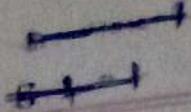
\* \* \* \* → star → n - How

\* \*

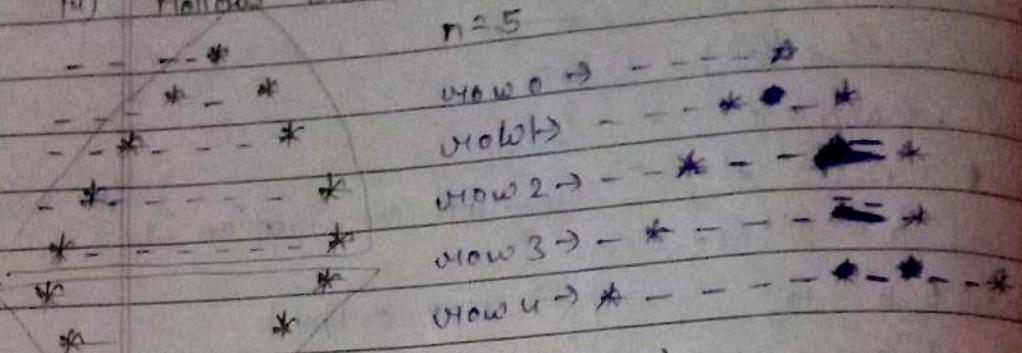
\*

Done by us code

$$\begin{aligned} * \text{Even } n &= 2 * m \\ * \text{odd } n &= 2 * m + 1 \\ &= 2 * m + 1 \end{aligned}$$



(ii) Hollow Diamond



space  $\rightarrow n - \text{row} - 1$

start space  $\rightarrow 2 * \text{row} + 1$

if ( $\text{col} == 0$ )  $\rightarrow *$   $\rightarrow$  first char

else if ( $\text{col} == 2 * \text{row}$ )  $\rightarrow *$   $\rightarrow$  last char

else ()  $\rightarrow -$

1<sup>st</sup> part four (int row = 0; row < n; row++) {

four (int col = 0; col < n - row - 1; col++) {

cout << " " ;

four (int start = 0; start < 2 \* row + 1; start++) {

if ( $\text{col} == 0$ )

cout << "\*";

}

else if ( $\text{col} == 2 * \text{row}$ )

cout << "\*";

}

else {

cout << " " ;

}

row 0  $\rightarrow$  0 space & 7 char

row 1  $\rightarrow$  2 space & 5 char

row 2  $\rightarrow$  2 space & 3 char

row 3  $\rightarrow$  3 space & 1 char

space  $\rightarrow$  row

(\*) row spaces  $= 2n - 2 * \text{row} - 1$

$\rightarrow$  if ( $\text{col} = 0 \text{ || } \text{col} = 2n - 2 * \text{row} - 2$ )  $\rightarrow *$

$\rightarrow$  else ()  $\rightarrow$  space

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

{ for (int space = 0; space < row; space++)

{ cout << " ";

}

for (int stow = 0; stow < (2 \* n - 2 \* row - 1); stow++)

. { if (col == 0 || col == 2n - 2 \* row - 2)

{ cout << "\*";

}

else {

cout << " ";

}

}

cout << endl;

}

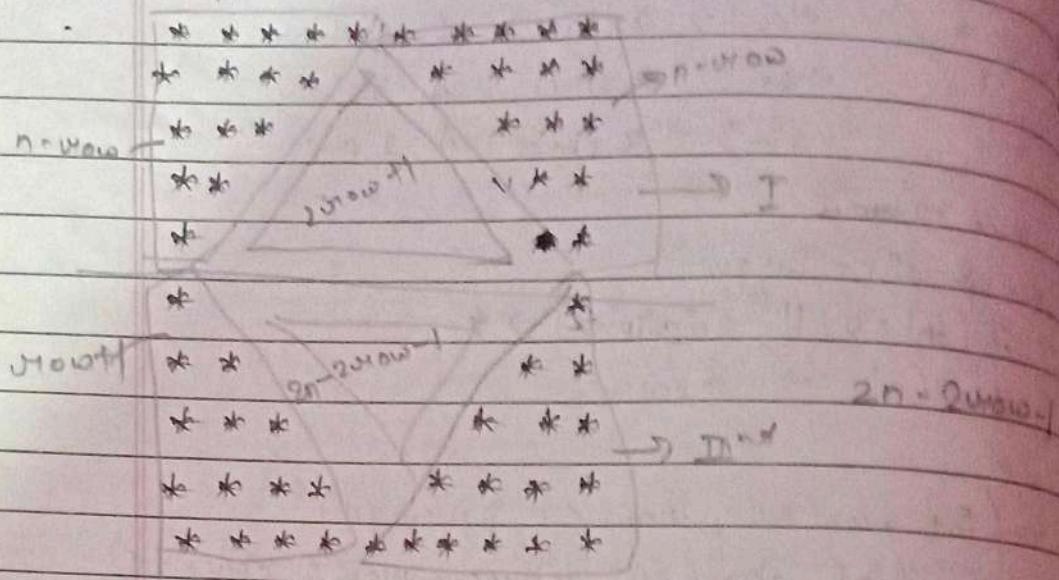
Done in vscode

QnP

$$1, 3, 5, 7 \rightarrow 2\text{row} + 1$$

$$1, 5, 3, 1 \rightarrow 2n - 2\text{row} - 1$$

15) Flipped solid Diamond



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

    11 half pyramid

    for (int col=0; col<n-row; col+1){

        cout << "\*";

}

    11 space wise full pyramid

    for (int col=0; col<2\*n-row+1; col+1){

        cout << " ";

}

    for (int col=0; col<n-row; col+1){

        cout << "\*";

}

    cout << endl;

}

Done

⑥ Fancy Pattern 2

a man 0 → 2 point

2 \* 2 man 2 → 2 point \*

3 \* 3 \* 3 man 3 → 3

4 \* 4 \* 4 \* 4 man 4 → 4

4 \* 4 \* 4 \* 4

3 \* 3 \* 3 \*

2 \* 2

1

for (int man=0; man<n; man++) {

for (int col=0; col<man; col++) { → number

cout << man+1;

if (col+man) {

cout << "\*"; } else blank

}

cout << endl;

}

for (int man=0; man<n; man++) {

for (int col=0; col<n-man; col++) {

cout << "n-man";

if (col+man) {

cout << "\*"; }

}

} cout << endl;

}

Dans

Q 1

row+1

2 2

3 3 3

4 4 4 4

5 5 5 5 5

for (int row = 0; row < n; row++) {

    for (int col = 0; col < row + 1; col++) {

        cout << row + 1;

    }

    cout << endl;

3

3

Q 2 Alphabets Polindromic Pyramid :-

Q 3 A

A B A

A B C B A

A B C D C B A

A B C D E D C B A

for (int row = 0; row < n; row++) {

    for (int col = 0; col < row + 1; col++) {

        int ans = col + 1;

        char ch = ans + 'A' - 1;

        cout << ch;

    }

for (int col = n - 1; col >= 0; col--) {

    int ans = col;

    char ch = ans + 'A' - 1;

    cout << ch;

3

3 Out-Second;

cone (Backside)

1  
1 2  
1 2 3  
1 2 3 4

⑩ ~~1  
1 2 1  
1 2 3 2 1  
1 2 3 4 3 2 1~~ <sup>reverse counting</sup>

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

for (col = 0; col < row + 1; col = col + 1) {  
 cout << col + 1;

}

col = col - 1; OR (int col = row)

for ( ; col >= 1; col--) {

cout << col;

}

cout << endl;

g

⑪ 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

row 0 → 0 elem

row 1 → 1 "

row 2 → 2 " (row)

row 3 → 3 " (row2)

row 4 → 4 " (row3)

row + 1  
row + 1 + 1

```

for (int row=0; row < n; row++) {
    for (int col=0; col < n-row-1; col++) {
        cout << " ";
    }
    for (int col=0; col < row+1; col++) {
        cout << row+col+1;
    }
    if (int start = 2*row);
    for (int col=0; col < row; col++) {
        cout << start;
        start = start - 2;
    }
    cout << endl;
}

```

20

### ⑪ Numeric Hollow Pyramid (Do it 5 times)

---	1	row 0 → 1 element
' ---	1 2	row 1 → 3 elem
' --	1 .. 3	row 2 → 5 elem
' -	1 . . . , 4	row 3 → 7 elem
' -	1 2 3 4 5	row 4 → 9 elem 2 * row + 1

```

for (int row=0; row < n; row++) {
    for (int col=0; col < n-row-1; col++) {
        cout << " ";
    }
    if (col == 0) {
        if (row < n-1) {
            cout << " ";
        } else {
            cout << " ";
        }
    }
}

```

space {

if ← if (col == 0) { } else { }

condition if (row < n-1) { }

next page

```

* (del) { } 
cout << 1; }
else if (col == 2 * (row)) {
    cout << "H" + 1;
else {
    cout << " ";
}
}
cout << endl;
}
}

```

if (row ==  
n-1),

```

four(int col = 1;
col <= n; col++)
cout << col << " ";
}
}
}

```

## ② Hollow Square Pattern

* * * * *	n=5
* . . . *	row 0 → 5 *
* * * * *	row 1 → 3 space, 2 star
* * * * *	row 2 → 3 space, 2 star
* * * * *	row 3 → 3 space, 2 star
* * * * *	row 4 → 5 *

```

four(int row = 0; row < n; row++) {
    four(int col = 0; col < n; col++) {
        if (col == 0 || col == n-1) {
            cout << "*";
        }
        else if (row == 0 || row == n-1) {
            cout << "*";
        }
        else {
            cout << " ";
        }
    }
    cout << endl;
}
}

```

(21)

Hollow Inverted Half Pyramid:-

\* \* \* \* \*  $\rightarrow$  row  $\rightarrow$  5 \*

\* \* \* \* \*  $\rightarrow$  2 \* , 2 space  $\rightarrow$  col = 2

\* \* \* \* \*  $\rightarrow$  2 \* , 2 space  $\rightarrow$  col = 3

\* \* \* \* \*  $\rightarrow$  2 \*  $\rightarrow$  col = 4

\* \* \* \* \*  $\rightarrow$  2 \*  $\rightarrow$  col = 5

\* \* \* \* \*  $\rightarrow$  2 \*  $\rightarrow$  col = 6

for (int row = 0; row < n; row++) {

    for (int col = 0; col < n; col++) {

        if ( $\text{row} == 0 \text{ || } \text{col} == 0 \text{ || } \text{col} == n - \text{row}$ )

            cout << "\*";

        }

    else {

        cout << " ";

}

Done

(22)

4 4 4 4

3 3 3

2 2

1

for (int row = 0; row < n; row++) {

    for (int col = 0; col < n - row; col++) {

        }

        cout << n - row << " ";

    }

    cout << endl;

}

Done

Same as Question 17 but code diff<sup>th</sup>

Hollow Full Pyramid :- n=5

- - *	view 0 $\rightarrow$ 4 space, 3 stars, 1 elem
- * *	view 1 $\rightarrow$ 3 space, 2 stars, 3 elem
- * *	view 2 $\rightarrow$ 2 space, 2 stars, 5 ele
* *	view 3 $\rightarrow$ 2 space, 2 stars, 7 ele
* * *	view 4 $\rightarrow$ 0 space, 5 stars, 9 ele $n = \text{view} + 1$ $n = \text{view} + 1$

Space  $\rightarrow$  n-view-1

if (view == n-1)  $\rightarrow$  last line

for (int view=0; view<n;

if (view+1, 2=0)  $\rightarrow$  \*

else  $\rightarrow$  space

else if (2\*view+1)  $\rightarrow$  first 3 line

if (col==0 || col==2\*view)  $\rightarrow$ \*

else  $\rightarrow$  space

14th  
Ques. (2) Numeric Hollow Half Pyramid :-

1

2 \*

1 3

2 4

3 4 5

for (int view=0; view<n;  
view++) {

for (int col=0; col<view+1;  
col++) {

if (col==0 || col==view ||  
view==n-1) {

cout << col+1;

3

else {

cout << " ";

cout << endl;

else {  
cout << "

}

{

cout << endl;

}

(25)

\*

\* \*

\* \*

\* \*

\* \* \* \*

for (int row = 0; row < n; row++) {

for (int col = 0; col < row + 1; col++) { (27)

if (col == 0 || col == row ||

row == n - 1) {

cout << "\*";

}

else {

cout << " ";

}

{

cout << endl;

}

3

### Numeric Hollow Inverted Half Pyramid

```

27.    1 2 3 4 5      * * * *
        2           5      *   *
        3           5      *   *
        4           5      *   *
        5           *      *
  
```

for (int row = 0; row < n; row++) {

for (int col = 0; col < n - row; col++) {

if (row == 0 || col == 0 || col == n - row - 1) {

cout << " " << col + row + 1;

}

} else {

cout << " ";

}

cout << endl;

}

### 27. Numeric Palindrome Equilateral pyramid.

			1					
		1	2	1				
	1	2	3	2	1			
-	1	2	3	4	3	2	1	
1	2	3	4	5	4	3	2	1

for (int row = 0; row < n; row++) {

for (int col = 0; col < n - row - 1; col++) {

cout << " ";

}

for (int col = 0; col < row + 1; col++) {

cout << col + 1;

## Language printing

Same (Int. val = max; col >= 0; col -  
curr <= col);

一

Can't be added

2

卷之三

卷之三

15

Castles

四

Fancy Patterns 1

125

Four (Part 1)  $\omega_{\text{HO}} = 0^\circ$ ,  $\psi_{\text{HO}} < 0^\circ$ ,  $\psi_{\text{HO}} + \phi$ )

The first col = 0; col < n + 3 - width; left  
last is "\*" ;

2

`for (int col=0; col<(2*x+1); col++) {`

if (col%2 == 0) {

can't be meeting

1

else {

cout << "\*";

}

}

for (int col = 0; col < n + s - max; col++) {

cout << "\*";

}

cout << endl;

}

}

Done

In

vs code

② Solid Half Diamond :-

\* now 0 → 2 (view +)

\*\* now 2 → 2

\*\*\* now 2 → 3

\*\*\*\* now 3 → 4

\*\*\*\*\* now 0 → 4 (n - view)

\*\*\* now 2 → 3

\*\* now 2 → 2

\* now 2 → 2

for (int view = 0; view < n; view++) {

for (int col = 0; col < view + 1; col++) {

cout << "\*";

}

cout << endl;

}

```
for (int row = 0; row < n; row++) {  
    for (int col = 0; col < n - row; col++) {  
        cout << "*";
```

{

cout &lt;&lt; endl;

{

{

③  
Double

fancy Pattern 3

\* row 0 → 1

\* 1 \* row 1 → 3

\* 1 2 1 \* row 2 → 5

\* 1 2 3 2 1 \* row 3 → 7

\* 1 2 1 \* row 0 → 5

\* 1 \* row 2 → 3

\* row 2 → 1

{ 2 \* row + 1 }

{ 2n - 2 \* row }

Spiral

for (int row = 0; row &lt; n; row++) {

int count = 0;

for (int col = 0; col &lt; 2 \* row + 1; col++) {

if (col == 0 || col == 2 \* row) {

cout &lt;&lt; "\*";

if (count &lt; 1) {

for (int col1 = 0; col1 &lt; row; col1++) {

cout &lt;&lt; col1 + 1;

col1++;

{

{

if (count + 2) {

for (int col2 = 0; col2 < max - 1; col2++) {

cout << max - 1 - col2;

col2++;

3

count++;

3

### (23) Fancy Pattern 2

1

max 0 → 1

2 \* 3

max 1 → 3

2 max + 1

4 \* 5 \* 6

max 2 → 5

7 \* 8 \* 9 \* 10 max 3 → 7

7 \* 8 \* 9 \* 10 max 0 → 7

9 \* 5 \* 6

max 2 → 5

2n + 2 max + 1

2 \* 3

max 2 → 3

1

max 4 → 2

int count = 2;

for (int max = 0; max < n; max++) {

for (int col = 0; col < 2 max + 1; col++) {

If (col % 2 == 0) {

cout << count;

count++;

3

else {

cout << count;

} 3 cout << endl;

3

int count2 = count + n;

for (int max = 0; max < n; max++) {

int count2 = count + n;

for (int col = 0; col < 2 n - 2 max + 1; col++) {

If (col % 2 == 0) {

cout << count2;

count = 0; }  
else { count = 4 \* n / 3;

93 count = 2, count = (n - 4 \* m - 1);  
cout << endl;

3) Floyd's Triangle Pattern

count = 1	1
count = 2	1 2
count = 3	1 2 3
count = 4	1 2 3 4
count = 5	1 2 3 4 5
count = 6	1 2 3 4 5 6
count = 7	1 2 3 4 5 6 7

n = 7

count = 8	1 2 3 4 5 6 7 8
count = 9	1 2 3 4 5 6 7 8 9
count = 10	1 2 3 4 5 6 7 8 9 10
count = 11	1 2 3 4 5 6 7 8 9 10 11
count = 12	1 2 3 4 5 6 7 8 9 10 11 12
count = 13	1 2 3 4 5 6 7 8 9 10 11 12 13
count = 14	1 2 3 4 5 6 7 8 9 10 11 12 13 14
count = 15	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
count = 16	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
count = 17	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
count = 18	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
count = 19	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
count = 20	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
count = 21	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21

count = 22	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
count = 23	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
count = 24	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
count = 25	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
count = 26	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
count = 27	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27
count = 28	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28

400 + 1

int count = 1;

for (int row = 0; row < n; row++) {

    for (int col = 0; col < row + 1; col++) {

        cout << count << " ";

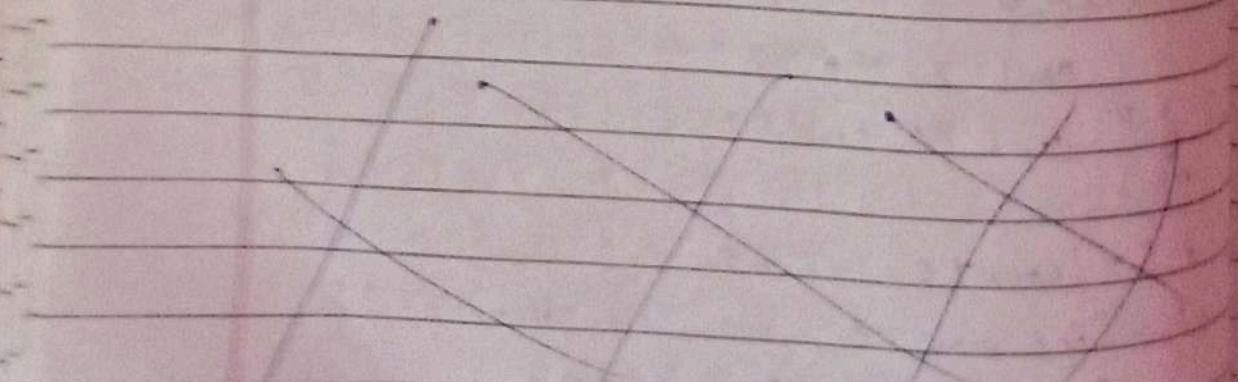
        count++;

}

    cout << endl;

}

}



	Row	Column	Value
1	1	1	1
2	2	2	2
3	3	3	3
4	4	4	6
5	5	5	10 10 5
6	6	6	15 20 15 6

n=7

Pascal Triangle make Binomial Coefficient  
 formula:  $c = c * (\text{row} - \text{col})/\text{col};$   
~~count~~

for (int row = 1; row <= n; row++) {

    int count = 1;

    for (int col = 1; col <= row; col++)

        cout << count << " "

        count = count \* (row - col) / col;

}

    cout << endl;

}

$$1+1=2$$

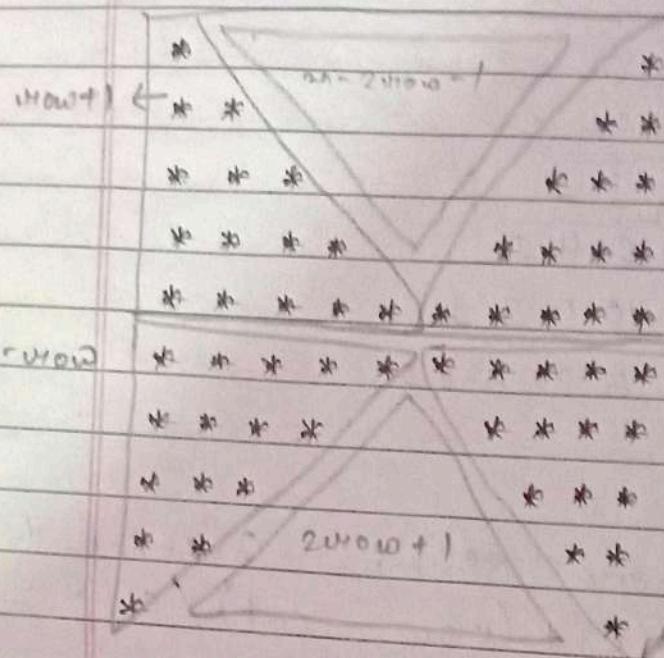
$$1+2=3$$

1 2 1

1 3 3 1

1 4 6 4 1

(34) Butterfly Pattern



four (int row = 0; row < n; row++) {

    four (int col1 = 0; col1 < row + 1; col1++) {

        cout << "\*"; }

    four (int col2 = 0; col2 < 2 \* n - 2 \* row - 1; 0

    { cout << " "; }

    four (int col3 = 0; col3 < row + 1; col3++) {

        cout << "\*"; }

    cout << endl;

}

    four (int row = 0; row < n; row++) {

        four (int col1 = 0; col1 < row + 1; col1++) {

            cout << "\*"; }

        four (int col2 = 0; col2 < 2 \* row + 1; col2++) {

            cout << " "; }

        four (int col3 = 0; col3 < n - row; col3++) {

            cout << "\*"; }

    cout << endl;

}

~~→ ← ← ← ← ←~~  
~~← → ← → ← ←~~  
~~Operations, function f for move~~

⑤ 1 1 1  
2 2 2      `for(int row=1; row<=n; row++) {`  
3 3 3      `for(int col=1; col<=n; col++) {`  
              `cout << move;`  
              }  
              `cout << endl;`  
              }

⑥ 1 2 3 4      `for(int row=1; row<=n; row++) {`  
1 2 3 4      }  
j 2 3 4      `for(int col=1; col<=n; col++) {`  
1 2 3 4      `cout << col;`  
              }  
              `cout << endl;`  
              }

⑦ 3 2 1      `for(int row=1; row<=n; row++) {`  
3 2 1      `for(int col=1; col<=n; col++) {`  
3 2 1      `cout << col;`  
              }  
              `cout << endl;`  
              }

(38)

1 2 3  
4 5 6

7 8 9      int count = 2;

for (int row = 0; row &lt; n; row++)

for (int col = 0; col &lt; n; col++)

{ cout &lt;&lt; count;

count++;

}

cout &lt;&lt; endl;

}

(39)

1      0 → row+1

2 2

3 3 3

4 4 4 4

for (int row = 0; row &lt; n; row++)

for (int col = 0; col &lt; row; col++)

{

cout &lt;&lt; row;

}

} } cout &lt;&lt; endl;

(40)

1

2 3      int count = 1;

4 5 6      for (int row = 1; row &lt; n; row++)

7 8 9 10 ... }

for (int col = 0; col &lt; row; col++)

{ cout &lt;&lt; count &lt;&lt; " ";

count++;

}

cout &lt;&lt; endl;

}

}

$0 \rightarrow \text{view} + 1$

(1) 1  
2 3  
3 4 5  
4 5 6 7

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

```

(2) 1  
2 1  
3 2 1  
4 3 2 1

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cout << "Enter number" << endl;
    cin >> n;
    for (int view = 1; view <= n; view++) {
        for (int col = view; view col >= 1; col--) {
            cout << col << " ";
        }
        cout << endl;
    }
}
```

(13) A A A

B B B

C C C

for (int row = 0; row < n; row++)  
for (int col = 0; col < n; col++)  
char ans = row + 'A';  
cout << ans;

}

cout << endl;

}

}

(14) A B C

A B C for (int row = 0; row < n; row++)  
A B C { for (int col = 0; col < n; col++)  
{ char ans = col + 'A';  
cout << ans;

}

cout << endl;

}

}

int count = 0;

(15)

A B C

D E F

G H I

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

{

for (int col = 0; col < n; col++)

{

char ans = count + 'A';

cout << ans;

count ++;

}

cout << endl;

}

}

'A' + row \* col - 2

(16) A B C

B C D for (int row = 0; row < n; row++)  
C D E if for (int col = 0; col < n; col++) {  
cout charr ans = 'A' + col + row - 2;  
cout << ans;

}

cout << endl;

3

3

(17) 1 2 3

2 3 4 for (int row = 0; row < n; row++) {  
3 4 5 for (int col = 0; col < n; col++) {  
cout << col + row;

3

cout << endl;

3

3

(18) A

B B

C C C

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

char ans = ('A' + row);

cout << ans;

3

cout << endl;

3

(11)

A  
B C  
D E F  
G H I J

int count = 0;

for (int row = 1; row < n; row++) {

    for (int col = 1; col < row; col++) {

        char ans = 'A' + count;

        cout << ans;

        count++;

}

    cout << endl; }

(50)

A  
B C  
C D E  
D E F G  
E F G H I

for (int row = 0; row < n; row++) {

    for (int col = 0; col < row+1; col++) {

        char ans = 'A' + col + row;

        cout << ans;

}

    cout << endl;

}

51 D

C D

B C D

A B C D

char ans =

'A' + n - row;

for (int view = 0; view < n; view++) {

for (int col = 0; col < view; col++)

{ cout << ans << " ";

ans++;

}

cout << endl;

}

52

- - - \*

- - \* \*

- \* \* \*

\* \* \* \*

for (int view = 0; view < n; view++)

{ for (int col = 0; col < n - view - 1; col++)

cout << " ";

}

for (int col2 = 0; col2 < view + 1; col2++)

cout << "\*";

cout << endl;

?

53 \* \* \* \*

- \* \* \*

- \* \* \*

- - - \*

for (int view = 0; view < n; view++)

{ for (int col2 = 0; col2 < n - view; col2++)

cout << " ";

?

for (int col2 = 0; col2 < n - view; col2++)

cout << "\*";

?

cout << endl;

?

(54)

1 1 1 1  
2 2 2

3 3 four (int view = 0; view < n; view++) {  
4     for (int col = 0; col < n; col++) {  
5         cout << " ";

}

four (int col1 = 0; col1 < n - view; col1++) {  
cout << view + 3;

2  
cout << endl;

(55)

3 1

2 2 four (int view = 0; view < n; view++) {  
3 3 3     for (int col = 0; col < n - view - 1; col++) {  
4 4 4 4             cout << " ";

5  
cout << endl;

four (int col1 = 0; col1 < view + 1; col1++) {  
cout << view + 3;

6

cout << endl;

(56)

1 2 3 4

2 3 4 four (int view = 0; view < n; view++) {  
3 4     for (int col = 0; col < view; col++) {

4         cout << " ";

5     four (int col1 = 1; col1 < n - view; col1++) {

6         cout << col1 + view;

7

cout << endl;

8

57

1      int count = 0;  
2 3      for (int row=0; row < n; row++) {  
4 5 6      for (int col=0; col < n-row; col++) {  
7 8 9 10      cout << " ";  
              }  
              }  
              for (int col2=1; col2 < n-row+1; col2++) {  
              cout << count;  
              count++;  
              }  
              cout << endl;  
              }

58

1 2 3 4 5 \* 5 4 3 2 1  
1 2 3 4 \*\*\* \* 4 3 2 1  
1 2 3 \* \*\*\* \* 3 2 1  
1 2 \* \* \* \* \* 2 1  
1 \* \* \* \* \* \* \* 1

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

}

for (int col=0; col < 2\*row+3; col++) {  
cout << "\*";  
}

}

for (int col3=n-row; col3 >= 1; col3--) {  
cout << col3;  
}

}

cout << endl;

}

(5)

9 8 7

6 5 4

3 2 | int count = n \* n;

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

    for (int col = 0; col < n; col++) {

        cout << count << " "

        count = -3;

    }

cout << endl;

}

(6)

\* \* \*

\* \* \*

\* \* \*

\* \* \*

\* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \*

\* \* \*

\* \* \*

\* \* \*

\* \* \*

\* \* \*

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

    for (int col = 0; col < n; col++) {

        cout << "\* ";

    }

for (int col = 0; col < n; col++) {

    cout << " ";

}

```
for (int col=0; col<n; col++) {  
    cout << col << " ";
```

? cout << endl; ?

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

? ?

```
    for (int col=0; col<n; col++) {  
        cout << "* " ; ?
```

```
    for (int col=0; col<n; col++) {  
        cout << "*" ; ?
```

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

```
        for (int col=0; col<n; col++) {  
            cout << "*" ;
```

? ?

① 3 3 3 3 3 n=3

3 2 2 2 3

3 2 1 2 3

3 2 2 2 3

3 3 2 3 3

In vs code

code

15 ↵

Done

```
for (int col=0; col<n;  
     col++) {
```

cout << " ";

```
for (int col=0; col<n; col++) {  
    cout << "*" ;
```

? ?

cout << endl; ?

? ?