Day 5:- Pattern Solving

19/4/2022

⇒ Pattern solving → 25

① I/P → n = 4 → column.

O/P → 1st 2nd 3rd 4th

1st → * * * *
2nd → * * * *
3rd → * * * *
4th → * * * *

row

4

4

Print this square pattern

new line so,

cout << endl;

approach :- row = 4

1st row = 4 columns
2nd = "
3rd = "
4th = "

given data → row
→ column
→ n

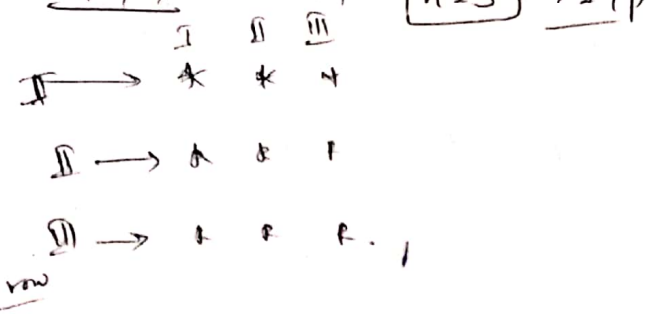
[4 rows hai] → each row → 4 stars.
has

so, col = row.

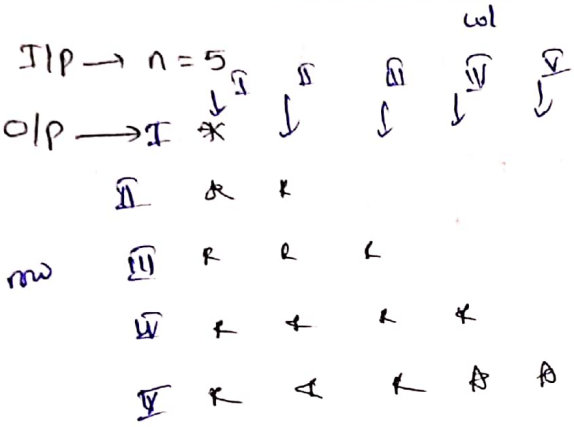
code:-

```
int main () {
    int n=4;
    for(int row=1; row<=n; row++) {
        // for each row, print n stars or we have n col
        for(int col=1; col<=n; col++) {
            cout << "*";
        }
        cout << endl; // for new line.
        // cout << '\n';
    }
    return 0;
}
```

Pattern 2:-



Pattern 2:- Half-pyramid



n=5

approach:-

for (row → 1 to n)

{

// for each row

stars count = row no

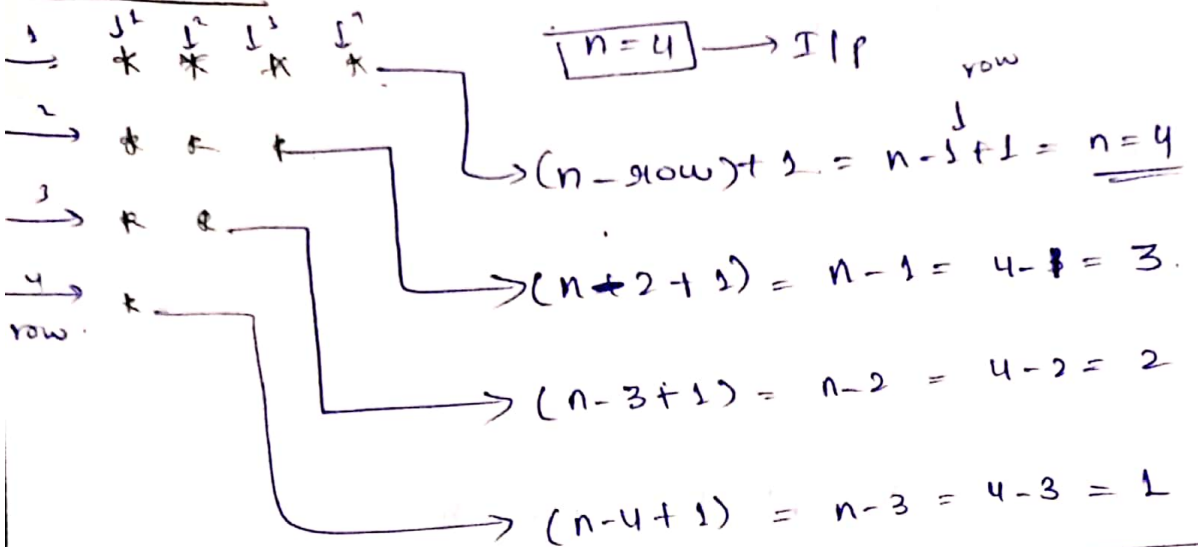
for (int col → row no)

}

code:-

```
int main() {
    int n=5;
    for(int row=1;row<=n; row++) {
        // for each row, print stars = row ka no.
        for(int col=1; col<=row; col++) {
            cout<<"*";
        }
        //after every row
        cout<<endl;
    }
}
```

Pattern 3:-



approach

1st row → 4 stars

2nd → 3 stars

3rd → 2 stars

4th → 1 star

NOTE: no of stars → depends on row, col, n

=> therefore, NO of stars to print is (n - row) + 1 .

Dry run:-

```
for(row 1 → n)
{
    // for each row
    for(col 1 to n - row + 1)
    {
        cout "*"
    }
    cout << endl
}
```

approach 2:-

1 → * * * * n [dependent on n]

2 → * * * n - 1

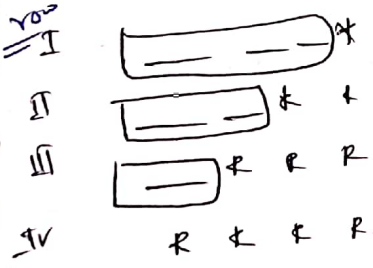
3 → * * n - 2

4 → * n - 3

code:-

```
int main() {
    int n=5;
    for (int row=1; row<=n; row++) {
        // for each row
        for(int col=1; col<= n-row+1; col++) {
            cout << "*";
        }
        // after each row
        cout << endl;
    }
    return 0;
}
```

Pattern 4:-



int n=4

(n-row) → space.

4-1=3.

4-2=2

4-3=1.

Approach

1st row → spaces → n-row

stars → row.

Dry run:-

```
for(int row → 1 to n)
{
    for(col 1 to n-row)
    {
        cout << " ";
    }
    for (col → 1 to row)
    {
        cout << "*"
    }
    cout << endl;
}
```

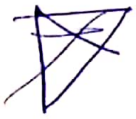
Code:- int main() {

```
int n=5;
for(int row=1; row <=n ; row++) {
    // for each row

    //spaces
    for (int col=1; col <=n-row; col++) {
        cout << " ";
    }
    // stars
    for (int col=1; col <=row; col++) {
        cout << "*";
    }
    //after each row
    cout << endl;
}
return 0;
}
```

row → 1 to n

col → 1 to row



row → 1 to n

col → n - row + 1

row → 1 to n

col → space → n - row

star → row

Pattern 5 :-

I → * * * *

II → — * * *

III → — — * *

IV → — — — *

row.

approach

spaces → 0 → (row − 1) stars →

1

2

3

n = 4 approach.

row	spaces	stars
I	0	4
II	1	3
III	2	2
IV	3	1

Spaces ke liye

I → 0

II → 1

III → 2

IV → 3

spaces = row − 1

stars ke liye

row 1 → 4

2 → 3

3 → 2

4 → 1

n = 4, row = 1 → [n − row + 1] = n = 1 + 1 = n = 4

n = 4, row = 2 → n = 2 + 1 = 3 (stars)

n = 4, row = 3 → 2

n = 4, row = 4 → 1

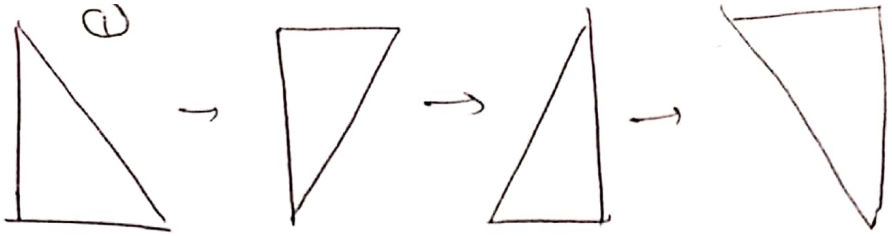
code

```
int main(){
    int n=5;
    for(int row=1; row<=n; row++){
        //for each row
        //spaces
        for(int col=1; col<=row-1; col++){
            cout<<" ";
        }
        //stars
        for(int col=1; col<=n-row+1; col++){
            cout<<"*";
        }
        //after each row
        cout<<endl;
    }
    return 0;
}
```

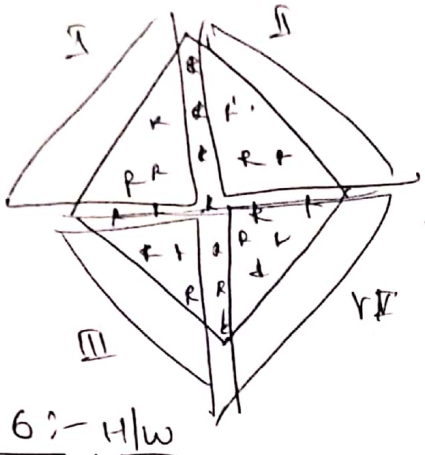
NOTE :- (=) → assignment

V = 23 → [23]

<= → compalision → (a<=b) → T/F



H/w:-



→ solid By Diamond

Pattern 6:- H/w

NOTE:- ① If, $n = 5$; $j = 2$ and we want to obtain 3

$n - j = 3$ → we obtained pattern

② $i = 2$; $j = 5$

to obtain 7, → $i + j =$

① H/w 1st row → 0 star → row - 1

row

2nd → 1 star

3rd → 2 star

4th → 3 starts.

④ row = 1, $n = 4$.

$n - row + 1 = n - 1 + 1 = n = 4$ → obtained

Pattern 7:- Full pyramid :-

Approach n=4

space		star n=4
1st (row) →	3	1
2nd →	2	2
3rd →	1	3
4th →	0	4

↓ n-row

↳ row.

n=4, row=1 → 1 star aana hai

n=4, row=2 → 2 star aana hai

n=4, row=3 → 3 " "

n=4, row=4 → 4 " "

[n=4
row=1
1st row=3]
spaces

Code:-

```
int main() {
    int n=5;
    for (int row=1; row<=n; row++) {
        // for each row
        //spaces
        for (int col=1; col<=n-row; col++) {
            cout<<" ";
        //stars
        for (int icol=1; col<=row; col++) {
            cout<<"*";
        }
        // after ro each row
        cout<<endl;
    }
```

space

"*"

↳ star ke baad

→ Pattern 8: 1 ☆ ☆ ☆ ☆

H/W:- II ☆ ☆ ☆

III ☆ ☆

IV ☆

n=4

approach

	space	star
row 1 →	0	4
2 →	1	3
3 →	2	2
4 →	3	1
	(row-1)	

(row=1, n=4)

4 banana hai
so,
(n-row+1)

Dry run:-

row → 1 to n

↳ space → 1 to row-1

stars → 1 to n-row+1

Pattern 9:- (Hallow inverted Half pyramid)

o/p:- ~~* * *~~ n=6

1 → * * * * * *

2 → * — — +

3 → * — — *

4 → * — *

5 → * *

↳ 6 → *

low

Dry run:-

for(row → 1 → n)

↓

if (row==1 || row==n)

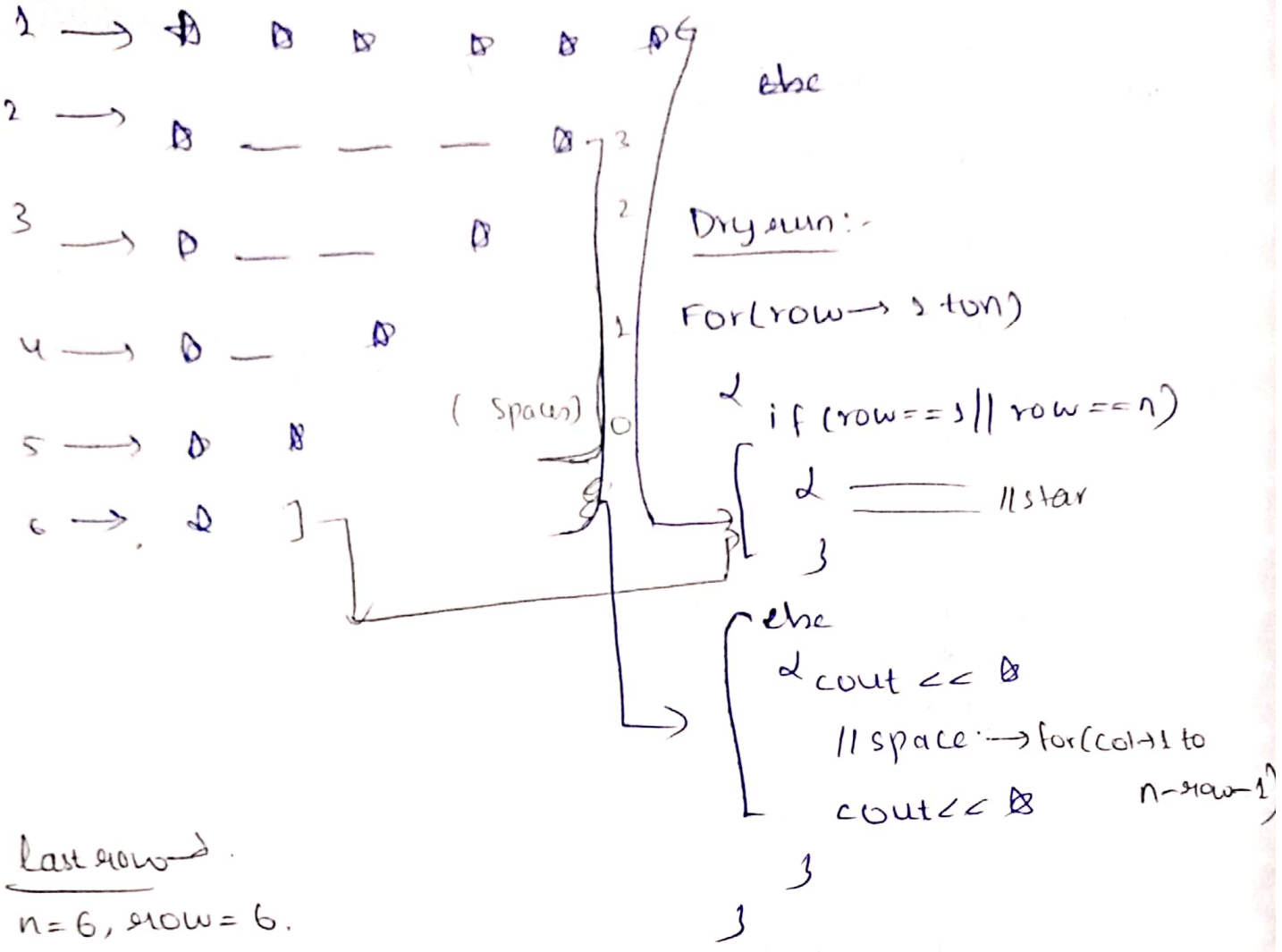
{ ——

}

else {

}

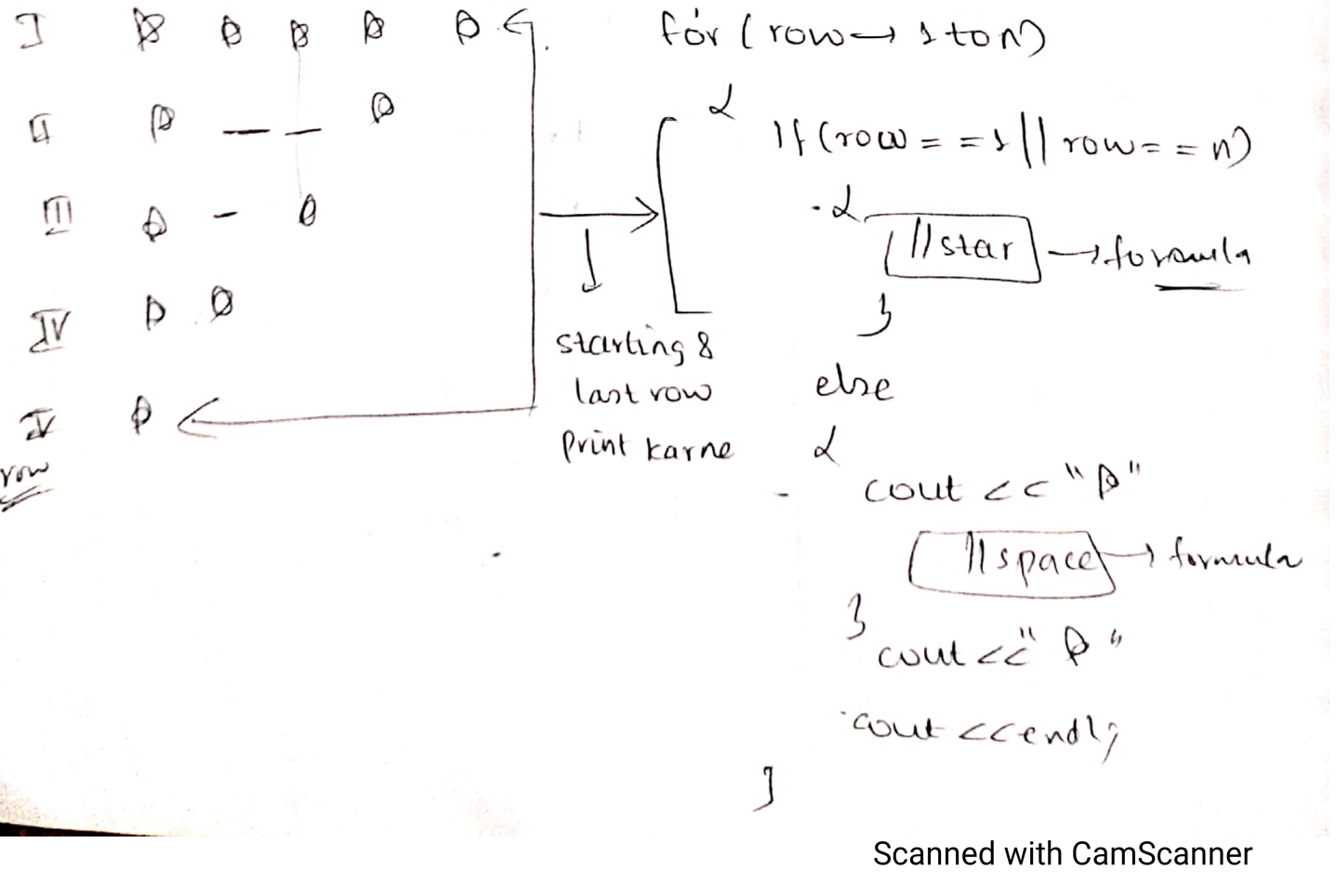
Scanned with CamScanner



Last row →

n = 6, row = 6.

So, `n-row+1` → to print stars in If block



row = 1
n = 5

5 banana hai

so, 5-1+1 = 5 → n-row+1 = stars to print in If block

row = 2
n = 5

2 banana hai

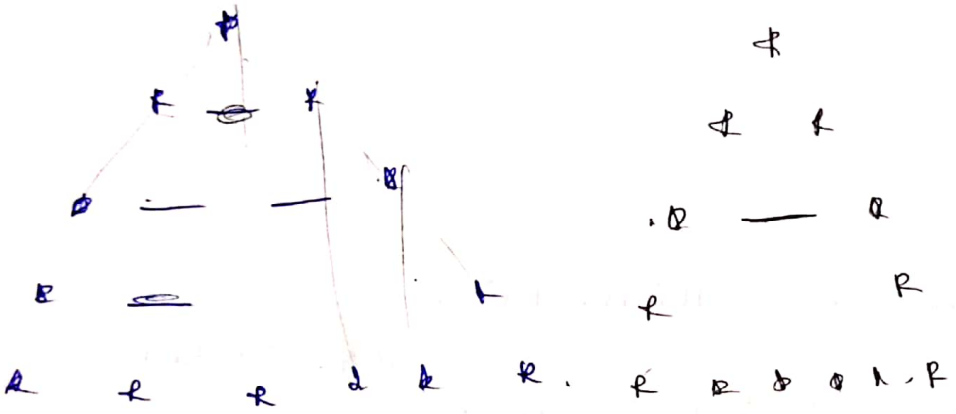
so, n-row-1 = 5-2-1 = 3 = spaces to print in [else block]

code:-

```
int main() {
    int n=6;
    for(int row=1; row<=n; row++) {
        if(row == 1 || row == n) {        // for 1st & last row.
            //stars
            for(int col=1; col<=n-row+1; col++) {
                cout << "*";
            }
        }
        else {
            cout << "*";
            // space
            for(int col=1; col<=n-row-1; col++) {
                cout << " ";
            }
        }
```

```
        cout << "*";
    }
    //after each row
    cout << endl;
}
row
return 0;
}
```

Pattern 10:- [full hallow pyramid]



Pattern 11:- Half pyramid:-

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

Pattern 12

Inverted Half Pyramid

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

Pattern 13:-

Hollow Half pyramid

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

Pattern 14

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
```

Pattern 15

Hollow full pyramid

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

Pattern 16

Hollow Inverted Half Pyramid

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

derive link

Pattern 11:- Half pyramid

```
int main(){
    int n=5;
    for(int row=1;row<=n;row++){
        // for each row, print stars = row ka no.
        for<<col<<" ";
    }
    // after each row
    cout<<endl;
}
```

o/p:-

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