Week-2

Patterns:-

1.

```
- - - *        ← row 0
- - *  *       ← row 1
- *  *  *      ← row 2
*  *  *  *     ← row 3
```

```
for (
{
        //space

}
for (
{
      //star


}
```

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

Q. Inverted Full Pyramid

```
row0 → *  *  *  *
row1 → _  *  *  *
row2 → _  _  *  *
row3 → _  _  _  *
```

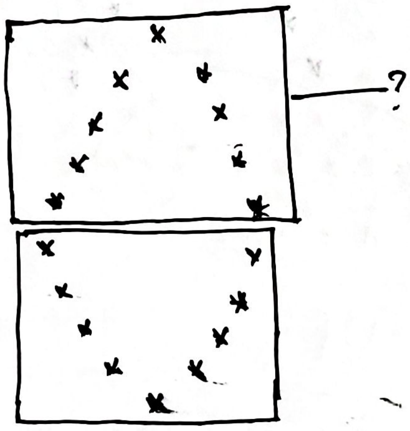
n=4
↓
outer loop

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

        //space

        //star

}
```

	spaces	stars	
row= 0	0	4	
row = 1	1	3	n=4
row=2	2	2	n-2
row = 3	3	1	n-3
	= row	n- row	n-1

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

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

}
```

Q. Hollow Diamond :-



row 0 → - - - - *
row 1 → - - - * _ *
row 2 → - * _ _ - *
row 3 → * _ _ _ _ _ *

n = 4

	space	n-[1+row]
row=0	3	
row=1	2	
row=2	1	
row=3	0	

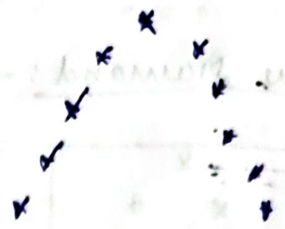
n-1-row

row → 0 1 ch → 1
row → 1 3 ch → 3
row → 2 5 ch → 5
row → 3 7 ch → 7

2row + 1 → character print krane hain

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

    // stars

    for (int col=0;  col < 2 * row + 1; col++) {
        // if first character of if last character
        if(col==0){
            // first character
            cout <<"*";
        }
        if(col == 2* row){
            // last character
            cout <<"*";
        }
else{
        cout<<" ";
    }
  }
  cout<<endl;
```

O/P



```
row0 * - - - - - *
row1  * - - - *
row2   * - *
row3    *
```

row→0	0 sp
row→1	1 sp
row→2	2 sp
row→3 →	3 sp

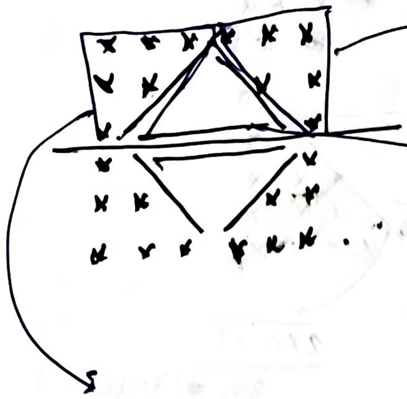
space = row

row 0 →	7 ch
row 1 →	5 ch
row 2 →	3 ch
row 3 →	1 ch

2n − 2row − 1

```
for (int row=0; row<n; row++){
        // spaces
        for (int col=0; col<row; col++){
                cout<<" ";
        }

        // star
        for (int col=0; col<2*n-2*row-1; col+=1){
                // if first or last character
                if (col==0 || 2*n-2*row-2){
                        cout<<"*";
                }
                else{
                        cout<<" ";
                }
        }
}
```

Flipped solid Diamond :-



~~row~~ same

```
      -        1
    - - -      3      space
  - - - - -    5
- - - - - - -  7
```

2row + 1

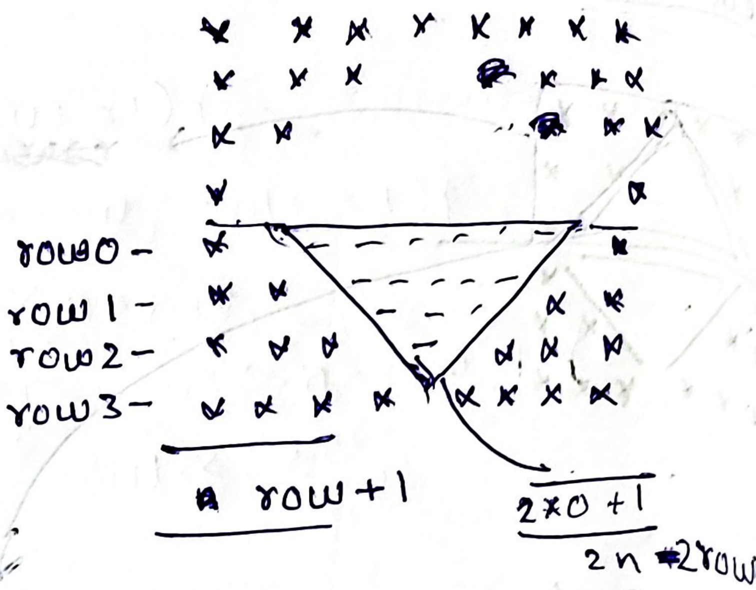
row0 → 4
row 1 → 3
row 2 → 2
row 3 → 3

n - row

For upper part :-

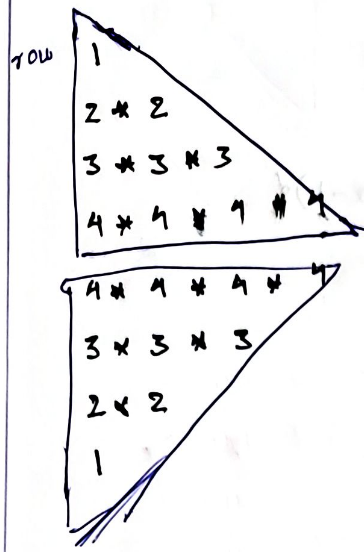
```
for (int row = 0; row < n; row++){
        // half pyramid 1
        for(int col = 0; col < n-row; col+ = 1){
                cout << " * ";
        }

        // spaces

        for (int col = 0; col < 2row + 1 ; col+ = 1){
                cout << "  ";
        }
        // half pyramid 2 (same code) of 1
        for (int col = 0; col < ~~2*row~~ n - row; col++){
                cout << " * ";
        }
        cout << endl;
}
```

For lower part



```
for (int row = 0; row < n; row += 1){
    // half pyramid
    for (int col = 0; col < row + 1; col = col + 1){
            cout << "*";
    }
    // space k liye
    for (int col = 0; col < 2*n-2*row - 1; col += 1){
            cout << " ";
    }
    // half pyramid
    for (int col = 0; col < row + 1; col += 1){
            cout << "*";
    }
    cout << endl;
}
```

Fancy pattern



```
        0  1  2  3
row0 →  *
   1    *  *
   2    *  *  *
   3    *  *  *  *
```

row 0	1
row 1	2
row 2	3
row 3	4

row + 1   row + 1

```
1
2 2
1 2 3
```

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

o/p

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

~~for (row = 0;~~ ;

```
for (int row = 0 ; row < n ; row + = 1 ) {
        for ( int col = 0 ; col < row + 1 ; col + = 1 ) {      (row + 1 → m)
            cout << row + 1 ;
            if ( col != row ) {          (row → last no. (m - 1))
                cout << " * " ;
            }
        }
        cout << endl ;
}
```

```
1
2 * 2
3 * 3 * 3
4 * 4 * 4 * 4
```

~~for rows~~

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

Counting ⟶ "m" numbers

1 start ⟶	1, 2, 3, 4 - - - - m
0 start ⟶	0, 1, 2, 3 - - m-1

$$\text{for (int col=0; col<}\underbrace{\text{n-row}}_{m}\text{; col++)}$$

$$\overset{m-1}{\text{n-row-1}}$$ ⟵ last number hai

Q.
```
1
1 2
1 2 3
1 2 3 4
```

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

Q.
```
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
```

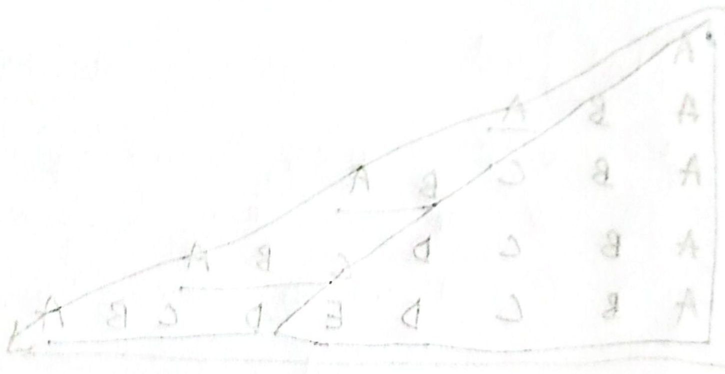
Jo bhi no. hai usee pehle tak aao.

Next page ✓

```
for (int row=0; row<n; row+=1){
        int col;
        for(col=0; col<row+1; col+=1){ q
                cout << col + 1;
        }
        // reverse counting
        for (int col=row; col>=1; col=col-1){
                cout << col;
        }
        cout << endl;
}
```

output

```
A
1
1 2 1
1 2 3 2 1
1 2 3 4 3 2 1
```



```
for (int row=0; row<n; row+=1){
        int col;
        for (col=0; col<row+1; col+=1){
                int ans = col+1;
                ~~cout << col +1.~~
                char ch = ans +'A'-1;
                cout << ch;
        }
        // reverse counting
        for (int col=row; col>=1; col=col-1){
                int ans = col;
                char ch = ans + 'A'-1;
                cout << ch;
        }
        cout<< endl;
```

O/P

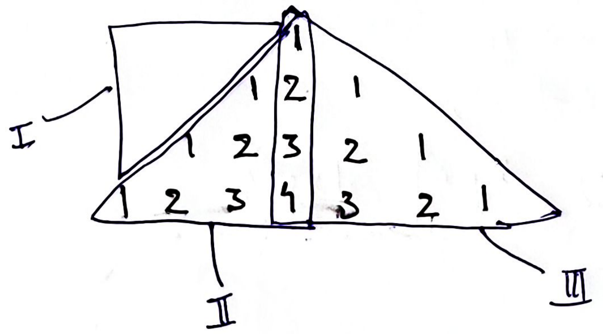
```
A
A B A
A B C B A
A B C D C B A
```

H.W

1) solid square
2) Hollow square
3) inverted half pyramid (Hollow)
4) Hollow full pyramid
5) Numeric hollow half pyramid + inverted

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

6)



7) Fancy.

8) Solid Half diamond

9) F. Pattern

10) F. Pattern

11) Floyd's Triangle

12) Pascals Triangle

13) Butterfly.