

1. Observation
2. Break the pattern into smaller pattern
3. Solve individuals pattern
  - a. Are we solved the pattern already
  - b. Think to relate with old any approach

1 → 1  
 2 3 → 2  
 4 5 6 → 3  
 7 8 9 10 → 4  
 11 12 13 14 15 → 5

1. Observation
2. Break it
  - a. Print 1 to 15
3. Find the relevant solution for each part
  - a. Take a variable name num and increase it

```

for (int i = 1; i <= 6; i++) {
    cout << i << " ";
}
  
```

1 2 3 4 5 6.

1

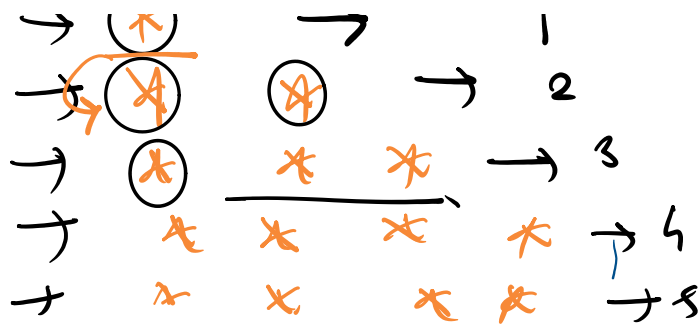
2 3

4 5 6

num = 1

→ (X)  
 → (X)  
 → (X)

row, col ⇒ X

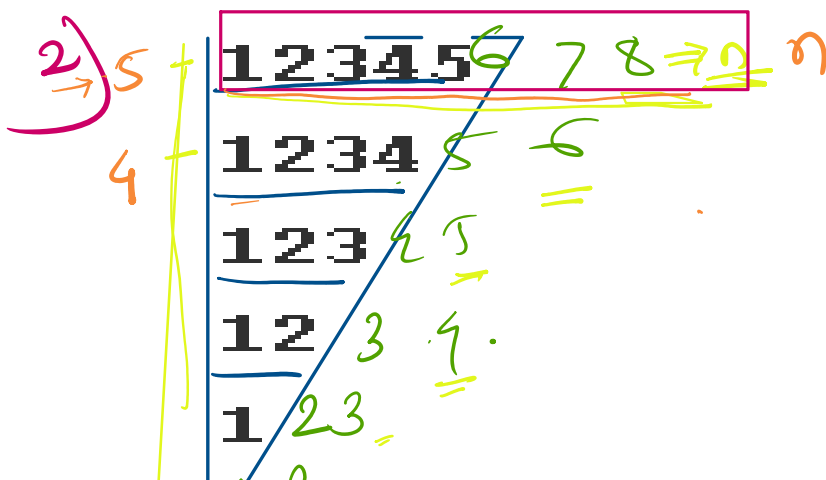


row, col  $\Rightarrow$  X

for(rep = 1; rep <= n; rep++) {

for (start = 1; start <= n; start++)  
 {  
     cout << \* << "  
 }  
 print (n);

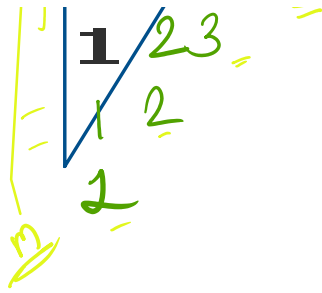
```
1 public class PrintNumberRect {
2     new *
3     public static void main(String[] args) {
4         int n = 5;
5         int num = 1;
6         for(int rep = 1; rep <= n; rep++) {
7             for(int start = 1; start <= rep; start++) {
8                 System.out.print(num + " ");
9                 num++;
10            }
11            System.out.println();
12        }
13    }
```



1. Observation

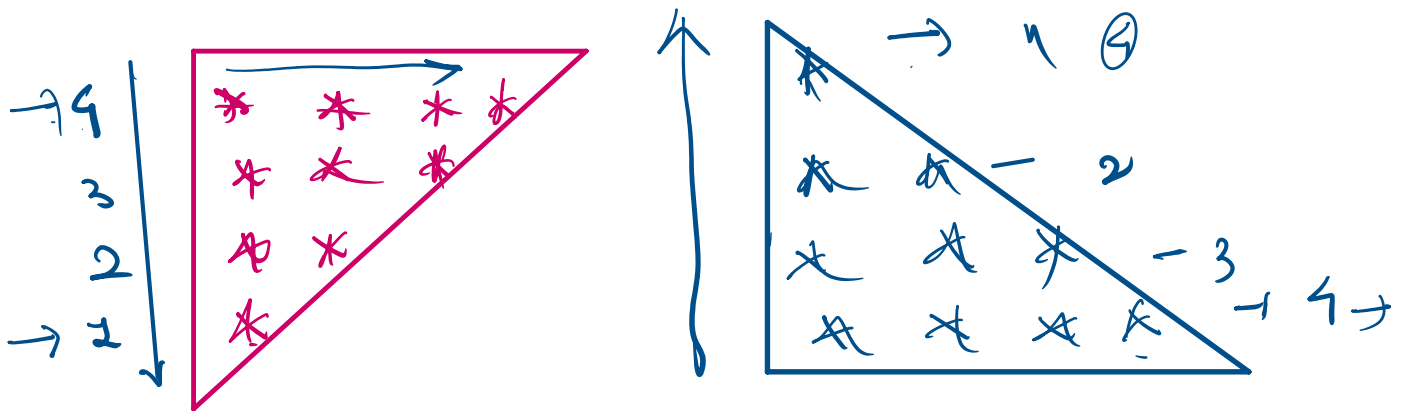
2. Break it

- In first it  $\rightarrow$  we have to print 1 to n
- Till n time we have reduce n by 1



1 — 6.

```
for (int num = 1; num <= 6; num++) {
    cout << num << "
"
```



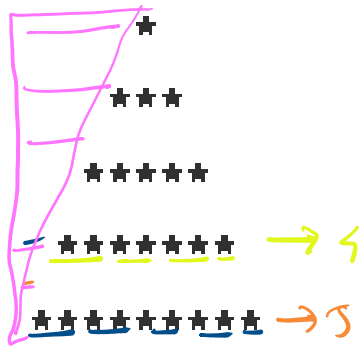
```
for (rep = n; rep >= 1; rep--) {
```

```
for (num = 1; num <= rep; num++)
```

```
    p(num)
}
```

3)

Maximum col  
Width

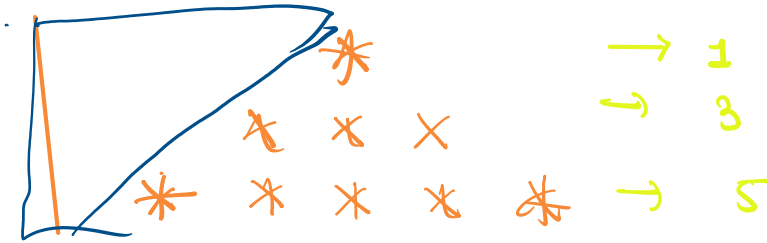


1. Observation

2. Break it

- We are printing the star  $2 * it - 1$  / (printing \* the in odd manner)
- Print the  $n - it$  space

It  $\rightarrow$  iterations



$$it = 1$$

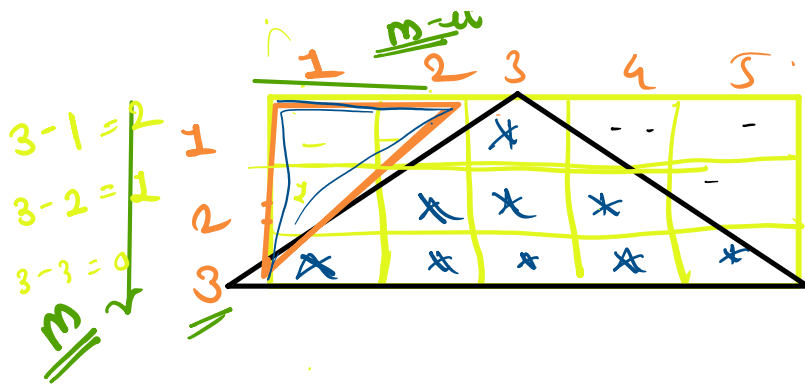
$$it = 2$$

$$it = 3 \Rightarrow \frac{2 * it - 1}{2 * 3 - 1} \\ 6 - 1 = 5$$

odd

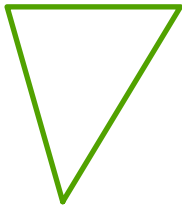
it {  
 {  
 for (star 1; star <=  $2 * it - 1$ ; star++)  
   cout << "\*";  
 }  
 }  
 $odd += 2$





$$\Rightarrow 1\ 2 = \text{space } 3*$$

$$\Rightarrow m-i = \text{space}$$



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

```

for (int space = 1; space <= m; space++)
{
    cout << " ";
}

```

1 to n odd.

```

{
    for (int num = 1; num <= n; num++)
    {
        if (num % 2 == 0)
        {
            // even
        }
        else
        {
            // odd
        }
    }
}

```

for (num = 1; num <= n; num += 2)

1 3 5 7 9 ...

1 3 5 7 9 - .

num = 1

while (num <= n) {  
    num += 2  
}

```
new ~
public class PrintPyramid {
    new *
    public static void main(String[] args) {
        int n = 5;
        int odd = 1;
        for(int rep = 1; rep <= n; rep++){ // -> odd <= n
            for(int space = 1; space <= n - rep; space++){ // problem b ->
                System.out.print(" ");
            }
            for(int star = 1; star <= odd; star++){ // star <= 2 * rep - 1; // Problem
                System.out.print("* ");
            }
            odd += 2;
            System.out.println();
        }
        // for(int i = 1 ; i <= 9; i+=2) System.out.print(i + " ");
    }
}
```

\*\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*

\*\*\*

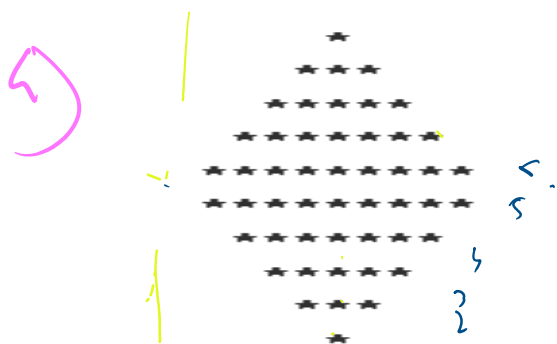
\*

Reverse pyramid

```

public class ReversetPyramid {
    new *
    public static void main(String[] args) {
        int n = 5;
        for(int rep = n; rep >= 1; rep--){
            for(int space = 1; space <= n - rep; space++){
                System.out.print(" ");
            }
            for(int star = 1; star <= 2 * rep - 1; star++){
                System.out.print("* ");
            }
            System.out.println();
        }
    }
}

```



Handwritten equations and notes:

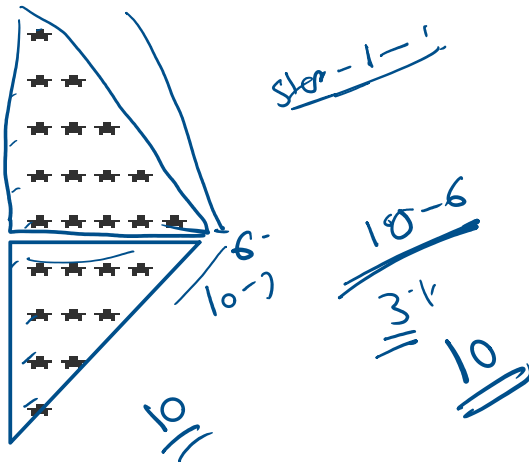
$n-1$    
 $\underline{\underline{6-5}}$    
 $n-1$    
 $\underline{\underline{n=8}}$    
 $n-6$

```

new *
1 public class DiamondPattern {
new *
2     public static void main(String[] args) {
3         int n = 5;
4         int odd = 1;
5         for(int rep = 1; rep <= n; rep++){ // -> odd <= n
6             for(int space = 1; space <= n - rep; space++){ // problem b ->
7                 System.out.print(" ");
8             }
9             for(int star = 1; star <= odd; star++){ // star <= 2 * rep - 1; // Problem
10                System.out.print("*");
11            }
12            odd += 2;
13            System.out.println();
14        }
15        n = 5;
16        for(int rep = n; rep >= 1; rep--){
17            for(int space = 1; space <= n - rep; space++){
18                System.out.print(" ");
19            }
20            for(int star = 1; star <= 2 * rep - 1; star++){
21                System.out.print("*");
22            }
23            System.out.println();
24        }
25    }
26 }

```

S



```

new *
1 public class halfDiamond {
new *
2     public static void main(String[] args) {
3         int n = 10;
4         for (int rep = 1; rep <= n; rep++) {
5             if (rep <= 5) {
6                 for (int star = 1; star <= rep; star++) {
7                     System.out.printf(" * "); // -> printing the start
8                 }
9             }
10            } else {
11                for (int star = 1; star <= n - rep; star++) { // n - rep + 1;
12                    System.out.print(" * ");
13                }
14            }
15            System.out.println();
16        }
17    }
18    // n = 4;
19    // for(int rep = n; rep >= 1; rep--){
20    //     for(int star = 1; star <= rep; star++){ // n - rep + 1;
21    //         System.out.print(" * ");
22    //     }
23    //     System.out.println();
24    // }
25 }
26

```