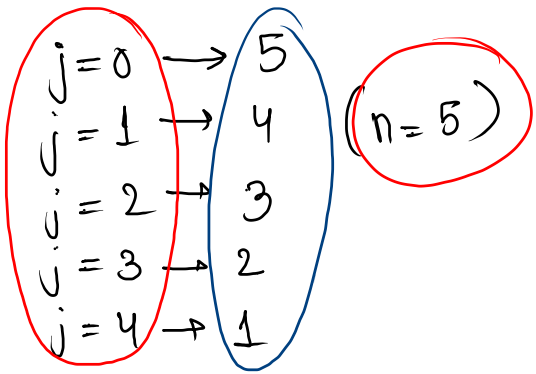
HW_Print Inverted Right Angled Triangle reversed whole numbers

n = 5

	0	1	2	3	4
0	5	4	3	2	1
1	5	4	3	2	
2	5	4	3		
3	5	4			
4	5				



template

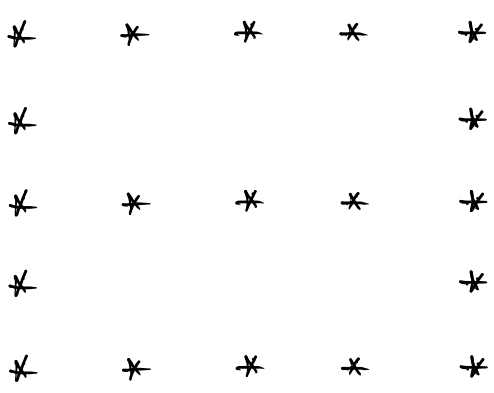
```
int st = n;
for(int i = 0; i < n; i++){
    for(int j = 0; j < st; j++){
        Syso((n-j)+" ");
    }
    st--;
    Sysoln();
}
```

n = 5, j = 0
n = 5, j = 1
n = 5, j = 2

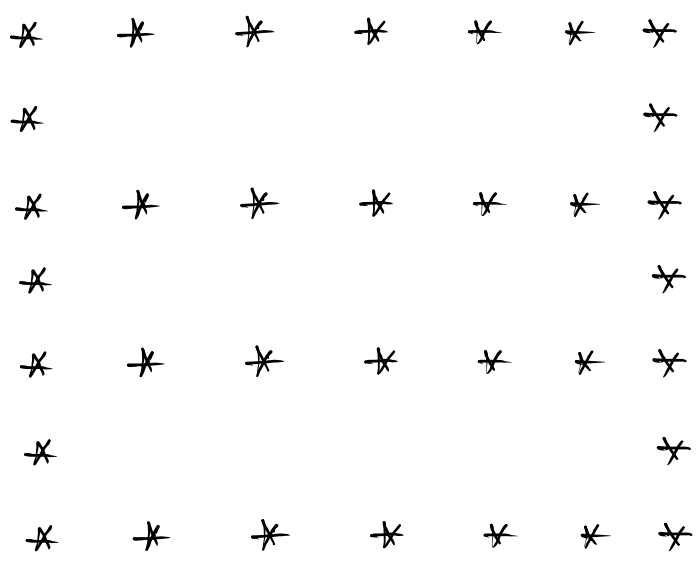
ans = ? = 5
ans = 4
ans = 3

Pattern 9 - Square Ladder with top and bottom

n = 5



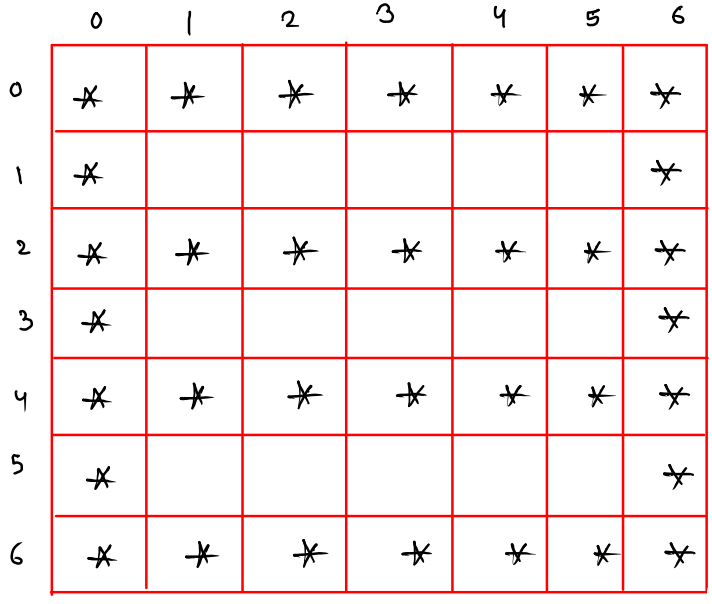
n = 7



n = 4 ✗

Note:- Input will always be odd

$n = 7$



check cond^n for
beginning & ending column

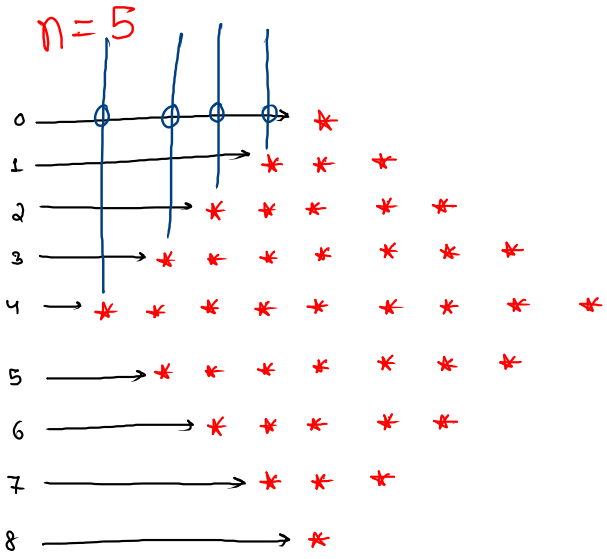
$j == 0 \quad || \quad j == n-1$

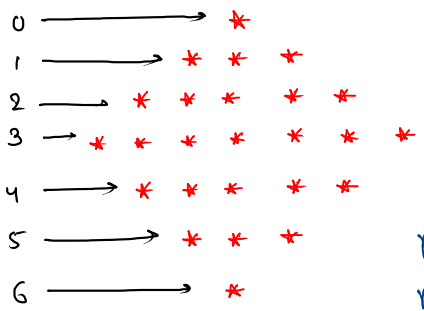
code

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            if ( i % 2 == 0 ) {
                System.out.print("*\t");
            } else {
                if ( j == 0 || j == n - 1 ) {
                    System.out.print("*\t");
                } else {
                    System.out.print("\t");
                }
            }
        }
        System.out.println();
    }
}
```

GKSTR29_Pattern_12_Diamond





rows = 2 * n - 1

n = 5 ⟶ rows = 9
n = 4 ⟶ rows = 7
n = 3 ⟶ rows = 5
n = 2 ⟶ rows = 3
n = 1 ⟶ rows = 1

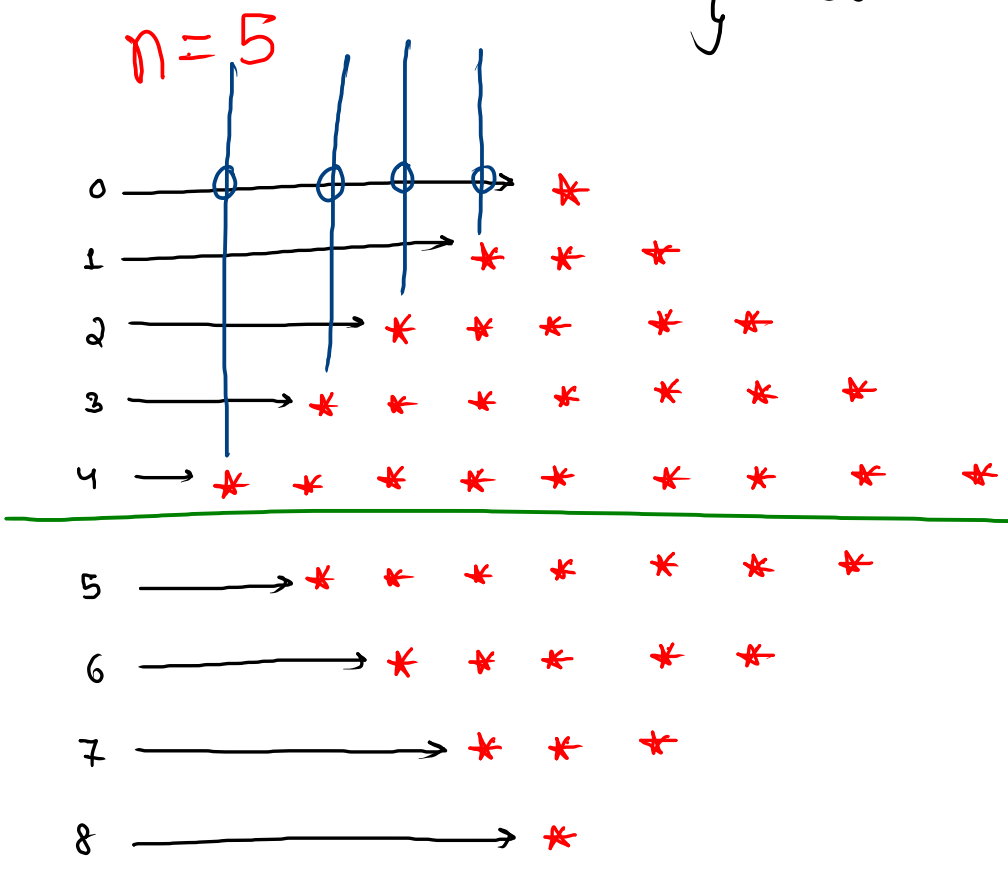
template

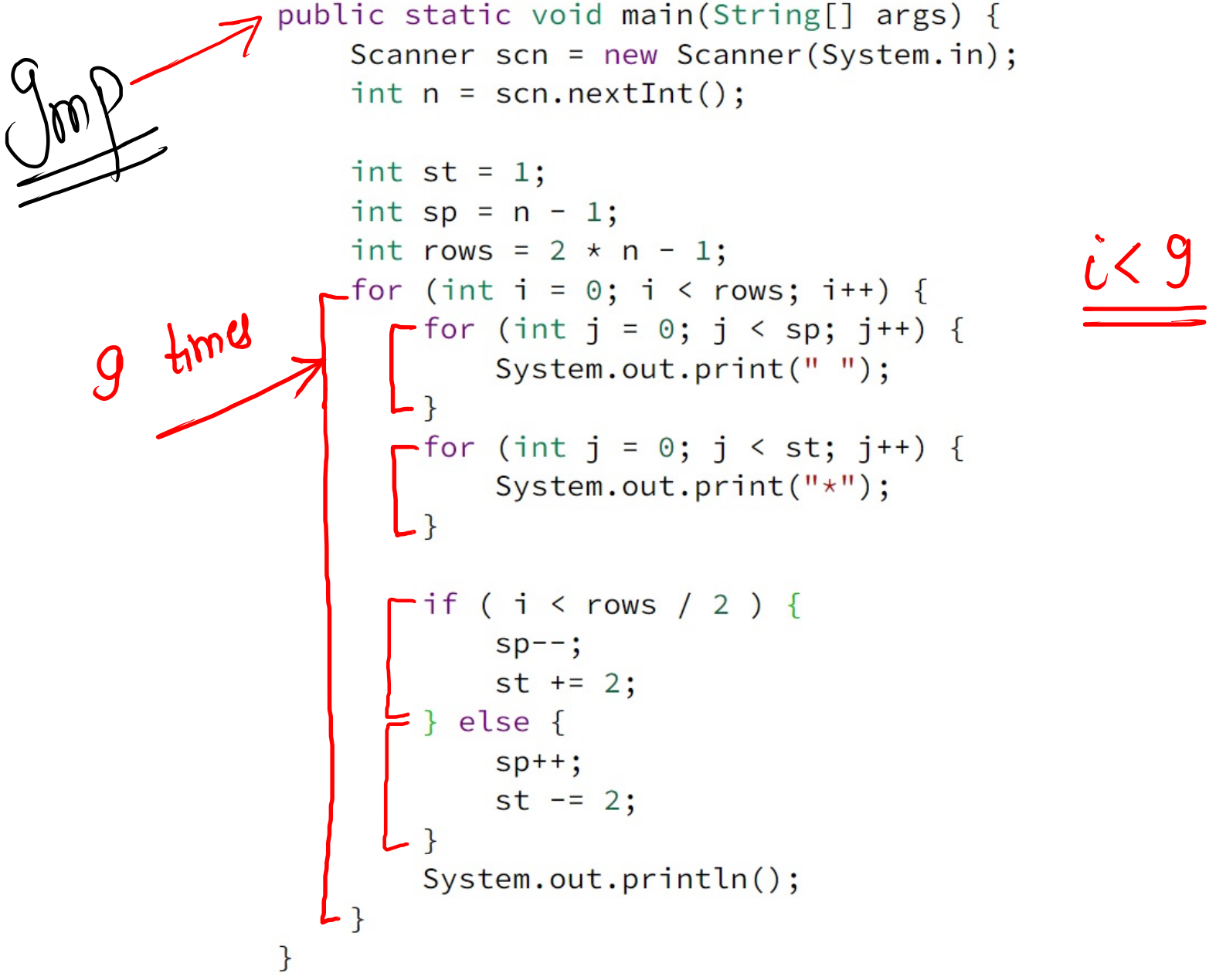
```
int st = 1;
int sp = n-1;
int rows = 2*n-1;
for(int i=0; i<rows; i++){
    for(int j=0; j<sp; j++){
        Syso(" ");
    }
    for(int j=0; j<st; j++){
        Syso("* ");
    }
    if(first half){
        sp--;
        st+=2
    }else{
        sp++;
        st-=2;
    }
    Sysoln();
}
```

i < rows/2

```
if ( first half) {
    sp--;
    st += 2;
} else {
    sp++;
    st -= 2;
}
```

rows






```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();

    int st = 1;
    int sp = n - 1;
    int rows = 2 * n - 1;
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < sp; j++) {
            System.out.print(" ");
        }
        for (int j = 0; j < st; j++) {
            System.out.print("*");
        }

        if ( i < rows / 2 ) {
            sp--;
            st += 2;
        } else {
            sp++;
            st -= 2;
        }
        System.out.println();
    }
}
```

⇒ Functions

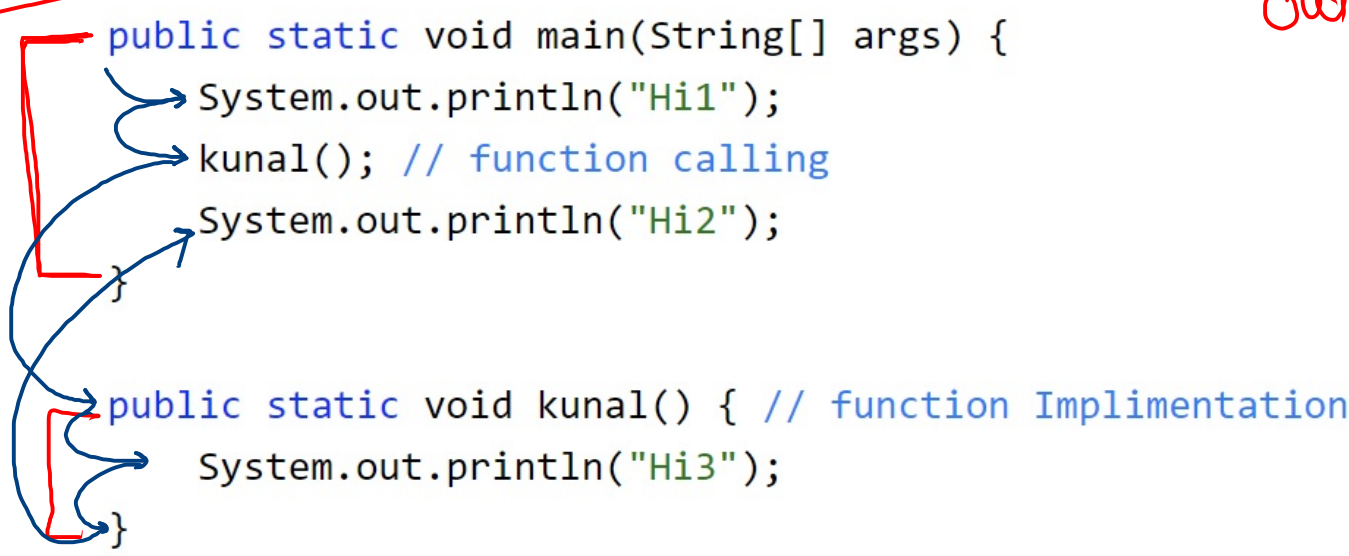
↳ saperate piece of code which can be re-used as many times as we want

advantage :- Rewsability & Readability

1) Function declaration / calling :- where we want to use that piece of code

2) Function Implimentation :- actual piece of code which will be called

functions



output

Hi1
Hi3
Hi2

Note:- main function will always be the first function to be called

Code

```
3   public static void main(String[] args) {
4     1) System.out.println("Hi1");
5     2) kunal(); // function calling
6     7) System.out.println("Hi2");
7     8) fun2();
8   }
9   public static void kunal() { // function Implimentation
10    3) System.out.println("Hi3");
11    4) fun1();
12  }
13  public static void fun2() {
14    9) System.out.println("Hi12");
15    10) System.out.println("Hi13");
16  }
17  public static void fun1() {
18    5) System.out.println("Hi5");
19    6) System.out.println("Hi7");
20  }
```

output

```
Hi1
Hi3
Hi5
Hi7
Hi2
Hi12
Hi13
```

Find sum using a function

```
int x = 5;
int y = 6;

// function calling
findSum(x, y);

// function implimentation
public static void findSum(int x, int y){
    Syso(x+y);
}
```

parameters

Note :-

There are 2 types of functions

1) paramaterised :- which accepts some parameter

Ex :- fun2 (2 , 'a' , true , 3) ;

2) non-paramaterised :- which doesn't

Ex :- fun1 ( ) ;

code

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int T = scn.nextInt();
    for (int i = 0; i < T; i++) {
        int x = scn.nextInt();
        int y = scn.nextInt();
        findSum(x, y);
    }
}
// main logic
public static void findSum(int x, int y) {
    int ans = x + y;
    System.out.println(ans);
}
```

these variables are not same