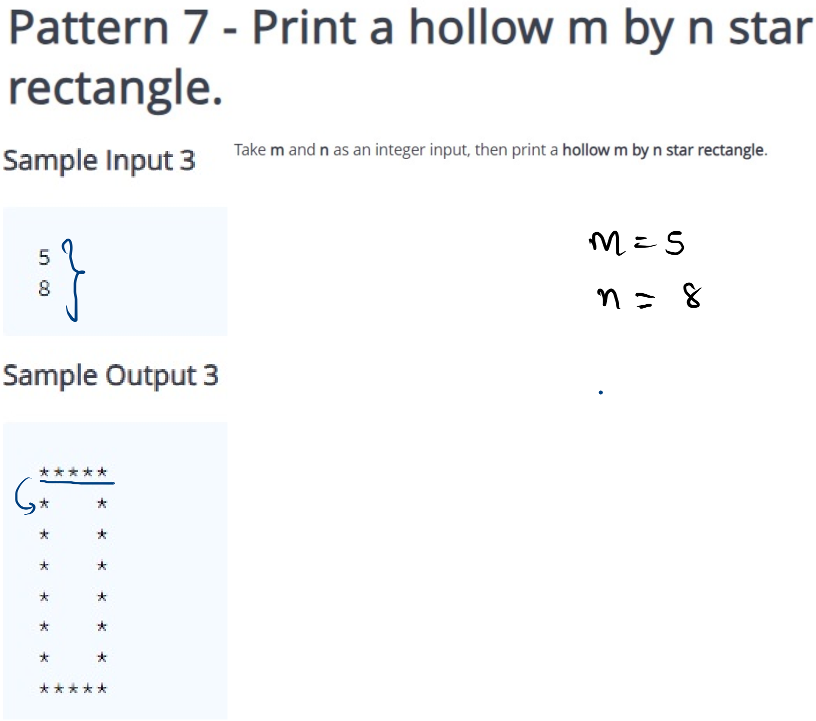


Pattern 7 - Print a hollow m by n star rectangle.

Take **m** and **n** as an integer input, then print a **hollow m by n star rectangle**.

Sample Input 3

```
5
8
```

Sample Output 3

```
*****
*   *
*   *
*   *
*   *
*   *
*   *
*****
```

m = 5

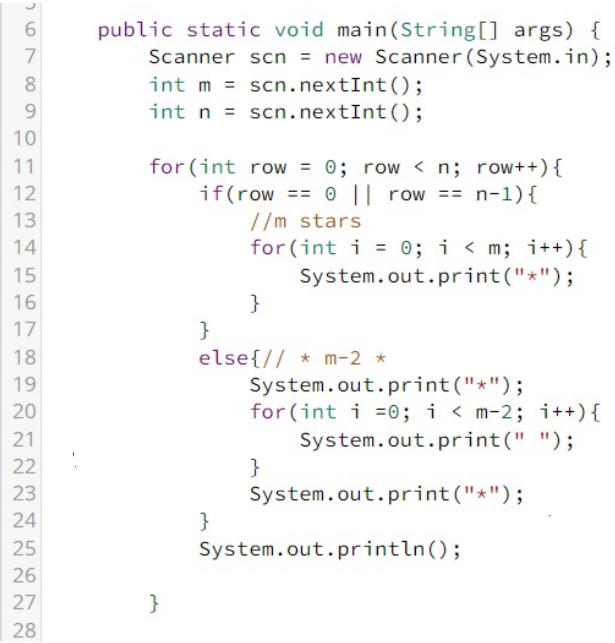
n = 8

1. rows → n

0th
n-1 → m stars.

space

rest ★ m-2 ★



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

        for(int row = 0; row < n; row++){
            if(row == 0 || row == n-1){
                //m stars
                for(int i = 0; i < m; i++){
                    System.out.print("*");
                }
            }
            else{// * m-2 *
                System.out.print("*");
                for(int i =0; i < m-2; i++){
                    System.out.print(" ");
                }
                System.out.print("*");
            }
            System.out.println();

        }

```

m = 5

n = 4

row = ~~0~~

1 ✓

2 ✓

3

4

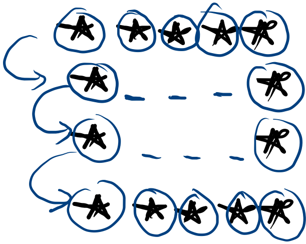
1 < 4 ✓

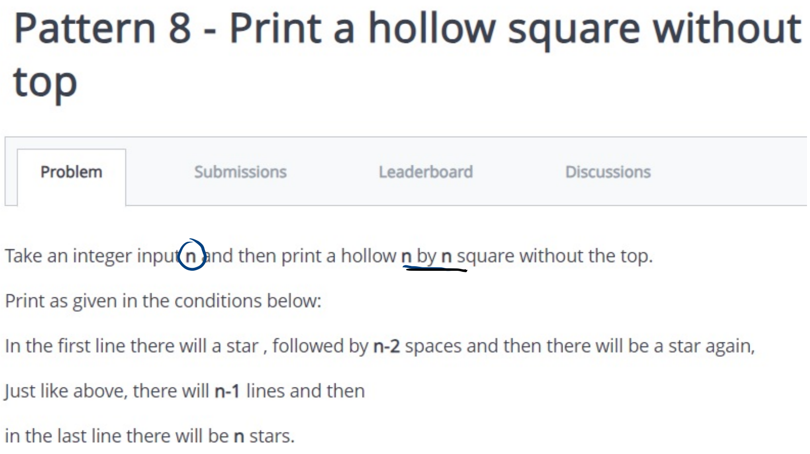
2 < 4 ✓

3 < 4 ✓

3 == 0 || 3 == 4-1 ✓

4 < 4 ✗





Pattern 8 - Print a hollow square without top

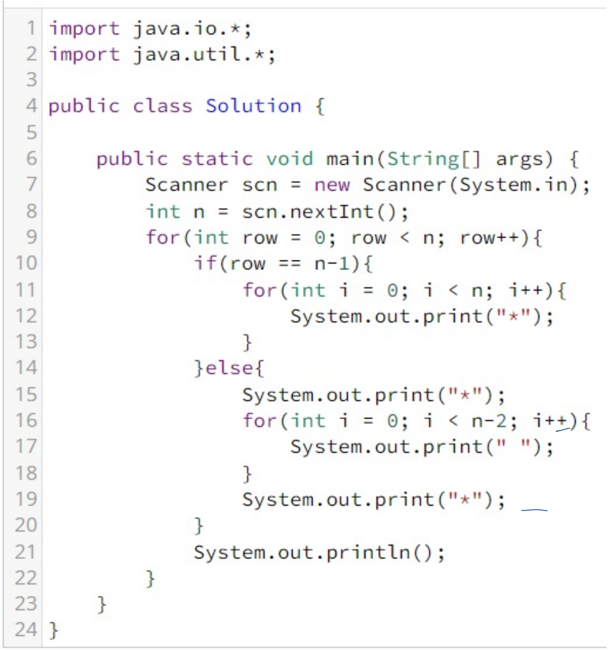
Take an integer input n and then print a hollow n by n square without the top.

Print as given in the conditions below:

In the first line there will a star , followed by **n-2** spaces and then there will be a star again,

Just like above, there will **n-1** lines and then

in the last line there will be **n** stars.

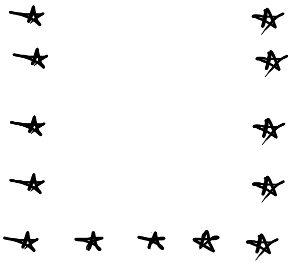


```java
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        for(int row = 0; row < n; row++){
            if(row == n-1){
                for(int i = 0; i < n; i++){
                    System.out.print("*");
                }
            }else{
                System.out.print("*");
                for(int i = 0; i < n-2; i++){
                    System.out.print(" ");
                }
                System.out.print("*");
            }
            System.out.println();
        }
    }
}
```

n = 5

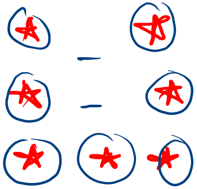


1. rows → n

2. last → n stars
   rest → ★ n-2 ★

row = 0   0<3 ✓
1   1<3
2   2<3
3   3<3 ✗

n=3.



Pattern 9 - Square Ladder with top and bottom

Take **n** as an integer input, then

print **n** tab separated stars in the first line,

then in the second line print a star, then **n-2** tabs, then print a star.

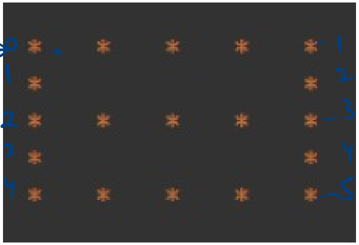
then print **n** tab separated stars in the third line.

then in the **fourth** line print a star, then **n-2** tabs, then print a star .

.

n=5

rows → n

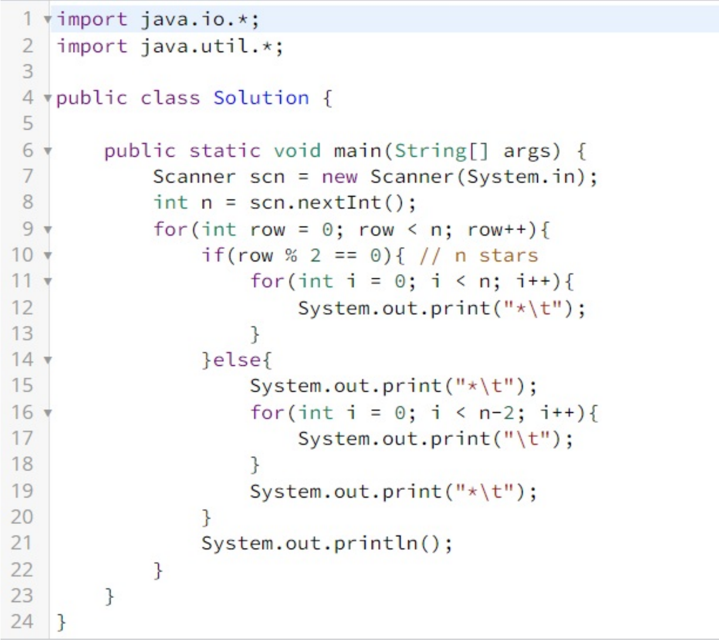


```
*    *    *    *    *
*                   *
*    *    *    *    *
*                   *
*    *    *    *    *
```

even → n stars.

odd → * _ _ _ *

```
* * * * *
*       *
* * * * *
*       *
* * * * *
```



```java
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        for(int row = 0; row < n; row++){
            if(row % 2 == 0){ // n stars
                for(int i = 0; i < n; i++){
                    System.out.print("*\t");
                }
            }else{
                System.out.print("*\t");
                for(int i = 0; i < n-2; i++){
                    System.out.print("\t");
                }
                System.out.print("*\t");
            }
            System.out.println();
        }
    }
}
```

n=5

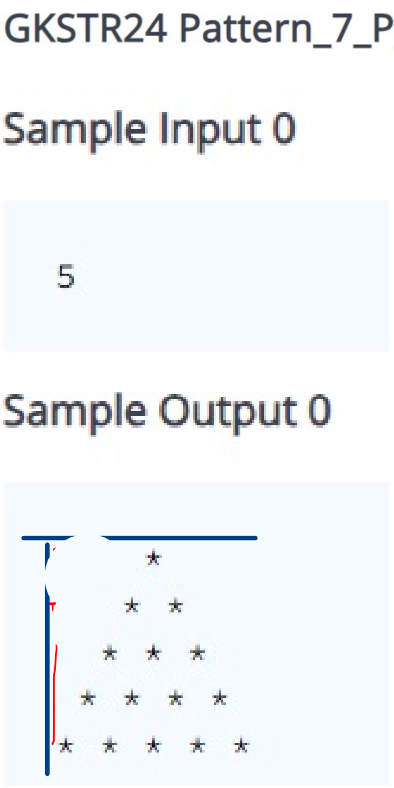
0 — * * * * *

1 — * _ _ _ *

2 — * * * * *

3 — * _ _ _ *

4 — * * * * *



GKSTR24 Pattern_7_Pyramid

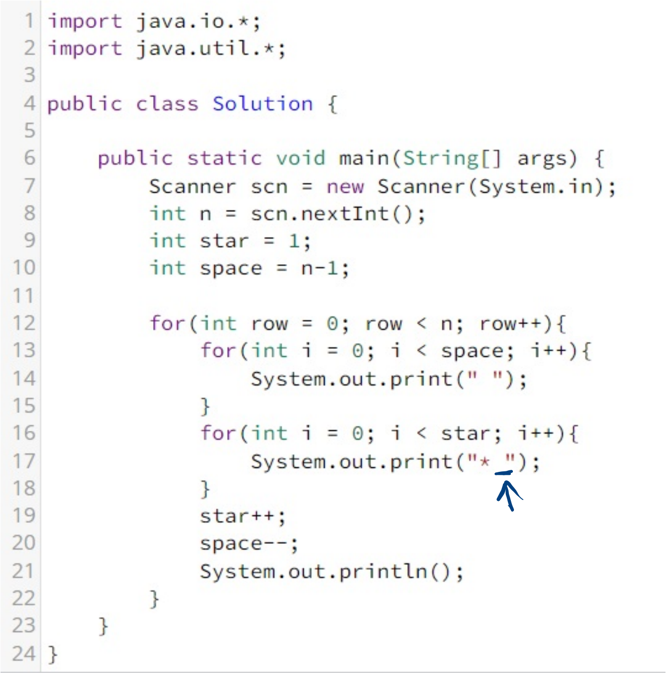
Sample Input 0

```
5
```

Sample Output 0

```
    *
   * *
  * * *
 * * * *
* * * * *
```

n = 5



```java
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int star = 1;
        int space = n-1;

        for(int row = 0; row < n; row++){
            for(int i = 0; i < space; i++){
                System.out.print(" ");
            }
            for(int i = 0; i < star; i++){
                System.out.print("* ");
            }
            star++;
            space--;
            System.out.println();
        }
    }
}
```

rows → n

Star = 1

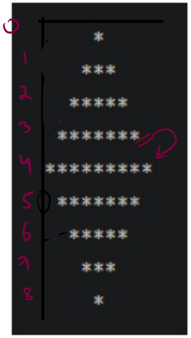
Space = n-1

Star++

Space--

Diamond.

n=5



```
    *
   ***
  *****
 *******
*********
 *******
  *****
   ***
    *
```

1. rows → ? $\boxed{2n-1}$

1st

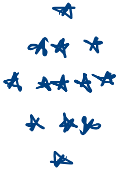
stars = 1 → star +=2

space = n-1 → space - 1

star -2

Space+1

n=3



row = 0

row = 1

row = 2

row 3 → 4 → +2 star, -1 space

row 4 → 5 → -2 star, +1 space

rows 5
6
7
8

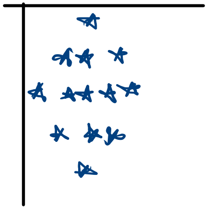
+2 star: 0, 1, 2

-1 space: 3 < 4

if (row < n-1)

else

n=5



star = 1, 3, 5, 3, 1

space = 2, 1, 0, 1, 2

n=3

rows 2n-1

n=3

```
0  - - *
1  - * * *
2  * * * * *
3  - * * *
4  - - *
```

Star +2
space -1] A

row=0 → 1 (A)

row=1 → 2 (A)

→ row=2 → 3 (B)

row=3 → 4 (B)

n=5 → 2n-1 =9

Star -2
space +1] B

```
0  - - - - *
1  - - - * * *
2  - - * * * * *
3  - * * * * * * *
4  * * * * * * * * *
5  - * * * * * * *
6  - - * * * * *
7  - - - * * *
8  - - - - *
```

row 0 → 1 (A)

row=1 → 2 (A)

2 → 3 (A)

3 → 4 (A)

4 → 5 (B)

5 → 6 (B)

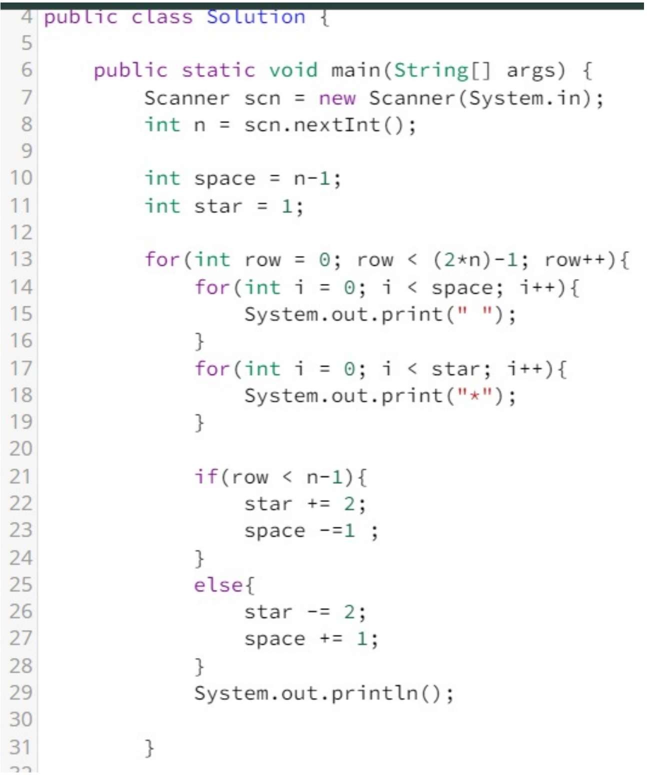
6 → 7 (B)

7 → 8 (B)

n=5

```
if (row < n-1)
    {
        A
    }
else {
        B
    }
```

row < 4



```java
public class Solution {

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

        int space = n-1;
        int star = 1;

        for(int row = 0; row < (2*n)-1; row++){
            for(int i = 0; i < space; i++){
                System.out.print(" ");
            }
            for(int i = 0; i < star; i++){
                System.out.print("*");
            }

            if(row < n-1){
                star += 2;
                space -=1 ;
            }
            else{
                star -= 2;
                space += 1;
            }
            System.out.println();

        }
```

n=3

2n-1

row=5

sp = ~~2~~ ~~1~~ ~~0~~ 1

st = ~~1~~ ~~3~~ ~~5~~ 3

✓

row = 1 < 2

0 → _ _ *

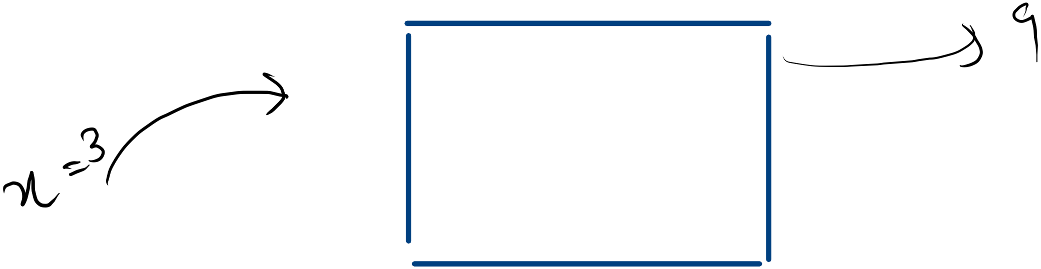
1 → _ * * *

2 → * * * * *

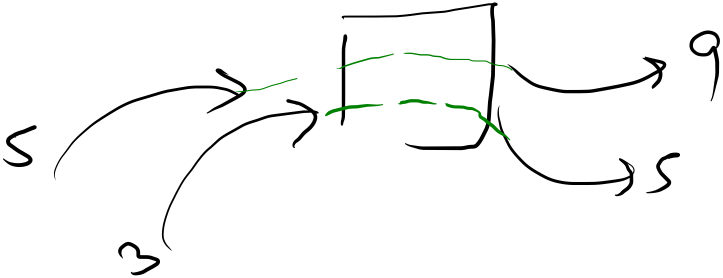
3 → _ * * *

4 →

fuction $\{ \quad f(x) = x^2$



function $\{ \quad f(x) = 2x - 1$



fuctions / Methods in Java.

↳ piece of code / block of code which perform specefic task.

$f(x) = x^2$

2 type of function

- inbuilt
  - Math.max()
  - Sysl print()
  - main()
  - nextInt()
- user-defined
  - we can create our func^n

Why? fuctions.

Reuse code.