```
for (int i=1; i<=10; i=i+2) {
        sop (i);
```

3

1 3 5 7 9

i ↷ [1,10]

i
1
3
5
7
9
11

breaks out from loop.

Loop to print * N times ?

no spaces

N = 3 output * * *

N = 5 output * * * * *

N = 7 output * * * * * * *

	iteration	
for (int i=1; i<=N; i++)	i=[1,N]	N
for (int i=1 ; i<N ; i++)	i= [1,N-1]	N-1
for (int i=0; i<=N; i++)	i= [0,N]	N+1

(1, 5) → 2, 3, 4

(→ exclusive [1, 5) → 1, 2, 3, 4

[→ inclusive (1, 5] → 2, 3, 4, 5

[1, 5] → 1, 2, 3, 4, 5

	range	count of no
[a, b] →	a to b	b − a + 1

① Given N, print a square of N* N stars.

N = 3

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

N = 4

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

N = 6

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

```
for (int i=1; i<=N; i++) {
    for (int j=1; j<=N; j++) {
            Sop("*");
    }
    sopln();
}
```

N = 3

desired output = * * *
* * *
* * *

i	j
1	1
	2
	3
	4 (break)
2	1
	2
	3
	4 (break)
3	1
	2
	3
	4 (break)
4	

```
* * * ↴
* * * ↴
* * * ↴
```

② . Given N, print the following pattern.

N = 3

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

N = 4

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

N = 5

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

```
for (int  i = 1; i <= N; i++) {
   for (int  j=1;  j <=(N-i+1); j++) {
            SOP("*");
   }
   SOPln();
}
```

N = 5

```
i
1 | *  *  *  *  *
2 | *  *  *  *
3 | *  *  *
4 | *  *
5 | *
```

i	stars	
1	N	N-1+1
2	N-1	N-2+1
3	N-2	N-3+1
4	N-3	N-4+1
⋮	⋮	
N	1	

ith row = N-i+1 stars

Q. 3 Given N, print the following pattern.

N = 3

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

N = 4

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

N = 5

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

```
for (int i=1; i<=N; i++) {
    for (int j=1; j<=i; j++) {
        sop("*");
    }
    SOPln();
}
```

i	stars (i)
1	1
2	2
3	3
⋮	⋮
N	N

Patterns with spaces:

①. Given N, print the following pattern.

N = 3

```
* _ _ *
* _ _ *
* _ _ *
```

N = 4

```
* _ _ _ *
* _ _ _ *
* _ _ _ *
* _ _ _ *
```

N = 5

```
* _ _ _ _ *
* _ _ _ _ *
* _ _ _ _ *
* _ _ _ _ *
* _ _ _ _ *
```

_ → space

Each row = "*" + $\underbrace{\text{spaces}}_{n-1}$ + "*"

```
for (int i=1; i<=n; i++) {
     SOP ("*");
     for(int j=1; j<= n-1; j++) {
          SOP ("  ");
     }
     SOP("*");
     SOPln();
}
```

(2). Given N, print the following pattern.

N = 3

```
* _ _ *
* _ *
* *
```

N=4

```
* _ _ _ *
* _ _ *
* _ *
* *
```

N = 5

```
* _ _ _ _ *
* _ _ _ *
* _ _ *
* _ *
* *
```

each row = "*" + some spaces + "*"

i	spaces (n-i)
1	N-1
2	N-2
3	N-3
⋮	⋮
N	0

N = 5

```
* _ _ _ _ *
* _ _ _ *
* _ _ *
* _ *
*
```

```
for (int i=1; i<=N; i++) {
        sop("*");
        for (int j=1; j<=n-i; j++) {
            sop(" ");
        }
        sop("*");
        sopln();
}
```

GCD / HCF

↳ greatest common divisor

gcd (a, b) → c

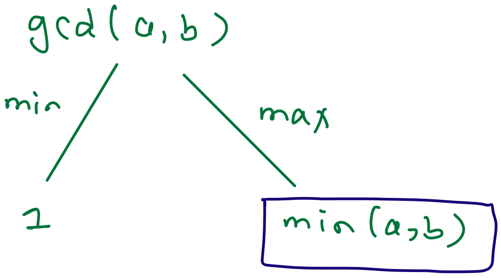
c is largest no. which can divide both a and b.

gcd (8,12) → 8

gcd (25, 50) → 25

gcd (36, 48) → 12

gcd (25, 49) → 1



```
gcd = 1;
for(int i=1; i<=min; i++){
        if(a%i==0 && b%i==0){
                gcd = i;
        }
}
```

```
min = 0;
if (a<b){
    min = a;
}
else {
    min = b;
}
```

a = 8   b = 12

min (8,12) = 8

i = [1, 8]

i	gcd
1	1
2	2
3	2
4	4
5	4
6	4
7	4
8	4

Instead of travelling from [1, min] we will travel from [min, 1].

i > 0

i >= 1

```
gcd = 1;
for ( i = min ; i >= 1; i--) {

    if ( a % i == 0 && b % i == 0) {
            gcd = i;
            break;
    }

}
```

a = 8 , b = 12

min (8,12) = 8

i = [8,1]

i	gcd
8	1
7	1
6	1
5	1
4	4

```
gcd = 1;
for ( i = min ; i >= 1; i--) {

    if ( a % i == 0 && b % i == 0) {
            gcd = i;
            break;
    }

}
```

a = 12

b = 18

min (12, 18) = 12

i = [12, 1]

i	gcd
12	1
11	1
10	1
9	1
8	1
7	1
6	6

a = 36

b = 48

gcd = 12

lcm = 144

$$\boxed{LCM = \frac{a * b}{GCD}}$$

$$lcm = \frac{36 * \overset{4}{\cancel{48}}}{\cancel{12}} = 144$$