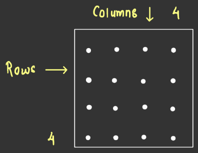
Pattern printing

Square pattern



pattern has two loop ⟶ Nested loop

Outer loop always works for rows or no of lines

Inner loop always works for columns and connect them with rows

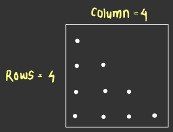
print anything that you want should be inside the inner loop.

observe symmetry ⟶ Optional not mandatory for all pattern

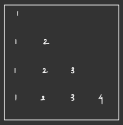
Code for above pattern

```
Rows = 4 ,  Cols = 4

     for i in range(4)

           for j in range(4):

                 print(*)

     print( )
```

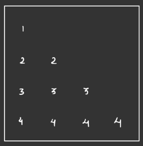
Right angle triangle pattern



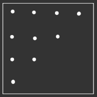
```
for i in range(1, n+1):

          for j in range(i)

                 print(*)

     print()
```



```
for i in range(1, n+1):

      for j in range(1, i+1)

             print(j)

      print()
```



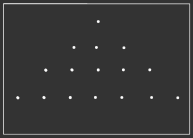
```
for i in range(1,n+1):

     for j in range(1,i+1):

           print(i, end=' ')

     print()
```



```
for i in range(n,-1,-1):
        for j in range(i):
                print(x)
        print()
```

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

```
for i in range(n,0,-1):
    for j in range(1,i+1):
          print(*)
    print()
```



```
| 3 , 1 , 3 |
| 2 , 3 , 2 |
| 1 , 5 , 1 |
| 0 , 7 , 0 |
n-i-1        n-i-1
```

Row 0-3   Col 0-6

4   7

Pattern ⟶ space print space

Row - 4

4-0-1 = 3
4-1-1 = 2
4-2-1 = 1
4-3-1 = 0

Suppose row ⟶ 4

```
for i in range(1,n+1):
      print(' ' * (n-i) + '*' * (2*i-1)
```

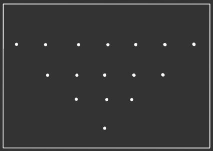
n = 4

Space * 3 + '*'

Space * 2 + '* * *'

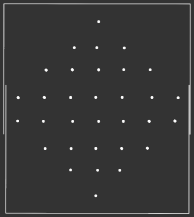
Space * 1 + '* * * * *'

space * 0 + '* * * * * * *'



Rows ⟶ 4

```
for i in range(4,0,-1):
      print(' ' * (n-i) + '*' * (2*i-1))
```

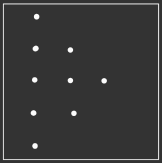


pattern combination above two pattern

pattern name ⟶ diamond pattern

Rotated triangle pattern

if r→3 mean's we are talking about number of columns.



```
for i in range(1,n+1)
      print('*'* i)
for i in range(n-1,0,-1)
      print('*'* i)
```

r→3

1 2 3

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

2 1 0

```
* *
*
```

```
1
0 1
1 0 1
0 1 0 1
1 0 1 0 1
```

All the even rows start with 1

```
r = 5

for i in range (r):
    for j in range(i+1):
        print (j % 2 , end='')
    print()
```

```
0
0 % 2 = 0
1
0 1 % 2 = 0 1
2
0 1 2 % 2 = 0 1 0
```

But above approach not giving desired result

we want $\longrightarrow$

```
1
0 1
1 0 1
```

```
for i in range(n):
    if i%2==0:
        start =1
    else:
        start = 0
    for j in range(i+1):
        print((start +j)%2, end='')
    print()
```

But we are getting result

```
0
0 1
1 0 1
```

$\longrightarrow$ Not correct

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

```
for i in range(1,n+1)
    for j in range(1,i+1):
        print()
    for _ in range (2*(n-i)):
        print()
    for j in range (i,0,-1):
        print()
```

$\longrightarrow$ 6 4 2 0 reverse table

r = 4 $\longrightarrow$ No of rows

```
1
2 3
4 5 6
7 8 9 10
```

```
Sum=1
for i in range(1,n+1):
    for j in range(1,i+1):
        print(sum,end='')
        sum +=1
    print()
```

```
i=  1  2  3  4
j-  1
    1  2
    1  2  3
    1  2  3  4
```

at each iteration the value of j

```
1
2 3
4 5 6
7 8 9 10
```

output

n = 3

```
A

A B

A B C
```

We know the concept just need to print alphabets instead of numbers like 1 2 3

$\text{chr}(65) \longrightarrow A$

```
for i in range(1,n+1):
    for j in range(65, 65+i):
        print(char(j))
print()
```

```
for i in range(n,0,-1)
```

Reverse printing

```
A

B   B

C   C   C
```

```
for i in range(1,n+1):
    for _ in range(i):        ⟶ # columns
        print(chr(64+i), end='')
    print()
```

n = 3

```
        A

    A   B   A

A   B   C   B   A
```

if we observe the symmetry

$\frac{5}{2}+1 = 3$

$\underline{A\ \ B}\ \boxed{C}\ \underline{B\ \ A}$

for spaces ⟶ (n-i)

for char ⟶ (2* (i-1))

Printing Space

```
for i in range(n):
    for j in range(n-i-1):
        print('', end='')
```

Printing pattern

```
ch = 'A'
breakpoint = (2*i+1)/2

    for j in range(1, 2*i+1):
        print(ch, end='')
        if j<= breakpoint:
            ch = chr(ord(ch)+1)
        else:
            ch = chr(ord(ch)-1)
    print()
```

N = 5

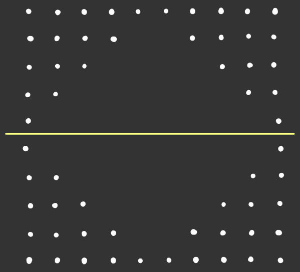
In alphabets at 5th position what comes ?

A B C D $\boxed{E}$

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

```
for i → 0,n
    for j → E, A          → psuedo code
        print (character)
```

j → range(ord(A)+ n-1, ord(A) + n-1-i, -1)



Stars	space	stars
5	0	5
4	2	4
3	4	3
2	6	2
1	8	1
n-i	0, +2	

To print this reverse

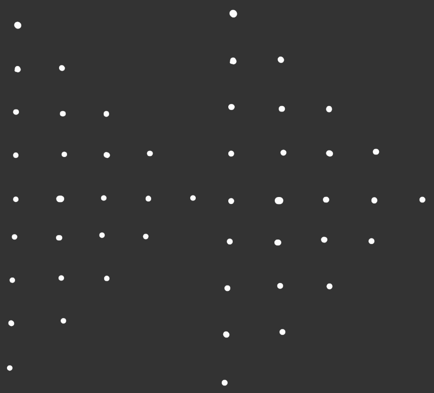
2 * n - 2   2 × 5 - 2 = 8

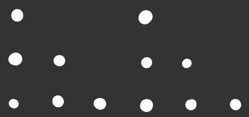
8 - 2   6

6 - 2   4

4 - 2   2

N = 5





stars	spaces	stars
1	4	1
2	2	2
3	0	3

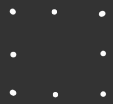
```
for i ⟶ (1,n+1)
    star = '*' * i
    spaces = ' '* formula
    print ( star , space, star )
```

formula = (2* n-2 * i)

To print lower part of pattern we can reverse loop

$(n-1, 0, -1) \longrightarrow$ for loop

N=3



```
for i in range(n):
    if i==0 or i==n-1:
        print(*)
    else:
        print('*' + ' ' * (n-2) + '*')
```

space $\longrightarrow$ n-2 * ' '

at 2nd row and 2nd col at index 1 print space

1	2	3	4	5	6	7
4	4	4	4	4	4	4
4	3	3	3	3	3	4
4	3	2	2	2	3	4
4	3	2	1	2	3	4
4	3	2	2	2	3	4
4	3	3	3	3	3	4
4	4	4	4	4	4	4

$\longrightarrow$ subtract 4 from each value

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 2 & 2 & 2 & 1 & 0 \\ 0 & 1 & 2 & 3 & 2 & 1 & 0 \\ 0 & 1 & 2 & 2 & 2 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{matrix} \longrightarrow$$

we got this after subtracting 4 from each value

current matrix $- n \longrightarrow$ To get new matrix

$n -$ new matrix $\longrightarrow$ To get current matrix

ncols, nrows $\longrightarrow 2 * n - 1$

$= 2 * 4 - 1 = 7$

Code

```
for i in range(2*n-1):          Rows
        for j in range(2*n-1):  Columns
```

Top = i

left = j

right = (2n-1)-1-j

bottom = (2n-1)-1-i