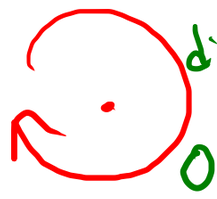
h = 4

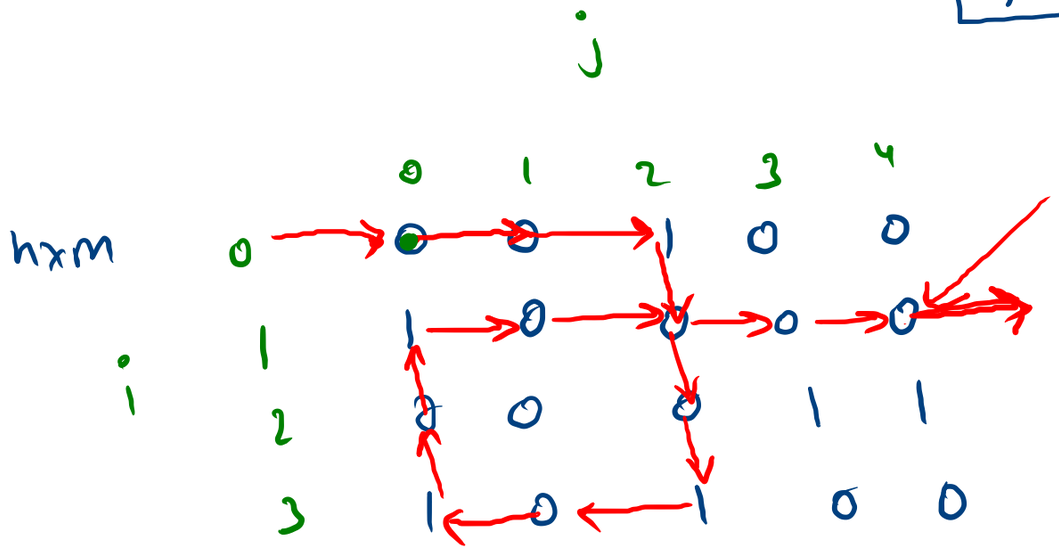
m = 5

0/1



j++

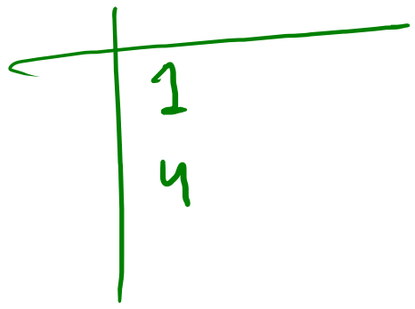
i++



0→ move

1→ turn

move



if (1) turn

move

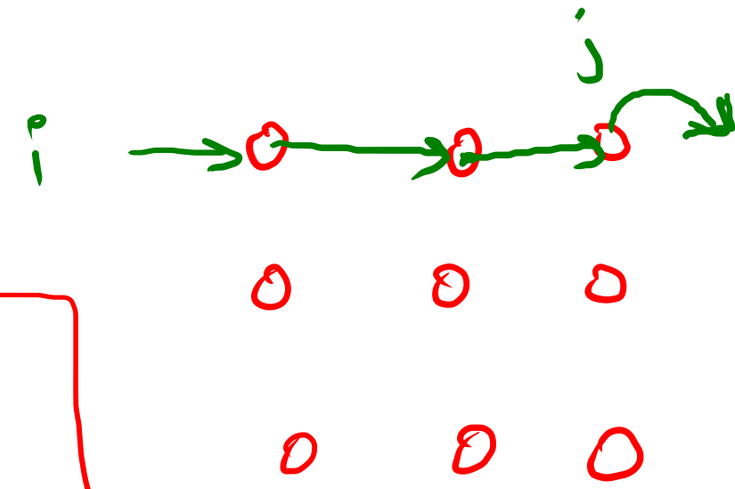
j--

2→ west

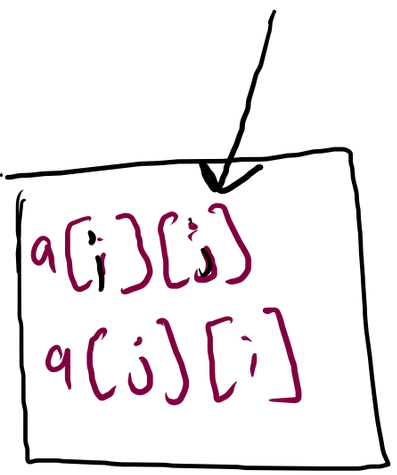
i--

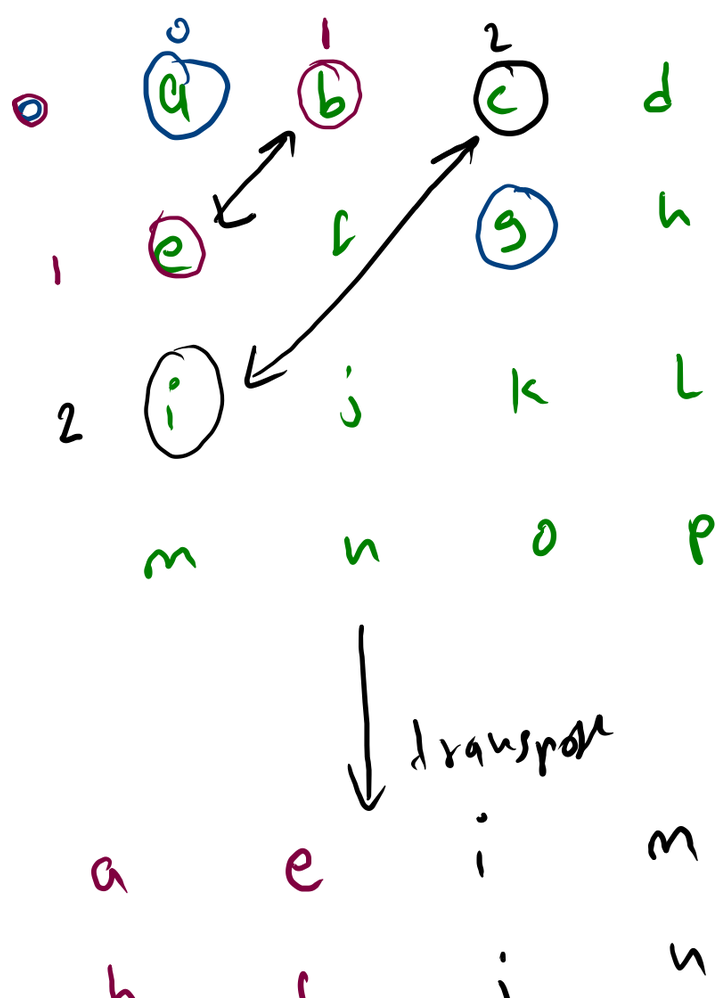
3→ north

```
i = 0
j = 0
dir = 0;
while (                ) {
    if ( arr[i][j] == 1 ) {
        change dir by 90°/o // turn
    }
    //move
    if ( east )
    else if ( west
        south
        north
}
```



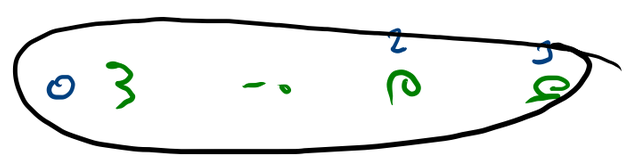
n = 4





transpose

a e i m
b f j n



s t r q

2 o k g r

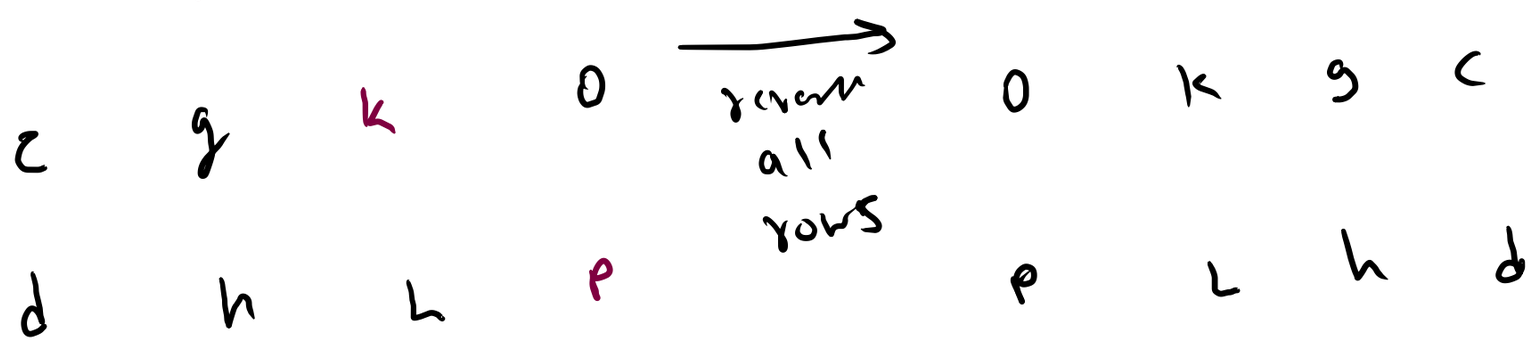
p l s p

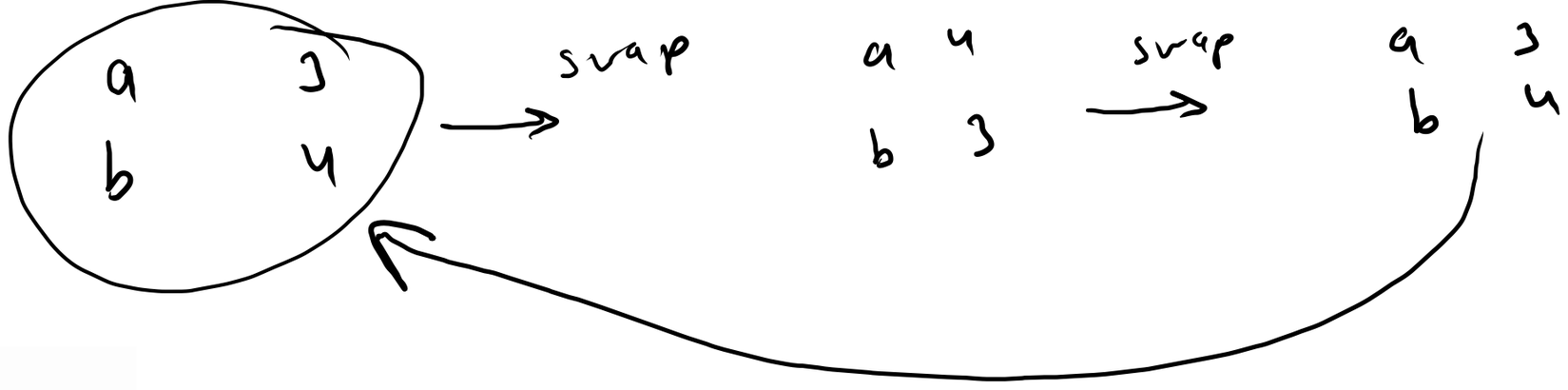
transpose

//check display

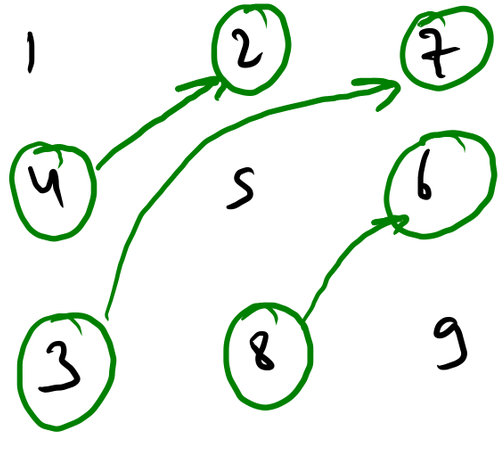
reverse display

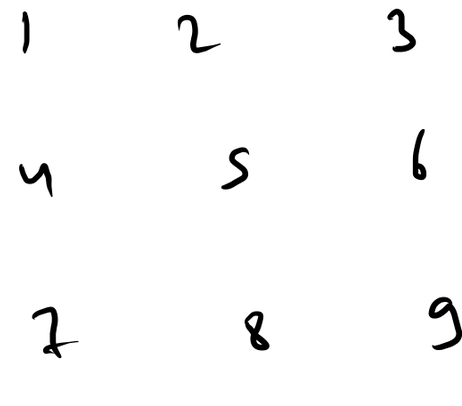
m i e a
h j f b

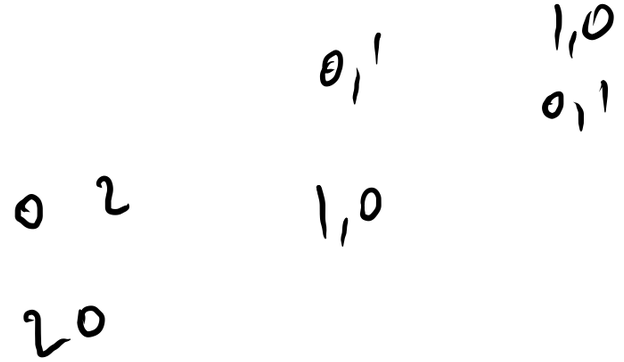


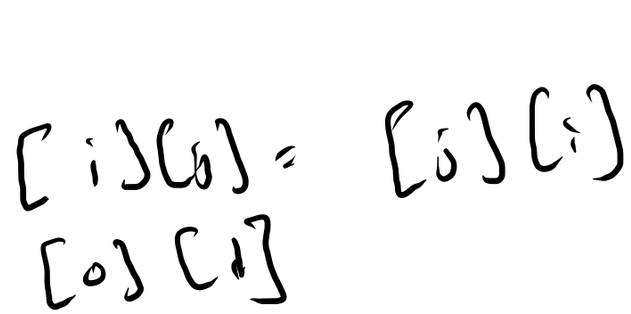


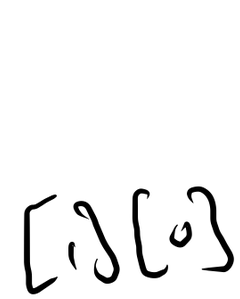

```java
for(int i=0;i<n;i++){
    for(int j=0;j<n;j++){
        // swap [i][j] with [j][i]
        int tmp = arr[i][j];
        arr[i][j] = arr[j][i];
        arr[j][i] = tmp;
    }
}
```





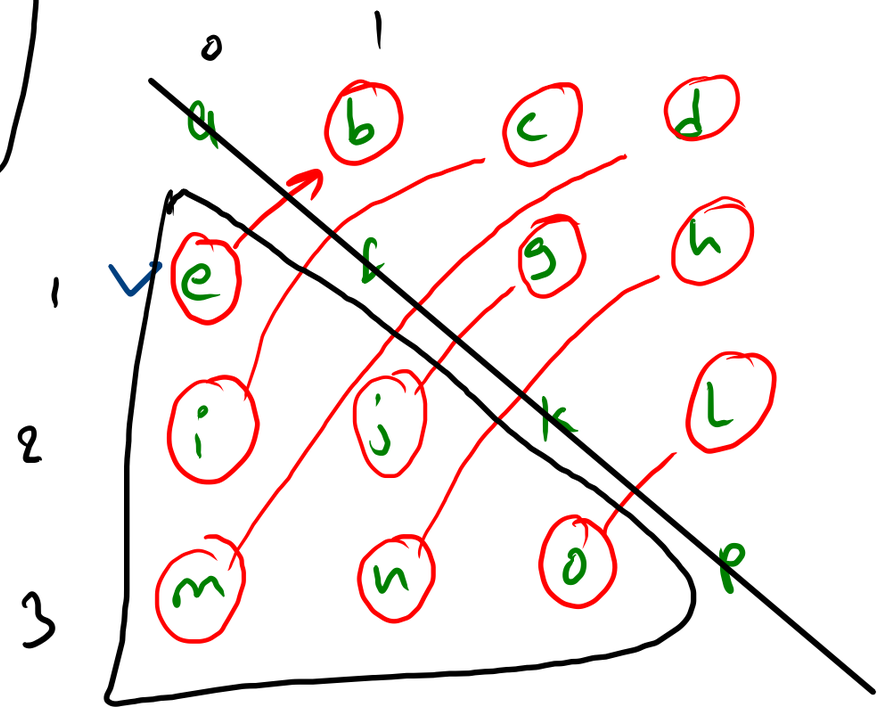






a b → b a

b a → a b



i ← 1 do n-1

j ← 0 do i-1

1 0 → 0 to 0

2 0 , 2, 1 → 0 to 1

30 , 31 , 32 → 0 to 2

i → 0 to i-1

h = 4

m = 5

s = 2

$\gamma = 1$

$\gamma = 2$

$\gamma = -2$

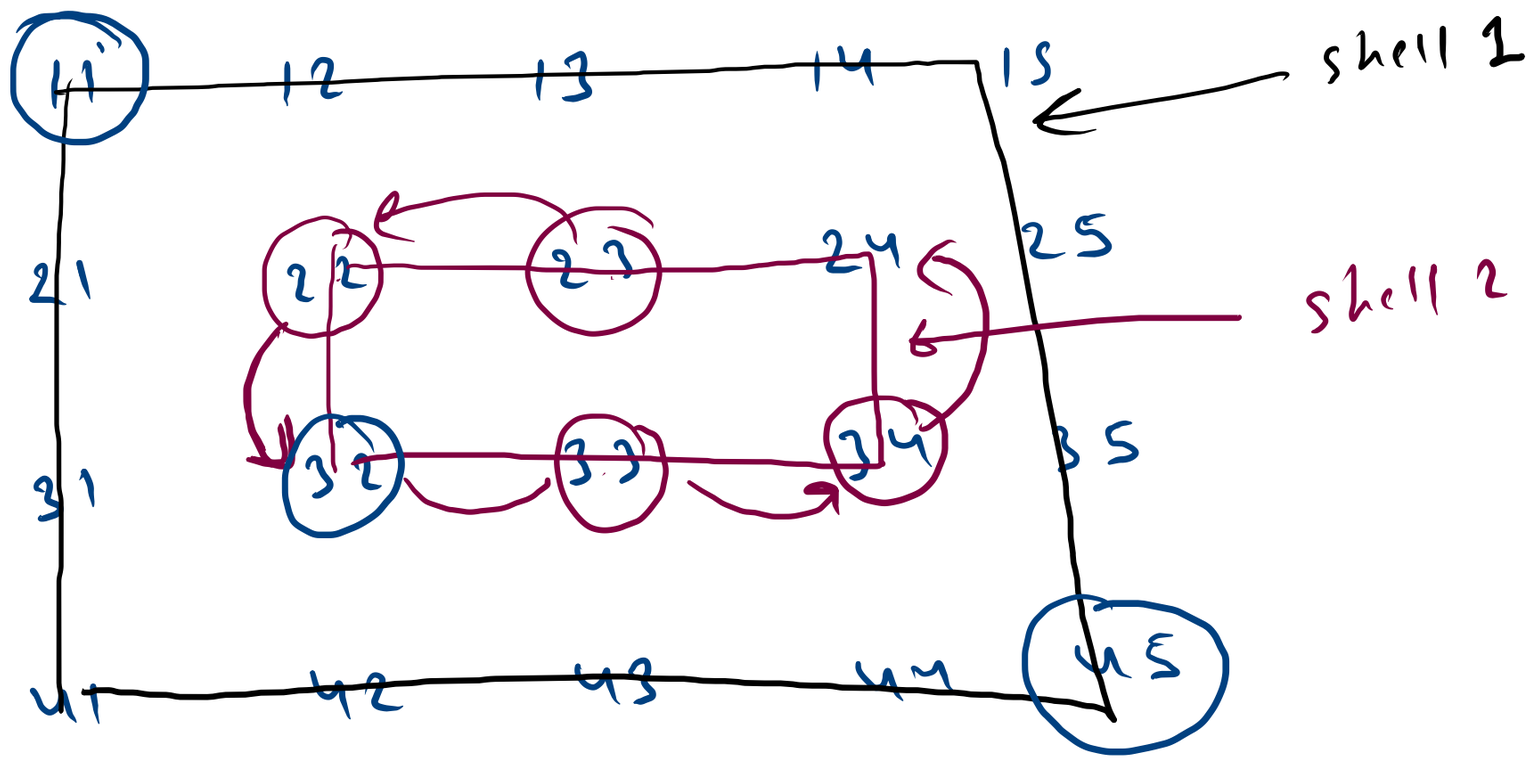
+v

-v

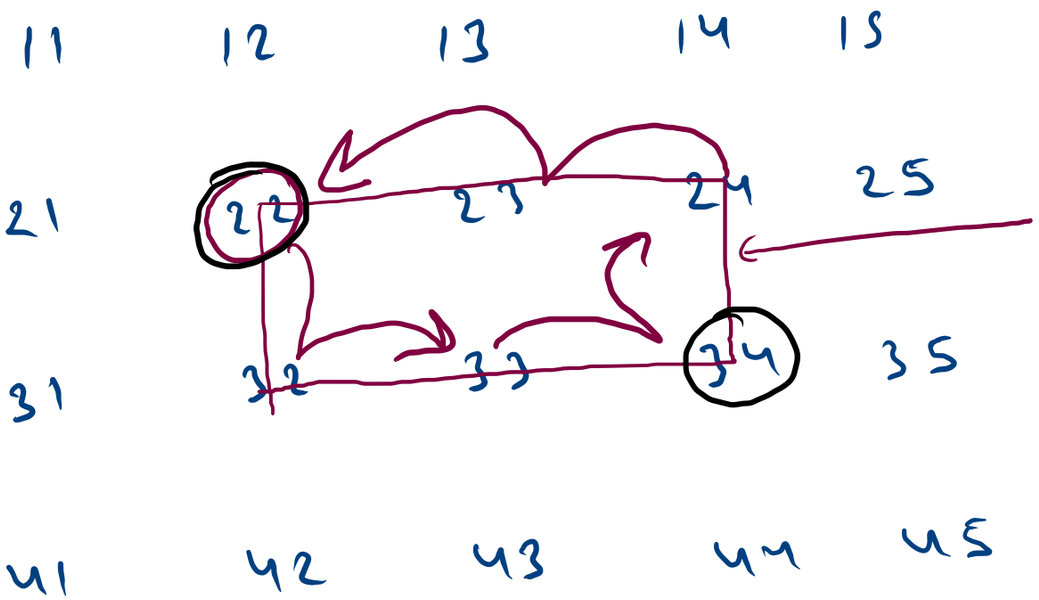
One D

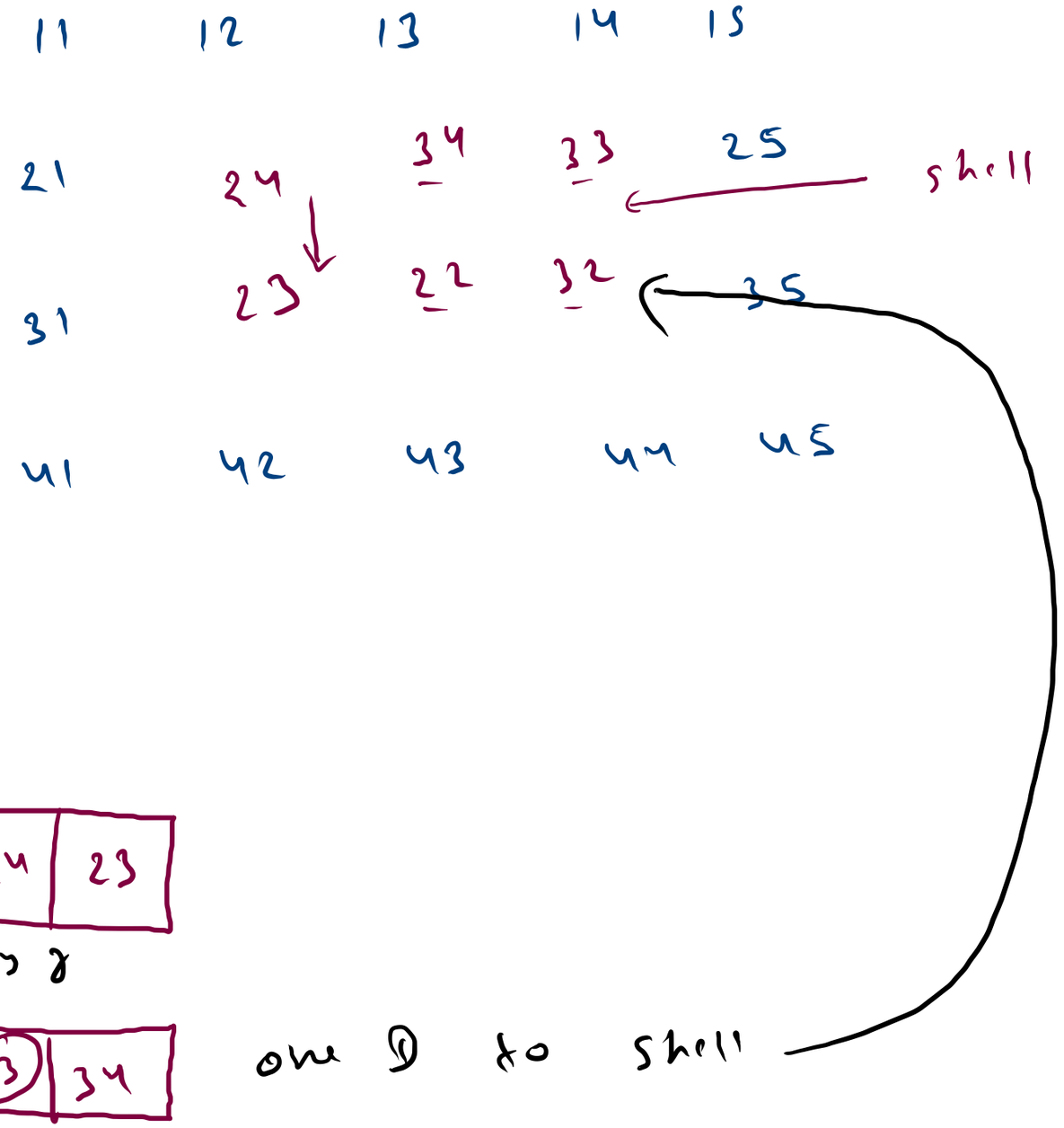
bor travel

shell



11	12	13	14	15
21	23	24	34	25
31	22	32	33	35
41	42	43	44	45



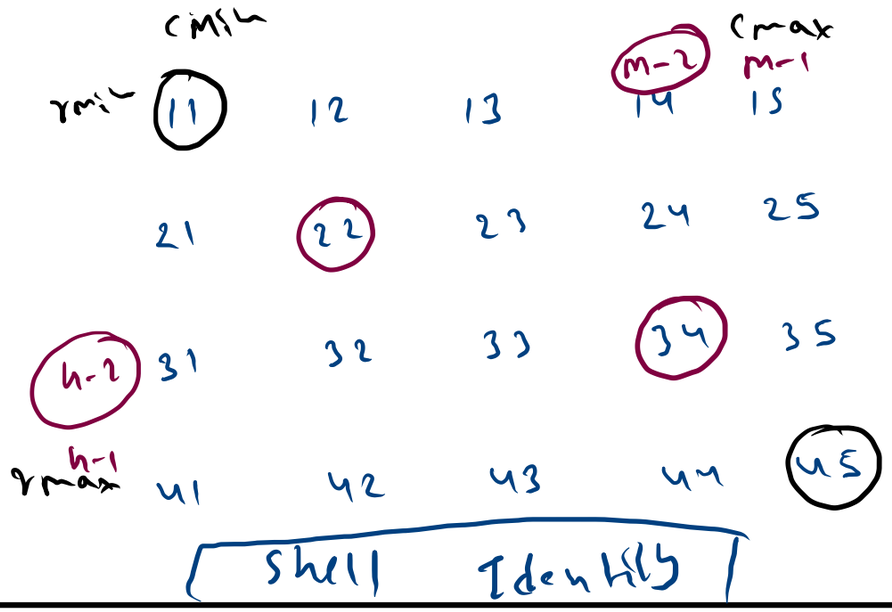


s = 2

r = 2

k : -∞ to ∞

shell to one D



identify shell

size of shell

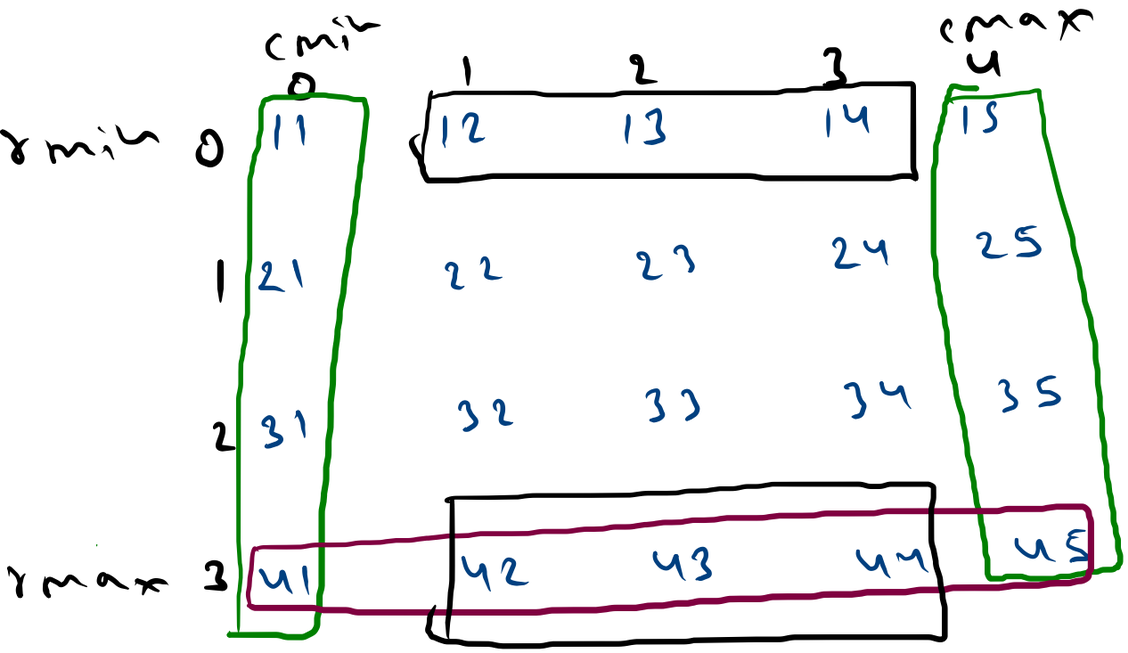
shell to one D array

rotate by r

one D to shell

Displays

	rmin	cmin	rmax	cmax
s = 1	0	0	h-1	m-1
s = 2	1	1	h-2	m-2
s = 3	2	2	h-3	m-3
s	s-1	s-1	h-s	m-s



7 8 9 10

10 20 30 40

↑ ↑

10 hi

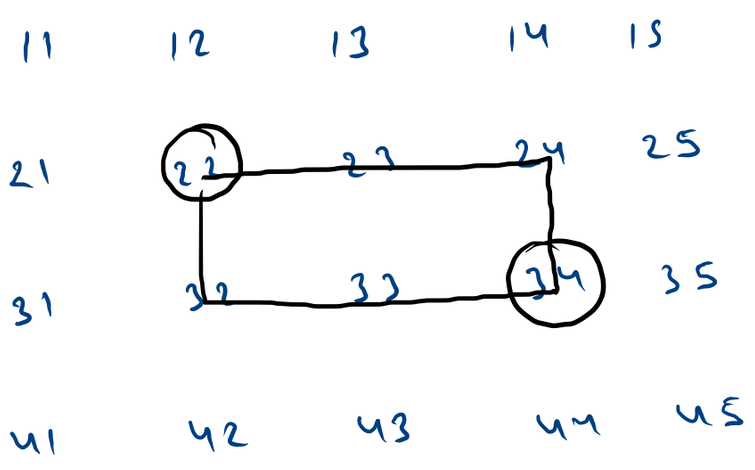
hi - 10 + 1

10 - 7 + 1 = 4

$2(r_{max} - r_{min} + 1) + 2(c_{max} - c_{min} + 1 - 2)$

$2r_{max} - 2r_{min} + 2 + 2c_{max} - 2c_{min} + 2 - 4$

$2(r_{max} - r_{min} + c_{max} - c_{min})$

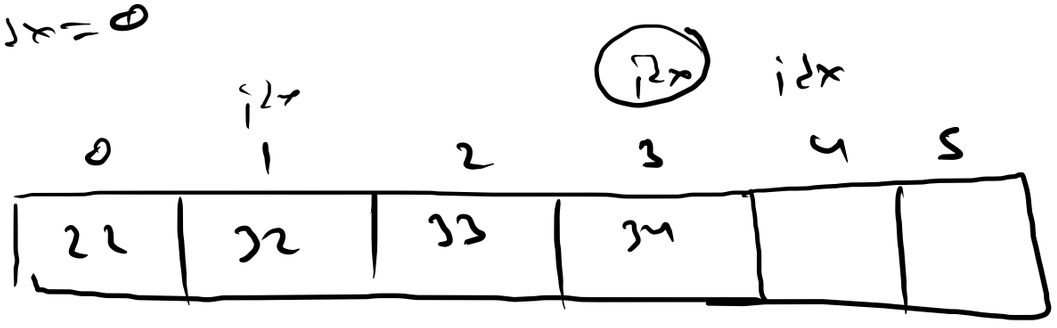


size = 2(rmax - rmin + cmax - cmin)

oneD to shell

```
int [r][c]
{arr[r][c] = oneD[idx]
 idx++;}
```

idx = 0



shell to oneD

```
for( r c )
    int arr[r][c]; → oneD[idx] = arr[r][c]
                     idx++;
```

n = 4

4 11 12 13 14 21
22 23 24 31 32
33 34 41 42 43
44 43

d = 43

n x n

d = 12

	0	1	2
0	11	12	13
1	21	22	23
2	31	32	33
3	41	42	43

14

24 34 44

3
2

d = 55

Not found

d = 32

j

j=1 j=2 3

i 0 11 12 13 14

i=1 21 22 23 24

i=2 31 32 33 34

41 42 43 44

d

d = 32

i==n
j==-1
No bound

	i	j=-1	j	j	j
	i	~~11~~	~~12~~	13	~~14~~
	i	~~21~~	22	~~23~~	~~24~~
	i	31	33	34	35
n-1	i	41	42	43	44

i = 0

j = n-1

if (arr[i][j] < d)

i++

else if(arr[i][j] > d)

j--

d2 ss n i

0 1

0 a_{00} a_{01}

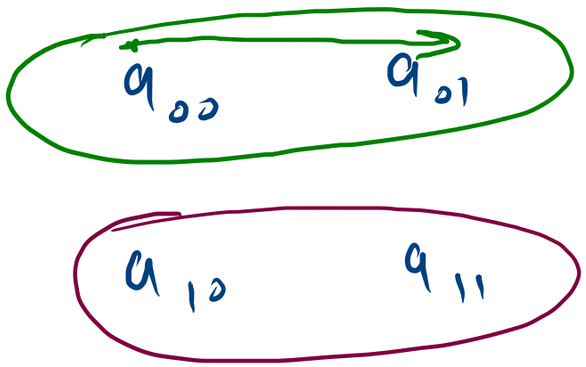
1 a_{10} a_{11}

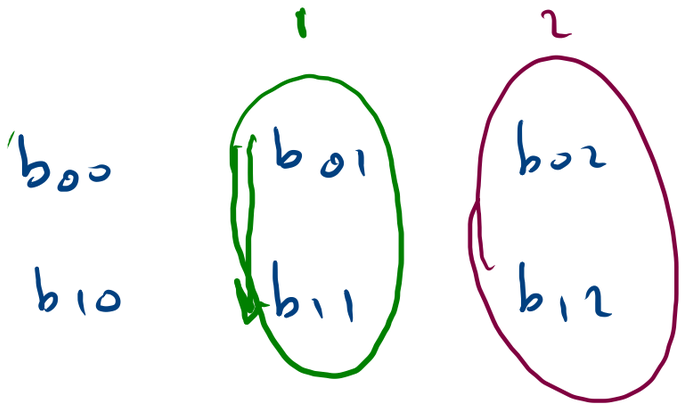
$$\begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \\ b_{20} & b_{21} \end{bmatrix} = \begin{bmatrix} & \end{bmatrix}$$

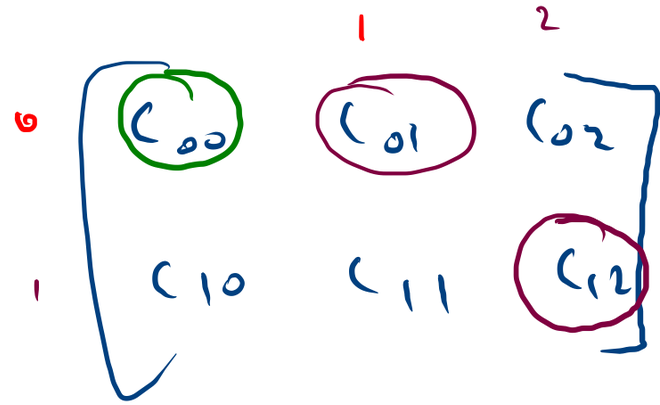
2×2
$n_1 \quad m_1$

3×2
$n_2 \quad m_2$

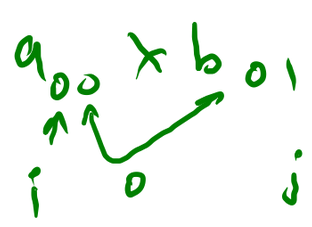
number of column $=$ no: of row

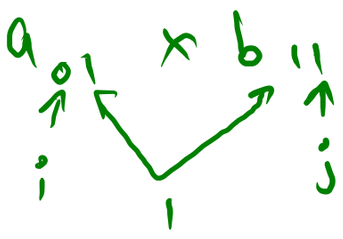






$$C_{01} = a_{00} \times b_{01} + a_{01} \times b_{11}$$
$$i\ j$$

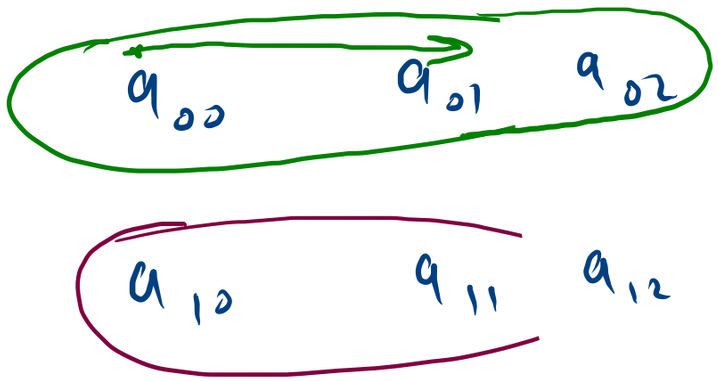




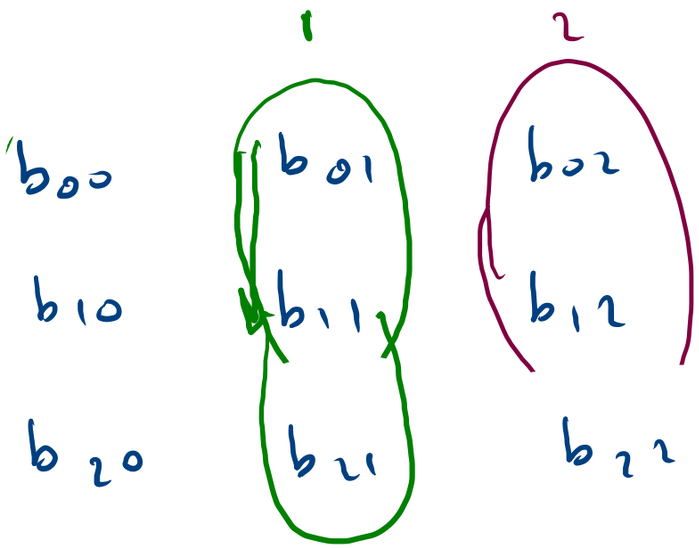
i ← 0 to h1-1

j ← 0 to m2-1

```
int sum=0
k ← 0 to n2-1
    A[i][k] x B[k][j]
ans[i][j] = sum;
```



n_1 m_1



$n_2 \times m_2$

vary 0 to $n_2 - 1$

C_{01} = $a_{00} \times b_{01}$ + $a_{01} \times b_{11}$ + $a_{02} \times b_{21}$

i j

i 0 j — 0

i 1 j

i $\overbrace{\quad}^{2}$ j — $\boxed{n_2 - 1}$