SMART

kapp

2D Matrix

	1	3	5	7 ← l
	10	11	16	20 ↓ m
m	23	30	34	60 ↓ h

t = 16

- $O(m \times n)$ → brute force.
- $(m+n)$ or $(m \times n)$? — Rohit's approach.
- — $m + \log n$
- — $\log m + \log n = \log(mn)$

100×100

$\log(m \times n)$

4

200

1 2 8 9 - l

10 11 13 18 - m

27 29 32 37 - h

4 5 6 8

8 9 12 95.

```
def bs(arr, l, h, t):
    while l <= h:
        mid = l + (h-l)//2        # to avoid integer overflow
        if (arr[mid] == t):
            return mid
        elif (arr[mid] < t):
            bs(arr, mid+1, h, t)
        else:
            bs(arr, l, mid-1, t)
    return -1
```

```
main():
    arr = [1, 2, 3, 8, 9, 11]
    l = 0
    h = len(arr)-1
    T = 9
    bs(arr, l, h, t)
```

Binary Search

1)
arr = [4, 5, 6, 7, 0, 1, 2, 3]
t = 2

approach1 - Linear Search - O(n)

left side is sated

if arr[l] < arr[m]:

if l <= t && t < m

h = mid - 1

else

l = mid + 1

arr = [5, 0, 1, 2, 3]
l = 0, h = len(arr) - 1
t = 2

while(l <= h):

mid = (h+1)/2

if arr[mid] == t:

return mid

else:

right sid e t = 5.

if m < t && t <= h:

l = mid + 1

else.

h = m - 1

arr = [2, 5, 7, 9, 18, 23, 35]
T = 7

while (l <= h)
mid = (l+h)/2
if arr[mid] ==

return mid

elif arr[mid] < T:

l = mid + 1

else:

h = mid - 1

return -1