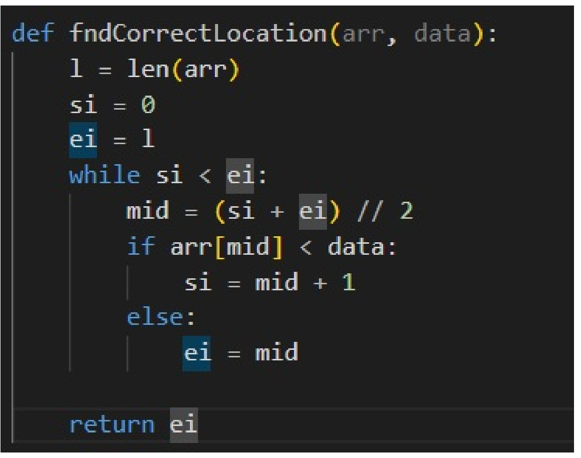


```
def fndCorrectLocation(arr, data):
    l = len(arr)
    si = 0
    ei = l
    while si < ei:
        mid = (si + ei) // 2
        if arr[mid] < data:
            si = mid + 1
        else:
            ei = mid

    return ei
```

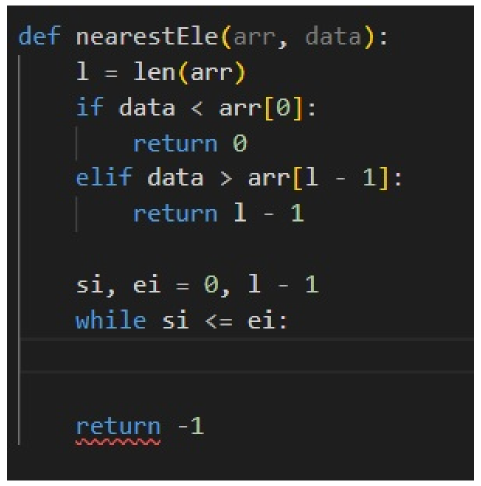
2	2	3	4	10	12	15	18	20	29	40	47	50	53	
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

-1000

① index of data.

9	4
11	5
70	14
1	0
18	7
29	9
41	11

2	2	3	4	10	12	15	18	20	29	40	47	50	53
0	1	2	3	4	5	6	7	8	9	10	11	12	13



```
def nearestEle(arr, data):
    l = len(arr)
    if data < arr[0]:
        return 0
    elif data > arr[l - 1]:
        return l - 1

    si, ei = 0, l - 1
    while si <= ei:


    return -1
```

13 → 12
14 → 15
19 → [18, 20] → 18
1 → 2
70 → 53

data = 19

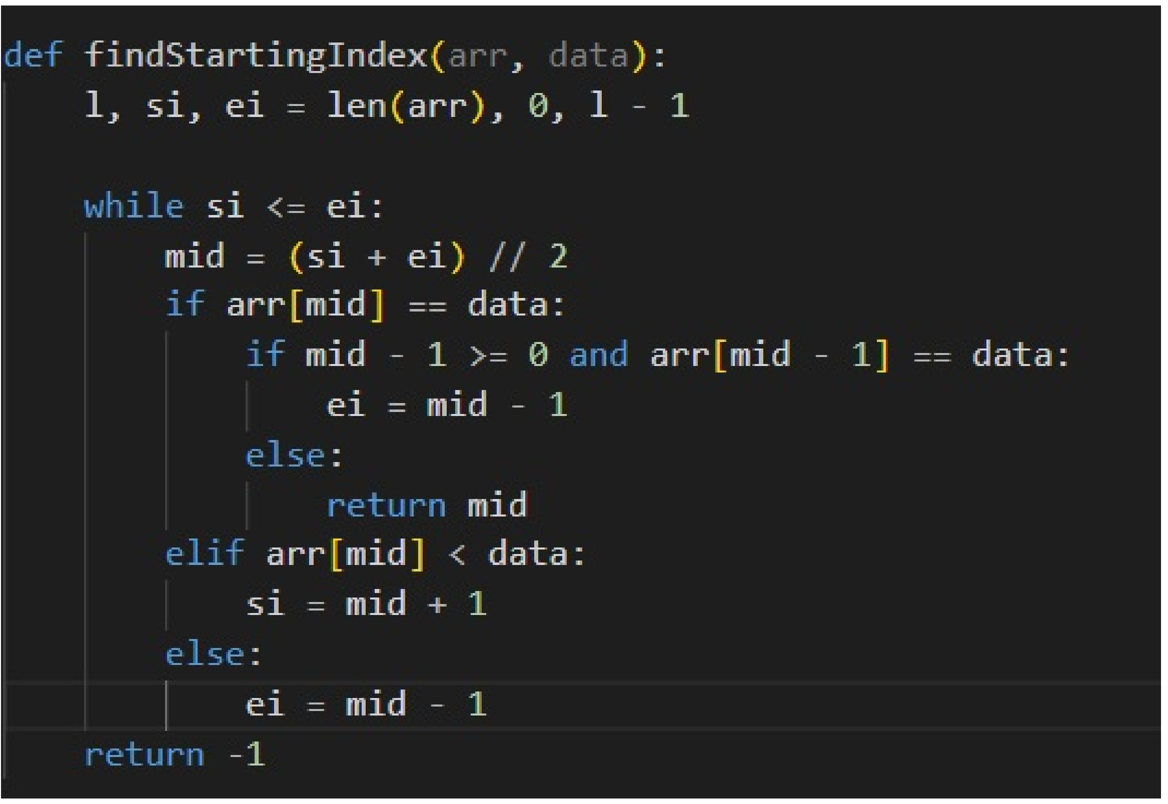
23

19 - 12 < 15 - 13

23 - 20

⓪ < $\cancel{3}$

2	2	3	3	3	3	3	3	3	3	3	3	5	6	6	6	6	12	18
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18



```python
def findStartingIndex(arr, data):
    l, si, ei = len(arr), 0, l - 1

    while si <= ei:
        mid = (si + ei) // 2
        if arr[mid] == data:
            if mid - 1 >= 0 and arr[mid - 1] == data:
                ei = mid - 1
            else:
                return mid
        elif arr[mid] < data:
            si = mid + 1
        else:
            ei = mid - 1
    return -1
```

arr1:

2	3	10	12	14	18	22

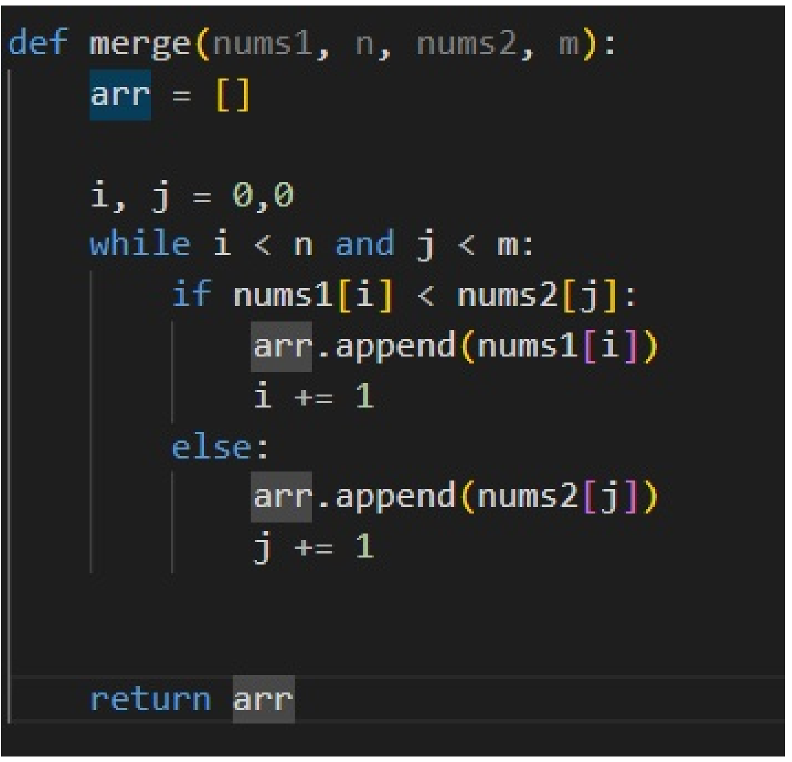
7

i

arr2:

-2	-1	0	0	19	25	28	29	42	45

10



```python
def merge(nums1, n, nums2, m):
    arr = []

    i, j = 0,0
    while i < n and j < m:
        if nums1[i] < nums2[j]:
            arr.append(nums1[i])
            i += 1
        else:
            arr.append(nums2[j])
            j += 1


    return arr
```