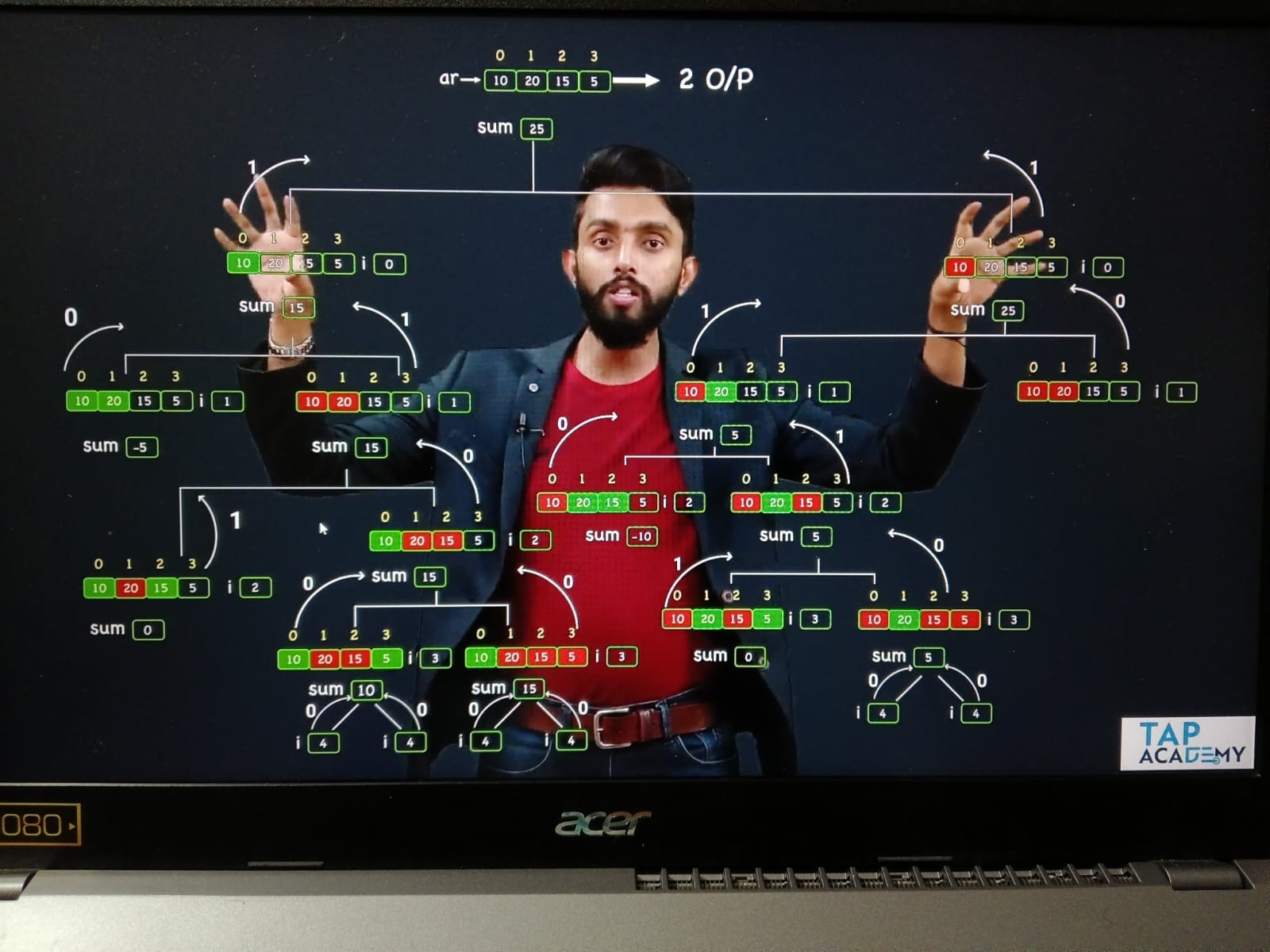
Python – DSA

Recursion

53.SubSet Sum



def count\_subsets(arr, sum, i):

    if sum == 0:

        return 1

    if sum < 0 :

        return 0

    if i == len(arr):

        return 0

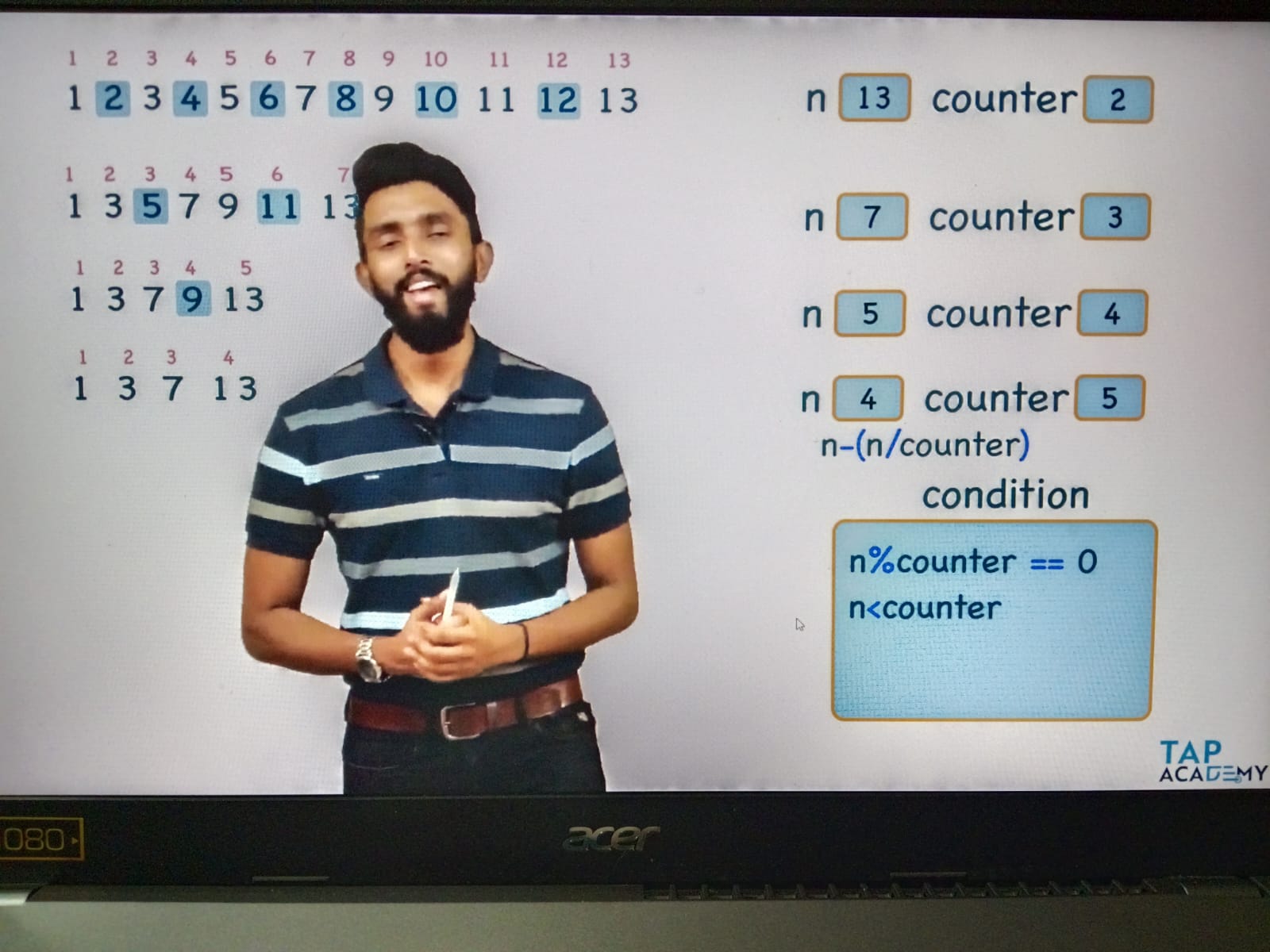
    return count\_subsets(arr , sum - arr[i] , i+1) + count\_subsets(arr, sum , i+1)

def main():

    ar = [10,15,20,5]

    print(count\_subsets(ar,25,0))

54. Lucky Number



def is\_lucky\_number(n,counter):

    if n < counter:

       return True

    if n % counter == 0:

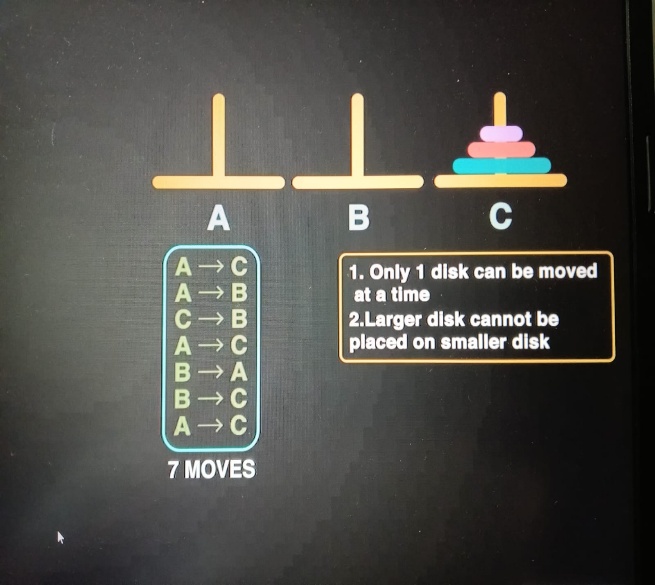
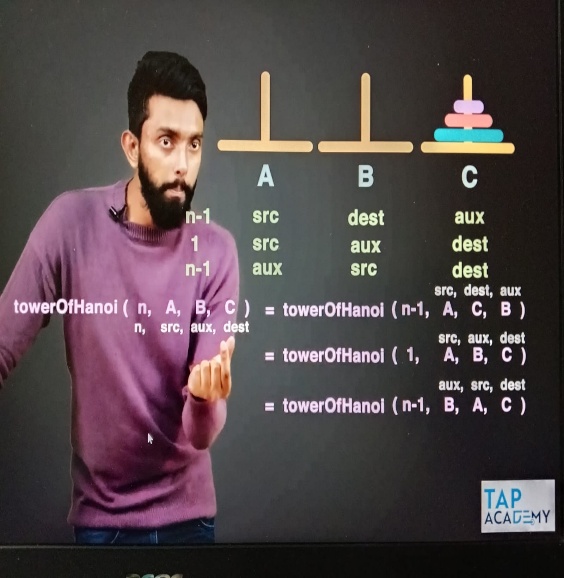
        return False

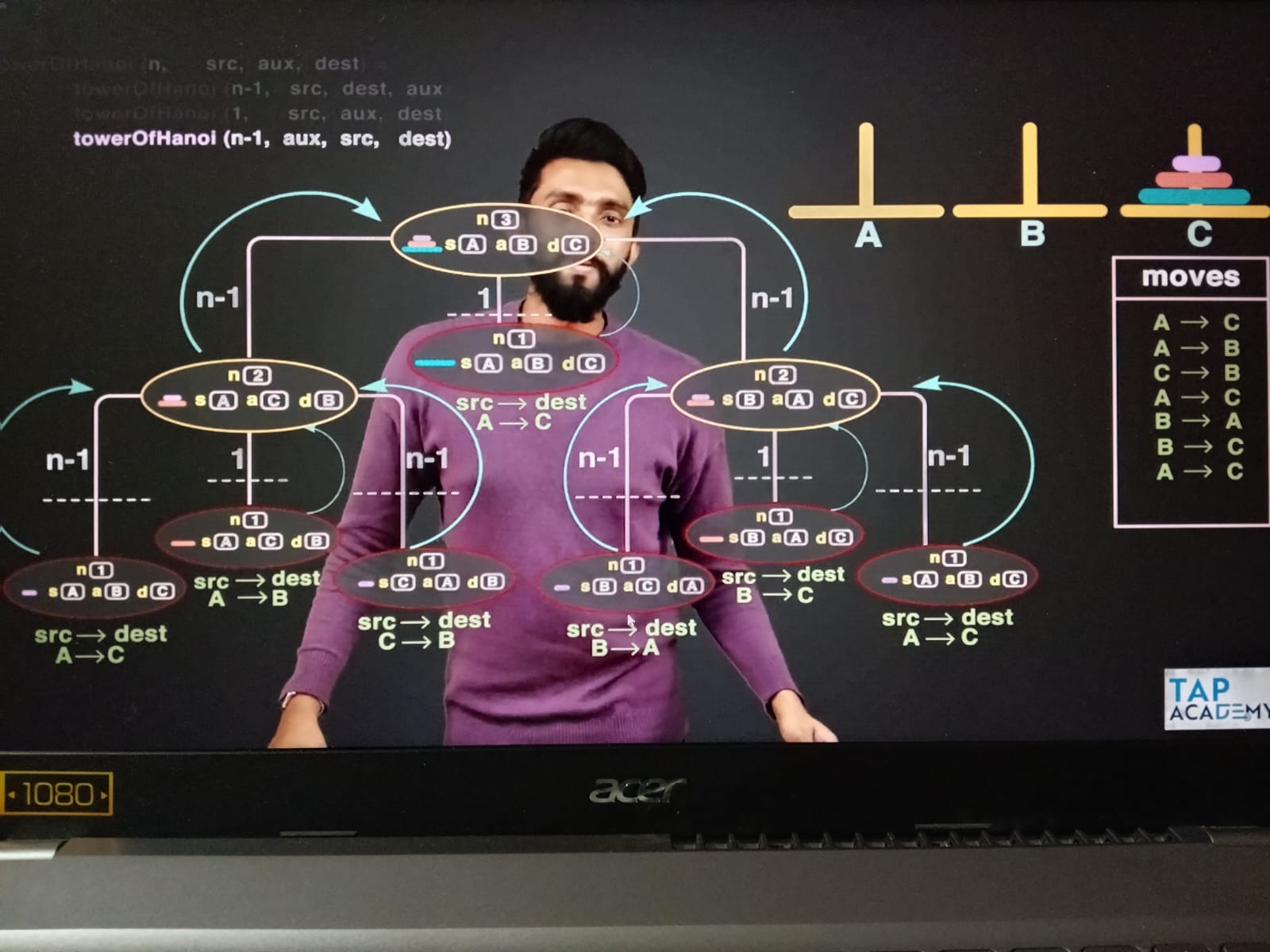
    return is\_lucky\_number( n-(n//counter) , counter+1)

def main():

    print(is\_lucky\_number(9,2))

55.Tower of Honai



def tower\_of\_honai(n , src , aux , dest):

    if n == 1:

       print(src,'-->',dest)

       return

    tower\_of\_honai(n-1,src,dest,aux)

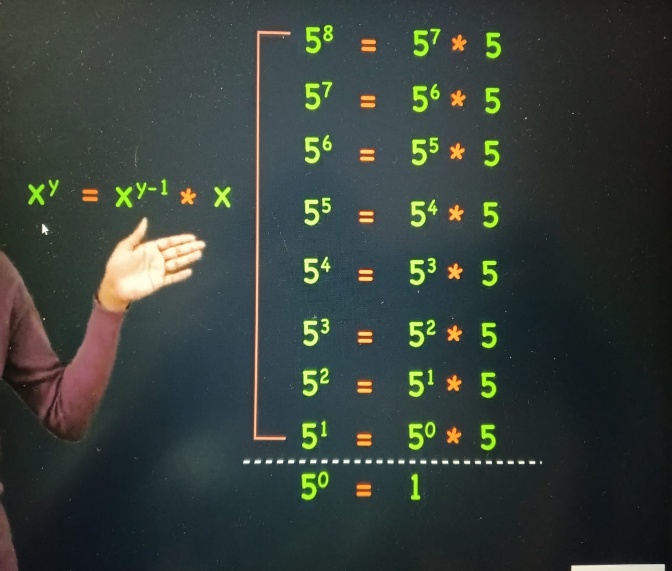
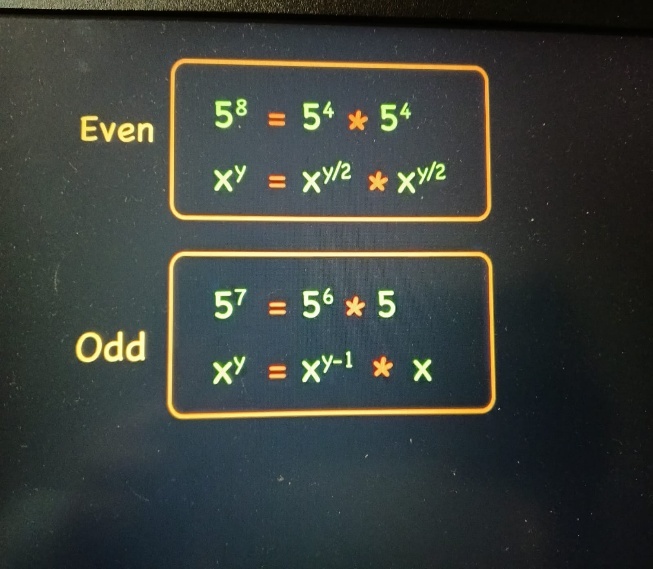
    tower\_of\_honai(1,src,aux,dest)

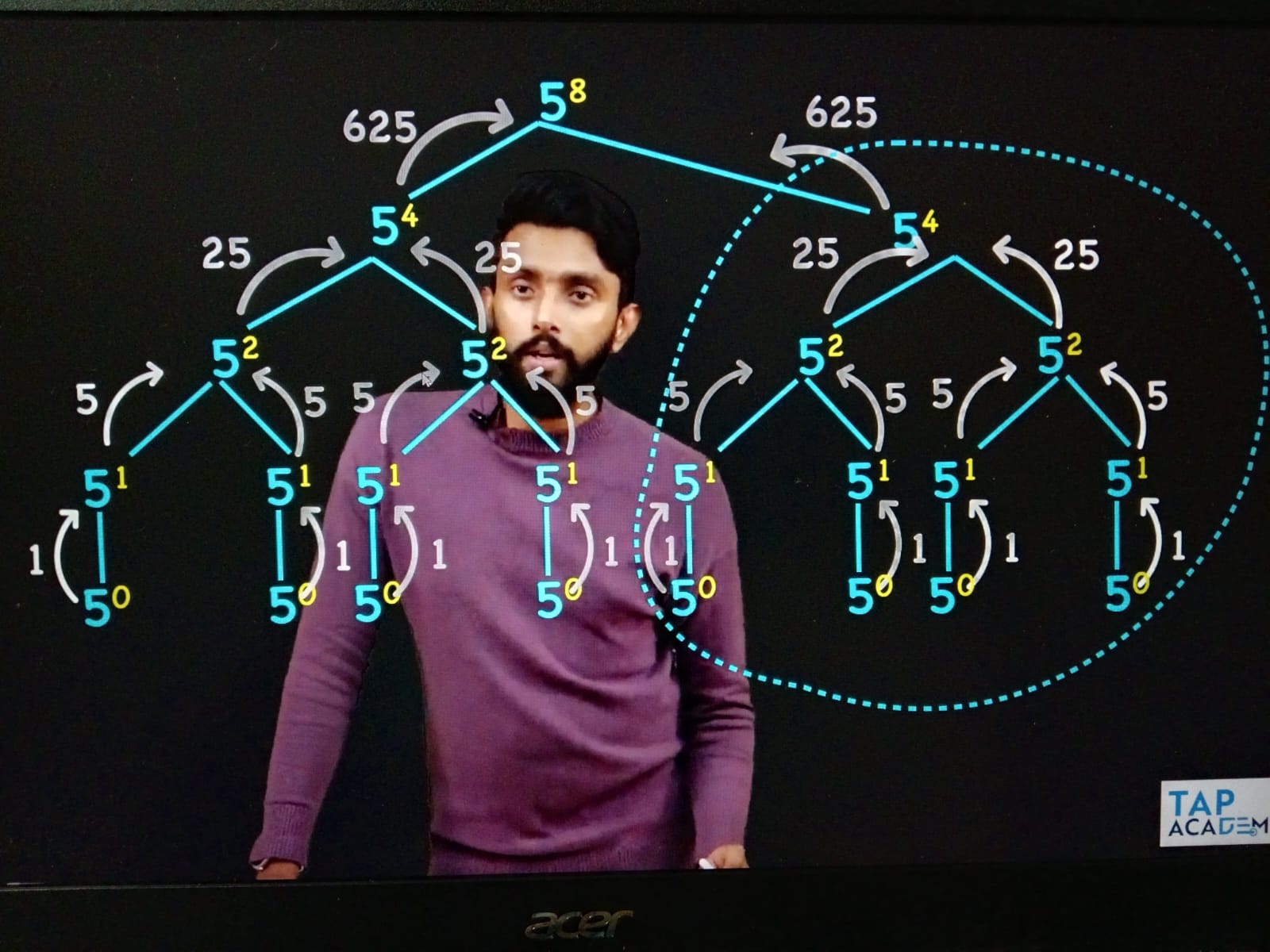
    tower\_of\_honai(n-1,aux,src,dest)

def main():

   tower\_of\_honai(4,'A','B','C')

56.power\_of



def power\_of(x,y):

    if y == 0:

       return 1

    if y % 2 == 0:

       res = power\_of(x,y//2)

       return res \* res

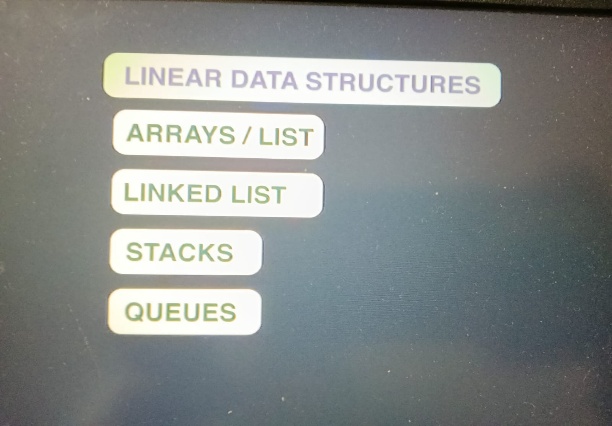
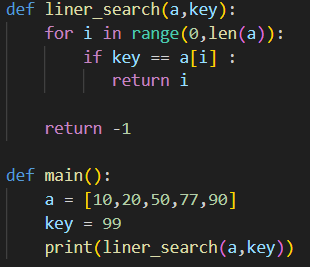
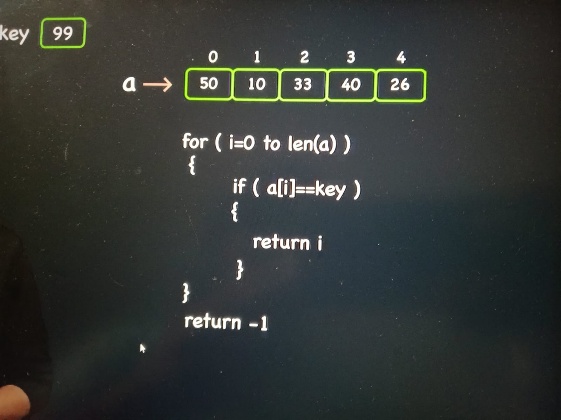
    else:

       return power\_of(x,y-1) \* x

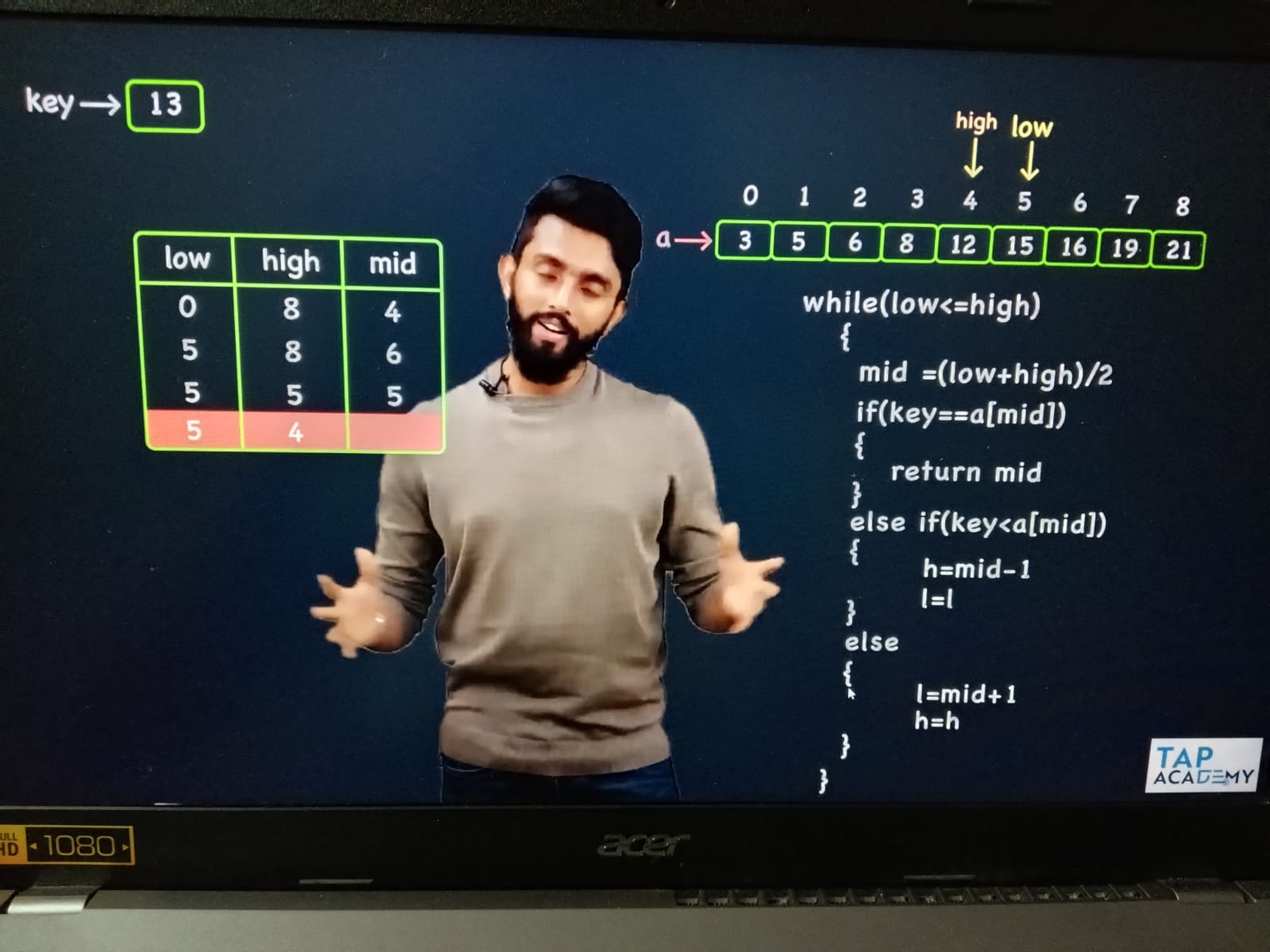
def main():

   print(power\_of(5,2))

6.Linear Data Structures – 57.Linear Search

58. Binaray Search



def binary\_search(arr,key):

    low , high , mid = 0 , len(arr)-1 , 0

    while(low<=high):

        #mid = (low+high)//2

        mid = low + (high - low) // 2

        if arr[mid] == key:

           return mid

        elif arr[mid] > key:

            high = mid - 1

            #low = low

        else:

            low = mid + 1

            #high = high

    return -1

def main():

    arr = [14,5,67,89,2,3,0]

    arr.sort()

    print("Sorted array:", arr)

    print(binary\_search(arr,89))

59. Span of List

Span = max – min

def span\_of\_list(a):

    max = a[0]

    min = a[0]

    for i in range(0,len(a)):

        if a[i] > max:

           max = a[i]

        if a[i] < min:

            min = a[i]

    return max - min

def main():

    a = [10,20,40,99,6]

    print(span\_of\_list(a))

60. Second Largest Element

def second\_largest(arr):

    max1 , max2 = 0 , 0

    if arr[0] > arr[1]:

        max1 , max2 = arr[0], arr[1]

    else:

        max1 , max2 = arr[1], arr[0]

    for i in range(2,len(arr)):

        if max1 < arr[i]:

           max2 , max1 = max1 , arr[i]

        elif max2 < arr[i]:

            max2 = arr[i]

    return max2

def main():

    arr = [20,42,6,25,30,88]

    print(second\_largest(arr))

61.Secnond Smallest Element

def second\_smallest(arr):

    max1 , max2 = 0 , 0

    if arr[0] < arr[1]:

        max1 , max2 = arr[0], arr[1]

    else:

        max1 , max2 = arr[1], arr[0]

    for i in range(2,len(arr)):

        if max1 > arr[i]:

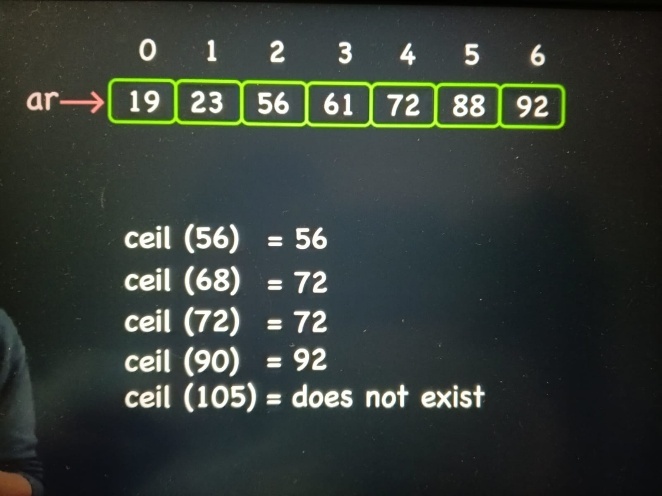
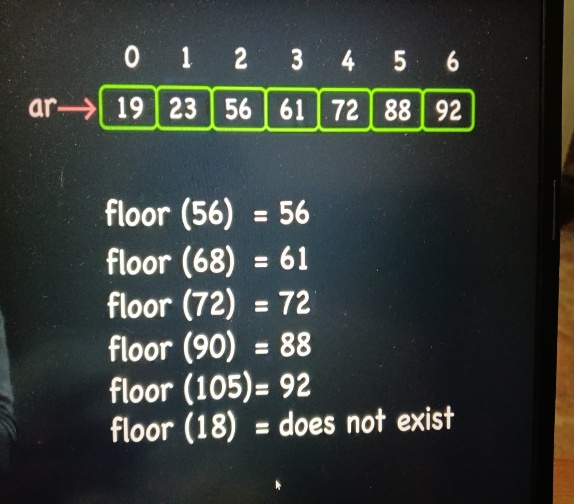
           max2 , max1 = max1 , arr[i]

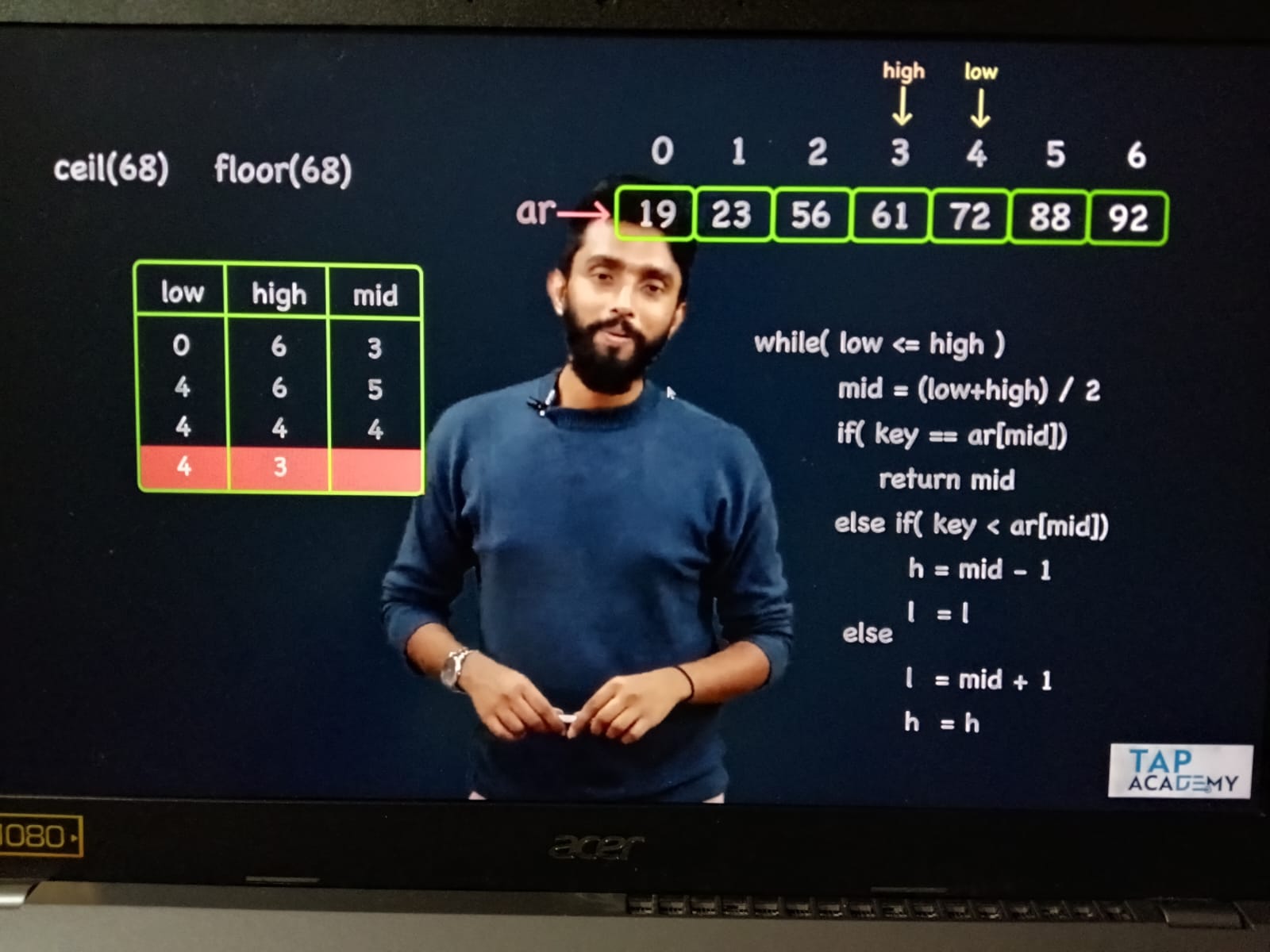
        elif max2 > arr[i]:

            max2 = arr[i]

    return max2

62. Ceil and Floor



def ceil(arr,key):

    low , high , mid = 0 , len(arr)-1 , 0

    while(low<=high):

        #mid = (low+high)//2

        mid = low + (high - low) // 2

        if arr[mid] == key:

           return arr[mid]

        elif arr[mid] > key:

            high = mid - 1

            #low = low

        else:

            low = mid + 1

            #high = high

    if low < len(arr):

        return arr[low]

    else:

        return -1

def floor(arr,key):

    low , high , mid = 0 , len(arr)-1 , 0

    while(low<=high):

        #mid = (low+high)//2

        mid = low + (high - low) // 2

        if arr[mid] == key:

           return arr[mid]

        elif arr[mid] > key:

            high = mid - 1

            #low = low

        else:

            low = mid + 1

            #high = high

    if high >= 0:

        return arr[high]

    else:

        return -1

def main():

    arr = [19,23,56,61,72,88,92]

    print(ceil(arr,68))

    print(floor(arr,70))