

WEEK-07-MCQ-Lists

1.What will be the output after the following statements?

```
m = [5, 10, 35]
```

```
del m[:]
```

```
print(m)
```

- a. [5, 10, 35] b. 5, 10, 35 c. [] d. [5, 35]

2. L=[1,5,9]

```
print(sum(L),max(L),min(L))
```

what will be the expected output for this following?

Answer:15 9 1

3. What will be the output after the following statements?

```
m = ['July', 'September', 'December']
```

```
n = m[0] + m[2]
```

```
print( n )
```

- a.SeptemberDecember b.JulyDecember c.JulySeptember d.July

4. L=['Amit','Anita','Zee','Longest Word']

```
print(max(L))
```

what will be the expected output for this following?

Answer: Longest Word

5. Fill in the blanks with same word in both places

```
>>> import _____
```

```
>>> list1 = [1,2,3,4,5]
```

```
>>> list2 = _____copy(list1)
```

```
>>> list2
```

- a.Pickle b.Math c.Copy

6. Write the output of the following :

```
def listchange(L):  
    L.append(45)  
    return  
    L1 = [1, 2, 3, 4]  
    listchange(L1)  
    print(L1)
```

what will be the expected output for this following?

Answer: [1,2,3,4,45]

7. What will be the output after the following statements?

```
m = 'A'  
n = 'B'  
o = 'C'  
p = [m, n, o]  
print(p)
```

- a. 'C', 'A', 'B' b. ['C', 'B', 'A'] c. ['A', 'B', 'C'] d. ['C', 'A', 'B']

8. What will be the output after the following statements?

```
m = [4, 8]  
print(m * 3)
```

- a. [4, 8, 4, 8, 4, 8] b. [4, 8, 4, 8] c. [4, 8] d. [4, 8] * 3

9. What will be the output after the following statements?

```
m = [75, 23, 64]  
n = m[0] + m[1]  
print(n)
```

- a.75 b.23 c.98 d.64

10. What is the data type of m after the following statement?

```
m = ['July', 'September', 'December']
```

- a.String b.Tuple c.Dictionary d.List

WEEK-07-CODING-Lists

1.Program to print all the distinct elements in an array. Distinct elements are nothing but the unique (non-duplicate) elements present in the given array.

Input Format:

First line take an Integer input from stdin which is array length n.

Second line take n Integers which is inputs of array.

Output Format:

Print the Distinct Elements in Array in single line which is space Separated

Example Input:

5

1

2

2

3

4

Output:

1 2 3 4

Example Input:

6

1

1

2

2

3

3

Output:

1 2 3

For example:

Input	Result
5 1 2 2 3	1 2 3 4

Input	Result
4	
6 1 1 2 2 3 3	1 2 3

Coding:

```
def ele():
    n=int(input().strip())
    a=[]
    for _ in range(n):
        while True:
            e=int(input().strip())
            a.append(e)
            break

    d=set(a)
    d=sorted(d)
    print(' '.join(map(str,d)))
ele()
```

Output:

	Input	Expected	Got	
✓	5 1 2 2 3 4	1 2 3 4	1 2 3 4	✓
✓	6 1 1 2 2 3 3	1 2 3	1 2 3	✓

Passed all tests! ✓

2. Given an integer n, return an list of length n + 1 such that for each i (0 <= i <= n), ans[i] is the number of 1's in the binary representation of i.

Example:

Input: n = 2

Output: [0,1,1]

Explanation:

0 --> 0

1 --> 1

2 --> 10

Example2:

Input: n = 5

Output: [0,1,1,2,1,2]

Explanation:

0 --> 0

1 --> 1

2 --> 10

3 --> 11

4 --> 100

5 --> 101

Note: Complete the given function alone

For example:

Test	Result
print(CountingBits(5))	[0, 1, 1, 2, 1, 2]

Coding:

```
def CountingBits(n):  
    res=[]  
    for i in range(n+1):  
        res.append(bin(i).count('1'))  
    return res
```

Output:

	Test	Expected	Got	
✓	print(CountingBits(2))	[0, 1, 1]	[0, 1, 1]	✓
✓	print(CountingBits(5))	[0, 1, 1, 2, 1, 2]	[0, 1, 1, 2, 1, 2]	✓

Passed all tests! ✓

3. Given an array A of sorted integers and another non negative integer k, find if there exists 2 indices i and j such that $A[i] - A[j] = k$, $i \neq j$.

Input Format

1. First line is number of test cases T. Following T lines contain:
2. N, followed by N integers of the array
3. The non-negative integer k

Output format

Print 1 if such a pair exists and 0 if it doesn't.

Example

Input

1

3

1

3

5

4

Output:

1

Input

1

3

1

3

5

99

Output

0

For example:

Input	Result
1 3 1 3 5 4	1
1 3	0

Input	Result
1 3 5 99	

Coding:

```
def pair(arr,n,k):
```

```
    i , j = 0, 1
```

```
    while i < n and j < n:
```

```
        if i != j and arr[j]-arr[i]==k:
```

```
            return 1
```

```
        elif arr[j] - arr[i] < k:
```

```
            j +=1
```

```
        else:
```

```
            i +=1
```

```
        if i == j:
```

```
            j +=1
```

```
    return 0
```

```
def main():
```

```
    t = int(input().strip())
```

```
    for _ in range(t):
```

```
        n = int(input().strip())
```

```
        arr = []
```

```
        for _ in range(n):
```

```
            arr.append(int(input().strip()))
```

```
        k = int(input().strip())
```

```
        print(pair(arr,n,k))
```

```
if __name__ == "__main__":
```

```
    main()
```

Output:

	Input	Expected	Got	
✓	1 3 1 3 5 4	1	1	✓
✓	1 3 1 3 5 99	0	0	✓

Passed all tests! ✓

4. Complete the program to count frequency of each element of an array. Frequency of a particular element will be printed once.

Sample Test Cases

Test Case 1

Input

7
23
45
23
56
45
23
40

Output

23 occurs 3 times
45 occurs 2 times
56 occurs 1 times
40 occurs 1 times

Coding:

```
def fry(ar):  
    f={}  
    for e in ar:  
  
        if e in f:  
            f[e] +=1  
        else:  
            f[e] = 1  
  
    for e, value in f.items():  
        print(f"{e} occurs {value} times")  
  
n = int(input())  
ar=[]  
for i in range(n):  
    e=int(input())  
    ar.append(e)  
  
fry(ar)
```

Output:

	Input	Expected	Got	
✓	7	23 occurs 3 times	23 occurs 3 times	✓
	23	45 occurs 2 times	45 occurs 2 times	
	45	56 occurs 1 times	56 occurs 1 times	
	23	40 occurs 1 times	40 occurs 1 times	
	56			
	45			
	23			
	40			

Passed all tests! ✓

5. An array is monotonic if it is either **monotone increasing** or **monotone decreasing**. An array A is monotone increasing if for all $i \leq j$, $A[i] \leq A[j]$. An array A is monotone decreasing if for all $i \leq j$, $A[i] \geq A[j]$.

Write a program if n array is monotonic or not. Print "True" if is monotonic or "False" if it is not. Array can be monotone increasing or decreasing.

Input Format:

First line n-get number of elements

Next n Lines is the array of elements

Output Format:

True ,if array is monotone increasing or decreasing.

otherwise False is printed

Sample Input1

4

5

6

7

8

Sample Output1

True

Sample Input2

4

6

5

4

3

Sample Output2

True

Sample Input 3

4

6

7

8

7

Sample Output3

False

For example:

Input	Result
4	True
6	
5	
4	
3	

Coding:

```
n=int(input())
```

```
arr =[int(input()) for _ in range(n)]
```

```
inc = True
```

```
dec = True
```

```
for i in range(1,n):  
    if arr[i] > arr[i-1]:  
        dec = False  
    if arr[i] < arr[i-1]:  
        inc = False
```

```
if inc or dec:  
    print("True")  
else:  
    print("False")
```

Output:

	Input	Expected	Got	
✓	4 6 5 4 3	True	True	✓
✓	4 3 5 7 4	False	False	✓
✓	4 1 6 9 2	False	False	✓
✓	4 9 6 4 2	True	True	✓
✓	3 2 1 4	False	False	✓

Passed all tests! ✓

6. Given two arrays of positive integers, for each element in the second array, find the total number of elements in the first array which are *less than or equal to* that element. Store the values determined in an array.

For example, if the first array is $[1, 2, 3]$ and the second array is $[2, 4]$, then there are 2 elements in the first array *less than or equal to* 2. There are 3 elements in the first array which are *less than or equal to* 4. We can store these answers in an array, $answer = [2, 3]$.

Program Description

The program must return an array of m positive integers, one for each $maxes[i]$ representing the total number of elements $nums[j]$ satisfying $nums[j] \leq maxes[i]$ where $0 \leq j < n$ and $0 \leq i < m$, in the given order.

The program has the following:

$nums[nums[0], \dots, nums[n-1]]$: first array of positive integers

$maxes[maxes[0], \dots, maxes[m-1]]$: second array of positive integers

Constraints

- $2 \leq n, m \leq 10^5$
- $1 \leq nums[j] \leq 10^9$, where $0 \leq j < n$.
- $1 \leq maxes[i] \leq 10^9$, where $0 \leq i < m$.

Input Format For Custom Testing

Input from stdin will be processed as follows and passed to the program.

The first line contains an integer n , the number of elements in $nums$.

The next n lines each contain an integer describing $nums[j]$ where $0 \leq j < n$.

The next line contains an integer m , the number of elements in $maxes$.

The next m lines each contain an integer describing $maxes[i]$ where $0 \leq i < m$.

Sample Case 0

Sample Input 0

```
4
1
4
2
4
2
3
5
```

Sample Output 0

```
2
4
```

Explanation 0

We are given $n = 4$, $nums = [1, 4, 2, 4]$, $m = 2$, and $maxes = [3, 5]$.

1. For $maxes[0] = 3$, we have 2 elements in $nums$ ($nums[0] = 1$ and $nums[2] = 2$) that are $\leq maxes[0]$.
2. For $maxes[1] = 5$, we have 4 elements in $nums$ ($nums[0] = 1$, $nums[1] = 4$, $nums[2] = 2$, and $nums[3] = 4$) that are $\leq maxes[1]$.

Thus, the program returns the array $[2, 4]$ as the answer.

Sample Case 1

Sample Input 1

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

Sample Output 1

```
1
0
```

3
4

Explanation 1

We are given, $n = 5$, $nums = [2, 10, 5, 4, 8]$, $m = 4$, and $maxes = [3, 1, 7, 8]$.

1. For $maxes[0] = 3$, we have 1 element in $nums$ ($nums[0] = 2$) that is $\leq maxes[0]$.
2. For $maxes[1] = 1$, there are 0 elements in $nums$ that are $\leq maxes[1]$.
3. For $maxes[2] = 7$, we have 3 elements in $nums$ ($nums[0] = 2$, $nums[2] = 5$, and $nums[3] = 4$) that are $\leq maxes[2]$.
4. For $maxes[3] = 8$, we have 4 elements in $nums$ ($nums[0] = 2$, $nums[2] = 5$, $nums[3] = 4$, and $nums[4] = 8$) that are $\leq maxes[3]$.

Thus, the program returns the array $[1, 0, 3, 4]$ as the answer.

Coding:

```
import bisect
n= int(input())
nums =[int(input()) for _ in range(n)]
m= int(input())
maxes = [int(input()) for _ in range(m)]

nums.sort()

result = []
for maxval in maxes:
    count = bisect.bisect_right(nums, maxval)
    result.append(count)

for res in result:
    print(res)
```

Output:

	Input	Expected	Got	
✓	4	2	2	✓
	1	4	4	
	4			
	2			
	4			
	2			
	3			
	5			
✓	5	1	1	✓
	2	0	0	
	10	3	3	
	5	4	4	
	4			
	8			
	4			
	3			
	1			
	7			
	8			

Passed all tests! ✓

7.The program must accept **N** integers and an integer **K** as the input. The program must print every K integers in descending order as the output.

-

Note: If $N \% K \neq 0$, then sort the final $N\%K$ integers in descending order.

Boundary Condition(s):

$1 \leq N \leq 10^4$

$-99999 \leq \text{Array Element Value} \leq 99999$

Input Format:

The first line contains the values of N and K separated by a space.
The second line contains N integers separated by space(s).

Output Format:

The first line contains N integers.

Example Input/Output 1:

Input:

```
7 3
48 541 23 68 13 41 6
```

Output:

```
541 48 23 68 41 13 6
```

Explanation:

The first three integers are 48 541 23, after sorting in descending order the integers are **541 48 23**.

The second three integers are 68 13 41, after sorting in descending order the integers are **68 41 13**.

The last integer is **6**.

The integers are **541 48 23 68 41 13 6**
Hence the output is **541 48 23 68 41 13 6**.

Coding:

```
N, K = map(int, input().split())

a = list(map(int, input().split()))

if len(a) < N:
    a.extend([0] * (N - len(array)))

result = []

for i in range(0, N, K):
    chunk = a[i:i + K]
    result.extend(sorted(chunk, reverse=True))

print(" ".join(map(str, result)))
```

Output:

	Input	Expected	Got	
✓	7 3 48 541 23 68 13 41 6	541 48 23 68 41 13 6	541 48 23 68 41 13 6	✓

Passed all tests! ✓

8. Assume you have an array of length n initialized with all **0**'s and are given k update operations.

Each operation is represented as a triplet: **[startIndex, endIndex, inc]** which increments each element of subarray **A[startIndex ... endIndex]** (startIndex and endIndex inclusive) with **inc**.

Return the modified array after all k operations were executed.

Example:

Input:

5
3
1 3 2
2 4 3
0 2 -2

Output:

-2 0 3 5 3

Explanation:

Initial state:

length = 5, updates = [[1,3,2],[2,4,3],[0,2,-2]]

[0,0,0,0,0]

After applying operation [1,3,2]:

[0,2,2,2,0]

After applying operation [2,4,3]:

[0,2,5,5,3]

After applying operation [0,2,-2]:

[-2,0,3,5,3]

Coding:

```
n= int(input())
```

```
k= int(input())
```

```
a = [0]* n
```

```
for _ in range(k):
```

```
    Sl, El, inc= map(int, input().split())
```

```
    for i in range(Sl, El +1):
```

```
        a[i] += inc
```

```
print(" ".join(map(str,a)))
```

Output:

	Input	Expected	Got	
✓	5 3 1 3 2 2 4 3 0 2 -2	-2 0 3 5 3	-2 0 3 5 3	✓

Passed all tests! ✓

9. Given a matrix mat where every row is sorted in **strictly increasing** order, return the **smallest common element** in all rows.

If there is no common element, return -1.

Example 1:

Input:

```
4 5
1 2 3 4 5
2 4 5 8 10
3 5 7 9 11
1 3 5 7 9
```

Output:

5

Constraints:

- $1 \leq \text{mat.length}, \text{mat}[i].\text{length} \leq 500$
- $1 \leq \text{mat}[i][j] \leq 10^4$
- $\text{mat}[i]$ is sorted in strictly increasing order.

Coding:

```
import sys
from collections import Counter
f = input().strip()
n = int(f.strip()[0])

m = []

for _ in range(n):
    r = list(map(int, input().split()))
    m.append(r)

counter = Counter()

for r in m:
    u = set(r)
    for e in u:
        counter[e] += 1

small = float('inf')
```

```

for e, count in counter.items():
    if count == n and e < small:
        small = e
if small == float('inf'):
    print(-1)
else:
    print(small)

```

Output:

	Input	Expected	Got	
✓	4 5 1 2 3 4 5 2 4 5 8 10 3 5 7 9 11 1 3 5 7 9	5	5	✓

Passed all tests! ✓

10. Determine the factors of a number (i.e., all positive integer values that evenly divide into a number) and then return the p^{th} element of the list, sorted ascending. If there is no p^{th} element, return 0.

Example

$n = 20$

$p = 3$

The factors of 20 in ascending order are {1, 2, 4, 5, 10, 20}. Using 1-based indexing, if $p = 3$, then 4 is returned. If $p > 6$, 0 would be returned.

Constraints

$1 \leq n \leq 10^{15}$

$1 \leq p \leq 10^9$

The first line contains an integer n , the number to factor.

The second line contains an integer p , the 1-based index of the factor to return.

Sample Case 0

Sample Input 0

10

3

Sample Output 0

5

Explanation 0

Factoring $n = 10$ results in {1, 2, 5, 10}. Return the $p = 3^{\text{rd}}$ factor, 5, as the answer.

Sample Case 1

Sample Input 1

10

5

Sample Output 1

0

Explanation 1

Factoring $n = 10$ results in $\{1, 2, 5, 10\}$. There are only 4 factors and $p = 5$, therefore 0 is returned as the answer.

Sample Case 2**Sample Input 2**

1

1

Sample Output 2

1

Explanation 2

Factoring $n = 1$ results in $\{1\}$. The $p = 1$ st factor of 1 is returned as the answer.

For example:

Input	Result
10 3	5
10 5	0
1 1	1

Coding:

```
def fa(n,p):  
    f=[i for i in range(1,n+1) if n%i ==0]  
    f.sort()
```

```
    if p <= len(f):  
        return f[p-1]  
    else:  
        return 0
```

```
n = int(input())  
p = int(input())  
print(fa(n,p))
```

Output:

	Input	Expected	Got	
✓	10 3	5	5	✓
✓	10 5	0	0	✓
✓	1 1	1	1	✓

Passed all tests! ✓